

# 4) Tables

Vitor Kamada

December 2019

Tables, Graphics, and Figures from  
**Computational and Inferential Thinking:  
The Foundations of Data Science**

Adhikari & DeNero (2019): Ch 6. Tables

<https://www.inferentialthinking.com>

# U.S. Census Bureau, Population Division

```
from datascience import *  
import numpy as np
```

```
path_data = 'http://www2.census.gov/programs-surveys/popdiv/metadata/tables/nc-est2015-agesex-res.csv'  
data = path_data + 'nc-est2015-agesex-res.csv'  
full_census_table = Table.read_table(data)  
full_census_table
```

SEX	AGE	CENSUS2010POP	ESTIMATESBASE2010
0	0	3944153	3944160
0	1	3978070	3978090
0	2	4096929	4096939
0	3	4119040	4119051

# Annual Estimates of the Resident Population by Single Year for the United States: 2010 to 2015

<b>SEX</b>	0 = Total, 1 = Male, 2 = Female
<b>AGE</b>	999 = total population
<b>CENSUS2010POP</b>	4/1/2010 resident Census 2010 population
<b>ESTIMATESBASE2010</b>	4/1/2010 resident population estimates base
<b>POPESTIMATE2010</b>	7/1/2010 resident population estimate
<b>POPESTIMATE2011</b>	7/1/2011 resident population estimate

```
full_census_table.num_rows
```

306

# Column Labels

```
full_census_table.num_columns
```

10

```
full_census_table.labels
```

```
('SEX',  
 'AGE',  
 'CENSUS2010POP',  
 'ESTIMATESBASE2010',  
 'POPESTIMATE2010',  
 'POPESTIMATE2011',  
 'POPESTIMATE2012',  
 'POPESTIMATE2013',  
 'POPESTIMATE2014',  
 'POPESTIMATE2015')
```

# Select Relevant Columns

```
partial_census_table = full_census_table.select('SEX', 'AGE',  
                                                'POPESTIMATE2010', 'POPESTIMATE2014')  
partial_census_table
```

SEX	AGE	POPESTIMATE2010	POPESTIMATE2014
0	0	3951330	3949775
0	1	3957888	3949776
0	2	4090862	3959664

# Rename Labels

```
us_pop = partial_census_table.relabeled('POPESTIMATE2010',  
                                         '2010').relabeled('POPESTIMATE2014', '2014')  
us_pop
```

SEX	AGE	2010	2014
0	0	3951330	3949775
0	1	3957888	3949776
0	2	4090862	3959664
0	3	4111920	4007079

# Create Variables

```
change = us_pop.column('2014') - us_pop.column('2010')
census = us_pop.with_columns(
    'Change', change,
    'Percent Change', change/us_pop.column('2010')
)
census.set_format('Percent Change', PercentFormatter)
```

SEX	AGE	2010	2014	Change	Percent Change
0	0	3951330	3949775	-1555	-0.04%
0	1	3957888	3949776	-8112	-0.20%
0	2	4090862	3959664	-131198	-3.21%



# Sorting the Data

```
census.sort('Change', descending=True)
```

SEX	AGE	2010	2014	Change	Percent Change
0	999	309346863	318907401	9560538	3.09%
1	999	152088043	156955337	4867294	3.20%
2	999	157258820	161952064	4693244	2.98%

# Two Conditions

```
us_pop.where('SEX', are.equal_to(0)).where('AGE', are.between(97, 101))
```

SEX	AGE	2010	2014
0	97	68893	83089
0	98	47037	59726
0	99	32178	41468
0	100	54410	71626

# Convert and Format %

```
pop_2014 = all_ages.column('2014').item(0)
all_ages.with_column(
    'Proportion', all_ages.column('2014')/pop_2014
).set_format('Proportion', PercentFormatter)
```

SEX	AGE	2014	Proportion
0	999	318907401	100.00%
1	999	156955337	49.22%
2	999	161952064	50.78%

# Proportions of Boys and Girls among Infants

```
infants = us_pop_2014.where('AGE', are.equal_to(0))
infants_2014 = infants.column('2014').item(0)
infants.with_column(
    'Proportion', infants.column('2014')/infants_2014
).set_format('Proportion', PercentFormatter)
```

SEX	AGE	2014	Proportion
0	0	3949775	100.00%
1	0	2020326	51.15%
2	0	1929449	48.85%

# Female:Male Ratio at Each Age

```
females_all_rows = us_pop_2014.where('SEX', are.equal_to(2))  
females = females_all_rows.where('AGE', are.not_equal_to(999))  
females
```

```
males_all_rows = us_pop_2014.where('SEX', are.equal_to(1))  
males = males_all_rows.where('AGE', are.not_equal_to(999))  
males
```

SEX	AGE	2014
2	0	1929449
2	1	1931375
2	2	1935991
2	3	1957483

SEX	AGE	2014
1	0	2020326
1	1	2018401
1	2	2023673
1	3	2049596

# Compare Array

```
males.column('AGE')
```

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12,  
       13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25,  
       26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38,  
       39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51,  
       52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64,  
       65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77,  
       78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90,  
       91, 92, 93, 94, 95, 96, 97, 98, 99, 100])
```

```
females.column('AGE')
```

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12,  
       13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25,  
       26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38,  
       39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51,  
       52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64,  
       65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77,  
       78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90,  
       91, 92, 93, 94, 95, 96, 97, 98, 99, 100])
```

# Elementwise Division

```
ratios = Table().with_columns(  
    'AGE', females.column('AGE'),  
    '2014 F:M RATIO', females.column('2014')/males.column('2014')  
)  
ratios
```

```
ratios.where('AGE', are.above(75)).show()
```

		AGE	2014 F:M RATIO
AGE	2014 F:M RATIO		
		76	1.23487
0	0.955019	77	1.25797
1	0.956884	78	1.28244
2	0.956672	79	1.31627

At ages 98 and 99, there were about 3.5 to 4 times as many women as men

```
males.where('AGE', are.between(98, 100))
```

```
females.where('AGE', are.between(98, 100))
```

SEX	AGE	2014
-----	-----	------

1	98	13518
---	----	-------

1	99	8951
---	----	------

SEX	AGE	2014
-----	-----	------

2	98	46208
---	----	-------

2	99	32517
---	----	-------



# Gender Imbalance

```
%matplotlib inline
import matplotlib.pyplot as plots
plots.style.use('fivethirtyeight')
ratios.plot('AGE')
```

