In [1]:
```
#importing the pandas library and reading the dataset to a dataframe
import pandas as pd
data = pd.read_csv(r"C:\Users\Chinenye Claire\Downloads\FoodBalanceSheets_E_Africa_NOFLAG.csv", encod
data.head()
```

Out[1]:

	Area Code	Area	Item Code	Item	Element Code	Element	Unit	Y2014	Y2015	Y2016	Y2017	Y2018
0	4	Algeria	2501	Population	511	Total Population - Both sexes	1000 persons	38924.00	39728.00	40551.00	41389.00	42228.00
1	4	Algeria	2501	Population	5301	Domestic supply quantity	1000 tonnes	0.00	0.00	0.00	0.00	0.00
2	4	Algeria	2901	Grand Total	664	Food supply (kcal/capita/day)	kcal/capita/day	3377.00	3379.00	3372.00	3341.00	3322.00
3	4	Algeria	2901	Grand Total	674	Protein supply quantity (g/capita/day)	g/capita/day	94.90	94.35	94.72	92.82	91.83
4	4	Algeria	2901	Grand Total	684	Fat supply quantity (g/capita/day)	g/capita/day	80.06	79.36	77.40	80.19	77.28

In [2]:
```
data.shape
```

Out[2]:
```
(60943, 12)
```

In [3]:
```
#random sampling of the dataset
data.sample(10)
```

Out[3]:

	Area Code	Area	Item Code	Item	Element Code	Element	Unit	Y2014	Y2015	Y2016	Y2017	Y2018
55063	222	Tunisia	2848	Milk - Excluding Butter	5072	Stock Variation	1000 tonnes	0.00	-1.00	-1.00	0.00	2.00
30883	129	Madagascar	2532	Cassava and products	5511	Production	1000 tonnes	2930.00	2677.00	2629.00	2523.00	2500.00
40825	147	Namibia	2576	Palmkernel Oil	5170	Residuals	1000 tonnes	0.00	0.00	0.00	0.00	0.00
45391	184	Rwanda	2630	Coffee and products	674	Protein supply quantity (g/capita/day)	g/capita/day	0.01	0.04	0.03	0.02	0.01
50196	202	South Africa	2551	Nuts and products	5511	Production	1000 tonnes	20.00	21.00	21.00	22.00	23.00
35133	136	Mauritania	2543	Sweeteners, Other	5170	Residuals	1000 tonnes	0.00	0.00	0.00	0.00	0.00
57653	215	United Republic of Tanzania	2641	Pimento	684	Fat supply quantity (g/capita/day)	g/capita/day	0.04	0.04	0.03	0.03	0.03
34589	133	Mali	2943	Meat	5511	Production	1000 tonnes	379.00	388.00	433.00	441.00	461.00
10682	37	Central African Republic	2737	Fats, Animals, Raw	5611	Import Quantity	1000 tonnes	0.00	0.00	0.00	0.00	0.00
57447	215	United Republic of Tanzania	2611	Oranges, Mandarines	5911	Export Quantity	1000 tonnes	14.00	123.00	37.00	22.00	24.00

In [5]:
```
#getting the descriptive statistics foe each numerical column
data.describe()
```

Out[5]:

	Area Code	Item Code	Element Code	Y2014	Y2015	Y2016	Y2017	Y2018
count	60943.000000	60943.000000	60943.000000	59354.000000	59395.000000	59408.000000	59437.000000	59507.000000
mean	134.265576	2687.176706	3814.856456	134.196282	135.235966	136.555222	140.917765	143.758381
std	72.605709	146.055739	2212.007033	1567.663696	1603.403984	1640.007194	1671.862359	1710.782658
min	4.000000	2501.000000	511.000000	-1796.000000	-3161.000000	-3225.000000	-1582.000000	-3396.000000
25%	74.000000	2562.000000	684.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	136.000000	2630.000000	5142.000000	0.090000	0.080000	0.080000	0.100000	0.070000
75%	195.000000	2775.000000	5511.000000	8.340000	8.460000	8.430000	9.000000	9.000000
max	276.000000	2961.000000	5911.000000	176405.000000	181137.000000	185960.000000	190873.000000	195875.000000

In [6]:
```
#getting the unique values in the Element column
data['Element'].unique()
```

Out[6]:
```
array(['Total Population - Both sexes', 'Domestic supply quantity',
       'Food supply (kcal/capita/day)',
       'Protein supply quantity (g/capita/day)',
       'Fat supply quantity (g/capita/day)', 'Production',
       'Import Quantity', 'Stock Variation', 'Export Quantity', 'Feed',
       'Seed', 'Losses', 'Processing', 'Other uses (non-food)',
       'Residuals', 'Food', 'Food supply quantity (kg/capita/yr)',
       'Tourist consumption'], dtype=object)
```

In [8]:
```
#getting the unique values in the Item column
data['Item'].unique()
```

Out[8]:
```
array(['Population', 'Grand Total', 'Vegetal Products', 'Animal Products',
       'Cereals - Excluding Beer', 'Wheat and products',
       'Rice and products', 'Barley and products', 'Maize and products',
       'Rye and products', 'Oats', 'Millet and products',
       'Sorghum and products', 'Cereals, Other', 'Starchy Roots',
       'Cassava and products', 'Potatoes and products', 'Sweet potatoes',
       'Yams', 'Roots, Other', 'Sugar Crops', 'Sugar cane', 'Sugar beet',
       'Sugar & Sweeteners', 'Sugar (Raw Equivalent)',
       'Sweeteners, Other', 'Honey', 'Pulses', 'Beans', 'Peas',
       'Pulses, Other and products', 'Treenuts', 'Nuts and products',
       'Oilcrops', 'Soyabeans', 'Groundnuts (Shelled Eq)',
       'Sunflower seed', 'Rape and Mustardseed', 'Cottonseed',
       'Coconuts - Incl Copra', 'Sesame seed',
       'Olives (including preserved)', 'Oilcrops, Other',
       'Vegetable Oils', 'Soyabean Oil', 'Groundnut Oil',
       'Sunflowerseed Oil', 'Rape and Mustard Oil', 'Cottonseed Oil',
       'Palmkernel Oil', 'Palm Oil', 'Coconut Oil', 'Sesameseed Oil',
       'Olive Oil', 'Maize Germ Oil', 'Oilcrops Oil, Other', 'Vegetables',
       'Tomatoes and products', 'Onions', 'Vegetables, Other',
       'Fruits - Excluding Wine', 'Oranges, Mandarines',
       'Lemons, Limes and products', 'Grapefruit and products',
       'Citrus, Other', 'Bananas', 'Plantains', 'Apples and products',
       'Pineapples and products', 'Dates',
       'Grapes and products (excl wine)', 'Fruits, Other', 'Stimulants',
       'Coffee and products', 'Cocoa Beans and products',
       'Tea (including mate)', 'Spices', 'Pepper', 'Pimento', 'Cloves',
       'Spices, Other', 'Alcoholic Beverages', 'Wine', 'Beer',
       'Beverages, Fermented', 'Beverages, Alcoholic',
       'Alcohol, Non-Food', 'Meat', 'Bovine Meat', 'Mutton & Goat Meat',
       'Pigmeat', 'Poultry Meat', 'Meat, Other', 'Offals',
       'Offals, Edible', 'Animal fats', 'Butter, Ghee', 'Cream',
       'Fats, Animals, Raw', 'Fish, Body Oil', 'Fish, Liver Oil', 'Eggs',
       'Milk - Excluding Butter', 'Fish, Seafood', 'Freshwater Fish',
       'Demersal Fish', 'Pelagic Fish', 'Marine Fish, Other',
       'Crustaceans', 'Cephalopods', 'Molluscs, Other',
       'Aquatic Products, Other', 'Aquatic Animals, Others',
       'Aquatic Plants', 'Miscellaneous', 'Infant food',
       'Sugar non-centrifugal', 'Palm kernels', 'Ricebran Oil'],
      dtype=object)
```

In [ ]:
```
Answering The Quiz Questions
```

In [9]:
```
#Perform a groupby operation on 'Element'.  What is the total number of the sum of Processing in 2017
Answer = data.groupby('Element').sum().loc['Processing']['Y2017']
print('The sum of processing in 2017 is', (Answer))
```

```
The sum of processing in 2017 is 292836.0

C:\Users\Chinenye Claire\AppData\Local\Temp\ipykernel_6996\3503391954.py:2: FutureWarning: The defau
lt value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only wi
ll default to False. Either specify numeric_only or select only columns which should be valid for th
e function.
  Answer = data.groupby('Element').sum().loc['Processing']['Y2017']
```

In [11]:
```
#What is the total number and percentage of missing data in 2014 to 3 decimal places?
missing_values = data['Y2014'].isnull().sum()
total_values= len(data)
missing_percentage= (missing_values/total_values)*100
print('total number of missing data for 2014:', (missing_values))
print('percentage of missing data in 2014:', round((missing_percentage), 3), '%')
```

```
total number of missing data for 2014: 1589
percentage of missing data in 2014: 2.607 %
```

In [12]:
```
#What is the total sum of Wine produced in 2015 and 2018 respectively?
Production_A = data.groupby('Item')['Y2015'].sum().loc['Wine']
Production_B = data.groupby('Item')['Y2018'].sum().loc['Wine']
print('The total sum of wine produced in 2015 and 2018 respectively is', (Production_A, Production_B))
```

```
The total sum of wine produced in 2015 and 2018 respectively is (4251.81, 4039.32)
```

In [13]:
```
#Which year had the least correlation with 'Element Code'?
correlations = data[['Element Code', 'Y2014', 'Y2015', 'Y2016', 'Y2017', 'Y2018']].corr()
least_correlated_year = correlations['Element Code'].abs().idxmin()
print('The year with the least correlation with "Element Code" is', (least_correlated_year))
```

```
The year with the least correlation with "Element Code" is Y2016
```

In [14]:
```
#Select columns 'Y2017' and 'Area', Perform a groupby operation on 'Area'.  Which of these Areas had t
selected_columns = data[['Y2017', 'Area']]
grouped_data = selected_columns.groupby('Area')['Y2017'].sum().reset_index()
sorted_data = grouped_data.sort_values('Y2017')
seventh_lowest_area = sorted_data.iloc[6]['Area']
print('The area with the seventh lowest sum in 2017 is', (seventh_lowest_area))
```

```
The area with the seventh lowest sum in 2017 is Guinea-Bissau
```

In [21]:
```
#Perform a groupby operation on 'Element'.  What year has the highest sum of Stock Variation?
stock_variation_sum = data.groupby('Element')[['Y2014', 'Y2015', 'Y2016', 'Y2017', 'Y2018']].sum()
sorted_data = stock_variation_sum.loc['Stock Variation']
print(sorted_data)
```

```
Y2014    58749.83
Y2015    34910.99
Y2016    33140.12
Y2017    54316.91
Y2018    20577.91
Name: Stock Variation, dtype: float64
```

In [22]:
```
#What is the mean and standard deviation across the whole dataset for the year 2017 to 2 decimal place
details = data.describe()
Mean = details.loc['mean']
print(Mean)
```

```
Area Code        134.265576
Item Code       2687.176706
Element Code    3814.856456
Y2014            134.196282
Y2015            135.235966
Y2016            136.555222
Y2017            140.917765
Y2018            143.758381
Name: mean, dtype: float64
```

In [23]:
```
Standard_Deviation = details.loc['std']
print(Standard_Deviation)
```

```
Area Code         72.605709
Item Code        146.055739
Element Code    2212.007033
Y2014           1567.663696
Y2015           1603.403984
Y2016           1640.007194
Y2017           1671.862359
Y2018           1710.782658
Name: std, dtype: float64
```

In [24]:
```
#Select columns 'Y2017' and 'Area', Perform a groupby operation on 'Area'.  Which of these Areas had t
selected_columns = data[['Y2017', 'Area']]
grouped_data = selected_columns.groupby('Area')['Y2017'].sum()
area_with_the_highest_sum = grouped_data.idxmax()
print('The area with the highest sum is', (area_with_the_highest_sum))
```

```
The area with the highest sum is Nigeria
```

In [25]:
```
#What is the total number of unique countries in the dataset?
unique_countries = data['Area'].nunique()
print('There are', (unique_countries), 'unique countries')
```

```
There are 49 unique countries
```

In [26]:
```
#What is the total Protein supply quantity in Madagascar in 2015?
madagascar_2015 = data[(data['Area'] == 'Madagascar') & (data['Element'] == 'Protein supply quantity
total_protein_supply = madagascar_2015.sum()
print('The total protein supply for Madagascar in 2015 is', round((total_protein_supply), 2))
```

```
The total protein supply for Madagascar in 2015 is 173.05
```