In [5]:
```
# Assignment Week 7-8: Data Cleaning and Transforming
'''
Name : Karthikeyan Chellamuthu

Date : 05-08-2022
'''
```

Out[5]:
```
' \nName : Karthikeyan Chellamuthu \n\nDate : 05-08-2022\n'
```

In [26]:
```
import pandas as pd
import numpy as np
import os
from datetime import datetime
```

In [27]:
```
# Chapter 7



# Define column names
cols = ('Object_Number','Is_Highlight','Is_Public_Domain','Object_ID','Department','
```

In [28]:
```
# Read csv file MetObjects
metobjects =pd.read_csv('MetObjects.csv',sep=",", skipinitialspace = True, quotechar
metobjects
```

Out[28]:

	Object_Number	Is_Highlight	Is_Public_Domain	Object_ID	Department	Object
0	Object Number,Is Highlight,Is Public Domain,Ob...	NaN	NaN	NaN	NaN	
1	1979.486.1,False,False,1,The American Wing,Coi...	NaN	NaN	NaN	NaN	
2	1980.264.5,False,False,2,The American Wing,Coi...	NaN	NaN	NaN	NaN	
3	67.265.9,False,False,3,The American Wing,Coin,...	NaN	NaN	NaN	NaN	
4	67.265.10,False,False,4,The American Wing,Coin...	NaN	NaN	NaN	NaN	
...	...	...	...	...	...	
12083	1974.356.1 recto,False,False,11814,The America...	NaN	NaN	NaN	NaN	
12084	54.143.8,False,False,11815,The American Wing,W...	NaN	NaN	NaN	NaN	
12085	1976.201.4,False,False,11816,The American Wing...	NaN	NaN	NaN	NaN	
12086	64.118,False,False,11817,The American Wing,Wat...	NaN	NaN	NaN	NaN	
12087	4	NaN	NaN	NaN	NaN	

12088 rows × 44 columns

In [29]:
```
# Get rows and column details
metobjects.shape
```

Out[29]:
```
(12088, 44)
```

In [30]:
```
# Find out the data types
metobjects.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12088 entries, 0 to 12087
Data columns (total 44 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Object_Number            12088 non-null  object 
 1   Is_Highlight             0 non-null      float64
 2   Is_Public_Domain         0 non-null      float64
 3   Object_ID                0 non-null      float64
 4   Department               0 non-null      float64
 5   Object_Name              0 non-null      float64
 6   Title                    0 non-null      float64
 7   Culture                  0 non-null      float64
 8   Period                   0 non-null      float64
 9   Dynasty                  0 non-null      float64
 10  Reign                    0 non-null      float64
 11  Portfolio                0 non-null      float64
 12  Artist_Role              0 non-null      float64
 13  Artist_Prefix            0 non-null      float64
 14  Artist_Display_Name      0 non-null      float64
 15  Artist_Display_Bio       0 non-null      float64
 16  Artist_Suffix            0 non-null      float64
 17  Artist_Alpha_Sort        0 non-null      float64
 18  Artist_Nationality       0 non-null      float64
 19  Artist_Begin_Date        0 non-null      float64
 20  Artist_End_Date          0 non-null      float64
 21  Object_Date              0 non-null      float64
 22  Object_Begin_Date        0 non-null      float64
 23  Object_End_Date          0 non-null      float64
 24  Medium                   0 non-null      float64
 25  Dimensions               0 non-null      float64
 26  Credit_Line              0 non-null      float64
 27  Geography_Type           0 non-null      float64
 28  City                     0 non-null      float64
 29  State                    0 non-null      float64
 30  County                   0 non-null      float64
 31  Country                  0 non-null      float64
 32  Region                   0 non-null      float64
 33  Subregion                0 non-null      float64
 34  Locale                   0 non-null      float64
 35  Locus                    0 non-null      float64
 36  Excavation               0 non-null      float64
 37  River                    0 non-null      float64
 38  Classification           0 non-null      float64
 39  Rights_and_Reproduction  0 non-null      float64
 40  Link_Resource            0 non-null      float64
 41  Metadata_Date            0 non-null      float64
 42  Repository               0 non-null      float64
 43  Tags                     0 non-null      float64
dtypes: float64(43), object(1)
memory usage: 4.1+ MB
```

In [31]:
```
# Find missing value for all the columns of the dataframe
metobjects.isna().sum().sort_values(ascending = False)
```

Out[31]:
```
Object_Begin_Date          12088
Is_Highlight               12088
Medium                     12088
Dimensions                 12088
Credit_Line                12088
Geography_Type             12088
City                       12088
State                      12088
County                     12088
Country                    12088
Region                     12088
Subregion                  12088
Locale                     12088
Locus                      12088
Excavation                 12088
River                      12088
Classification             12088
Rights_and_Reproduction    12088
Link_Resource              12088
Metadata_Date              12088
Repository                 12088
Object_End_Date            12088
Tags                       12088
Object_Date                12088
Reign                      12088
Is_Public_Domain           12088
Object_ID                  12088
Department                 12088
Object_Name                12088
Title                      12088
Culture                    12088
Period                     12088
Dynasty                    12088
Portfolio                  12088
Artist_End_Date            12088
Artist_Role                12088
Artist_Prefix              12088
Artist_Display_Name        12088
Artist_Display_Bio         12088
Artist_Suffix              12088
Artist_Alpha_Sort          12088
Artist_Nationality         12088
Artist_Begin_Date          12088
Object_Number                  0
dtype: int64
```

In [32]:
```
#  identify the duplicates
metobjects.duplicated(['Object_Number']).sum()
```

Out[32]:
```
165
```

In [33]:
```
# Clean or Remove the duplicates and gets it rows and column details
Nodup_met_objects = metobjects.drop_duplicates(subset='Object_Number')
Nodup_met_objects.shape
```

Out[33]:
```
(11923, 44)
```

In [34]:
```python
# Chapter 8
# Hierarchical index
# Create a subset dataframe
subset_objects = Nodup_met_objects[['Object_Number', 'Department', 'Title', 'Object_

# Select random sample of 2000 rows
random_objects = subset_objects.sample(2000)

# Reset Index of random dataframe
random_objects = random_objects.reset_index()
random_objects.head()
```

Out[34]:

	index	Object_Number	Department	Title	Object_Name	Medium	Classification	I
0	9064	33.120.481,False,True,8438,The American Wing,S...	NaN	NaN	NaN	NaN	NaN	
1	9425	1978.302.80,False,False,8911,The American Wing...	NaN	NaN	NaN	NaN	NaN	
2	4830	Inst.68.8.36,False,True,4633,The American Wing...	NaN	NaN	NaN	NaN	NaN	
3	9269	33.120.591,False,False,8746,The American Wing,...	NaN	NaN	NaN	NaN	NaN	
4	4662	50.187.66,False,True,4466,The American Wing,Mi...	NaN	NaN	NaN	NaN	NaN	

In [35]:
```python
# Create Hierarchical indexing using set_index
random_H_objects = random_objects.set_index(['Department', 'Medium'])
random_H_objects.head()
```

Out[35]:

		index	Object_Number	Title	Object_Name	Classification	Is_Pu
Department	**Medium**						
NaN	**NaN**	9064	33.120.481,False,True,8438,The American Wing,S...	NaN	NaN	NaN	
	NaN	9425	1978.302.80,False,False,8911,The American Wing...	NaN	NaN	NaN	
	NaN	4830	Inst.68.8.36,False,True,4633,The American Wing...	NaN	NaN	NaN	
	NaN	9269	33.120.591,False,False,8746,The American Wing,...	NaN	NaN	NaN	
	NaN	4662	50.187.66,False,True,4466,The American Wing,Mi...	NaN	NaN	NaN	

In [36]:
```python
# Reshaping the sample dataframe
stack_objects = random_objects.stack()
stack_objects
```

Out[36]:
```
0     index                                                       9064
      Object_Number    33.120.481,False,True,8438,The American Wing,S...
1     index                                                       9425
```

```
      Object_Number    1978.302.80,False,False,8911,The American Wing...
2     index                                                      4830
                                               ...
1997  Object_Number    "64.36.2a, b",False,True,5618,The American Win...
1998  index                                                      4578
      Object_Number    17.108.9,False,False,4386,The American Wing,Ho...
1999  index                                                      5739
      Object_Number    1982.439.23,False,True,5444,The American Wing,...
Length: 4000, dtype: object
```

In [37]:

```
# We will now reshape the rows into the columns using unstack
stack_objects.unstack()
```

Out[37]:

	index	Object_Number
0	9064	33.120.481,False,True,8438,The American Wing,S...
1	9425	1978.302.80,False,False,8911,The American Wing...
2	4830	Inst.68.8.36,False,True,4633,The American Wing...
3	9269	33.120.591,False,False,8746,The American Wing,...
4	4662	50.187.66,False,True,4466,The American Wing,Mi...
...	...	...
1995	3449	11.60.157aÐc,False,True,3184,The American Wing...
1996	6617	60.111.65,False,False,6310,The American Wing,P...
1997	5927	"64.36.2a, b",False,True,5618,The American Win...
1998	4578	17.108.9,False,False,4386,The American Wing,Ho...
1999	5739	1982.439.23,False,True,5444,The American Wing,...

2000 rows × 2 columns

In [38]:

```
candy_2015=pd.read_excel('CANDY-HIERARCHY-2015-SURVEY-Responses.xlsx')
candy_2016=pd.read_excel('BOING-BOING-CANDY-HIERARCHY-2016-SURVEY-Responses.xlsx')
candy_2017=pd.read_excel('candyhierarchy2017.xlsx')
```

```
C:\Users\LENOVO\anaconda3\lib\site-packages\openpyxl\worksheet\_reader.py:312: UserW
arning: Unknown extension is not supported and will be removed
  warn(msg)
```

In [39]:

```
# Merge data sets - inner join (shouldn't return results as the survey data is from
pd.merge(candy_2015, candy_2016, how= 'inner')
```

Out[39]:

	Timestamp	How old are you?	Are you going actually going trick or treating yourself?	[Butterfinger]	[100 Grand Bar]	[Anonymous brown globs that come in black and orange wrappers]	[Any full-sized candy bar]	[Black Jacks]	[Bonkers]	[Bottl Caps

0 rows × 155 columns

In [40]:

```
# Merge data sets - left join (should return 2015 results )
pd.merge(candy_2015, candy_2016, how= 'left')
```

Out[40]:

	Timestamp	How old are you?	Are you going actually going trick or treating yourself?	[Butterfinger]	[100 Grand Bar]	[Anonymous brown globs that come in black and orange wrappers]	[Any full-sized candy bar]	[Black Jacks]	[Bonker
0	2015-10-23 08:46:20.451	35.0	No	JOY	NaN	DESPAIR	JOY	NaN	Na
1	2015-10-23 08:46:51.583	41.0	No	JOY	JOY	DESPAIR	JOY	DESPAIR	DESPAI
2	2015-10-23 08:47:34.285	33.0	No	DESPAIR	DESPAIR	DESPAIR	JOY	DESPAIR	DESPAI
3	2015-10-23 08:47:58.964	31.0	No	JOY	JOY	DESPAIR	JOY	DESPAIR	DESPAI

4	2015-10-23 08:48:11.719	30.0	No	NaN	JOY	DESPAIR	JOY	NaN	Na
...	...	...	...	...	...	...	...	...	
5625	2015-10-31 05:23:40.526	50.0	No	DESPAIR	DESPAIR	DESPAIR	JOY	DESPAIR	DESPAI
5626	2015-10-31 05:29:26.937	43.0	No	JOY	JOY	DESPAIR	JOY	DESPAIR	DESPAI
5627	2015-10-31 06:13:29.083	35.0	Yes	NaN	JOY	DESPAIR	JOY	NaN	Na
5628	2015-10-31 06:26:52.566	38.0	No	JOY	JOY	JOY	JOY	JOY	JC
5629	2015-10-31 06:41:31.904	44.0	No	DESPAIR	JOY	DESPAIR	JOY	DESPAIR	DESPAI

5630 rows × 155 columns

In [10]:
```
metobjects
```

Out[10]:

	Object_Number	Is_Highlight	Is_Public_Domain	Object_ID	Departn
0	version https://git-lfs.github.com/spec/v1	NaN	NaN	NaN	
1	oid sha256:fd00b55c6d3a7ea8eded8b832b47e4f7e50...	NaN	NaN	NaN	
2	size 310397416	NaN	NaN	NaN	

3 rows × 45 columns

In [164...
```
#Drop the first column
metobjects.drop([0], inplace=True)
metobjects.head()
```

Out[164...

	Object_Number	Is_Highlight	Is_Public_Domain	Object_ID	Department	Object_Name
1	1979.486.1,False,False,1,The American Wing,Coi...	False	False	1	The American Wing	Coin
2	1980.264.5,False,False,2,The American Wing,Coi...	False	False	2	The American Wing	Coin
3	67.265.9,False,False,3,The American Wing,Coin,...	False	False	3	The American Wing	Coin

	Object_Number	Is_Highlight	Is_Public_Domain	Object_ID	Department	Object_Name
4	67.265.10,False,False,4,The American Wing,Coin...	False	False	4	The American Wing	Coin
5	67.265.11,False,False,5,The American Wing,Coin...	False	False	5	The American Wing	Coin

5 rows × 45 columns

In [165…

```
#Drop the first column
metobjects.drop(columns=['Object_Number'], inplace=True)
metobjects.head()
```

Out[165…

	Is_Highlight	Is_Public_Domain	Object_ID	Department	Object_Name	Title	Culture	Period	D
1	False	False	1	The American Wing	Coin	One-dollar Liberty Head Coin			
2	False	False	2	The American Wing	Coin	Ten-dollar Liberty Head Coin			
3	False	False	3	The American Wing	Coin	Two-and-a-Half Dollar Coin			
4	False	False	4	The American Wing	Coin	Two-and-a-Half Dollar Coin			
5	False	False	5	The American Wing	Coin	Two-and-a-Half Dollar Coin			

5 rows × 44 columns

In [166…

```
#identify nulls
metobjects.isnull()
```

Out[166…

	Is_Highlight	Is_Public_Domain	Object_ID	Department	Object_Name	Title	Culture	Period

	Is_Highlight	Is_Public_Domain	Object_ID	Department	Object_Name	Title	Culture	Period
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False
5	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...
12083	False	False	False	False	False	False	False	False
12084	False	False	False	False	False	False	False	False
12085	False	False	False	False	False	False	False	False
12086	False	False	False	False	False	False	False	False
12087	True	True	True	True	True	True	True	True

12087 rows × 44 columns

In [167…

```
#Convert blank space into nulls
metobjects = metobjects.apply(lambda x: x.str.strip() if isinstance(x, str) else x).
```

In [168…

```
# Check for nulls
metobjects.isnull()
```

Out[168…

	Is_Highlight	Is_Public_Domain	Object_ID	Department	Object_Name	Title	Culture	Period
1	False	False	False	False	False	False	True	True
2	False	False	False	False	False	False	True	True
3	False	False	False	False	False	False	True	True
4	False	False	False	False	False	False	True	True
5	False	False	False	False	False	False	True	True
...	...	...	...	...	...	...	...	...
12083	False	False	False	False	False	False	False	False
12084	False	False	False	False	False	False	False	True
12085	False	False	False	False	False	False	False	False
12086	False	False	False	False	False	False	False	True
12087	True	True	True	True	True	True	True	True

12087 rows × 44 columns

In [149…

```
#Filter out missing data
metobjects_df1=metobjects
# dropping rows with null
```

```
metobjects_df1.dropna(inplace=True)
metobjects_df1
```

Out[149…

Is_Highlight	Is_Public_Domain	Object_ID	Department	Object_Name	Title	Culture	Period	Dyna

0 rows × 44 columns

In [171…

```
#Filter out missing data
metobjects_df2=metobjects
# drops the row with all null values in the row
metobjects_df2.dropna(how='all',inplace=True)
metobjects_df2
```

Out[171…

	Is_Highlight	Is_Public_Domain	Object_ID	Department	Object_Name	Title	Culture
1	False	False	1	The American Wing	Coin	One-dollar Liberty Head Coin	NaN
2	False	False	2	The American Wing	Coin	Ten-dollar Liberty Head Coin	NaN
3	False	False	3	The American Wing	Coin	Two-and-a-Half Dollar Coin	NaN
4	False	False	4	The American Wing	Coin	Two-and-a-Half Dollar Coin	NaN
5	False	False	5	The American Wing	Coin	Two-and-a-Half Dollar Coin	NaN
...	...	...	...	...	...	...	...
12083	False	False	11814	The American Wing	Watercolor	"Rialto Bridge (Covered Bridge	Venice)"
12084	False	False	11815	The American Wing	Watercolor	The Rider	American
12085	False	False	11816	The American Wing	Watercolor	"Umbrellas in the Rain	Venice"
12086	False	False	11817	The American Wing	Watercolor	Worship of Moloch (The Golden Idol)	American
12087	None	None	None	None	None	None	None

12087 rows × 44 columns

In [191…

```
#Fill in missing data
#  Fill with mean value
metobjects["Object_End_Date"] = pd.to_numeric(metobjects.Object_End_Date, errors='co
meanval= metobjects['Object_End_Date'].mean() # Determine the mean value
metobjects['Object_End_Date'].fillna(value=meanval, inplace=True) # Fill the mean va
metobjects.Object_End_Date
```

Out[191…

```
1        1794.000000
2        1901.000000
3        1927.000000
4        1927.000000
5        1927.000000
            ...
12083    1858.000000
12084    1924.000000
12085    1858.000000
12086    1950.000000
12087    1842.789876
Name: Object_End_Date, Length: 12087, dtype: float64
```

In [193…

```
#Fill in missing data
#  Fill with chosen default
metobjects['Culture'] = metobjects.Culture.fillna('Unknown')
metobjects.Culture
```

Out[193…

```
1          Unknown
2          Unknown
3          Unknown
4          Unknown
5          Unknown
            ...
12083     Venice)"
12084     American
12085      Venice"
12086     American
12087      Unknown
Name: Culture, Length: 12087, dtype: object
```

In [190…

```
metobjects.Object_End_Date
```

Out[190…

```
1        1794.000000
2        1901.000000
3        1927.000000
4        1927.000000
5        1927.000000
            ...
12083    1858.000000
12084    1924.000000
12085    1858.000000
12086    1950.000000
12087    1842.789876
Name: Object_End_Date, Length: 12087, dtype: float64
```

In [195…

```
# Remove duplicates
#  Drop the duplicates in the dataframe; an entire duplicated row gets dropped
metobjects.duplicated()
metobjects.drop_duplicates(inplace=True)
metobjects
```

Out[195…

	Is_Highlight	Is_Public_Domain	Object_ID	Department	Object_Name	Title	Culture
1	False	False	1	The American Wing	Coin	One-dollar Liberty Head Coin	Unknown
2	False	False	2	The American Wing	Coin	Ten-dollar Liberty Head Coin	Unknown
3	False	False	3	The American Wing	Coin	Two-and-a-Half Dollar Coin	Unknown
4	False	False	4	The American Wing	Coin	Two-and-a-Half Dollar Coin	Unknown
5	False	False	5	The American Wing	Coin	Two-and-a-Half Dollar Coin	Unknown
...	...	...	...	...	...	...	...
12083	False	False	11814	The American Wing	Watercolor	"Rialto Bridge (Covered Bridge	Venice)"
12084	False	False	11815	The American Wing	Watercolor	The Rider	American
12085	False	False	11816	The American Wing	Watercolor	"Umbrellas in the Rain	Venice"
12086	False	False	11817	The American Wing	Watercolor	Worship of Moloch (The Golden Idol)	American
12087	None	None	None	None	None	None	Unknown

11922 rows × 44 columns

In [202…

```
# Remove duplicates
# Drop the duplicates in a particular field in the dataframe; keep = last returns la
metobjects_df3= metobjects
metobjects_df3.drop_duplicates(['Department','Object_Name'], keep='last')
metobjects_df3.head()
```

Out[202…

	Is_Highlight	Is_Public_Domain	Object_ID	Department	Object_Name	Title	Culture	Period
1	False	False	1	The American Wing	Coin	One-dollar Liberty Head Coin	Unknown	NaN

	Is_Highlight	Is_Public_Domain	Object_ID	Department	Object_Name	Title	Culture	Period
2	False	False	2	The American Wing	Coin	Ten-dollar Liberty Head Coin	Unknown	NaN
3	False	False	3	The American Wing	Coin	Two-and-a-Half Dollar Coin	Unknown	NaN
4	False	False	4	The American Wing	Coin	Two-and-a-Half Dollar Coin	Unknown	NaN
5	False	False	5	The American Wing	Coin	Two-and-a-Half Dollar Coin	Unknown	NaN

5 rows × 44 columns

In [227…

```
# Data Transformation
# Data Transformation using mapping

metobjects_df3=metobjects
object_nm = {'Coin': 'COIN',
                'Ring': 'RING'
                 }

ob_nm=metobjects_df3['Object_Name']
metobjects_df3['Object_Name']=ob_nm.map(object_nm)

metobjects_df3
```

Out[227…

	Is_Highlight	Is_Public_Domain	Object_ID	Department	Object_Name	Title	Culture
1	False	False	1	The American Wing	NaN	One-dollar Liberty Head Coin	Unknown
2	False	False	2	The American Wing	NaN	Ten-dollar Liberty Head Coin	Unknown
3	False	False	3	The American Wing	NaN	Two-and-a-Half Dollar Coin	Unknown
4	False	False	4	The American Wing	NaN	Two-and-a-Half Dollar Coin	Unknown

	Is_Highlight	Is_Public_Domain	Object_ID	Department	Object_Name	Title	Culture
5	False	False	5	The American Wing	NaN	Two-and-a-Half Dollar Coin	Unknown
...	...	...	...	...	...	...	...
12083	False	False	11814	The American Wing	NaN	"Rialto Bridge (Covered Bridge	Venice)"
12084	False	False	11815	The American Wing	NaN	The Rider	American
12085	False	False	11816	The American Wing	NaN	"Umbrellas in the Rain	Venice"
12086	False	False	11817	The American Wing	NaN	Worship of Moloch (The Golden Idol)	American
12087	None	None	None	None	NaN	None	Unknown

11922 rows × 44 columns

In [230…

```
# Data Transformation
#Replace values
metobjects_df3.Department.replace('The American Wing', 'THE AMERICAN WING', inplace=
metobjects_df3.Department
```

Out[230…

```
1        THE AMERICAN WING
2        THE AMERICAN WING
3        THE AMERICAN WING
4        THE AMERICAN WING
5        THE AMERICAN WING
               ...
12083    THE AMERICAN WING
12084    THE AMERICAN WING
12085    THE AMERICAN WING
12086    THE AMERICAN WING
12087                 None
Name: Department, Length: 11922, dtype: object
```

In [233…

```
# Data Transformation
#Dicretization & Binning
metobjects_df3.Object_End_Date
bins=[1500,1600,1700,1800,1900,2000,2100]
metobjects_df3["ObjectRange"]=pd.cut(metobjects_df3.Object_End_Date,bins)
metobjects_df3.ObjectRange
```

Out[233…

```
1        (1700, 1800]
2        (1900, 2000]
3        (1900, 2000]
4        (1900, 2000]
5        (1900, 2000]
             ...
12083    (1800, 1900]
12084    (1900, 2000]
12085    (1800, 1900]
12086    (1900, 2000]
12087    (1800, 1900]
Name: ObjectRange, Length: 11922, dtype: category
Categories (6, interval[int64]): [(1500, 1600] < (1600, 1700] < (1700, 1800] < (180
0, 1900] < (1900, 2000] < (2000, 2100]]
```

In [5]:

```
#Dataset 2 - Candy data ingestion

candy_2015=pd.read_excel('CANDY-HIERARCHY-2015-SURVEY-Responses.xlsx')
candy_2016=pd.read_excel('BOING-BOING-CANDY-HIERARCHY-2016-SURVEY-Responses.xlsx')
candy_2017=pd.read_excel('candyhierarchy2017.xlsx')
```

In [6]:

```
candy_2016.head()
```

Out[6]:

	Timestamp	Are you going actually going trick or treating yourself?	Your gender:	How old are you?	Which country do you live in?	Which state, province, county do you live in?	[100 Grand Bar]	[Anonymous brown globs that come in black and orange wrappers]	[Any full-sized candy bar]	[Black Jacks]
0	2016-10-24 05:09:23.033	No	Male	22	Canada	Ontario	JOY	DESPAIR	JOY	MEH
1	2016-10-24 05:09:54.798	No	Male	45	usa	il	MEH	MEH	JOY	JOY
2	2016-10-24 05:13:06.734	No	Female	48	US	Colorado	JOY	DESPAIR	JOY	MEH

3	2016-10-24 05:14:17.192	No	Male	57	usa	il	JOY	MEH	JOY	MEH
4	2016-10-24 05:14:24.625	Yes	Male	42	USA	South Dakota	MEH	DESPAIR	JOY	DESPAIR

5 rows × 123 columns

In [7]:

```python
# Hierachical indexing

candy_2017_index=candy_2017.set_index(['Q2: GENDER', 'Q10: DRESS']).sort_index() # A
candy_2017_index.iloc[3] # Access the data using index location
```

Out[7]:

```
Internal ID                           90273060
Q1: GOING OUT?                              No
Q3: AGE                                     37
Q4: COUNTRY                                USA
Q5: STATE, PROVINCE, COUNTY, ETC            DC
                                       ...
Q12: MEDIA [Daily Dish]                    NaN
Q12: MEDIA [Science]                         1
Q12: MEDIA [ESPN]                          NaN
Q12: MEDIA [Yahoo]                         NaN
Click Coordinates (x, y)               (72, 4)
Name: (Female, Blue and black), Length: 118, dtype: object
```

In [281…

```python
candy_2017_index.iloc[:3] # Access the rows using index location
```

Out[281…

		Internal ID	Q1: GOING OUT?	Q3: AGE	Q4: COUNTRY	Q5: STATE, PROVINCE, COUNTY, ETC	Q6 | 100 Grand Bar	Q6 | Anonymous brown globs that come in black and orange wrappers\t(a.k.a. Mary Janes)	Q6 | Any full-sized candy bar
Q2: GENDER	**Q10: DRESS**								
	Blue and black	90272868	No	37	Canada	Ontario	MEH	JOY	JOY
Female	Blue and black	90272948	No	50	United States	Illinois	MEH	DESPAIR	MEH
	Blue and black	90272995	No	40	Canada	yukon	MEH	DESPAIR	JOY

3 rows × 118 columns

In [8]:

```
# Merge data sets - inner join (shouldn't return results as the survey data is from
pd.merge(candy_2015, candy_2016, how= 'inner')
```

Out[8]:

Timestamp	How old are you?	Are you going actually going trick or treating yourself?	[Butterfinger]	[100 Grand Bar]	[Anonymous brown globs that come in black and orange wrappers]	[Any full-sized candy bar]	[Black Jacks]	[Bonkers]	[Bottl Caps

0 rows × 155 columns

In [9]:

```
# Merge data sets - left join (should return 2015 results )
pd.merge(candy_2015, candy_2016, how= 'left')
```

Out[9]:

	Timestamp	How old are you?	Are you going actually going trick or treating yourself?	[Butterfinger]	[100 Grand Bar]	[Anonymous brown globs that come in black and orange wrappers]	[Any full-sized candy bar]	[Black Jacks]	[Bonkers
0	2015-10-23 08:46:20.451	35	No	JOY	NaN	DESPAIR	JOY	NaN	Na
1	2015-10-23 08:46:51.583	41	No	JOY	JOY	DESPAIR	JOY	DESPAIR	DESPAI
2	2015-10-23 08:47:34.285	33	No	DESPAIR	DESPAIR	DESPAIR	JOY	DESPAIR	DESPAI

3	2015-10-23 08:47:58.964	31	No	JOY	JOY	DESPAIR	JOY	DESPAIR	DESPAI
4	2015-10-23 08:48:11.719	30	No	NaN	JOY	DESPAIR	JOY	NaN	Na
...	...	...	...	...	...	...	...	...	.
5625	2015-10-31 05:23:40.526	50	No	DESPAIR	DESPAIR	DESPAIR	JOY	DESPAIR	DESPAI
5626	2015-10-31 05:29:26.937	43	No	JOY	JOY	DESPAIR	JOY	DESPAIR	DESPAI
5627	2015-10-31 06:13:29.083	35	Yes	NaN	JOY	DESPAIR	JOY	NaN	Na
5628	2015-10-31 06:26:52.566	38	No	JOY	JOY	JOY	JOY	JOY	JO
5629	2015-10-31 06:41:31.904	44	No	DESPAIR	JOY	DESPAIR	JOY	DESPAIR	DESPAI

5630 rows × 155 columns

In [41]:

```
# Pivot tables -

# Create a data frame with subset of fields
pvtdf =candy_2015.iloc[:,0:4]

# Convert timestamp into date
pvtdf["survey_dt"] = pd.to_datetime(pvtdf['Timestamp']).apply(lambda x: x.date())

#Rename field names
pvtdf.set_axis(['Timestamp', 'Age', 'Trick_Treat_Participation', 'Butterfinger', 'Da
pvtdf1=pvtdf[['Date', 'Butterfinger', 'Trick_Treat_Participation']]

#Convert Joy/Despair values to numeric
pvtdf1["Butterfinger"].replace({"JOY": "1", "DESPAIR": "2", "Nan": "3"}, inplace=Tru
pvtdf1.Butterfinger = pvtdf1.Butterfinger.astype(float)

#Pivot rows into columns with avg
pvtdf2 = pvtdf1.pivot_table(index=['Date'], columns='Trick_Treat_Participation', val
pvtdf2.columns = ['_'.join(col).strip() for col in pvtdf2.columns.values]

pvtdf2=pvtdf2.reset_index()

pvtdf2
```

```
C:\Users\LENOVO\anaconda3\lib\site-packages\pandas\core\generic.py:6619: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  return self._update_inplace(result)
C:\Users\LENOVO\anaconda3\lib\site-packages\pandas\core\generic.py:5516: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
```

```
ser_guide/indexing.html#returning-a-view-versus-a-copy
  self[name] = value
```

Out[41]:

	Date	Butterfinger_No	Butterfinger_Yes
0	2015-10-23	1.213618	1.210526
1	2015-10-24	1.178082	1.205128
2	2015-10-25	1.157447	1.142857
3	2015-10-26	1.204724	1.454545
4	2015-10-27	1.153846	1.466667
5	2015-10-28	1.217997	1.272727
6	2015-10-29	1.215101	1.219048
7	2015-10-30	1.186869	1.571429
8	2015-10-31	1.375000	NaN

In [42]:

```
# Pivot table with totals

pvtdf3 = pd.pivot_table(pvtdf1, index=['Date'], columns=['Trick_Treat_Participation'
            aggfunc=np.sum, fill_value=0, margins=True)

pvtdf3
```

Out[42]:

	Butterfinger		
Trick_Treat_Participation	**No**	**Yes**	**All**
Date			
2015-10-23	909	115	1024.0
2015-10-24	430	47	477.0
2015-10-25	272	32	304.0
2015-10-26	153	32	185.0
2015-10-27	75	22	97.0
2015-10-28	2274	224	2498.0
2015-10-29	1384	128	1512.0
2015-10-30	235	22	257.0
2015-10-31	11	0	11.0
All	5743	622	6365.0

In [43]:

```
# Chapter 10



# Group By Dictionaries

pvtdf

pvtdf1index=pvtdf.set_index(['Date']).sort_index()
pvtdf1index["Butterfinger"].replace({"JOY": "1", "DESPAIR": "2", "Nan": "3"}, inplac
pvtdf1index.Butterfinger = pvtdf1index.Butterfinger.astype(float)
mapping={'Butterfinger': 'Butterfinger'}
pvtdf4=pvtdf1index.groupby(mapping, axis=1)
pvtdf4.sum()
```

Out[43]:

	Butterfinger
Date	
2015-10-23	1.0
2015-10-23	1.0
2015-10-23	2.0
2015-10-23	1.0
2015-10-23	0.0
...	...
2015-10-31	2.0
2015-10-31	1.0
2015-10-31	0.0
2015-10-31	1.0
2015-10-31	2.0

5630 rows × 1 columns

In [44]:

```
# Group By Series

map_series = pd.Series(mapping)
pvtdf1index.groupby(map_series, axis=1).sum()
```

Out[44]:

	Butterfinger
Date	
2015-10-23	1.0
2015-10-23	1.0
2015-10-23	2.0
2015-10-23	1.0
2015-10-23	0.0
...	...
2015-10-31	2.0
2015-10-31	1.0
2015-10-31	0.0
2015-10-31	1.0
2015-10-31	2.0

5630 rows × 1 columns

In [42]:
```
# Group By Functions
pvtdf1index.groupby('Butterfinger').min()
```

Out[42]:

	Timestamp	Trick_Treat_Participation
Butterfinger		
1.0	2015-10-23 08:46:20.451	No
2.0	2015-10-23 08:47:34.285	No

In [47]:
```
# Split/Apply/Combine
pvtdf1index["Age"] = pd.to_numeric(pvtdf1index.Age, errors='coerce')
pvtdf1index

def old(df, n=5, column='Age'):
    return df.sort_values(by=column)[-n:]

old(pvtdf1index, n=6)


pvtdf1index.groupby('Trick_Treat_Participation').apply(old)
```

Out[47]:

		Timestamp	Age	Trick_Treat_Participation	Butterfinger
Trick_Treat_Participation	**Date**				
No	**2015-10-30**	2015-10-30 14:12:27.299	NaN	No	1.0
	2015-10-30	2015-10-30 15:39:39.356	NaN	No	1.0
	2015-10-30	2015-10-30 17:34:01.613	NaN	No	1.0
	2015-10-30	2015-10-30 20:51:09.502	NaN	No	1.0
	2015-10-31	2015-10-31 05:15:32.494	NaN	No	NaN
Yes	**2015-10-29**	2015-10-29 09:07:25.335	NaN	Yes	1.0
	2015-10-29	2015-10-29 12:57:56.042	NaN	Yes	2.0
	2015-10-29	2015-10-29 17:26:57.566	NaN	Yes	1.0
	2015-10-30	2015-10-30 06:44:52.414	NaN	Yes	1.0
	2015-10-30	2015-10-30 14:29:40.318	NaN	Yes	2.0

In [48]:
```
# crosstab

pd.crosstab(pvtdf1index.Trick_Treat_Participation, pvtdf1index.Butterfinger,margins=
```

Out[48]:

Butterfinger	1.0	2.0	All

Butterfinger	1.0	2.0	All
Trick_Treat_Participation			
No	3763	990	4753
Yes	366	128	494
All	4129	1118	5247

In [49]:
```
# Chapter 11



# Convert timestamp to string

from datetime import datetime
dt_object = pvtdf1index.Timestamp.dt.strftime('%Y-%m-%d')
dt_object
```

Out[49]:
```
Date
2015-10-23    2015-10-23
2015-10-23    2015-10-23
2015-10-23    2015-10-23
2015-10-23    2015-10-23
2015-10-23    2015-10-23
                 ...
2015-10-31    2015-10-31
2015-10-31    2015-10-31
2015-10-31    2015-10-31
2015-10-31    2015-10-31
2015-10-31    2015-10-31
Name: Timestamp, Length: 5630, dtype: object
```

In [50]:
```
# Convert string to timestamp
pd.to_datetime(dt_object)
```

Out[50]:
```
Date
2015-10-23   2015-10-23
2015-10-23   2015-10-23
2015-10-23   2015-10-23
2015-10-23   2015-10-23
2015-10-23   2015-10-23
                ...
2015-10-31   2015-10-31
2015-10-31   2015-10-31
2015-10-31   2015-10-31
2015-10-31   2015-10-31
2015-10-31   2015-10-31
Name: Timestamp, Length: 5630, dtype: datetime64[ns]
```

In [51]:
```
# Generate date range

datelist = pd.date_range(datetime.today(), periods=10).tolist()
datelist
```

Out[51]:
```
[Timestamp('2022-05-08 22:10:22.434667', freq='D'),
 Timestamp('2022-05-09 22:10:22.434667', freq='D'),
 Timestamp('2022-05-10 22:10:22.434667', freq='D'),
 Timestamp('2022-05-11 22:10:22.434667', freq='D'),
 Timestamp('2022-05-12 22:10:22.434667', freq='D'),
 Timestamp('2022-05-13 22:10:22.434667', freq='D'),
 Timestamp('2022-05-14 22:10:22.434667', freq='D'),
 Timestamp('2022-05-15 22:10:22.434667', freq='D'),
 Timestamp('2022-05-16 22:10:22.434667', freq='D'),
 Timestamp('2022-05-17 22:10:22.434667', freq='D')]
```

In [52]:

```
#Frequencies
pd.date_range('2020-01-01', '2020-01-03 23:59', freq='4h')
```

Out[52]:

```
DatetimeIndex(['2020-01-01 00:00:00', '2020-01-01 04:00:00',
               '2020-01-01 08:00:00', '2020-01-01 12:00:00',
               '2020-01-01 16:00:00', '2020-01-01 20:00:00',
               '2020-01-02 00:00:00', '2020-01-02 04:00:00',
               '2020-01-02 08:00:00', '2020-01-02 12:00:00',
               '2020-01-02 16:00:00', '2020-01-02 20:00:00',
               '2020-01-03 00:00:00', '2020-01-03 04:00:00',
               '2020-01-03 08:00:00', '2020-01-03 12:00:00',
               '2020-01-03 16:00:00', '2020-01-03 20:00:00'],
              dtype='datetime64[ns]', freq='4H')
```

In [53]:

```
# offseting
pd.Series(np.random.randn(4), index=pd.date_range('1/1/2020', periods=4, freq='M'))
```

Out[53]:

```
2020-01-31    0.635155
2020-02-29    0.102822
2020-03-31    0.200000
2020-04-30   -0.437259
Freq: M, dtype: float64
```

In [54]:

```
# offseting
from pandas.tseries.offsets import Day, MonthEnd
now=datetime(2011,11,17)
now+MonthEnd()
```

Out[54]:

```
Timestamp('2011-11-30 00:00:00')
```

In [55]:

```
# Date to periods
pd.DataFrame(pd.date_range('2014-01-01', freq='2w', periods=12))
```

Out[55]:

	0
0	2014-01-05
1	2014-01-19
2	2014-02-02
3	2014-02-16
4	2014-03-02
5	2014-03-16
6	2014-03-30
7	2014-04-13
8	2014-04-27
9	2014-05-11
10	2014-05-25

	0
11	2014-06-08

In [56]:
```
#Period Frequency Conversions
period = pd.Period(freq="S", year = 2021, month = 4, day = 16, hour = 2, minute = 35
period
```

Out[56]:
```
Period('2021-04-16 02:35:15', 'S')
```