Datasets

For this project, three datasets were used:

1. **Nutrients Intake DataSet**

Extracted from - U.S department of Agriculture, Link - https://www.ers.usda.gov/data-products/food-consumption-and-nutrient-intakes.aspx This dataset was downloaded as an excel, we converted into a CSV file by an online converter(https://cloudconvert.com/xlsx-to-csv)

This dataset is a combination of sub-tables.

Below is the python code for reading the multiheader CSV file and processing it.

In [2]:
```
import csv

with open('nutrient_table1.csv') as csvfile:

    myCSVReader = csv.reader(csvfile, delimiter=",", quotechar='"')

    first_row = next(myCSVReader)   # reading the first row of the CSV file - "Nutrient intake by food..."
    second_row = next(myCSVReader)  # reading the second row - "Away from home"
    third_row = next(myCSVReader)   # reading the third row - "Nutrient group,Total,At home,..."
    fourth_row = next(myCSVReader)  # reading the fourth row - "2015-16, 2017-18"

    # Getting the index for the 2017-2018 as we want to process the latest data only
    year_index = fourth_row.index('2017-18')

    # Removing the headers of the 2015-16 data
    for item in third_row:
        third_row.remove(item)
        if (item == "Other"):
            break

    # Changing the "Total" headers into unique header - because there are two of them present

    total_index1 = third_row.index("Total") # reading the 1st total
    # replacing the 'Total' with a meaningful name by reading the values before and after its index
    third_row =  third_row[:total_index1] + ['Total - Overall'] + third_row[total_index1+1:]

    total_index2 = third_row.index("Total") # reading the 2nd total
    # replacing the 'Total' with a meaningful name by reading the values before and after its index
    place =  third_row[:total_index2] + ['Total - Away from home'] + third_row[total_index2+1:]

    # Defining variables to be used while reading the entire CSV
    outrows = []
    i = 0
    nutrient = ''
    age_range = ''

    # Starting to read the CSV file
    for row in myCSVReader:

        # Checking if we have reached to the end of CSV where Notes and Guidelines are present or no data
        if ('Notes:' in row[0] or not any(row)):
            break # breaking the loop as we have reached the end of the file

        else:
            nutrient = row[0] # reading the first nutrient i.e. Energy

            # Starting to read rest of the data present in CSV
            for row in myCSVReader:

                # Checking for the nutrient name that appears in Orange rows
                # Our method for reading those is that we check if the 1st item in the row is not empty & the 2nd item
is definity empty
                if (row[0] != '' and row[1] == ''):
                    nutrient = row[0]
                    # We are cleaning the nutrient data here a big my removing the '(mg)'
                    if('(' in nutrient): # Checking if the nutrient contains a parentheesis
                        i = nutrient.index('(')
                        # replacing the nutrient name with the value present before the '('
                        nutrient = nutrient[:i].strip()

                else:
                    # Processing rest of the file
                    for index,item in enumerate(row):
                        # We only want the age group data so omitting the rest of it
                        if (row[0] != '' and 'Household income' not in row[0]):
                            # Cleaning the data a bit to remove the extra no.s present after the age groups
                            age_range_copy = row[0].strip()
                            if('Total' in row[0] or 'Adults' in row[0] or 'Seniors' in row[0]):
                                age_range_copy = age_range_copy[:-1]
                            age_range = age_range_copy

                        # Checking place to start insert values because we only need the data of 2017-18
                        if (index > year_index-1):
                            outrow = { "nutrient": nutrient,
                                       "age_range": age_range,
                                       "place": place[index-1], # inorder to get the correct index of place
                                       "daily_intake": item }
                            outrows.append(outrow)


out_headers = ['nutrient','age_range','place','daily_intake']

# Writing our data into a new CSV file
with open('nutrient_intake_processed.csv', 'w') as csvfile:
    myCsvWriter = csv.DictWriter(csvfile,
                                 fieldnames=out_headers)

    myCsvWriter.writeheader()

    for row_dict in outrows:
        myCsvWriter.writerow(row_dict)

print("Successfully written Nutrient_intake file")
```

```
Successfully written Nutrient_intake file
```

The Nutrient Intake file after processing

In [3]:
```
%%sql

SELECT *
FROM 'nutrient_intake_processed.csv';
```

Out[3]:

	column0	column1	column2	column3
0	nutrient	age_range	place	daily_intake
1	Energy (calories)	Total population	Total - Overall	2093.14
2	Energy (calories)	Total population	At home	1402.55
3	Energy (calories)	Total population	Total - Away from home	690.60
4	Energy (calories)	Total population	Restaurant	173.28
...	...	...	...	...
220	Sodium	Seniors age 65 and above	Total - Away from home	895.83
221	Sodium	Seniors age 65 and above	Restaurant	399.11
222	Sodium	Seniors age 65 and above	Fast food	286.35
223	Sodium	Seniors age 65 and above	School	NA
224	Sodium	Seniors age 65 and above	Other	210.37

225 rows × 4 columns

1. **Ingredient Value Dataset**

Extracted from - U.S department of Agriculture, Link - https://www.ars.usda.gov/northeast-area/beltsville-md-bhnrc/beltsville-human-nut rition-research-center/food-surveys-research-group/docs/fndds-download-databa ses/
This dataset was downloaded as an excel, we converted into a CSV file by an online converter(https://cloudconvert.com/xlsx-to-csv)

We plan to streamline the processing of the CSV file by reducing its size since not all columns are necessary for our analysis. The file contains a substantial number of rows, and unnecessary columns could potentially lead to longer loading times when creating SQL tables.

Below is the python code for reading the CSV file and processing it

In [4]:
```
import csv

with open('ingredient_values.csv') as csvfile:

    myCSVReader = csv.reader(csvfile, delimiter=",", quotechar='"')

    first_row = next(myCSVReader)  # reading the first row of the csv - 'Ingredient Nutrient...'
    second_row = next(myCSVReader) # reading the headers present in the 2nd row

    columns = []

    # Removing the headers after 'Nutrient value source' as we don't need it for our analysis
    for item in second_row:
        if(item == 'Nutrient value source'):
            item_index = second_row.index('Nutrient value source')
            break
        columns.append(item)

    outrows = []

    # Reading rest of the data present in the file and making a dictionary
    for row in myCSVReader:
        row = row[:item_index] # removing the data from the unwanted columns
        outrow = {
            "Ingredient code": row[0],
            "Ingredient description": row[1],
            "Nutrient code": row[2],
            "Nutrient description": row[3],
            "Nutrient value": row[4]
        }
        outrows.append(outrow)

# Writing our data into a new CSV file
with open('ingredient_values_processed.csv', 'w') as csvfile:
    myCsvWriter = csv.DictWriter(csvfile,
                                 fieldnames=columns)

    myCsvWriter.writeheader()

    for row in outrows:
        myCsvWriter.writerow(row)

print("Successfully written Ingredient_values file")
```

```
Successfully written Ingredient_values file
```

The Ingredient Value file after processing

In [5]:
```
%%sql

SELECT *
FROM 'ingredient_values_processed.csv';
```

Out[5]:

Ingredient code	Ingredient description	Nutrient code	Nutrient description	Nutrient value
0	1001	Butter, stick, salted	203	Protein	0.85
1	1001	Butter, stick, salted	204	Total Fat	82.20
2	1001	Butter, stick, salted	205	Carbohydrate	0.06
3	1001	Butter, stick, salted	208	Energy	743.00
4	1001	Butter, stick, salted	221	Alcohol	0.00
...	...	...	...	...	...
122325	999431	Folic acid as ingredient	629	20:5 n-3	0.00
122326	999431	Folic acid as ingredient	630	22:1	0.00
122327	999431	Folic acid as ingredient	631	22:5 n-3	0.00
122328	999431	Folic acid as ingredient	645	Fatty acids, total monounsaturated	0.00
122329	999431	Folic acid as ingredient	646	Fatty acids, total polyunsaturated	0.00

122330 rows × 5 columns

1. **Obesity Age Dataset**

Extracted from - 2017-2018 Crude obesity percentage statistics from National Institutes of Health(NIH) Link:https://www.niddk.nih.gov/health-information/health-statistics/overweight-obesity This dataset was downloaded as a PDF. We converted into a CSV using an online converter(https://www.zamzar.com/convert/pdf-to-csv).

This dataset was downloaded from a different data source and it was initially present as a data pdf for a chart displayed on their website. We are going to process it too in order to clean the data (obesity percentage) and read only the required values instead of all the sentances present in it.

Below is the python code for reading the CSV file and processing it.

In [6]:
```
import csv

with open('obesity_ages.csv') as csvfile:

    myCSVReader = csv.reader(csvfile, delimiter=",", quotechar='"')

    first_row = next(myCSVReader)  # reading the 1st row - 'Data brief 360...'
    second_row = next(myCSVReader) # reading the 2nd row -  which is blank
    third_row = next(myCSVReader)  # reading the 3rd row - 'Data table for...'
    fourth_row = next(myCSVReader) # reading the 4th row - 'Age group'
    fifth_row = next(myCSVReader)  # reading the 5th row - 'Sex, 20 and over, ...'
    sixth_row = next(myCSVReader)  # reading the 6th row - 'percent (standard error)'

    # Define the age_group variable
    age_group = fifth_row[1:]

    # Change "20 and over" into "20 and under" (data cleaning)
    if '20 and over' in age_group:
        age_group[age_group.index('20 and over')] = '20 and under'

    outrows = []

    # starting to read rest of the data
    for row in myCSVReader:
        if ('Men' in row[0]): # we are not doing gender specific analysis so don't need it
            break

        for index,item in enumerate(row[1:]):
            # Cleaning parenthesis present in obesity %
            if('(' in item):
                i = item.index('(') # getting the index for the parenthesis
                item = item[:i] # getting the values before the parenthesis index
                outrow = {
                    "age_group":age_group[index],
                    "obesity_percentage": item
                }
                outrows.append(outrow)

columns = ['age_group','obesity_percentage']

# Writing our data into a new CSV file
with open('obesity_age_processed.csv', 'w') as csvfile:
    myCsvWriter = csv.DictWriter(csvfile,
                                 fieldnames=columns)

    myCsvWriter.writeheader()

    for row in outrows:
        myCsvWriter.writerow(row)

print("Successfully written Obesity_age file")
```

```
Successfully written Obesity_age file
```

The Obesity Age file after processing

In [7]:
```
%%sql

SELECT *
FROM 'obesity_age_processed.csv';
```

Out[7]:

column0	column1
0 age_group	obesity_percentage
1 20 and under	42.4
2 20–39	40.0
3 40–59	44.8
4 60 and over	42.8