Khalil MASMOUDI

Journée Statistique, Mathématique et Data Science 2019

The Series Data Structure

In [1]:

```
import pandas as pd
pd.Series?
```

In [2]:

```
animals = ['Tiger', 'Bear', 'Moose']
pd.Series(animals)
```

Out[2]:

```
0    Tiger
1     Bear
2    Moose
dtype: object
```

In [4]:

```
numbers = [1, 2, 3]
pd.Series(numbers)
```

Out[4]:

```
0    1
1    2
2    3
dtype: int64
```

In [5]:

```
animals = ['Tiger', 'Bear', None]
pd.Series(animals)
```

Out[5]:

```
0    Tiger
1     Bear
2     None
dtype: object
```

In [6]:

```
numbers = [1, 2, None]
pd.Series(numbers)
```

Out[6]:

```
0    1.0
1    2.0
2    NaN
dtype: float64
```

In [4]:

```
import numpy as np
np.nan == None
```

Out[4]:

```
False
```

In [2]:

```
np.nan == np.nan
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-2-b8a58fdd7199> in <module>()
----> 1 np.nan == np.nan

NameError: name 'np' is not defined
```

In [5]:

```
import numpy as np
np.isnan(np.nan)
```

Out[5]:

```
True
```

In [6]:

```
sports = {'Archery': 'Bhutan',
          'Golf': 'Scotland',
          'Sumo': 'Japan',
          'Taekwondo': 'South Korea'}
s = pd.Series(sports)
s
```

Out[6]:

```
Archery           Bhutan
Golf            Scotland
Sumo               Japan
Taekwondo    South Korea
dtype: object
```

In [11]:

```
s.index
```

Out[11]:

```
Index(['Archery', 'Golf', 'Sumo', 'Taekwondo'], dtype='object')
```

In [12]:

```
s = pd.Series(['Tiger', 'Bear', 'Moose'], index=['India', 'America', 'Canada'])
s
```

Out[12]:

```
India      Tiger
America     Bear
Canada     Moose
dtype: object
```

In [13]:

```
sports = {'Archery': 'Bhutan',
          'Golf': 'Scotland',
          'Sumo': 'Japan',
          'Taekwondo': 'South Korea'}
s = pd.Series(sports, index=['Golf', 'Sumo', 'Hockey'])
s
```

Out[13]:

```
Golf      Scotland
Sumo         Japan
Hockey         NaN
dtype: object
```

Querying a Series

In [10]:

```
sports = {'Archery': 'Bhutan',
          'Golf': 'Scotland',
          'Sumo': 'Japan',
          'Taekwondo': 'South Korea'}
s = pd.Series(sports)
s
```

Out[10]:

```
Archery           Bhutan
Golf            Scotland
Sumo               Japan
Taekwondo    South Korea
dtype: object
```

In [11]:

```
s.iloc[3]
```

Out[11]:

```
'South Korea'
```

In [14]:

```
s.loc['Golf']
s.iloc[1]
```

Out[14]:

```
'Scotland'
```

In [17]:

```
s[3]
```

Out[17]:

```
'South Korea'
```

In [18]:

```
s['Golf']
```

Out[18]:

```
'Scotland'
```

In [15]:

```
sports = {99: 'Bhutan',
          100: 'Scotland',
          101: 'Japan',
          102: 'South Korea'}
s = pd.Series(sports)
```

In [17]:

```
s.iloc[0] #This won't call s.iloc[0] as one might expect, it generates an error instead
```

Out[17]:

```
'Bhutan'
```

In [23]:

```
s = pd.Series([100.00, 120.00, 101.00, 3.00])
```

```
s
```

Out[23]:

```
0    100.0
1    120.0
2    101.0
3      3.0
dtype: float64
```

In [24]:

```
total = 0
for item in s:
    total+=item
print(total)
```

```
324.0
```

In [25]:

```
import numpy as np

total = np.sum(s)
print(total)
```

```
324.0
```

In [26]:

```
#this creates a big series of random numbers
s = pd.Series(np.random.randint(0,1000,10000))
s.head()
```

Out[26]:

```
0     55
1    613
2    785
3    139
4    833
dtype: int32
```

In [27]:

```
len(s)
```

Out[27]:

```
10000
```

In [28]:

```
%%timeit -n 100
summary = 0
for item in s:
    summary+=item
```

```
1.56 ms ± 140 µs per loop (mean ± std. dev. of 7 runs, 100 loops each)
```

In [29]:

```
%%timeit -n 100
summary = np.sum(s)
```

```
245 µs ± 38.9 µs per loop (mean ± std. dev. of 7 runs, 100 loops each)
```

In [30]:

```
s+=2 #adds two to each item in s using broadcasting
s.head()
```

Out[30]:

```
0     57
1    615
2    787
3    141
4    835
dtype: int32
```

In [32]:

```
for label, value in s.iteritems():
    s.set_value(label, value+2)
s.head()
```

```
C:\Users\Khalil\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: FutureWarning: set_value is deprecated and will be removed in a future release. Please use .at[] or .iat[] accessors instead
```

Out[32]:

```
0     61
1    619
2    791
3    145
4    839
dtype: int32
```

In [5]:

```
%%timeit -n 10
s = pd.Series(np.random.randint(0,1000,1000))
for label, value in s.iteritems():
    s.loc[label]= value+2
```

```
330 ms ± 45.4 ms per loop (mean ± std. dev. of 7 runs, 10 loops each)
```

In [6]:

```
%%timeit -n 10
s = pd.Series(np.random.randint(0,1000,1000))
s+=2
```

```
The slowest run took 6.63 times longer than the fastest. This could mean that an intermediate result is being cached.
802 µs ± 679 µs per loop (mean ± std. dev. of 7 runs, 10 loops each)
```

In [7]:

```
s = pd.Series([1, 2, 3])
s.loc['Animal'] = 'Bears'
s
```

Out[7]:

```
0             1
1             2
2             3
Animal    Bears
dtype: object
```

In [8]:

```
original_sports = pd.Series({'Archery': 'Bhutan',
                             'Golf': 'Scotland',
                             'Sumo': 'Japan',
                             'Taekwondo': 'South Korea'})
cricket_loving_countries = pd.Series(['Australia',
                                      'Barbados',
                                      'Pakistan',
                                      'England'],
                                     index=['Cricket',
                                            'Cricket',
```

```
                                            'Cricket',
                                            'Cricket'])
all_countries = original_sports.append(cricket_loving_countries)
```

In [9]:

```
original_sports
```

Out[9]:

```
Archery           Bhutan
Golf            Scotland
Sumo               Japan
Taekwondo    South Korea
dtype: object
```

In [10]:

```
cricket_loving_countries
```

Out[10]:

```
Cricket    Australia
Cricket     Barbados
Cricket     Pakistan
Cricket      England
dtype: object
```

In [11]:

```
all_countries
```

Out[11]:

```
Archery           Bhutan
Golf            Scotland
Sumo               Japan
Taekwondo    South Korea
Cricket        Australia
Cricket         Barbados
Cricket         Pakistan
Cricket          England
dtype: object
```

In [12]:

```
all_countries.loc['Cricket']
```

Out[12]:

```
Cricket    Australia
Cricket     Barbados
Cricket     Pakistan
Cricket      England
dtype: object
```

The DataFrame Data Structure

In [18]:

```
import pandas as pd
purchase_1 = pd.Series({'Name': 'Chris',
                        'Item Purchased': 'Dog Food',
                        'Cost': 22.50})
purchase_2 = pd.Series({'Name': 'Kevyn',
                        'Item Purchased': 'Kitty Litter',
                        'Cost': 2.50})
purchase_3 = pd.Series({'Name': 'Vinod',
                        'Item Purchased': 'Bird Seed',
                        'Cost': 5.00})
df = pd.DataFrame([purchase_1, purchase_2, purchase_3], index=['Store 1', 'Store 1', 'St
```

```
ore 2'])
df.head()
```

Out[18]:

	Name	Item Purchased	Cost
Store 1	Chris	Dog Food	22.5
Store 1	Kevyn	Kitty Litter	2.5
Store 2	Vinod	Bird Seed	5.0

In [19]:

```
df.loc['Store 2']
```

Out[19]:

```
Name                   Vinod
Item Purchased    Bird Seed
Cost                      5
Name: Store 2, dtype: object
```

In [20]:

```
type(df.loc['Store 2'])
```

Out[20]:

```
pandas.core.series.Series
```

In [16]:

```
df.loc['Store 1']
```

Out[16]:

	Name	Item Purchased	Cost
Store 1	Chris	Dog Food	22.5
Store 1	Kevyn	Kitty Litter	2.5

In [17]:

```
df.loc['Store 1', 'Cost']
```

Out[17]:

```
Store 1    22.5
Store 1     2.5
Name: Cost, dtype: float64
```

In [18]:

```
df.T
```

Out[18]:

	Store 1	Store 1	Store 2
Name	Chris	Kevyn	Vinod
Item Purchased	Dog Food	Kitty Litter	Bird Seed
Cost	22.5	2.5	5

In [19]:

```
df.T.loc['Cost']
```

Out[19]:

```
Store 1    22.5
Store 1     2.5
Store 2       5
Name: Cost, dtype: object
```

In [20]:

```
df['Cost']
```

Out[20]:

```
Store 1    22.5
Store 1     2.5
Store 2     5.0
Name: Cost, dtype: float64
```

In [21]:

```
df.loc['Store 1']['Cost']
```

Out[21]:

```
Store 1    22.5
Store 1     2.5
Name: Cost, dtype: float64
```

In [22]:

```
df.loc[:,['Name', 'Cost']]
```

Out[22]:

	Name	Cost
Store 1	Chris	22.5
Store 1	Kevyn	2.5
Store 2	Vinod	5.0

In [23]:

```
df.drop('Store 1')
```

Out[23]:

	Name	Item Purchased	Cost
Store 2	Vinod	Bird Seed	5.0

In [24]:

```
df
```

Out[24]:

	Name	Item Purchased	Cost
Store 1	Chris	Dog Food	22.5
Store 1	Kevyn	Kitty Litter	2.5
Store 2	Vinod	Bird Seed	5.0

In [30]:

```
copy_df = df.copy()
copy_df = copy_df.drop('Store 1')
copy_df
```

Out[30]:

	Name	Item Purchased	Cost
Store 2	Vinod	Bird Seed	5.0

In [31]:

```
copy_df.drop?
```

In [32]:

```
del copy_df['Name']
copy_df
```

Out[32]:

	Item Purchased	Cost
Store 2	Bird Seed	5.0

In [33]:

```
df['Location'] = None
df
```

Out[33]:

	Name	Item Purchased	Cost	Location
Store 1	Chris	Dog Food	22.5	None
Store 1	Kevyn	Kitty Litter	2.5	None
Store 2	Vinod	Bird Seed	5.0	None

Dataframe Indexing and Loading

In [34]:

```
costs = df['Cost']
costs
```

Out[34]:

```
Store 1    22.5
Store 1     2.5
Store 2     5.0
Name: Cost, dtype: float64
```

In [35]:

```
costs+=2
costs
```

Out[35]:

```
Store 1    24.5
Store 1     4.5
Store 2     7.0
Name: Cost, dtype: float64
```

In [36]:

```
df
```

Out[36]:

	Name	Item Purchased	Cost	Location
Store 1	Chris	Dog Food	24.5	None

	Name	Item Purchased	Cost	Location
Store 1	Kevyn	Kitty Litter	4.5	None
Store 2	Vinod	Bird Seed	7.0	None

In [37]:

```
!cat olympics.csv
```

```
'cat' n'est pas reconnu en tant que commande interne
ou externe, un programme ex,cutable ou un fichier de commandes.
```

In [21]:

```
df = pd.read_csv('olympics.csv')
df.head()
```

Out[21]:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	NaN	№ Summer	01 !	02 !	03 !	Total	№ Winter	01 !	02 !	03 !	Total	№ Games	01 !	02 !	03 !	Combined total
1	Afghanistan (AFG)	13	0	0	2	2	0	0	0	0	0	13	0	0	2	2
2	Algeria (ALG)	12	5	2	8	15	3	0	0	0	0	15	5	2	8	15
3	Argentina (ARG)	23	18	24	28	70	18	0	0	0	0	41	18	24	28	70
4	Armenia (ARM)	5	1	2	9	12	6	0	0	0	0	11	1	2	9	12

In [23]:

```
df = pd.read_csv('olympics.csv', index_col = 0, skiprows=1)
df.head()
```

Out[23]:

	№ Summer	01 !	02 !	03 !	Total	№ Winter	01 !.1	02 !.1	03 !.1	Total.1	№ Games	01 !.2	02 !.2	03 !.2	Combined total
Afghanistan (AFG)	13	0	0	2	2	0	0	0	0	0	13	0	0	2	2
Algeria (ALG)	12	5	2	8	15	3	0	0	0	0	15	5	2	8	15
Argentina (ARG)	23	18	24	28	70	18	0	0	0	0	41	18	24	28	70
Armenia (ARM)	5	1	2	9	12	6	0	0	0	0	11	1	2	9	12
Australasia (ANZ) [ANZ]	2	3	4	5	12	0	0	0	0	0	2	3	4	5	12

In [24]:

```
df.columns
```

Out[24]:

```
Index(['№ Summer', '01 !', '02 !', '03 !', 'Total', '№ Winter', '01 !.1',
       '02 !.1', '03 !.1', 'Total.1', '№ Games', '01 !.2', '02 !.2', '03 !.2',
       'Combined total'],
      dtype='object')
```

In [25]:

```
for col in df.columns:
    if col[:2]=='01':
        df.rename(columns={col:'Gold' + col[4:]}, inplace=True)
    if col[:2]=='02':
        df.rename(columns={col:'Silver' + col[4:]}, inplace=True)
    if col[:2]=='03':
        df.rename(columns={col:'Bronze' + col[4:]}, inplace=True)
    if col[:1]=='№':
        df.rename(columns={col:'#' + col[1:]}, inplace=True)
```

```
df.head()
```

Out[25]:

	# Summer	Gold	Silver	Bronze	Total	# Winter	Gold.1	Silver.1	Bronze.1	Total.1	# Games	Gold.2	Silver.2	B
Afghanistan (AFG)	13	0	0	2	2	0	0	0	0	0	13	0	0	
Algeria (ALG)	12	5	2	8	15	3	0	0	0	0	15	5	2	
Argentina (ARG)	23	18	24	28	70	18	0	0	0	0	41	18	24	
Armenia (ARM)	5	1	2	9	12	6	0	0	0	0	11	1	2	
Australasia (ANZ) [ANZ]	2	3	4	5	12	0	0	0	0	0	2	3	4	

Querying a DataFrame

In [26]:

```
df[df['Gold'] > 0]
```

Out[26]:

	# Summer	Gold	Silver	Bronze	Total	# Winter	Gold.1	Silver.1	Bronze.1	Total.1	# Games	Gold.2	Silver
Algeria (ALG)	12	5	2	8	15	3	0	0	0	0	15	5	
Argentina (ARG)	23	18	24	28	70	18	0	0	0	0	41	18	
Armenia (ARM)	5	1	2	9	12	6	0	0	0	0	11	1	
Australasia (ANZ) [ANZ]	2	3	4	5	12	0	0	0	0	0	2	3	
Australia (AUS) [AUS] [Z]	25	139	152	177	468	18	5	3	4	12	43	144	15
Austria (AUT)	26	18	33	35	86	22	59	78	81	218	48	77	1
Azerbaijan (AZE)	5	6	5	15	26	5	0	0	0	0	10	6	
Bahamas (BAH)	15	5	2	5	12	0	0	0	0	0	15	5	
Belarus (BLR)	5	12	24	39	75	6	6	4	5	15	11	18	
Belgium (BEL)	25	37	52	53	142	20	1	1	3	5	45	38	
Brazil (BRA)	21	23	30	55	108	7	0	0	0	0	28	23	
Bulgaria (BUL) [H]	19	51	85	78	214	19	1	2	3	6	38	52	
Burundi (BDI)	5	1	0	0	1	0	0	0	0	0	5	1	
Cameroon (CMR)	13	3	1	1	5	1	0	0	0	0	14	3	
Canada (CAN)	25	59	99	121	279	22	62	56	52	170	47	121	15
Chile (CHI) [I]	22	2	7	4	13	16	0	0	0	0	38	2	
China (CHN) [CHN]	9	201	146	126	473	10	12	22	19	53	19	213	1
Colombia (COL)	18	2	6	11	19	1	0	0	0	0	19	2	
Costa Rica (CRC)	14	1	1	2	4	6	0	0	0	0	20	1	
Croatia (CRO)	6	6	7	10	23	7	4	6	1	11	13	10	
Cuba (CUB) [Z]	19	72	67	70	209	0	0	0	0	0	19	72	
Czech Republic (CZE) [CZE]	5	14	15	15	44	6	7	9	8	24	11	21	
Czechoslovakia (TCH) [TCH]	16	49	49	45	143	16	2	8	15	25	32	51	
Denmark (DEN) [Z]	26	43	68	68	179	13	0	1	0	1	39	43	

Dominican Republic (DOM)	# Summer	Gold	Silver	Bronze	Total	# Winter	Gold.1	Silver.1	Bronze.1	Total.1	# Games	Gold.2	Silver
Ecuador (ECU)	13	1	1	0	2	0	0	0	0	0	13	1	
Egypt (EGY) [EGY] [Z]	21	7	9	10	26	1	0	0	0	0	22	7	
Estonia (EST)	11	9	9	15	33	9	4	2	1	7	20	13	
Ethiopia (ETH)	12	21	7	17	45	2	0	0	0	0	14	21	
Finland (FIN)	24	101	84	117	302	22	42	62	57	161	46	143	14
...	...	...	...	...	...	...	...	...	...	...	...	...	
Russia (RUS) [RUS]	5	132	121	142	395	6	49	40	35	124	11	181	16
Russian Empire (RU1) [RU1]	3	1	4	3	8	0	0	0	0	0	3	1	
Soviet Union (URS) [URS]	9	395	319	296	1010	9	78	57	59	194	18	473	37
Unified Team (EUN) [EUN]	1	45	38	29	112	1	9	6	8	23	2	54	4
Serbia (SRB) [SRB]	3	1	2	4	7	2	0	0	0	0	5	1	
Serbia and Montenegro (SCG) [SCG]	3	2	4	3	9	3	0	0	0	0	6	2	
Slovakia (SVK) [SVK]	5	7	9	8	24	6	2	2	1	5	11	9	
Slovenia (SLO)	6	4	6	9	19	7	2	4	9	15	13	6	
South Africa (RSA)	18	23	26	27	76	6	0	0	0	0	24	23	2
Spain (ESP) [Z]	22	37	59	35	131	19	1	0	1	2	41	38	5
Suriname (SUR) [E]	11	1	0	1	2	0	0	0	0	0	11	1	
Sweden (SWE) [Z]	26	143	164	176	483	22	50	40	54	144	48	193	20
Switzerland (SUI)	27	47	73	65	185	22	50	40	48	138	49	97	11
Syria (SYR)	12	1	1	1	3	0	0	0	0	0	12	1	
Chinese Taipei (TPE) [TPE] [TPE2]	13	2	7	12	21	11	0	0	0	0	24	2	
Thailand (THA)	15	7	6	11	24	3	0	0	0	0	18	7	
Trinidad and Tobago (TRI) [TRI]	16	2	5	11	18	3	0	0	0	0	19	2	
Tunisia (TUN)	13	3	3	4	10	0	0	0	0	0	13	3	
Turkey (TUR)	21	39	25	24	88	16	0	0	0	0	37	39	2
Uganda (UGA)	14	2	3	2	7	0	0	0	0	0	14	2	
Ukraine (UKR)	5	33	27	55	115	6	2	1	4	7	11	35	2
United Arab Emirates (UAE)	8	1	0	0	1	0	0	0	0	0	8	1	
United States (USA) [P] [Q] [R] [Z]	26	976	757	666	2399	22	96	102	84	282	48	1072	85
Uruguay (URU)	20	2	2	6	10	1	0	0	0	0	21	2	
Uzbekistan (UZB)	5	5	5	10	20	6	1	0	0	1	11	6	
Venezuela (VEN)	17	2	2	8	12	4	0	0	0	0	21	2	
Yugoslavia (YUG) [YUG]	16	26	29	28	83	14	0	3	1	4	30	26	3
Zimbabwe (ZIM) [ZIM]	12	3	4	1	8	1	0	0	0	0	13	3	
Mixed team (ZZX) [ZZX]	3	8	5	4	17	0	0	0	0	0	3	8	
Totals	27	4809	4775	5130	14714	22	959	958	948	2865	49	5768	57

100 rows × 15 columns

In [44]:

```
only_gold = df.where(df['Gold'] > 0)
only_gold.head()
```

Out[44]:

	# Summer	Gold	Silver	Bronze	Total	# Winter	Gold.1	Silver.1	Bronze.1	Total.1	# Games	Gold.2	Silver.2	B
Afghanistan (AFG)	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
Algeria (ALG)	12.0	5.0	2.0	8.0	15.0	3.0	0.0	0.0	0.0	0.0	15.0	5.0	2.0	
Argentina (ARG)	23.0	18.0	24.0	28.0	70.0	18.0	0.0	0.0	0.0	0.0	41.0	18.0	24.0	
Armenia (ARM)	5.0	1.0	2.0	9.0	12.0	6.0	0.0	0.0	0.0	0.0	11.0	1.0	2.0	
Australasia (ANZ) [ANZ]	2.0	3.0	4.0	5.0	12.0	0.0	0.0	0.0	0.0	0.0	2.0	3.0	4.0	

In [45]:

```
only_gold['Gold'].count()
```

Out[45]:

```
100
```

In [46]:

```
df['Gold'].count()
```

Out[46]:

```
147
```

In [47]:

```
only_gold = only_gold.dropna()
only_gold.head()
```

Out[47]:

	# Summer	Gold	Silver	Bronze	Total	# Winter	Gold.1	Silver.1	Bronze.1	Total.1	# Games	Gold.2	Silver.2	B
Algeria (ALG)	12.0	5.0	2.0	8.0	15.0	3.0	0.0	0.0	0.0	0.0	15.0	5.0	2.0	
Argentina (ARG)	23.0	18.0	24.0	28.0	70.0	18.0	0.0	0.0	0.0	0.0	41.0	18.0	24.0	
Armenia (ARM)	5.0	1.0	2.0	9.0	12.0	6.0	0.0	0.0	0.0	0.0	11.0	1.0	2.0	
Australasia (ANZ) [ANZ]	2.0	3.0	4.0	5.0	12.0	0.0	0.0	0.0	0.0	0.0	2.0	3.0	4.0	
Australia (AUS) [AUS] [Z]	25.0	139.0	152.0	177.0	468.0	18.0	5.0	3.0	4.0	12.0	43.0	144.0	155.0	

In [48]:

```
only_gold = df[df['Gold'] > 0]
only_gold.head()
```

Out[48]:

	# Summer	Gold	Silver	Bronze	Total	# Winter	Gold.1	Silver.1	Bronze.1	Total.1	# Games	Gold.2	Silver.2	Br

Algeria (ALG)	12	5	2	8	15	3	0	0	0	0	15	5	2
Argentina (ARG)	23	18	24	28	70	18	0	0	0	0	41	18	24
Armenia (ARM)	5	1	2	9	12	6	0	0	0	0	11	1	2
Australasia (ANZ) [ANZ]	2	3	4	5	12	0	0	0	0	0	2	3	4
Australia (AUS) [AUS] [Z]	25	139	152	177	468	18	5	3	4	12	43	144	155

In [49]:

```
len(df[(df['Gold'] > 0) | (df['Gold.1'] > 0)])
```

Out[49]:

101

In [50]:

```
df[(df['Gold.1'] > 0) & (df['Gold'] == 0)]
```

Out[50]:

	# Summer	Gold	Silver	Bronze	Total	# Winter	Gold.1	Silver.1	Bronze.1	Total.1	# Games	Gold.2	Silver.2
Liechtenstein (LIE)	16	0	0	0	0	18	2	2	5	9	34	2	2

Indexing Dataframes

In [31]:

```
df.describe()
```

Out[31]:

	time	playback position	volume
count	3.300000e+01	33.000000	4.00
mean	1.469976e+09	13.000000	8.75
std	1.420109e+03	11.696688	2.50
min	1.469974e+09	1.000000	5.00
25%	1.469975e+09	1.000000	8.75
50%	1.469975e+09	10.000000	10.00
75%	1.469977e+09	25.000000	10.00
max	1.469978e+09	33.000000	10.00

In [52]:

```
df['country'] = df.index
df = df.set_index('Gold')
df.head()
```

Out[52]:

Gold	# Summer	Silver	Bronze	Total	# Winter	Gold.1	Silver.1	Bronze.1	Total.1	# Games	Gold.2	Silver.2	Bronze.2	Combined total
0	13	0	2	2	0	0	0	0	0	13	0	0	2	2
5	12	2	8	15	3	0	0	0	0	15	5	2	8	15

	Gold	# Summer	Silver	Bronze	Total	# Winter	Gold.1	Silver.1	Bronze.1	Total.1	# Games	Gold.2	Silver.2	Bronze.2	Combined total
18	23	24	28	70	18	0	0	0	0	41	18	24	28	70	
Gold	1	5	2	9	12	6	0	0	0	0	11	1	2	9	12
3	2	4	5	12	0	0	0	0	0	2	3	4	5	12	

In [53]:

```
df = df.reset_index()
df.head()
```

Out[53]:

	Gold	# Summer	Silver	Bronze	Total	# Winter	Gold.1	Silver.1	Bronze.1	Total.1	# Games	Gold.2	Silver.2	Bronze.2	Combir to
0	0	13	0	2	2	0	0	0	0	0	13	0	0	2	
1	5	12	2	8	15	3	0	0	0	0	15	5	2	8	
2	18	23	24	28	70	18	0	0	0	0	41	18	24	28	
3	1	5	2	9	12	6	0	0	0	0	11	1	2	9	
4	3	2	4	5	12	0	0	0	0	0	2	3	4	5	

In [58]:

```
df = pd.read_csv('census.csv')
df.head()
```

Out[58]:

	SUMLEV	REGION	DIVISION	STATE	COUNTY	STNAME	CTYNAME	CENSUS2010POP	ESTIMATESBASE2010	POPESTIMA
0	40	3	6	1	0	Alabama	Alabama	4779736	4780127	
1	50	3	6	1	1	Alabama	Autauga County	54571	54571	
2	50	3	6	1	3	Alabama	Baldwin County	182265	182265	
3	50	3	6	1	5	Alabama	Barbour County	27457	27457	
4	50	3	6	1	7	Alabama	Bibb County	22915	22919	

5 rows × 100 columns

In [29]:

```
df['SUMLEV'].unique()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\Anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key, method, t
olerance)
   3062             try:
-> 3063                 return self._engine.get_loc(key)
   3064             except KeyError:

pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_i
tem()
```

```
pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'SUMLEV'

During handling of the above exception, another exception occurred:

KeyError                                  Traceback (most recent call last)
<ipython-input-29-68c20b5ca48a> in <module>()
----> 1 df['SUMLEV'].unique()

~\Anaconda3\lib\site-packages\pandas\core\frame.py in __getitem__(self, key)
   2683             return self._getitem_multilevel(key)
   2684         else:
-> 2685             return self._getitem_column(key)
   2686
   2687     def _getitem_column(self, key):

~\Anaconda3\lib\site-packages\pandas\core\frame.py in _getitem_column(self, key)
   2690         # get column
   2691         if self.columns.is_unique:
-> 2692             return self._get_item_cache(key)
   2693
   2694         # duplicate columns & possible reduce dimensionality

~\Anaconda3\lib\site-packages\pandas\core\generic.py in _get_item_cache(self, item)
   2484         res = cache.get(item)
   2485         if res is None:
-> 2486             values = self._data.get(item)
   2487             res = self._box_item_values(item, values)
   2488             cache[item] = res

~\Anaconda3\lib\site-packages\pandas\core\internals.py in get(self, item, fastpath)
   4113
   4114             if not isna(item):
-> 4115                 loc = self.items.get_loc(item)
   4116             else:
   4117                 indexer = np.arange(len(self.items))[isna(self.items)]

~\Anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key, method, tolerance)
   3063                 return self._engine.get_loc(key)
   3064             except KeyError:
-> 3065                 return self._engine.get_loc(self._maybe_cast_indexer(key))
   3066
   3067         indexer = self.get_indexer([key], method=method, tolerance=tolerance)

pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'SUMLEV'
```

In [30]:

```
df=df[df['SUMLEV'] == 50]
df.head()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\Anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key, method, tolerance)
   3062             try:
-> 3063                 return self._engine.get_loc(key)
   3064             except KeyError:
```

```
pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_i
tem()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_i
tem()

KeyError: 'SUMLEV'

During handling of the above exception, another exception occurred:

KeyError                                  Traceback (most recent call last)
<ipython-input-30-b634225f2bd8> in <module>()
----> 1 df=df[df['SUMLEV'] == 50]
      2 df.head()

~\Anaconda3\lib\site-packages\pandas\core\frame.py in __getitem__(self, key)
   2683             return self._getitem_multilevel(key)
   2684         else:
-> 2685             return self._getitem_column(key)
   2686
   2687     def _getitem_column(self, key):

~\Anaconda3\lib\site-packages\pandas\core\frame.py in _getitem_column(self, key)
   2690         # get column
   2691         if self.columns.is_unique:
-> 2692             return self._get_item_cache(key)
   2693
   2694         # duplicate columns & possible reduce dimensionality

~\Anaconda3\lib\site-packages\pandas\core\generic.py in _get_item_cache(self, item)
   2484         res = cache.get(item)
   2485         if res is None:
-> 2486             values = self._data.get(item)
   2487             res = self._box_item_values(item, values)
   2488             cache[item] = res

~\Anaconda3\lib\site-packages\pandas\core\internals.py in get(self, item, fastpath)
   4113
   4114             if not isna(item):
-> 4115                 loc = self.items.get_loc(item)
   4116             else:
   4117                 indexer = np.arange(len(self.items))[isna(self.items)]

~\Anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key, method, t
olerance)
   3063                 return self._engine.get_loc(key)
   3064             except KeyError:
-> 3065                 return self._engine.get_loc(self._maybe_cast_indexer(key))
   3066
   3067         indexer = self.get_indexer([key], method=method, tolerance=tolerance)

pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas\_libs\index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_i
tem()

pandas\_libs\hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_i
tem()

KeyError: 'SUMLEV'
```

In [61]:

```
columns_to_keep = ['STNAME',
                   'CTYNAME',
```

```
                    'BIRTHS2010',
                    'BIRTHS2011',
                    'BIRTHS2012',
                    'BIRTHS2013',
                    'BIRTHS2014',
                    'BIRTHS2015',
                    'POPESTIMATE2010',
                    'POPESTIMATE2011',
                    'POPESTIMATE2012',
                    'POPESTIMATE2013',
                    'POPESTIMATE2014',
                    'POPESTIMATE2015']
df = df[columns_to_keep]
df.head()
```

Out[61]:

	STNAME	CTYNAME	BIRTHS2010	BIRTHS2011	BIRTHS2012	BIRTHS2013	BIRTHS2014	BIRTHS2015	POPESTIMATE2010
1	Alabama	Autauga County	151	636	615	574	623	600	54660
2	Alabama	Baldwin County	517	2187	2092	2160	2186	2240	183193
3	Alabama	Barbour County	70	335	300	283	260	269	27341
4	Alabama	Bibb County	44	266	245	259	247	253	22861
5	Alabama	Blount County	183	744	710	646	618	603	57373

In [62]:

```
df = df.set_index(['STNAME', 'CTYNAME'])
df.head()
```

Out[62]:

		BIRTHS2010	BIRTHS2011	BIRTHS2012	BIRTHS2013	BIRTHS2014	BIRTHS2015	POPESTIMATE2010	PO
STNAME	CTYNAME								
Alabama	Autauga County	151	636	615	574	623	600	54660	
	Baldwin County	517	2187	2092	2160	2186	2240	183193	
	Barbour County	70	335	300	283	260	269	27341	
	Bibb County	44	266	245	259	247	253	22861	
	Blount County	183	744	710	646	618	603	57373	

In [28]:

```
df.loc['Michigan', 'Washtenaw County']
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\Anaconda3\lib\site-packages\pandas\core\indexing.py in _validate_key(self, key, axis)
   1789                 if not ax.contains(key):
-> 1790                     error()
   1791             except TypeError as e:

~\Anaconda3\lib\site-packages\pandas\core\indexing.py in error()
   1784                                 .format(key=key,
```

```
-> 1785                                    axis=self.obj._get_axis_name(axis)))
   1786
```

```
KeyError: 'the label [Michigan] is not in the [index]'
```

```
During handling of the above exception, another exception occurred:
```

```
KeyError                                  Traceback (most recent call last)
<ipython-input-28-948eb76ccb27> in <module>()
----> 1 df.loc['Michigan', 'Washtenaw County']

~\Anaconda3\lib\site-packages\pandas\core\indexing.py in __getitem__(self, key)
   1470             except (KeyError, IndexError):
   1471                 pass
-> 1472             return self._getitem_tuple(key)
   1473         else:
   1474             # we by definition only have the 0th axis

~\Anaconda3\lib\site-packages\pandas\core\indexing.py in _getitem_tuple(self, tup)
    868     def _getitem_tuple(self, tup):
    869         try:
--> 870             return self._getitem_lowerdim(tup)
    871         except IndexingError:
    872             pass

~\Anaconda3\lib\site-packages\pandas\core\indexing.py in _getitem_lowerdim(self, tup)
    996         for i, key in enumerate(tup):
    997             if is_label_like(key) or isinstance(key, tuple):
--> 998                 section = self._getitem_axis(key, axis=i)
    999
   1000                 # we have yielded a scalar ?

~\Anaconda3\lib\site-packages\pandas\core\indexing.py in _getitem_axis(self, key, axis)
   1909
   1910         # fall thru to straight lookup
-> 1911         self._validate_key(key, axis)
   1912         return self._get_label(key, axis=axis)
   1913

~\Anaconda3\lib\site-packages\pandas\core\indexing.py in _validate_key(self, key, axis)
   1796                 raise
   1797             except:
-> 1798                 error()
   1799
   1800     def _is_scalar_access(self, key):

~\Anaconda3\lib\site-packages\pandas\core\indexing.py in error()
   1783                 raise KeyError(u"the label [{key}] is not in the [{axis}]"
   1784                                .format(key=key,
-> 1785                                        axis=self.obj._get_axis_name(axis)))
   1786
   1787             try:
```

```
KeyError: 'the label [Michigan] is not in the [index]'
```

In [64]:

```
df.loc[ [('Michigan', 'Washtenaw County'),
         ('Michigan', 'Wayne County')] ]
```

Out[64]:

		BIRTHS2010	BIRTHS2011	BIRTHS2012	BIRTHS2013	BIRTHS2014	BIRTHS2015	POPESTIMATE2010	P
STNAME	**CTYNAME**								
Michigan	**Washtenaw County**	977	3826	3780	3662	3683	3709	345563	
	Wayne County	5918	23819	23270	23377	23607	23586	1815199	

Missing values

In [27]:

```
df = pd.read_csv('log.csv')
df
```

Out[27]:

	time	user	video	playback position	paused	volume
0	1469974424	cheryl	intro.html	5	False	10.0
1	1469974454	cheryl	intro.html	6	NaN	NaN
2	1469974544	cheryl	intro.html	9	NaN	NaN
3	1469974574	cheryl	intro.html	10	NaN	NaN
4	1469977514	bob	intro.html	1	NaN	NaN
5	1469977544	bob	intro.html	1	NaN	NaN
6	1469977574	bob	intro.html	1	NaN	NaN
7	1469977604	bob	intro.html	1	NaN	NaN
8	1469974604	cheryl	intro.html	11	NaN	NaN
9	1469974694	cheryl	intro.html	14	NaN	NaN
10	1469974724	cheryl	intro.html	15	NaN	NaN
11	1469974454	sue	advanced.html	24	NaN	NaN
12	1469974524	sue	advanced.html	25	NaN	NaN
13	1469974424	sue	advanced.html	23	False	10.0
14	1469974554	sue	advanced.html	26	NaN	NaN
15	1469974624	sue	advanced.html	27	NaN	NaN
16	1469974654	sue	advanced.html	28	NaN	5.0
17	1469974724	sue	advanced.html	29	NaN	NaN
18	1469974484	cheryl	intro.html	7	NaN	NaN
19	1469974514	cheryl	intro.html	8	NaN	NaN
20	1469974754	sue	advanced.html	30	NaN	NaN
21	1469974824	sue	advanced.html	31	NaN	NaN
22	1469974854	sue	advanced.html	32	NaN	NaN
23	1469974924	sue	advanced.html	33	NaN	NaN
24	1469977424	bob	intro.html	1	True	10.0
25	1469977454	bob	intro.html	1	NaN	NaN
26	1469977484	bob	intro.html	1	NaN	NaN
27	1469977634	bob	intro.html	1	NaN	NaN
28	1469977664	bob	intro.html	1	NaN	NaN
29	1469974634	cheryl	intro.html	12	NaN	NaN
30	1469974664	cheryl	intro.html	13	NaN	NaN
31	1469977694	bob	intro.html	1	NaN	NaN
32	1469977724	bob	intro.html	1	NaN	NaN

In [70]:

```
df.fillna?
```

In [71]:

```
df = df.set_index('time')
df = df.sort_index()
df
```

Out[71]:

time	user	video	playback position	paused	volume
1469974424	cheryl	intro.html	5	False	10.0
1469974424	sue	advanced.html	23	False	10.0
1469974454	cheryl	intro.html	6	NaN	NaN
1469974454	sue	advanced.html	24	NaN	NaN
1469974484	cheryl	intro.html	7	NaN	NaN
1469974514	cheryl	intro.html	8	NaN	NaN
1469974524	sue	advanced.html	25	NaN	NaN
1469974544	cheryl	intro.html	9	NaN	NaN
1469974554	sue	advanced.html	26	NaN	NaN
1469974574	cheryl	intro.html	10	NaN	NaN
1469974604	cheryl	intro.html	11	NaN	NaN
1469974624	sue	advanced.html	27	NaN	NaN
1469974634	cheryl	intro.html	12	NaN	NaN
1469974654	sue	advanced.html	28	NaN	5.0
1469974664	cheryl	intro.html	13	NaN	NaN
1469974694	cheryl	intro.html	14	NaN	NaN
1469974724	cheryl	intro.html	15	NaN	NaN
1469974724	sue	advanced.html	29	NaN	NaN
1469974754	sue	advanced.html	30	NaN	NaN
1469974824	sue	advanced.html	31	NaN	NaN
1469974854	sue	advanced.html	32	NaN	NaN
1469974924	sue	advanced.html	33	NaN	NaN
1469977424	bob	intro.html	1	True	10.0
1469977454	bob	intro.html	1	NaN	NaN
1469977484	bob	intro.html	1	NaN	NaN
1469977514	bob	intro.html	1	NaN	NaN
1469977544	bob	intro.html	1	NaN	NaN
1469977574	bob	intro.html	1	NaN	NaN
1469977604	bob	intro.html	1	NaN	NaN
1469977634	bob	intro.html	1	NaN	NaN
1469977664	bob	intro.html	1	NaN	NaN
1469977694	bob	intro.html	1	NaN	NaN
1469977724	bob	intro.html	1	NaN	NaN

In [72]:

```
df = df.reset_index()
df = df.set_index(['time', 'user'])
df
```

Out[72]:

time	user	video	playback position	paused	volume
1469974424	cheryl	intro.html	5	False	10.0
	sue	advanced.html	23	False	10.0
1469974454	cheryl	intro.html	6	NaN	NaN
	sue	advanced.html	24	NaN	NaN
1469974484	cheryl	intro.html	7	NaN	NaN
1469974514	cheryl	intro.html	8	NaN	NaN
1469974524	sue	advanced.html	25	NaN	NaN
1469974544	cheryl	intro.html	9	NaN	NaN
1469974554	sue	advanced.html	26	NaN	NaN
1469974574	cheryl	intro.html	10	NaN	NaN
1469974604	cheryl	intro.html	11	NaN	NaN
1469974624	sue	advanced.html	27	NaN	NaN
1469974634	cheryl	intro.html	12	NaN	NaN
1469974654	sue	advanced.html	28	NaN	5.0
1469974664	cheryl	intro.html	13	NaN	NaN
1469974694	cheryl	intro.html	14	NaN	NaN
1469974724	cheryl	intro.html	15	NaN	NaN
	sue	advanced.html	29	NaN	NaN
1469974754	sue	advanced.html	30	NaN	NaN
1469974824	sue	advanced.html	31	NaN	NaN
1469974854	sue	advanced.html	32	NaN	NaN
1469974924	sue	advanced.html	33	NaN	NaN
1469977424	bob	intro.html	1	True	10.0
1469977454	bob	intro.html	1	NaN	NaN
1469977484	bob	intro.html	1	NaN	NaN
1469977514	bob	intro.html	1	NaN	NaN
1469977544	bob	intro.html	1	NaN	NaN
1469977574	bob	intro.html	1	NaN	NaN
1469977604	bob	intro.html	1	NaN	NaN
1469977634	bob	intro.html	1	NaN	NaN
1469977664	bob	intro.html	1	NaN	NaN
1469977694	bob	intro.html	1	NaN	NaN
1469977724	bob	intro.html	1	NaN	NaN

In [73]:

```
df = df.fillna(method='ffill')
df.head()
```

Out[73]:

time	user	video	playback position	paused	volume
1469974424	cheryl	intro.html	5	False	10.0
	sue	advanced.html	23	False	10.0
1469974454	cheryl	intro.html	6	False	10.0

time	user	video	playback position	paused	volume
1469974484	cheryl	intro.html	0	False	10.0
1469974434	cheryl	intro.html	0	False	10.0
1469974434	sue	advanced.html	24	False	10.0
1469974484	cheryl	intro.html	7	False	10.0