ISYS613 Course Project Fall 2021

Project Overview

The objective of this project is to afford you the opportunity to demonstrate the skills acquired over the course of the semester. Specifically, you will be constructing an analytics-ready dataset using data from several disparate, but related data sources.

Your dataset is to be a pandas dataframe suitable for answering the proof-of-concept questions located at the end of this document.

The Scenario

You are the proprietor of a small tour company. You are interested in launching a new app that will allow users to undertake self-guided tours of noteworthy authors. To launch your app you need to assemble the authors' location information along with some other facts that you believe will be of interest to your customers.

Data Sources

You will be collecting and assembling data from several different sources and formats.

1. Source 1 - web scraping
2. Source 2 - delimitted file
3. Source 3 - direct download XML file
4. Source 4 - Zip Code API

Project Teams

Working on a team is not a requirement and you may complete the project on your own. Thus, project team(s) may be composed of 1 or **at most** 2 students. If you are working with another student please, ensure both of your names are clearly visible in your final solution JNB.

Source 1: Initial Author Data

The initial list of author names is to be web-scraped from a popular "famous quotes" website. There are 50 unique authors available from this site. Begin your exploration at the base URL and determine your own strategy to find and capture the required author data.

Depending on your approach, you may encounter the same author multiple times. In the end, your list of authors should be duplicate free.

```
Base Scrape URL: https://quotes.toscrape.com
```

Tasks

1. Beginning at the Base Scrape URL given above, you are to use a web scraping technique to collect the following author information:

- Full Name
- Date of birth
- Birth location

2. Create a pandas dataframe from (or using) the relevant data

In [1]:
```
import pandas as pd
import requests
import html5lib
from bs4 import BeautifulSoup as BS
```

In [2]:

```
URL = 'https://quotes.toscrape.com'
nameOfAuthors = {}
set_ =set()
flag = True
nextLink = URL
detailsOfAuthors =[]

def call(url):
    response = requests.get(url)
    #print('Response:',response)
    dta = response.content
    dta = BS(response.content,'html5lib')
    return dta

def nameScrapper(webUrl):
    data = call(webUrl)
    rawData = data.findAll('div',attrs={'class':'quote'})
    for i in rawData:
        if i.a['href'] not in set_:
            set_.add(f"{URL}{i.a['href']}")
            nameOfAuthors[i.small.text] = f"{URL}{i.a['href']}"

data = call(URL)

while(flag):
    data = call(nextLink)
    #print(nextLink)
    nameScrapper(nextLink)
    try:
        next_ = data.find('li',attrs={'class':'next'}).a.text.split
        nLink = data.find('li',attrs={'class':'next'}).a['href']
    except:
        flag = False
    finally:
        #print(nLink)
        if next_ == 'Next':
            nextLink = f"{URL}{nLink}"


for name_,li_nk in nameOfAuthors.items():
    rawData = call(li_nk)
    dob = rawData.find('span',attrs={'class':'author-born-date'}).t
    birthPlace = rawData.find('span',attrs={'class':'author-born-lo
    detailsOfAuthors.append([name_, dob ,birthPlace])

df1 = pd.DataFrame(detailsOfAuthors,columns=['Name of Author','Date
df1
```

Out[2]:

	Name of Author	Date of Birth	Place of Birth
0	Albert Einstein	March 14, 1879	Ulm, Germany
1	J.K. Rowling	July 31, 1965	Yate, South Gloucestershire, England, The Uni...
2	Jane Austen	December 16, 1775	Steventon Rectory, Hampshire, The United Kgdom
3	Marilyn Monroe	June 01, 1926	The United States
4	André Gide	November 22, 1869	Paris, France
5	Thomas A. Edison	February 11, 1847	Milan, Ohio, The United States

	Name of Author	Date of Birth	Place of Birth
6	Eleanor Roosevelt	October 11, 1884	The United States
7	Steve Martin	August 14, 1945	Waco, Texas, The United States
8	Bob Marley	February 06, 1945	Ne Mile, Sat Ann, Jamaica
9	Dr. Seuss	March 02, 1904	Sprgfield, MA, The United States
10	Douglas Adams	March 11, 1952	Cambridge, England, The United Kgdom
11	Elie Wiesel	September 30, 1928	Sighet, Romania
12	Friedrich Nietzsche	October 15, 1844	Röcken bei Lützen, Prussian Provce of Saxony,...
13	Mark Twain	November 30, 1835	Florida, Missouri, The United States
14	Allen Saunders	April 24, 1899	The United States
15	Pablo Neruda	July 12, 1904	Parral, Chile
16	Ralph Waldo Emerson	May 25, 1803	Boston, Massachusetts, The United States
17	Mother Teresa	August 26, 1910	Skopje, Macedonia, the Former Yugoslav Republ...
18	Garrison Keillor	August 07, 1942	Anoka, Mnesota, The United States
19	Jim Henson	September 24, 1936	Greenville, Mississippi, The United States
20	Charles M. Schulz	November 26, 1922	Mneapolis, MN, The United States
21	William Nicholson	January 12, 1948	Lewes, Sussex, The United Kgdom
22	Jorge Luis Borges	August 24, 1899	Buenos Aires, Argenta
23	George Eliot	November 22, 1819	South Farm, Arbury Hall, Nuneaton, Warwickshi...
24	George R.R. Martin	September 20, 1948	Bayonne, New Jersey, The United States
25	C.S. Lewis	November 29, 1898	Belfast, Ireland
26	Martin Luther King Jr.	January 15, 1929	Atlanta, Georgia, The United States
27	James Baldwin	August 02, 1924	Harlem, New York, The United States
28	Haruki Murakami	January 12, 1949	Kyoto, Japan
29	Alexandre Dumas fils	July 27, 1824	Paris, France
30	Stephenie Meyer	December 24, 1973	Connecticut, The United States
31	Ernest Hemingway	July 21, 1899	Oak Park, Illois, The United States
32	Helen Keller	June 27, 1880	Tuscumbia, Alabama, The United States
33	George Bernard Shaw	July 26, 1856	Dubl, Ireland
34	Charles Bukowski	August 16, 1920	Andernach, Germany
35	Suzanne Collins	August 11, 1962	Hartford, Connecticut, The United States
36	J.R.R. Tolkien	January 03, 1892	Bloemfonte, Mangaung, Free State, South Africa
37	Alfred Tennyson	August 06, 1809	Somersby, Lcolnshire, The United Kgdom
38	Terry Pratchett	April 28, 1948	Beaconsfield, Buckghamshire, England, The Uni...
39	J.D. Salinger	January 01, 1919	Manhattan, New York, The United States
40	George Carlin	May 12, 1937	New York, New York, The United States
41	John Lennon	October 09, 1940	Liverpool, England, The United Kgdom
42	W.C. Fields	January 29, 1880	Darby, Pennsylvania, The United States
43	Ayn Rand	February 02, 1905	St. Petersburg, Russian Federation

	Name of Author	Date of Birth	Place of Birth
44	Jimi Hendrix	November 27, 1942	Seattle, Washgton, The United States
45	J.M. Barrie	May 09, 1860	Kirriemuir, Angus, Scotland, The United Kgdom
46	E.E. Cummings	October 14, 1894	Cambridge, Massachusetts, The United States

Source 2: Author Key Data

For each of the 50 authors previously identified, you are to merge a key and gender value available from a CSV file with the author data from *Source 1*.

```
CSV File: *author_key_file.csv*
```

Tasks

1. The author names are unique in both data sources and thus may be used to associate the *key* and *gender* attribute values with the author.
2. Once you have completed the merge, convert the *key* column to be the dataframe's row index.

In [3]:

```
author_key = pd.read_csv('author_key_file.csv',names=["Name of Auth
authorDf =pd.merge(author_key,df1,on='Name of Author',how='right')
authorDf.set_index('key',inplace=True)
authorDf
```

Out[3]:

	Name of Author	gender	Date of Birth	Place of Birth
key				
QWxiZXJ0	Albert Einstein	M	March 14, 1879	Ulm, Germany
Si1LLVJv	J.K. Rowling	F	July 31, 1965	Yate, South Gloucestershire, England, The Uni...
SmFuZS1B	Jane Austen	F	December 16, 1775	Steventon Rectory, Hampshire, The United Kgdom
TWFyaWx5	Marilyn Monroe	F	June 01, 1926	The United States
QW5kcmUt	André Gide	M	November 22, 1869	Paris, France
VGhvbWFz	Thomas A. Edison	M	February 11, 1847	Milan, Ohio, The United States
RWxlYW5v	Eleanor Roosevelt	F	October 11, 1884	The United States
U3RldmUt	Steve Martin	M	August 14, 1945	Waco, Texas, The United States
Qm9iLU1h	Bob Marley	M	February 06, 1945	Ne Mile, Sat Ann, Jamaica
RHltU2V1	Dr. Seuss	M	March 02, 1904	Sprgfield, MA, The United States
RG91Z2xh	Douglas Adams	M	March 11, 1952	Cambridge, England, The United Kgdom
RWxpZS1X	Elie Wiesel	M	September 30, 1928	Sighet, Romania
RnJpZWRy	Friedrich Nietzsche	M	October 15, 1844	Röcken bei Lützen, Prussian Provce of Saxony,...

	Name of Author	gender	Date of Birth	Place of Birth
key				
TWFyay1U	Mark Twain	M	November 30, 1835	Florida, Missouri, The United States
QWxsZW4t	Allen Saunders	M	April 24, 1899	The United States
UGFibG8t	Pablo Neruda	M	July 12, 1904	Parral, Chile
UmFscGgt	Ralph Waldo Emerson	M	May 25, 1803	Boston, Massachusetts, The United States
TW90aGVy	Mother Teresa	F	August 26, 1910	Skopje, Macedonia, the Former Yugoslav Republ...
R2Fycmlz	Garrison Keillor	M	August 07, 1942	Anoka, Mnesota, The United States
SmltLUhl	Jim Henson	M	September 24, 1936	Greenville, Mississippi, The United States
Q3hhcmxl	Charles M. Schulz	M	November 26, 1922	Mneapolis, MN, The United States
V2lsbGlh	William Nicholson	M	January 12, 1948	Lewes, Sussex, The United Kgdom
Sm9yZ2Ut	Jorge Luis Borges	M	August 24, 1899	Buenos Aires, Argenta
R4Vvcmdl	George Eliot	F	November 22, 1819	South Farm, Arbury Hall, Nuneaton, Warwickshi...
R5Vvcmdl	George R.R. Martin	M	September 20, 1948	Bayonne, New Jersey, The United States
Qy1TLUxl	C.S. Lewis	M	November 29, 1898	Belfast, Ireland
TWFydGlu	Martin Luther King Jr.	M	January 15, 1929	Atlanta, Georgia, The United States
SmFtZXMt	James Baldwin	M	August 02, 1924	Harlem, New York, The United States
SGFydWtp	Haruki Murakami	M	January 12, 1949	Kyoto, Japan
QWxleGFu	Alexandre Dumas fils	M	July 27, 1824	Paris, France
U3RlcGhl	Stephenie Meyer	F	December 24, 1973	Connecticut, The United States
RXJuZXN0	Ernest Hemingway	M	July 21, 1899	Oak Park, Illois, The United States
SGVsZW4t	Helen Keller	F	June 27, 1880	Tuscumbia, Alabama, The United States
R2Vvcmdl	George Bernard Shaw	M	July 26, 1856	Dubl, Ireland
Q2hhcmxl	Charles Bukowski	M	August 16, 1920	Andernach, Germany
U3V6YW5u	Suzanne Collins	F	August 11, 1962	Hartford, Connecticut, The United States
Si1SLVIt	J.R.R. Tolkien	M	January 03, 1892	Bloemfonte, Mangaung, Free State, South Africa
QWxmcmVk	Alfred Tennyson	M	August 06, 1809	Somersby, Lcolnshire, The United Kgdom
VGVycnkt	Terry Pratchett	M	April 28, 1948	Beaconsfield, Buckghamshire, England, The Uni...

	Name of Author	gender	Date of Birth	Place of Birth
key				
Si1ELVNh	J.D. Salinger	M	January 01, 1919	Manhattan, New York, The United States
R3Vvcmdl	George Carlin	M	May 12, 1937	New York, New York, The United States
Sm9obi1M	John Lennon	M	October 09, 1940	Liverpool, England, The United Kgdom
Vy1DLUZp	W.C. Fields	M	January 29, 1880	Darby, Pennsylvania, The United States
QXluLVJh	Ayn Rand	F	February 02, 1905	St. Petersburg, Russian Federation
SmltaS1l	Jimi Hendrix	M	November 27, 1942	Seattle, Washgton, The United States
Si1NLUJh	J.M. Barrie	M	May 09, 1860	Kirriemuir, Angus, Scotland, The United Kgdom

In [16]:
```
len(authorDf)
```

Out[16]:
```
50
```

Source 3: Author Location Data

This direct download XML data source contains the location information that will be used to direct app users to the historical sites associated with each author. The *id* tag can be used to match the previously assembled author data (ie, the *key* attribute) with the author location data.

You are to retain all of the location attributes (excluding *id* - which will be a duplicate of the existing *key* attribute) from this data source to extend the previously collected author data.

```
Direct Download URL: https://www.drivehq.com/file/DFPublishFil
e.aspx/FileID7657515244/Keycqacws4cypvo/author_location_data.x
ml
Data Format: XML
Encoding: UTF-8
```

Tasks

1. Ingest the XML data into a pandas dataframe
2. Use the pandas dataframe merge method to join the new dataframe with the Source 1+2 dataframe.
3. *Source 3* contains location data for many public figures in addition to the 50 authors in which we are currently interested. You must restrict your final Source 3 dataframe to our 50 authors.

In [5]:
```
import xml.etree.ElementTree as et
from urllib.request import urlopen
```

In [6]:
```
xmldoc =urlopen('https://www.drivehq.com/file/DFPublishFile.aspx/Fi
tree= et.parse(xmldoc)
```

```
root = tree.getroot()
```

In [7]:
```
dataList =[]
for i in root.findall('./location'):
    id_ = i.find('id').text
    price = i.find('price').text
    bedrooms = i.find('bedrooms').text
    bathrooms = i.find('bathrooms').text
    sqft_living = i.find('sqft_living').text
    sqft_lot = i.find('sqft_lot').text
    floors = i.find('floors').text
    waterfront = i.find('waterfront').text
    grade= i.find('grade').text
    yr_built= i.find('yr_built').text
    lat= i.find('lat').text
    long= i.find('long').text
    dataList.append([id_,price,bedrooms,bathrooms,sqft_living,sqft_
```

In [8]:
```
detailDf = pd.DataFrame(dataList,columns = ["key","price","bedrooms
detailDf
```

Out[8]:

	key	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	grade
0	99744767	271310	2	1	870	5340	1.5	0	6
1	74681995	503500	3	2.5	1810	1750	2	0	7
2	53154276	2574000	4	3.75	4475	20424	2	1	12
3	Q3hhcmxl	134000	2	1.5	980	5000	2	0	7
4	SmltaS1l	284200	3	1.75	1540	6632	1	0	7
...	...	...	...	...	...	...	...	...	..
491	19602869	270000	3	1.75	1610	6205	1	0	7
492	18124298	291375	4	2.5	2220	6233	2	0	7
493	17897404	435000	3	1	1050	5500	1	0	6
494	U3RlcGhl	200000	4	2	1920	4822	1	0	6
495	95149109	279000	4	2	1980	10051	1	0	7

496 rows × 12 columns

In [9]:
```
source3 = pd.merge(authorDf,detailDf,on='key',how='inner')
source3
```

Out[9]:

	key	Name of Author	gender	Date of Birth	Place of Birth	price	bedrooms	bathrooms
0	QWxiZXJ0	Albert Einstein	M	March 14, 1879	Ulm, Germany	699999	3	0.7
1	Si1LLVJv	J.K. Rowling	F	July 31, 1965	Yate, South Gloucestershire, England, The Uni...	415000	3	2.
2	SmFuZS1B	Jane Austen	F	December 16, 1775	Steventon Rectory, Hampshire, The United Kgdom	300000	3	2.

	key	Name of Author	gender	Date of Birth	Place of Birth	price	bedrooms	bathroom
3	TWFyaWx5	Marilyn Monroe	F	June 01, 1926	The United States	2480000	4	

Source 4: Zip Code API

You may have observed that the *Source 3* data does not include the city and zip code information for the property locations for our 50 authors. Your task is the use the latitude and longitude values from the *Source 3* data to locate the city and zip code information and incorporate that information into your dataframe.

The USPS maintains data related to all US zip codes and their centroid latitude and longitude. The *Source 3* data contains the latitude and longitude information (the lat and long tags) for the author locations. Your task is to use the *Source 3* latitude and longitude values as an argument to the zip code api in order to to augment your dataframe with the city name and zip code of the authors' locations.

The ZIP code API contains ZIP codes for the continental United States, Alaska, Hawaii, Puerto Rico, and American Samoa. The API provides data in JSON format values for for ZIP code, city, latitude, longitude, timezone (offset from GMT).

The relevant zip code API URL and parameters are shown in the table below.

API Information and Parameters	Value
API Documentation	https://public.opendatasoft.com/explore/dataset/georef-united-states-of-america-zc-point/api/ (https://public.opendatasoft.com/explore/dataset/georef-united-states-of-america-zc-point/api/)
API URL	https://public.opendatasoft.com/api/records/1.0/search/ (https://public.opendatasoft.com/api/records/1.0/search/)
dataset (parameter)	georef-united-states-of-america-zc-point
rows (parameter)	100
geofilter.distance (parameter)	TBD by You. If used, distance value must by at least 10 kilometers
Geofilter Documentation	https://help.opendatasoft.com/platform/en/exploring_catalog_and_datasets/03_searching_the_data/search.html#geo-filtering (https://help.opendatasoft.com/platform/en/exploring_catalog_and_datasets/03_searching_the_data/search.html#geo-filtering)
q (parameter)	TBD by You. If used, distance argument must by at least 10 kilometers. **Hint:** use the #distance function.
Search Documentation	https://help.opendatasoft.com/platform/en/exploring_catalog_and_datasets/03_searching_the_data/search.html#full-text-search (https://help.opendatasoft.com/platform/en/exploring_catalog_and_datasets/03_searching_the_data/search.html#full-text-search)
Notes:	use only one of either the *q* or *geofilter.distance* parameters. Do not use both. The choice is yours.

Tasks

The long and lat values from *Source 3* are the exact longitude and latitude values for a single property location. As such, it is unlikely that you will find an exact match from the zip code API using these values. Instead, you must:

1. round the lat and long values to 2 decimal place
2. invoke the API requesting all zip code related data within 10 kilometers from the rounded lat and long values. (Refer to the *q* or *geofilter.distance* parameter documentation listed above.)
3. If the returned JSON results indicate more that one candidate record has been returned, you must determine which record's longitude and latitude are the **closest** to the *Source 3* long and lat values.

   There are several ways to accomplish this. For example, you could calculate the distance between the each result's longitude,latitude values and the the long, lat values from *Source 3*. The distance between the values can be calculated using the Pythagorean Theorem as follows:
   $\sqrt{(lon_1-lon_2)^2 + (lat_1-lat_2)^2}$

   Another, far better, approach involves a careful reading the documentation paying special attention to the *sort* parameter. This is a hint...
4. Use the city and zip code that are associated with the result having the minimum distance from (closest to) the *Source 3* long, lat values.

In [10]:
```python
import json
import numpy as np
```

In [11]:
```python
zipd = {}
cityList=[]
zipList=[]
def getzip(dic):
    #print("in getzip")
    min_ =min(dic.keys())
    #print(min_)
    for key,value in dic.items():
        if key == min_:
            #print(value)
            return value



def callonglat(lat1,long1,lat2,long2):
    #print("in callonglat")
    lat1,long1,lat2,long2 = np.round([float(lat1),float(long1),floa
    return np.sqrt(((long1)-long2)**2 + (lat1-lat2)**2)

def latlong(data_,a,o):
    #print("in latlong")
    zipd.clear()
    for len_ in range(len(data_['records'])):
        laT,lonG = data_['records'][len_]['geometry']['coordinates'
        z = data_['records'][len_]['fields']['zip_code']
        ci = data_['records'][len_]['fields']['coty_name']
        #print(z,"\t\t",ci)
        zipd[callonglat(a,o,laT,lonG)] = [z,ci]
    #print(zipd)
    name1 = getzip(zipd)
    #print(name1)
    cityName,zipCode = name1[1],name1[0]
    return cityName,zipCode

```

```
def calculate():
    for i in range(len(source3)):
        #print(f"{i}th Run")
        latitude = round(float(source3['lat'].iloc[i]),2)
        longitude = round(float(source3['long'].iloc[i]),2)
        #print(latitude,longitude)
        dis =10*1000
        api = f'https://public.opendatasoft.com/api/records/1.0/sea
        r = requests.get(api).text
        var = json.loads(r)
        #print(var)
        city,zipco = latlong(var,latitude,longitude)
        cityList.append(city)
        zipList.append(zipco)
calculate()
source3['City'] = cityList
source3['Zip Code'] = zipList
```

In [12]:

```
source3
```

Out[12]:

	key	Name of Author	gender	Date of Birth	Place of Birth	price	bedrooms	bathroo
0	QWxiZXJ0	Albert Einstein	M	March 14, 1879	Ulm, Germany	699999	3	0
1	Si1LLVJv	J.K. Rowling	F	July 31, 1965	Yate, South Gloucestershire, England, The Uni...	415000	3	
2	SmFuZS1B	Jane Austen	F	December 16, 1775	Steventon Rectory, Hampshire, The United Kgdom	300000	3	
3	TWFyaWx5	Marilyn Monroe	F	June 01, 1926	The United States	2480000	4	
4	QW5kcmUt	André Gide	M	November 22, 1869	Paris, France	425000	3	1
5	VGhvbWFz	Thomas A. Edison	M	February 11, 1847	Milan, Ohio, The United States	179900	2	
6	RWxlYW5v	Eleanor Roosevelt	F	October 11, 1884	The United States	420000	3	1
7	U3RldmUt	Steve Martin	M	August 14, 1945	Waco, Texas, The United States	761000	3	
8	Qm9iLU1h	Bob Marley	M	February 06, 1945	Ne Mile, Sat Ann, Jamaica	307150	3	
9	RHItU2V1	Dr. Seuss	M	March 02, 1904	Sprgfield, MA, The United States	550000	3	
10	RG91Z2xh	Douglas Adams	M	March 11, 1952	Cambridge, England, The United Kgdom	528000	3	
11	RWxpZS1X	Elie Wiesel	M	September 30, 1928	Sighet, Romania	418395	4	

	key	Name of Author	gender	Date of Birth	Place of Birth	price	bedrooms	bathroo
12	RnJpZWRy	Friedrich Nietzsche	M	October 15, 1844	Röcken bei Lützen, Prussian Provce of Saxony,...	630000	4	2
13	TWFyay1U	Mark Twain	M	November 30, 1835	Florida, Missouri, The United States	673200	5	
14	QWxsZW4t	Allen Saunders	M	April 24, 1899	The United States	264950	2	
15	UGFibG8t	Pablo Neruda	M	July 12, 1904	Parral, Chile	1295000	4	
16	UmFscGgt	Ralph Waldo Emerson	M	May 25, 1803	Boston, Massachusetts, The United States	230000	4	
17	TW90aGVy	Mother Teresa	F	August 26, 1910	Skopje, Macedonia, the Former Yugoslav Republ...	410000	4	
18	R2Fycmlz	Garrison Keillor	M	August 07, 1942	Anoka, Mnesota, The United States	247500	4	1
19	SmltLUhl	Jim Henson	M	September 24, 1936	Greenville, Mississippi, The United States	325000	4	
20	Q3hhcmxl	Charles M. Schulz	M	November 26, 1922	Mneapolis, MN, The United States	134000	2	
21	V2lsbGlh	William Nicholson	M	January 12, 1948	Lewes, Sussex, The United Kgdom	315000	3	1
22	Sm9yZ2Ut	Jorge Luis Borges	M	August 24, 1899	Buenos Aires, Argenta	390000	3	
23	R4Vvcmdl	George Eliot	F	November 22, 1819	South Farm, Arbury Hall, Nuneaton, Warwickshi...	260000	4	
24	R5Vvcmdl	George R.R. Martin	M	September 20, 1948	Bayonne, New Jersey, The United States	378000	4	
25	Qy1TLUxl	C.S. Lewis	M	November 29, 1898	Belfast, Ireland	351000	4	
26	TWFydGlu	Martin Luther King Jr.	M	January 15, 1929	Atlanta, Georgia, The United States	291600	3	1
27	SmFtZXMt	James Baldwin	M	August 02, 1924	Harlem, New York, The United States	660000	3	1
28	SGFydWtp	Haruki Murakami	M	January 12, 1949	Kyoto, Japan	292000	3	
29	QWxleGFu	Alexandre Dumas fils	M	July 27, 1824	Paris, France	576750	3	

	key	Name of Author	gender	Date of Birth	Place of Birth	price	bedrooms	bathroo
30	U3RlcGhl	Stephenie Meyer	F	December 24, 1973	Connecticut, The United States	200000	4	
31	RXJuZXN0	Ernest Hemingway	M	July 21, 1899	Oak Park, Illois, The United States	368000	3	1
32	SGVsZW4t	Helen Keller	F	June 27, 1880	Tuscumbia, Alabama, The United States	396000	3	1
33	R2Vvcmdl	George Bernard Shaw	M	July 26, 1856	Dubl, Ireland	525000	4	
34	Q2hhcmxl	Charles Bukowski	M	August 16, 1920	Andernach, Germany	590000	4	
35	U3V6YW5u	Suzanne Collins	F	August 11, 1962	Hartford, Connecticut, The United States	530000	3	
36	Si1SLVlt	J.R.R. Tolkien	M	January 03, 1892	Bloemfonte, Mangaung, Free State, South Africa	792500	3	
37	QWxmcmVk	Alfred Tennyson	M	August 06, 1809	Somersby, Lcolnshire, The United Kgdom	565000	3	1
38	VGVycnkt	Terry Pratchett	M	April 28, 1948	Beaconsfield, Buckghamshire, England, The Uni...	547000	2	
39	Si1ELVNh	J.D. Salinger	M	January 01, 1919	Manhattan, New York, The United States	291850	3	
40	R3Vvcmdl	George Carlin	M	May 12, 1937	New York, New York, The United States	245000	3	1
41	Sm9obi1M	John Lennon	M	October 09, 1940	Liverpool, England, The United Kgdom	515000	2	
42	Vy1DLUZp	W.C. Fields	M	January 29, 1880	Darby, Pennsylvania, The United States	425000	4	
43	QXluLVJh	Ayn Rand	F	February 02, 1905	St. Petersburg, Russian Federation	825000	4	2
44	SmltaS1I	Jimi Hendrix	M	November 27, 1942	Seattle, Washgton, The United States	284200	3	1
45	Si1NLUJh	J.M. Barrie	M	May 09, 1860	Kirriemuir, Angus, Scotland, The United Kgdom	515000	3	
46	RS1FLUN1	E.E. Cummings	M	October 14, 1894	Cambridge, Massachusetts, The United States	485000	4	

Proof of Concept

Use the dataframe that you constructed above to answer the following three questions.

Question 1

What is the mean and standard deviation of living space (sqft_living) by number of bedrooms? Use the pandas dataframe.agg() method to calculate these statistics in a single step.

Question 2

How many authors, by gender, live on lots that exceed the mean lot size (sqft_lot) by more than 1 standard deviation.

Question 3

Examine the documentation for the pandas dataframe.cut() method [Docs (https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.cut.html)](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.cut.html)

Use this method to convert the *price* attribute into to 3 approximately equal sized bins. Assign each bin specific labels of 'low', 'medium', 'high'. How many authors are there in each price category?

Question 1 What is the mean and standard deviation of living space (sqft_living) by number of bedrooms? Use the pandas dataframe.agg() method to calculate these statistics in a single step.

In [13]:
```
source3['sqft_living'] = source3['sqft_living'].apply(lambda x : ir
source3['bedrooms'] = source3['bedrooms'].apply(lambda x: int(x))
s3 = source3.groupby(['bedrooms'])
s3.agg({'sqft_living':['mean','std']})
```

Out[13]:

	sqft_living	
	mean	std
bedrooms		
2	972.000000	270.314631
3	1824.230769	504.271142
4	2454.777778	1066.275130
5	4180.000000	NaN

Question 2

How many authors, by gender, live on lots that exceed the mean lot size (sqft_lot) by more than 1 standard deviation.

In [14]:
```
source3['sqft_lot'] = source3['sqft_lot'].apply(lambda x:int(x))
s2 = source3.groupby('gender')
```

```
meanLotSize = s2.agg({'sqft_lot':['mean','std']})
print("Mean and Standard Deviation of Lot Size\n",meanLotSize)
female_mean = meanLotSize['sqft_lot']['mean'][0]
male_mean = meanLotSize['sqft_lot']['mean'][1]
female_std = meanLotSize['sqft_lot']['std'][0]
male_std = meanLotSize['sqft_lot']['std'][1]
male_data = []
female_data =[]
for i in range(len(source3)):
    if source3['gender'].iloc[i] == 'M':
        male_data.append(source3['sqft_lot'].iloc[i])
    else:
        female_data.append(source3['sqft_lot'].iloc[i])

def calZ(data_,mean,std):
    count = 0
    for i in data_:
        z = (i-mean)/std
        if z > 1:
            count+=1
    return count


no_of_male = calZ(male_data,male_mean,male_std)
no_of_female = calZ(female_data,female_mean,female_std)
print(f"\n\n{no_of_male} male authors and {no_of_female} Female aut
```

```
Mean and Standard Deviation of Lot Size
             sqft_lot
                 mean           std
gender
F         9051.250000   7063.398038
M         9723.710526  14505.911876


2 male authors and 2 Female authors, live on lots that exceed the me
an lot size (sqft_lot) by more than 1 standard deviation
```

Question 3

Examine the documentation for the pandas dataframe.cut() method [Docs (https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.cut.html)](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.cut.html)

Use this method to convert the *price* attribute into to 3 approximately equal sized bins. Assign each bin specific labels of 'low', 'medium', 'high'. How many authors are there in each price category?

In [15]:
```
source3['price'] = source3['price'].apply(lambda x: int(x))
price_category = pd.cut(source3['price'], bins=3, right=False,label
low = 0
medium =0
high = 0
for i in price_category:
    if i =='Low':
        low+=1
    elif i == 'Medium':
        medium+=1
    else:
        high+=1
print(f"\nAuthors in Price Categories\n\nLow\tMedium\tHigh\n{low}\t
```

```
Authors in Price Categories

Low     Medium  High
48      1       1
```

Deliverable

Due to the the implementation of the JNB app, it is very easy to create circular variable dependencies. Such dependencies will thwart my ability to run your entire JNB solution and result in the loss of valuable project points.

To ensure that your have not inadvertently created such dependencies, I **strongly** recommend you perform the following steps prior to submitting your JNB.

1. Clear all outputs from your JNB solution's code cells
2. Save you JNB solution
3. Stop and restart your JNB kernel via the *Kernel* Jupyter Notebook menu item. Use the

Restart & Run All Output menu item.

1. Inspect your output cells for any errors

Once you have completed your project, upload the JNB containing your solution to the *Course Project* assessment item located in the *Course Project* content area on our Blackboard site.

```
In [ ]: 
```

```
In [ ]: 
```