# Name:Viviyan Richards W

Roll no:205229133

# Lab4. Pandas Grouping and Aggregation

## Import necessary modules

In [1]:
```
import pandas as pd
```

In [2]:
```
data = pd.read_csv("thanksgiving-2015-poll-data.csv", encoding="Latin-1")
```

In [3]:
```
data.head()
```

Out[3]:

| | RespondentID | Do you celebrate Thanksgiving? | What is typically the main dish at your Thanksgiving dinner? | What is typically the main dish at your Thanksgiving dinner? - Other (please specify) | How is the main dish typically cooked? | How is the main dish typically cooked? - Other (please specify) | What kind of stuffing/dressing do you typically have? |
|---|---|---|---|---|---|---|---|
| **0** | 4337954960 | Yes | Turkey | NaN | Baked | NaN | Bread-based |
| **1** | 4337951949 | Yes | Turkey | NaN | Baked | NaN | Bread-based |
| **2** | 4337935621 | Yes | Turkey | NaN | Roasted | NaN | Rice-based |
| **3** | 4337933040 | Yes | Turkey | NaN | Baked | NaN | Bread-based |
| **4** | 4337931983 | Yes | Tofurkey | NaN | Baked | NaN | Bread-based |

5 rows × 65 columns

In [4]:
```
data.shape
```

Out[4]:
```
(1058, 65)
```

### What are unique values of "Do you celebrate Thanksgiving?" column?

In [5]:
```
data["Do you celebrate Thanksgiving?"].unique()
```

Out[5]:
```
array(['Yes', 'No'], dtype=object)
```

**View all column names (top 5)**

In [6]:
```
data.columns[:5]
```

Out[6]:
```
Index(['RespondentID', 'Do you celebrate Thanksgiving?',
       'What is typically the main dish at your Thanksgiving dinner?',
       'What is typically the main dish at your Thanksgiving dinner? - Other (p
lease specify)',
       'How is the main dish typically cooked?'],
      dtype='object')
```

## Applying functions to Sreies

**How many male, female and NaN in "What is your gender?" column**

In [7]:
```
data["What is your gender?"].unique()
data["What is your gender?"].value_counts(dropna=False)
```

Out[7]:
```
Female    544
Male      481
NaN        33
Name: What is your gender?, dtype: int64
```

Let apply a user defined function to each value in the What is your gender? column to transform Male to 0 and female to 1

In [8]:
```
import math

def gender_code(gender_val):
    if isinstance(gender_val, float) and math.isnan(gender_val):
        return gender_val
    return int(gender_val.lower().strip() == "female")
```

**Apply gender_code() to What is your gender? column**

In [9]:
```
gender_codes = data["What is your gender?"].apply(gender_code)
print(type(gender_codes), gender_codes.head())
```

```
<class 'pandas.core.series.Series'> 0    0.0
1    1.0
2    0.0
3    0.0
4    0.0
Name: What is your gender?, dtype: float64
```

**Now, count male and females as 0s and 1s. How many in "gender" column?**

In [10]:
```
data["gender"] = gender_codes
data["gender"].value_counts(dropna=False)
```

Out[10]:
```
1.0    544
0.0    481
NaN     33
Name: gender, dtype: int64
```

## Applying functions to DataFrames

**Check the data type of each column in data using a lambda function. Just visualize data types of first 5 columns**

In [11]:
```
def get_type(row):
    return row.dtype
data.apply(get_type).head()
```

Out[11]:
```
RespondentID
int64
Do you celebrate Thanksgiving?
object
What is typically the main dish at your Thanksgiving dinner?
object
What is typically the main dish at your Thanksgiving dinner? - Other (please sp
ecify)    object
How is the main dish typically cooked?
object
dtype: object
```

## DATA CLEANING - Let us clean up Income column

**We need to convert string values representing income in "How much total combined money did all members of your HOUSEHOLD earn last year" column into numeric values. Check the unique values first**

In [12]:
```
column_name = "How much total combined money did all members of your HOUSEHOLD ea
data[column_name].value_counts(dropna=False)
```

Out[12]:
```
$25,000 to $49,999      180
Prefer not to answer    136
$50,000 to $74,999      135
$75,000 to $99,999      133
$100,000 to $124,999    111
$200,000 and up          80
$10,000 to $24,999       68
$0 to $9,999             66
$125,000 to $149,999     49
$150,000 to $174,999     40
NaN                      33
$175,000 to $199,999     27
Name: How much total combined money did all members of your HOUSEHOLD earn last
year?, dtype: int64
```

In [13]:
```
import numpy as np

def clean_income(value):
    if value == "$200,000 and up":
        return 200000
    elif value == "Prefer not to answer":
        return np.nan
    elif isinstance(value, float) and math.isnan(value):
        return np.nan
    value = value.replace(",", "").replace("$", "")
    income_low, income_high = value.split(" to ")

    return (int(income_high) + int(income_low)) / 2
```

**Now apply this function to the "How much total combined money did all members of your HOUSEHOLD earn last year?" column and put it in new column "income"**

In [14]:
```
column_name = "How much total combined money did all members of your HOUSEHOLD ea
data["income"] = data[column_name].apply(clean_income)
data["income"].head()
```

Out[14]:
```
0     87499.5
1     62499.5
2      4999.5
3    200000.0
4    112499.5
Name: income, dtype: float64
```

## Grouping Data with Pandas

**Check unique values in column, "What type of cranberry saucedo you typically have?" first.**

In [15]:
```
data["What type of cranberry saucedo you typically have?"].value_counts()
```

Out[15]:
```
Canned                    502
Homemade                  301
None                      146
Other (please specify)     25
Name: What type of cranberry saucedo you typically have?, dtype: int64
```

**Create a datafrme by filtering values "Homemade"**

In [16]:
```
column_name = "What type of cranberry saucedo you typically have?"
homemade_mask = data[column_name] == "Homemade"
print(homemade_mask.head())

homemade = data[homemade_mask]
```

```
0    False
1    False
2     True
3     True
4    False
Name: What type of cranberry saucedo you typically have?, dtype: bool
```

**Create another datafrme by filtering values "Canned"**

In [17]:
```
canned = data[data[column_name] == "Canned"]
```

**Now print mean income of homemade_df and canned_df for these two groups of people**

In [18]:
```
print(homemade["income"].mean())
print(canned["income"].mean())
```

```
94878.1072874494
83823.40340909091
```

## Use groupby() and aggregation() to find out "Who earns more income?"

**Split dataset based on "What type of cranberry saucedo you typically have?" column automatically into groups based on unique values**

In [19]:
```
column_name = "What type of cranberry saucedo you typically have?"
grouped = data.groupby(column_name)
grouped
```

Out[19]:
```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x0000026CF97F2AC0>
```

**List out all groups that are created by groupby()**

In [20]:
```
grouped.groups
```

Out[20]:
```
{'Canned': [4, 6, 8, 11, 12, 15, 18, 19, 26, 27, 38, 43, 48, 53, 58, 59, 60, 6
8, 69, 71, 74, 76, 79, 80, 86, 87, 89, 90, 91, 97, 103, 106, 107, 109, 115, 11
6, 118, 119, 123, 127, 129, 130, 132, 135, 136, 137, 140, 141, 143, 144, 145, 1
50, 153, 155, 156, 157, 158, 159, 161, 162, 163, 166, 167, 168, 169, 173, 179,
180, 181, 182, 184, 186, 190, 192, 193, 195, 198, 199, 200, 204, 205, 207, 209,
210, 211, 212, 213, 215, 217, 218, 220, 222, 224, 226, 229, 230, 231, 239, 243,
245, ...], 'Homemade': [2, 3, 5, 7, 13, 14, 16, 20, 21, 23, 25, 28, 30, 32, 33,
37, 39, 42, 44, 46, 52, 54, 56, 57, 62, 64, 66, 70, 82, 83, 85, 88, 93, 94, 96,
98, 101, 102, 108, 110, 111, 112, 114, 120, 122, 128, 134, 138, 139, 152, 165,
171, 172, 174, 175, 176, 177, 178, 183, 188, 189, 194, 201, 202, 203, 208, 219,
223, 225, 232, 234, 235, 236, 238, 241, 242, 244, 246, 248, 254, 255, 256, 259,
261, 262, 263, 264, 268, 281, 285, 286, 287, 290, 291, 292, 295, 298, 300, 302,
303, ...], 'None': [0, 17, 24, 29, 34, 36, 40, 47, 49, 51, 55, 61, 67, 72, 73,
77, 78, 81, 92, 99, 100, 104, 105, 117, 121, 124, 126, 131, 133, 142, 146, 148,
149, 160, 164, 185, 187, 191, 197, 227, 228, 237, 240, 274, 275, 319, 321, 329,
337, 362, 370, 377, 391, 395, 406, 409, 414, 417, 421, 437, 439, 466, 480, 491,
492, 495, 505, 514, 526, 529, 532, 537, 540, 553, 560, 564, 571, 573, 580, 584,
591, 594, 598, 602, 605, 606, 609, 610, 618, 626, 631, 639, 647, 658, 672, 673,
684, 700, 701, 716, ...], 'Other (please specify)': [1, 9, 154, 216, 221, 233,
249, 265, 301, 336, 380, 435, 444, 447, 513, 550, 749, 750, 784, 807, 860, 872,
905, 1000, 1007]}
```

In [21]:
```
grouped.size()
```

Out[21]:
```
What type of cranberry saucedo you typically have?
Canned                        502
Homemade                      301
None                          146
Other (please specify)         25
dtype: int64
```

In [22]:
```
for name, group in grouped:
    print(name)
    print('\t', group.shape)
    print('\t', type(group))
```

```
Canned
         (502, 67)
         <class 'pandas.core.frame.DataFrame'>
Homemade
         (301, 67)
         <class 'pandas.core.frame.DataFrame'>
None
         (146, 67)
         <class 'pandas.core.frame.DataFrame'>
Other (please specify)
         (25, 67)
         <class 'pandas.core.frame.DataFrame'>
```

In [23]:
```
grouped["income"]
```

Out[23]:
```
<pandas.core.groupby.generic.SeriesGroupBy object at 0x0000026CF770A940>
```

In [24]:
```
grouped["income"].size()
```

Out[24]:
```
What type of cranberry saucedo you typically have?
Canned                        502
Homemade                      301
None                          146
Other (please specify)         25
Name: income, dtype: int64
```

## Aggregating values in groups

### Now, find out average income

In [25]:
```
grouped["income"].agg(np.mean)
```

Out[25]:
```
What type of cranberry saucedo you typically have?
Canned                        83823.403409
Homemade                      94878.107287
None                          78886.084034
Other (please specify)        86629.978261
Name: income, dtype: float64
```

### If you want to consider all numberic attributes and find the mean for each group for every column in data, you can do as below.

In [26]:
```
grouped.agg(np.mean)
```

Out[26]:

| | RespondentID | gender | income |
|---|---|---|---|
| **What type of cranberry saucedo you typically have?** | | | |
| **Canned** | 4.336699e+09 | 0.552846 | 83823.403409 |
| **Homemade** | 4.336792e+09 | 0.533101 | 94878.107287 |
| **None** | 4.336765e+09 | 0.517483 | 78886.084034 |
| **Other (please specify)** | 4.336763e+09 | 0.640000 | 86629.978261 |

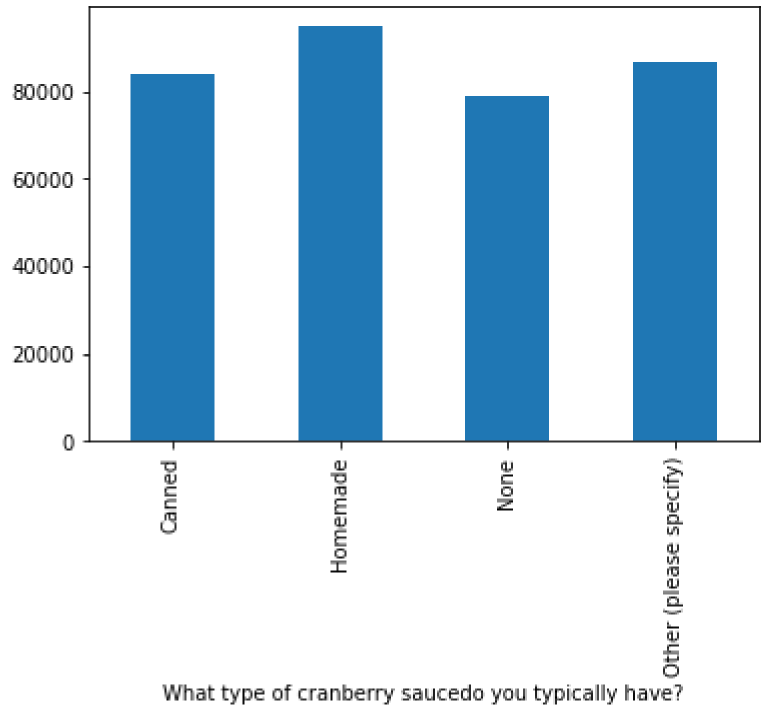
## Plotting the results

### What is the average income of each category?

In [27]:
```
%matplotlib inline

sauce = grouped.agg(np.mean)
sauce["income"].plot(kind="bar")
```

Out[27]:
```
<AxesSubplot:xlabel='What type of cranberry saucedo you typically have?'>
```



## Aggregation with multiple columns

### Find the average income of people who eat Homemade cranberry sauce and Tofurkey

In [28]:
```
grouped = data.groupby(
    ["What type of cranberry saucedo you typically have?",
     "What is typically the main dish at your Thanksgiving dinner?"])
grouped.agg(np.mean)
```

Out[28]:

| | | RespondentID | gender | income |
|---|---|---|---|---|
| **What type of cranberry saucedo you typically have?** | **What is typically the main dish at your Thanksgiving dinner?** | | | |
| **Canned** | **Chicken** | 4.336354e+09 | 0.333333 | 80999.600000 |
| | **Ham/Pork** | 4.336757e+09 | 0.642857 | 77499.535714 |
| | **I don't know** | 4.335987e+09 | 0.000000 | 4999.500000 |
| | **Other (please specify)** | 4.336682e+09 | 1.000000 | 53213.785714 |
| | **Roast beef** | 4.336254e+09 | 0.571429 | 25499.500000 |
| | **Tofurkey** | 4.337157e+09 | 0.714286 | 100713.857143 |
| | **Turkey** | 4.336705e+09 | 0.544444 | 85242.682045 |
| **Homemade** | **Chicken** | 4.336540e+09 | 0.750000 | 19999.500000 |
| | **Ham/Pork** | 4.337253e+09 | 0.250000 | 96874.625000 |
| | **I don't know** | 4.336084e+09 | 1.000000 | NaN |
| | **Other (please specify)** | 4.336863e+09 | 0.600000 | 55356.642857 |
| | **Roast beef** | 4.336174e+09 | 0.000000 | 33749.500000 |
| | **Tofurkey** | 4.336790e+09 | 0.666667 | 57916.166667 |
| | **Turducken** | 4.337475e+09 | 0.500000 | 200000.000000 |
| | **Turkey** | 4.336791e+09 | 0.531008 | 97690.147982 |
| **None** | **Chicken** | 4.336151e+09 | 0.500000 | 11249.500000 |
| | **Ham/Pork** | 4.336680e+09 | 0.444444 | 61249.500000 |
| | **I don't know** | 4.336412e+09 | 0.500000 | 33749.500000 |
| | **Other (please specify)** | 4.336688e+09 | 0.600000 | 119106.678571 |
| | **Roast beef** | 4.337424e+09 | 0.000000 | 162499.500000 |
| | **Tofurkey** | 4.336950e+09 | 0.500000 | 112499.500000 |
| | **Turducken** | 4.336739e+09 | 0.000000 | NaN |
| | **Turkey** | 4.336784e+09 | 0.523364 | 74606.275281 |
| **Other (please specify)** | **Ham/Pork** | 4.336465e+09 | 1.000000 | 87499.500000 |
| | **Other (please specify)** | 4.337335e+09 | 0.000000 | 124999.666667 |
| | **Tofurkey** | 4.336122e+09 | 1.000000 | 37499.500000 |
| | **Turkey** | 4.336724e+09 | 0.700000 | 82916.194444 |

## Aggregation with multiple functions

### Find sum, mean and standard deviation of each group in the income column of grouped dataframe

In [29]:
```
grouped["income"].agg([np.mean, np.sum, np.std]).head()
```

Out[29]:

| | | mean | sum | std |
|---|---|---|---|---|
| **What type of cranberry saucedo you typically have?** | **What is typically the main dish at your Thanksgiving dinner?** | | | |
| **Canned** | **Chicken** | 80999.600000 | 404998.0 | 75779.481062 |
| | **Ham/Pork** | 77499.535714 | 1084993.5 | 56645.063944 |
| | **I don't know** | 4999.500000 | 4999.5 | NaN |
| | **Other (please specify)** | 53213.785714 | 372496.5 | 29780.946290 |
| | **Roast beef** | 25499.500000 | 127497.5 | 24584.039538 |

### Find the number of people who live in each area type (Rural, Suburban, etc) who eat different kinds of main dishes for Thanksgiving

In [30]:
```
grouped = data.groupby("How would you describe where you live?")["What is typical
grouped.apply(lambda x: x.value_counts())
```

Out[30]:
```
How would you describe where you live?
Rural                                   Turkey                    189
                                        Other (please specify)      9
                                        Ham/Pork                    7
                                        I don't know                3
                                        Tofurkey                    3
                                        Turducken                   2
                                        Chicken                     2
                                        Roast beef                  1
Suburban                                Turkey                    449
                                        Ham/Pork                   17
                                        Other (please specify)     13
                                        Tofurkey                    9
                                        Roast beef                  3
                                        Chicken                     3
                                        I don't know                1
                                        Turducken                   1
Urban                                   Turkey                    198
                                        Other (please specify)     13
                                        Tofurkey                    8
                                        Chicken                     7
                                        Roast beef                  6
                                        Ham/Pork                    4
Name: What is typically the main dish at your Thanksgiving dinner?, dtype: int6
4
```