Business Case 5

PoS Appliance's Retail

Group members:

- Lorenzo Pigozzi --- m20200745
- Nguyen Huy Phuc --- m20200566
- Ema Mandura --- m20200647
- Xavier Goncalves --- m20201090

Expected outcomes

- **How can I understand each Point-of-Sale characteristics ?**\ Quarterly analysis of
  - Top products sold
  - Market Share (Family, Category), preferences
  - Product co-ocorrences
- **Point-of-Sales Clustering**
  - Value
  - Product preference
- **Forecasting**
  - Units Product forecast (6 weeks ahead)
  - Units Product forecast by Poit-of-Sale (6 weeks ahead)

Table of Contents

1. Importing data and libraries

In [2]:
```
import pandas as pd
import datetime
import time
```

In [1]:
```
dtype_dict = {
    'ProductFamily_ID':'category',
    'ProductCategory_ID':'category',
    'ProductBrand_ID':'category',
```

```
    'ProductName_ID':'category',
    'ProductPackSKU_ID':'category',
    'Point-of-Sale_ID':'category',
    'Measures':'category',
    'Value':'float32'
}

path = r"C:\Users\lorep\Documents\Master\Business
Cases\BC5\dataset\NOVAIMS_MAA_2020e21_BusinessCasesDataScience_MindOverData_R
```

In [3]:
```
df = pd.read_csv(path, dtype = dtype_dict)
```

2. Exploratory Data Analysis

In [4]:
```
df.head()
```

Out[4]:

	ProductFamily_ID	ProductCategory_ID	ProductBrand_ID	ProductName_ID	ProductPackSKU_ID	Poin c Sale_
0	Family_16	Category_11	ProductBrand_306	ProductName_649	ProductSKU_1970	POS
1	Family_16	Category_11	ProductBrand_306	ProductName_649	ProductSKU_1970	POS
2	Family_16	Category_11	ProductBrand_306	ProductName_649	ProductSKU_1970	POS
3	Family_16	Category_11	ProductBrand_306	ProductName_649	ProductSKU_1970	POS
4	Family_16	Category_11	ProductBrand_306	ProductName_649	ProductSKU_1970	POS

In [5]:
```
df.shape
```

Out[5]:
```
(182342304, 9)
```

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 182342304 entries, 0 to 182342303
Data columns (total 9 columns):
 #   Column              Dtype
---  ------              -----
 0   ProductFamily_ID    category
 1   ProductCategory_ID  category
```

```
 2   ProductBrand_ID     category
 3   ProductName_ID      category
 4   ProductPackSKU_ID   category
 5   Point-of-Sale_ID    category
 6   Date                object
 7   Measures            category
 8   Value               float32
dtypes: category(7), float32(1), object(1)
memory usage: 4.1+ GB
```

In [8]:
```
df.isna().sum()
```

Out[8]:
```
ProductFamily_ID      0
ProductCategory_ID    0
ProductBrand_ID       0
ProductName_ID        0
ProductPackSKU_ID     0
Point-of-Sale_ID      0
Date                  0
Measures              0
Value                 0
dtype: int64
```

Reducing the size of the dataset

Splitting the dataset in 2, based on "units" or "values" of the variable "Measure"

DF UNITS (QUANTITY)

In [7]:
```
df_units = df[df['Measures'] == "Sell-out units"]
df_values = df[df['Measures'] == "Sell-out values"]
```

In [8]:
```
df_units.head()
```

Out[8]:

	ProductFamily_ID	ProductCategory_ID	ProductBrand_ID	ProductName_ID	ProductPackSKU_ID	Point of Sale_
0	Family_16	Category_11	ProductBrand_306	ProductName_649	ProductSKU_1970	POS
2	Family_16	Category_11	ProductBrand_306	ProductName_649	ProductSKU_1970	POS
4	Family_16	Category_11	ProductBrand_306	ProductName_649	ProductSKU_1970	POS
6	Family_16	Category_11	ProductBrand_306	ProductName_649	ProductSKU_1970	POS
7	Family_16	Category_11	ProductBrand_306	ProductName_649	ProductSKU_1970	POS

In [10]:
```
df_units.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 91171152 entries, 0 to 182342299
Data columns (total 9 columns):
 #   Column              Dtype
---  ------              -----
 0   ProductFamily_ID    category
 1   ProductCategory_ID  category
 2   ProductBrand_ID     category
 3   ProductName_ID      category
 4   ProductPackSKU_ID   category
 5   Point-of-Sale_ID    category
 6   Date                object
 7   Measures            category
 8   Value               float32
dtypes: category(7), float32(1), object(1)
memory usage: 2.7+ GB
```

In [11]:
```
df_units = df_units[['ProductPackSKU_ID', 'Point-of-Sale_ID', 'Date',
'Value']]
```

In [12]:
```
df_units.head()
```

Out[12]:

	ProductPackSKU_ID	Point-of-Sale_ID	Date	Value
0	ProductSKU_1970	POS_1	2017-03-04	2.0
2	ProductSKU_1970	POS_1	2016-05-02	4.0
4	ProductSKU_1970	POS_1	2016-10-24	2.0
6	ProductSKU_1970	POS_1	2017-10-13	2.0
7	ProductSKU_1970	POS_1	2017-10-14	2.0

In [13]:
```
# slicing the string and keeping only the number of ID
df_units['ProductPackSKU_ID'] =
df_units['ProductPackSKU_ID'].str.replace(r"[^0-9]", "",regex=True)
df_units['Point-of-Sale_ID'] = df_units['Point-of-Sale_ID'].str.replace(r"
[^0-9]", "",regex=True)
```

In [15]:
```
df_units.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 91171152 entries, 0 to 182342299
Data columns (total 4 columns):
 #   Column              Dtype
---  ------              -----
 0   ProductPackSKU_ID   object
 1   Point-of-Sale_ID    object
 2   Date                object
 3   Value               float32
dtypes: float32(1), object(3)
memory usage: 3.1+ GB
```

In [16]:
```
# changing the datatype
```

```
df_units['ProductPackSKU_ID'] =
df_units['ProductPackSKU_ID'].astype('category')
df_units['Point-of-Sale_ID'] = df_units['Point-of-
Sale_ID'].astype('category')
df_units["Date"] = pd.to_datetime(df_units["Date"])
```

In [17]:
```
df_units.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 91171152 entries, 0 to 182342299
Data columns (total 4 columns):
 #   Column             Dtype
---  ------             -----
 0   ProductPackSKU_ID  category
 1   Point-of-Sale_ID   category
 2   Date               datetime64[ns]
 3   Value              float32
dtypes: category(2), datetime64[ns](1), float32(1)
memory usage: 2.0 GB
```

In [18]:
```
# saving the result as csv
df_units.to_csv(r"C:\Users\lorep\Documents\Master\Business
Cases\BC5\dataset\df_units.csv")
```

DF VALUES (PRICE)

In [19]:
```
df_values.head()
```

Out[19]:

	ProductFamily_ID	ProductCategory_ID	ProductBrand_ID	ProductName_ID	ProductPackSKU_ID	Po Sale
1	Family_16	Category_11	ProductBrand_306	ProductName_649	ProductSKU_1970	PO
3	Family_16	Category_11	ProductBrand_306	ProductName_649	ProductSKU_1970	PO
5	Family_16	Category_11	ProductBrand_306	ProductName_649	ProductSKU_1970	PO
9	Family_16	Category_11	ProductBrand_306	ProductName_649	ProductSKU_1970	PO
10	Family_16	Category_11	ProductBrand_306	ProductName_649	ProductSKU_1970	PO

In [20]:
```
df_values.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 91171152 entries, 1 to 182342303
Data columns (total 9 columns):
 #   Column              Dtype
```

```
---  ------                -----
 0   ProductFamily_ID      category
 1   ProductCategory_ID    category
 2   ProductBrand_ID       category
 3   ProductName_ID        category
 4   ProductPackSKU_ID     category
 5   Point-of-Sale_ID      category
 6   Date                  object
 7   Measures              category
 8   Value                 float32
dtypes: category(7), float32(1), object(1)
memory usage: 2.7+ GB
```

In [21]:

```
# slicing the string and keeping only the number of ID
df_values['ProductFamily_ID'] =
df_values['ProductFamily_ID'].str.replace(r"[^0-9]", "",regex=True)
df_values['ProductCategory_ID'] =
df_values['ProductCategory_ID'].str.replace(r"[^0-9]", "",regex=True)
df_values['ProductBrand_ID'] = df_values['ProductBrand_ID'].str.replace(r"
[^0-9]", "",regex=True)
```

```
<ipython-input-21-c97a5d9acead>:1: SettingWithCopyWarning:
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/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df_values['ProductFamily_ID'] = df_values['ProductFamily_ID'].str.replace(r"[^0-9]",
"",regex=True)
<ipython-input-21-c97a5d9acead>:2: SettingWithCopyWarning:
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/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df_values['ProductCategory_ID'] = df_values['ProductCategory_ID'].str.replace(r"[^0-
9]", "",regex=True)
<ipython-input-21-c97a5d9acead>:3: SettingWithCopyWarning:
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/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df_values['ProductBrand_ID'] = df_values['ProductBrand_ID'].str.replace(r"[^0-9]", "",
regex=True)
```

In [22]:

```
# slicing the string and keeping only the number of ID
df_values['ProductName_ID'] = df_values['ProductName_ID'].str.replace(r"
[^0-9]", "",regex=True)
df_values['ProductPackSKU_ID'] =
df_values['ProductPackSKU_ID'].str.replace(r"[^0-9]", "",regex=True)
df_values['Point-of-Sale_ID'] = df_values['Point-of-
Sale_ID'].str.replace(r"[^0-9]", "",regex=True)
```

```
<ipython-input-22-554a80768195>:1: SettingWithCopyWarning:
```

```
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/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df_values['ProductName_ID'] = df_values['ProductName_ID'].str.replace(r"[^0-9]", "",re
gex=True)
<ipython-input-22-554a80768195>:2: SettingWithCopyWarning:
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/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df_values['ProductPackSKU_ID'] = df_values['ProductPackSKU_ID'].str.replace(r"[^0-9]",
"",regex=True)
<ipython-input-22-554a80768195>:3: SettingWithCopyWarning:
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/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df_values['Point-of-Sale_ID'] = df_values['Point-of-Sale_ID'].str.replace(r"[^0-9]",
"",regex=True)
```

In [23]:

```
# changing the datatype
df_values['ProductFamily_ID'] =
df_values['ProductFamily_ID'].astype('category')
df_values['ProductCategory_ID'] =
df_values['ProductCategory_ID'].astype('category')
df_values['ProductBrand_ID'] =
df_values['ProductBrand_ID'].astype('category')
```

```
<ipython-input-23-8a7c0391443b>:1: SettingWithCopyWarning:
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/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df_values['ProductFamily_ID'] = df_values['ProductFamily_ID'].astype('category')
<ipython-input-23-8a7c0391443b>:2: SettingWithCopyWarning:
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/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df_values['ProductCategory_ID'] = df_values['ProductCategory_ID'].astype('category')
<ipython-input-23-8a7c0391443b>:3: SettingWithCopyWarning:
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/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df_values['ProductBrand_ID'] = df_values['ProductBrand_ID'].astype('category')
```

In [24]:

```
# changing the datatype
df_values['ProductName_ID'] =
df_values['ProductName_ID'].astype('category')
```

```
df_values['ProductPackSKU_ID'] =
df_values['ProductPackSKU_ID'].astype('category')
df_values['Point-of-Sale_ID'] = df_values['Point-of-
Sale_ID'].astype('category')
```

```
<ipython-input-24-c02d70d914f4>:1: SettingWithCopyWarning:
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/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df_values['ProductName_ID'] = df_values['ProductName_ID'].astype('category')
<ipython-input-24-c02d70d914f4>:2: SettingWithCopyWarning:
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/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df_values['ProductPackSKU_ID'] = df_values['ProductPackSKU_ID'].astype('category')
<ipython-input-24-c02d70d914f4>:3: SettingWithCopyWarning:
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/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df_values['Point-of-Sale_ID'] = df_values['Point-of-Sale_ID'].astype('category')
```

In [26]:

```
df_values["Date"] = pd.to_datetime(df_values["Date"])
```

```
<ipython-input-26-01a0fe5e25f1>:1: SettingWithCopyWarning:
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/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df_values["Date"] = pd.to_datetime(df_values["Date"])
```

In [27]:

```
df_values.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 91171152 entries, 1 to 182342303
Data columns (total 9 columns):
 #   Column              Dtype
---  ------              -----
 0   ProductFamily_ID    category
 1   ProductCategory_ID  category
 2   ProductBrand_ID     category
 3   ProductName_ID      category
 4   ProductPackSKU_ID   category
 5   Point-of-Sale_ID    category
 6   Date                datetime64[ns]
 7   Measures            category
 8   Value               float32
dtypes: category(7), datetime64[ns](1), float32(1)
memory usage: 2.7 GB
```

In [28]:

```
df_values.head()
```

Out[28]:

	ProductFamily_ID	ProductCategory_ID	ProductBrand_ID	ProductName_ID	ProductPackSKU_ID	Poi Sale
1	16	11	306	649	1970	
3	16	11	306	649	1970	
5	16	11	306	649	1970	
9	16	11	306	649	1970	
10	16	11	306	649	1970	

In [29]:
```
# saving the result as csv
df_units.to_csv(r"C:\Users\lorep\Documents\Master\Business Cases\BC5\dataset\df_values.csv")
```

Joining the units and Values

In [30]:
```
df_values.head()
```

Out[30]:

	ProductFamily_ID	ProductCategory_ID	ProductBrand_ID	ProductName_ID	ProductPackSKU_ID	Poi Sale
1	16	11	306	649	1970	
3	16	11	306	649	1970	
5	16	11	306	649	1970	
9	16	11	306	649	1970	
10	16	11	306	649	1970	

In [31]:
```
df_values.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 91171152 entries, 1 to 182342303
Data columns (total 9 columns):
 #   Column              Dtype
---  ------              -----
 0   ProductFamily_ID    category
```

```
 1   ProductCategory_ID  category
 2   ProductBrand_ID     category
 3   ProductName_ID      category
 4   ProductPackSKU_ID   category
 5   Point-of-Sale_ID    category
 6   Date                datetime64[ns]
 7   Measures            category
 8   Value               float32
dtypes: category(7), datetime64[ns](1), float32(1)
memory usage: 2.7 GB
```

In [32]:
```
df_units.head()
```

Out[32]:

	ProductPackSKU_ID	Point-of-Sale_ID	Date	Value
0	1970	1	2017-03-04	2.0
2	1970	1	2016-05-02	4.0
4	1970	1	2016-10-24	2.0
6	1970	1	2017-10-13	2.0
7	1970	1	2017-10-14	2.0

In [33]:
```
df_units.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 91171152 entries, 0 to 182342299
Data columns (total 4 columns):
 #   Column             Dtype
---  ------             -----
 0   ProductPackSKU_ID  category
 1   Point-of-Sale_ID   category
 2   Date               datetime64[ns]
 3   Value              float32
dtypes: category(2), datetime64[ns](1), float32(1)
memory usage: 2.0 GB
```

In [44]:
```
df_units.iloc[200:220,:]
```

Out[44]:

	ProductPackSKU_ID	Point-of-Sale_ID	Date	Value
399	6100	1	2017-01-20	2.0
400	6100	1	2017-01-21	2.0
401	6100	1	2017-01-23	2.0
402	6100	1	2017-01-31	2.0
408	6100	1	2017-07-01	2.0
409	6100	1	2017-07-07	5.0
410	6100	1	2017-07-08	2.0
411	6100	1	2017-07-12	2.0
412	6100	1	2017-07-13	4.0
418	6100	1	2019-06-12	2.0

	ProductPackSKU_ID	Point-of-Sale_ID	Date	Value
419	6100	1	2019-06-13	2.0
422	6100	1	2019-04-17	2.0
423	6100	1	2019-04-19	2.0
424	6100	1	2019-04-27	2.0
428	6100	1	2018-08-04	2.0
429	6100	1	2018-08-06	2.0
430	6100	1	2018-08-08	4.0
431	6100	1	2018-08-10	2.0
432	6100	1	2018-08-13	2.0
433	6100	1	2018-08-14	2.0

In [45]:
```
df_values[['ProductPackSKU_ID', 'Point-of-Sale_ID',
'Date']].iloc[200:220,:]
```

Out[45]:

	ProductPackSKU_ID	Point-of-Sale_ID	Date
404	6100	1	2017-01-20
405	6100	1	2017-01-21
406	6100	1	2017-01-23
407	6100	1	2017-01-31
413	6100	1	2017-07-01
414	6100	1	2017-07-07
415	6100	1	2017-07-08
416	6100	1	2017-07-12
417	6100	1	2017-07-13
420	6100	1	2019-06-12
421	6100	1	2019-06-13
425	6100	1	2019-04-17
426	6100	1	2019-04-19
427	6100	1	2019-04-27
434	6100	1	2018-08-04
435	6100	1	2018-08-06
436	6100	1	2018-08-08
437	6100	1	2018-08-10
438	6100	1	2018-08-13

	ProductPackSKU_ID	Point-of-Sale_ID	Date
439	6100	1	2018-08-14

Intuition

As the dataset provided is kind of a transactional system (aggregated by day and Point of Sale), the trasactions stored are ordered, thus both the 2 partial datasets created, df_units and df_values have the same order, and the "Value" variable of df_units for the record 'i' retrieves the quantity of the record 'i' in df_values. \ Thus, even though the best practice to do this merging of the 2 datasets would be a join operation based on "ProductPackSKU_ID", "Point-of-Sale_ID" and "Date", due to computational issues of the operation we can just store df_units["Value"] as df_values["Quantity"], based on the index resetted.\

In [61]:
```
# resetting the index
df_values = df_values.reset_index(drop = True)
df_units = df_units.reset_index(drop = True)
```

In [62]:
```
df_values.head()
```

Out[62]:

	ProductFamily_ID	ProductCategory_ID	ProductBrand_ID	ProductName_ID	ProductPackSKU_ID	Poin o Sale_I
0	16	11	306	649	1970	
1	16	11	306	649	1970	
2	16	11	306	649	1970	
3	16	11	306	649	1970	
4	16	11	306	649	1970	

In [64]:
```
df_units.head()
```

Out[64]:

	ProductPackSKU_ID	Point-of-Sale_ID	Date	Value
0	1970	1	2017-03-04	2.0
1	1970	1	2016-05-02	4.0
2	1970	1	2016-10-24	2.0
3	1970	1	2017-10-13	2.0
4	1970	1	2017-10-14	2.0

In [65]:
```
# storing the quantity of df_units in df_values
df_values["Quantity"] = df_units["Value"]
```

In [66]:
```
df_values.head()
```

Out[66]:

	ProductFamily_ID	ProductCategory_ID	ProductBrand_ID	ProductName_ID	ProductPackSKU_ID	Point-of-Sale_I
0	16	11	306	649	1970	
1	16	11	306	649	1970	
2	16	11	306	649	1970	
3	16	11	306	649	1970	
4	16	11	306	649	1970	

In [67]:
```
df_values.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 91171152 entries, 0 to 91171151
Data columns (total 10 columns):
 #   Column              Dtype
---  ------              -----
 0   ProductFamily_ID    category
 1   ProductCategory_ID  category
 2   ProductBrand_ID     category
 3   ProductName_ID      category
 4   ProductPackSKU_ID   category
 5   Point-of-Sale_ID    category
 6   Date                datetime64[ns]
 7   Measures            category
 8   Value               float32
 9   Quantity            float32
dtypes: category(7), datetime64[ns](1), float32(2)
memory usage: 2.4 GB
```

In [68]:
```
df_final = df_values.rename(columns={'Value': 'Price'})
```

In [72]:
```
del df_final['Measures']
```

In [73]:
```
df_final
```

Out[73]:

	ProductFamily_ID	ProductCategory_ID	ProductBrand_ID	ProductName_ID	ProductPackSKU_ID

	ProductFamily_ID	ProductCategory_ID	ProductBrand_ID	ProductName_ID	ProductPackSKU_ID
0	16	11	306	649	197(
1	16	11	306	649	197(
2	16	11	306	649	197(
3	16	11	306	649	197(
4	16	11	306	649	197(
...	...	...	...	...	.
91171147	4	34	279	577	181
91171148	4	34	279	577	181
91171149	4	34	279	577	181
91171150	4	34	279	577	181
91171151	4	34	279	577	181

91171152 rows × 9 columns

In [74]:
```
df_final.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 91171152 entries, 0 to 91171151
Data columns (total 9 columns):
 #   Column              Dtype
---  ------              -----
 0   ProductFamily_ID    category
 1   ProductCategory_ID  category
 2   ProductBrand_ID     category
 3   ProductName_ID      category
 4   ProductPackSKU_ID   category
 5   Point-of-Sale_ID    category
 6   Date                datetime64[ns]
 7   Price               float32
 8   Quantity            float32
dtypes: category(6), datetime64[ns](1), float32(2)
memory usage: 2.3 GB
```

In [76]:
```
# saving the result as csv
df_final.to_csv(r"C:\Users\lorep\Documents\Master\Business
Cases\BC5\dataset\df_final.csv")
```

In [ ]: