The cost of milk to make cheese

The Coronavirus outbreak has caused major economic disruption. The pandemic's impact on the tourism, travel, and hospitality industries has featured frequently in 2020 news reports, but the economic impact has not been isolated to these industries alone.

The UK dairy industry is one such affected industry and has had to work together to address current market challenges and avoid milk waste. The industry can adapt to the temporary decrease in demand by identifying opportunities for processing the milk into storable products such as butter, cheese, and skimmed milk powder.

The UK Department of Environment, Food, and Rural Affairs has contracted your company to perform a detailed analysis of the UK dairy industry to help them understand the industry's value.

Your team has divided the analysis project into several tasks. One of the tasks is to determine the value of the cheddar cheese industry. The revenue from cheese is much more than, for example, whole milk, but a kilogram of cheese requires almost ten liters of raw milk to make. Your first task is to determine the annual cost of milk needed to produce cheddar cheese over the past five years.

You have been provided with three datasets. ***Please note that 1000 kg = 1 metric tonne.***

datasets/milk_prices.csv - The monthly prices, volume, protein percentage and butterfat percentage of milk

Source: Defra Data Services Platform

- **Time:** The month and year when the value was recorded.
- **Measure Type:** The type of measure recorded.
- **Price:** The price of milk (in Pounds per liter).
- **Volume:** The amount of milk produced (in million liters).
- **Protein:** The protein content of the milk (percentage).
- **Butterfat:** The butterfat content of the milk (percentage).
- **Unit of Measure:** The units specific to the measure type.

datasets/Milk_products_production.csv - The monthly production figures of milk, cream and cheddar cheese

Source: GOV.UK

- **Unnamed: 0:** The year and month when the value was recorded.
- **Liquid Milk Production:** The total amount of milk produced (in million liters).
- **Cream Production:** The total amount of cream produced (in million liters).
- **Cheddar Cheese Production:**The total amount of cheddar cheese produced (in thousand tonnes).

datasets/conversion_factors.xls - Liters of milk used to make one kilogram of product

Source: GOV.UK

- **Product:** The type of product.
- **Conversion factor (litres/kg):** Liters of milk used to make one kilogram of product (in liters/kg).

In [26]:

```
# imports, pulling in data, and examining dataframes
import pandas as pd

milk_prices = pd.read_csv('datasets/milk_prices.csv')
milk_products = pd.read_csv('datasets/Milk_products_production.csv')
conversion = pd.read_excel('datasets/conversion_factors.xls')

display(milk_prices.head())
print(milk_prices.info())

display(milk_products.head())
print(milk_products.info())
```

```
display(conversion.head())
print(conversion.info())
```

	Time	Measure type	Price	Volume	Protein	Butterfat	Unit of Measure
0	1986-10	Price	0.1697	NaN	NaN	NaN	Pounds per litre
1	2000-04	Volume	NaN	1177.0	NaN	NaN	Million litres
2	1994-12	Protein	NaN	NaN	3.21	NaN	Percentage
3	1997-10	Price	0.2082	NaN	NaN	NaN	Pounds per litre
4	2002-10	Protein	NaN	NaN	3.41	NaN	Percentage

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1557 entries, 0 to 1556
Data columns (total 7 columns):
Time                1557 non-null object
Measure type        1557 non-null object
Price               611 non-null float64
Volume              313 non-null float64
Protein             313 non-null float64
Butterfat           320 non-null float64
Unit of Measure     1557 non-null object
dtypes: float64(4), object(3)
memory usage: 85.2+ KB
None
```

	Unnamed: 0	Liquid Milk Production	Cream Production	Cheddar Cheese Production
0	NaN	Million Litres	Million Litres	Thousand tonnes
1	Jan-15	571,2	25,7	24,2
2	Feb-15	524,1	22,5	21,3
3	Mar-15	583,0	25,7	26,2
4	Apr-15	542,0	26,4	29,3

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 73 entries, 0 to 72
Data columns (total 4 columns):
Unnamed: 0                  72 non-null object
Liquid Milk Production      73 non-null object
Cream Production            73 non-null object
Cheddar Cheese Production   73 non-null object
dtypes: object(4)
memory usage: 2.4+ KB
None
```

	Product	Conversion factor (litres/kg)
0	Butter (from Cream)	2.04
1	Cheddar	9.5
2	Other Long Life Territorials	9.2
3	Short life Territorials	8.1
4	Blue vein	9.1

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15 entries, 0 to 14
Data columns (total 2 columns):
Product                         15 non-null object
Conversion factor (litres/kg)   15 non-null object
dtypes: object(2)
memory usage: 320.0+ bytes
None
```

In [27]:

```
# cleaning up milk_prices dataframe
milk_prices.rename(columns={'Time': 'Date', 'Price': 'Milk Price'}, inplace=True)

cols_to_drop = ['Volume', 'Protein', 'Butterfat', 'Measure type', 'Unit of Measure']
milk_prices.drop(cols_to_drop, axis=1, inplace=True)

milk_prices.dropna(axis=0, how='any', inplace=True)

# converting columns from strings to integers
milk_prices['Milk Price'] = pd.to_numeric(milk_prices['Milk Price'])

display(milk_prices.head())
display(milk_prices.info())

# creating subset of milk_prices dataframe, 2015 - 2020
```

```
milk_prices['Date'] = pd.to_datetime(milk_prices['Date'])

start_date = '2015-01-01'
end_date = '2020-12-01'
five_years = (milk_prices['Date'] >= start_date) & (milk_prices['Date'] <= end_date)
milk_prices = milk_prices.loc[five_years]

milk_prices.set_index('Date', inplace=True)

display(milk_prices.head())
display(milk_prices.info())
```

	Date	Milk Price
0	1986-10	0.1697
3	1997-10	0.2082
16	2009-11	0.2487
24	2008-10	0.2736
26	2019-10	0.2940

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 611 entries, 0 to 1554
Data columns (total 2 columns):
Date          611 non-null object
Milk Price    611 non-null float64
dtypes: float64(1), object(1)
memory usage: 14.3+ KB
None
```

	Milk Price
Date	
2019-10-01	0.2940
2020-09-01	0.2901
2020-08-01	0.2805
2020-07-01	0.2773
2020-06-01	0.2715

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 71 entries, 2019-10-01 to 2019-11-01
Data columns (total 1 columns):
Milk Price    71 non-null float64
dtypes: float64(1)
memory usage: 1.1 KB
None
```

In [28]:

```
# cleaning up milk_products dataframe
milk_products.rename(columns={'Unnamed: 0': 'Date'}, inplace=True)
milk_products.dropna(axis=0, how='any', inplace=True)

cols_to_drop = ['Liquid Milk Production', 'Cream Production']
milk_products.drop(cols_to_drop, axis=1, inplace=True)

# converting columns from strings to integers
milk_products = milk_products.apply(lambda x: x.str.replace(',','.'))
milk_products['Cheddar Cheese Production'] = pd.to_numeric(milk_products['Cheddar Cheese Production'])

display(milk_products.head())
display(milk_products.info())

# creating subset of milk_prices dataframe, 2015 - 2020
milk_products['Date'] = pd.to_datetime(milk_products['Date'], format='%b-%y')

start_date = '2015-01-01'
end_date = '2020-12-01'
five_years = (milk_products['Date'] >= start_date) & (milk_products['Date'] <= end_date)
milk_products = milk_products.loc[five_years]

milk_products.set_index('Date', inplace=True)

display(milk_products.head())
display(milk_products.info())
```

	Date	Cheddar Cheese Production

	Date	Cheddar Cheese Production
1	Jan-15	24.2
2	Feb-15	21.3
3	Mar-15	26.2
4	Apr-15	29.3
5	May-15	31.0

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 72 entries, 1 to 72
Data columns (total 2 columns):
Date                          72 non-null object
Cheddar Cheese Production     72 non-null float64
dtypes: float64(1), object(1)
memory usage: 1.7+ KB
None
```

	Cheddar Cheese Production
Date	
2015-01-01	24.2
2015-02-01	21.3
2015-03-01	26.2
2015-04-01	29.3
2015-05-01	31.0

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 72 entries, 2015-01-01 to 2020-12-01
Data columns (total 1 columns):
Cheddar Cheese Production     72 non-null float64
dtypes: float64(1)
memory usage: 1.1 KB
None
```

In [29]:

```
# examining the conversion dataframe to make sure the correct product is selected for conversion
product_counts = conversion['Product'].value_counts()
print(product_counts)
```

```
Yoghurt                          1
Milk Powder (from Cream)         1
Cheddar                          1
Butter (from Cream)              1
Soft cheese                      1
Milk Powder (from Skim milk)     1
Milk Powder (from Whole milk)    1
Cream (from cream)               1
Cottage cheese /Fromage frais    1
Condensed Milk                   1
Short life Territorials          1
Mozarella                        1
Blue vein                        1
Non-specified cheese             1
Other Long Life Territorials     1
Name: Product, dtype: int64
```

In [30]:

```
# amount of milk needed to produce cheddar cheese
total_milk = milk_products['Cheddar Cheese Production'] / 0.000001 * 9.5

display(total_milk.head())

milk_products.insert(1, 'Total Milk Used', total_milk)
display(milk_products.head())
```

```
Date
2015-01-01    229900000.0
2015-02-01    202350000.0
2015-03-01    248900000.0
2015-04-01    278350000.0
2015-05-01    294500000.0
Name: Cheddar Cheese Production, dtype: float64
```

	Cheddar Cheese Production	Total Milk Used
Date		
2015-01-01	24.2	229900000.0
2015-02-01	21.3	202350000.0

Date	Cheddar Cheese Production	Total Milk Used
2015-03-01	26.2	248900000.0
2015-04-01	29.3	278350000.0
2015-05-01	31.0	294500000.0

In [31]:

```
# calculating cost of milk used for cheddar cheese production
production_price = pd.DataFrame(milk_prices['Milk Price']*milk_products['Total Milk Used'], columns=['Production
display(production_price.head())
display(production_price.info())

# totaling up price per year and creating df, annual_cost
annual_totals = production_price['Production Price'].groupby(production_price.index.year).sum()
annual_cost = pd.DataFrame({'Cost' : annual_totals})
display(annual_cost.head())
```

Date	Production Price
2015-01-01	60831540.0
2015-02-01	52732410.0
2015-03-01	62349450.0
2015-04-01	68724615.0
2015-05-01	71092300.0

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 72 entries, 2015-01-01 to 2020-12-01
Data columns (total 1 columns):
Production Price    71 non-null float64
dtypes: float64(1)
memory usage: 3.6 KB
None
```

Date	Cost
2015	702651920.0
2016	666635425.0
2017	887977635.0
2018	922553075.0
2019	896657880.0

In [32]:

```
%%nose

import pandas as pd
import re

test_solution = pd.DataFrame({'Date': ['2015-12-31', '2016-12-31', '2017-12-31', '2018-12-31',
       '2019-12-31', '2020-12-31'],\
                              'Cost': [7.02651920e+08, 6.66635425e+08, 8.87977635e+08, 9.22553075e+08,
       8.96657880e+08, 8.36934800e+08]}).set_index('Date')

convert_index = lambda x: [re.match('(\d{4})', date).group(0) for date in x.index.values.astype(str)]

def test_project():

    # Check whether the answer has been saved and is a DataFrame
    assert 'annual_cost' in globals() and type(annual_cost) == pd.core.frame.DataFrame, \
    "Have you assigned your answer to a DataFrame named `annual_cost`?"

    # Check whether they have the right column in their DataFrame
    assert annual_cost.columns.isin(['Cost']).any(), \
    "Your DataFrame is missing the required column!"

    # Check whether they have included the correct index
    assert annual_cost.index.name == 'Date', \
    "Your DataFrame is missing the required index!"

    # Check whether the values (converted to an integer) contain in the only column are correct
```

```
    # and check whether the index is identical
    assert annual_cost.Cost.astype('int64').values.all() == test_solution.Cost.astype('int64').values.all(), \
    "Your submitted DataFrame does not contain the correct values!"

    assert convert_index(test_solution) == convert_index(annual_cost), \
    "Your submitted DataFrame does not contain the correct values!"
```

Out[32]: 1/1 tests passed