Home Work 4 – Report

Neel Haria

```
from pandas import DataFrame
import pandas as pd
import numpy as np
```

```
#1 What is total amount spending captured in this dataset?

res_purchase = pd.read_csv('res_purchase_2014.csv')
#pd.set_option('display.max_columns',  None)
res_purchase['Amount'] = res_purchase['Amount'].str.replace("$",'')
res_purchase['Amount'] = res_purchase['Amount'].str.replace(",",'')
res_purchase['Amount'] = res_purchase['Amount'].str.replace("(",'')
res_purchase['Amount'] = res_purchase['Amount'].str.replace(")",'')
res_purchase['Amount'] = res_purchase['Amount'].str.replace("zero",'')
res_purchase['Amount'] = res_purchase['Amount'].fillna(0)
res_purchase['Amount'] = res_purchase['Amount'].astype('float')
print("The total amount spent is ${}".format(res_purchase['Amount'].sum().round(2)))
#res_purchase['Amount']
```

```
The total amount spent is $19101987.62
```

```
#2 & 3
def total_vendor(Vendor):
    print("Total amount at {} is:".format(Ven), res_purchase[res_purchase['Vendor']==Vendor]["Amount"].
```

```
#2. How much was spend at WW GRAINGER?
total_vendor('WW GRAINGER')
```

```
Total amount at WW GRAINGER is: 192149.72
```

```
#3. How much was spend at WM SUPERCENTER?
total_vendor('WM SUPERCENTER')
```

```
Total amount at WM SUPERCENTER is: 0.0
```

```
#4. How much was spend at GROCERY STORES?
def total_category(category):
    print("Total amount in {} category is:".format(category),res_purchase[res_purchase['Merchant Categ
total_category("GROCERY STORES,AND SUPERMARKETS")
```

```
Total amount in GROCERY STORES,AND SUPERMARKETS category is: 285952.67
```

```
#2Data Processing with Pandas (60 points)
```

```
#1. Read 'Energy.xlsx' and 'EnergyRating.xlsx' as BalanceSheet and Ratings(dataframe).
#2. drop the column if more than 90% value in this colnmn is 0 (or missing value).
#3. replace all None or NaN with average value of each column.


BalanceSheet = pd.read_excel('Energy.xlsx')
pd.set_option('display.max_columns',  None)
BalanceSheet = BalanceSheet.drop(columns=BalanceSheet.columns[((BalanceSheet==1).mean()>0.9)])
BalanceSheet = BalanceSheet.fillna(BalanceSheet.mean())
BalanceSheet = BalanceSheet.dropna(axis = 1, how = 'all')
print(BalanceSheet)
```

	Global Company Key	Data Date	Fiscal Year	Fiscal Quarter	Fiscal Year-end Month	Industry Format	Level of Consolidation - Company Interim Descriptor	Population Source	Data Format	Ticker Symbol	CUSIP	Comp Na
0	1380	20100331	2010	1	12	INDL	C	D	STD	HES	42809H107	HESS CC
1	1380	20100630	2010	2	12	INDL	C	D	STD	HES	42809H107	HESS CC
2	1380	20100930	2010	3	12	INDL	C	D	STD	HES	42809H107	HESS CC
3	1380	20101231	2010	4	12	INDL	C	D	STD	HES	42809H107	HESS CC
4	1380	20110331	2011	1	12	INDL	C	D	STD	HES	42809H107	HESS CC
...	...	...	...	...	...	...	...	...	...	...	...	
839	186989	20151231	2015	4	12	INDL	C	D	STD	MPC	56585A102	MARATH PETROLE CC
840	186989	20160331	2016	1	12	INDL	C	D	STD	MPC	56585A102	MARATH PETROLE CC
841	186989	20160630	2016	2	12	INDL	C	D	STD	MPC	56585A102	MARATH PETROLE CC
842	186989	20160930	2016	3	12	INDL	C	D	STD	MPC	56585A102	MARATH PETROLE CC
843	186989	20161231	2016	4	12	INDL	C	D	STD	MPC	56585A102	MARATH PETROLE CC

844 rows × 344 columns

```
#1. Read 'Energy.xlsx' and 'EnergyRating.xlsx' as BalanceSheet and Ratings(dataframe).
#2. drop the column if more than 90% value in this colnmn is 0 (or missing value).
#3. replace all None or NaN with average value of each column.


Ratings = pd.read_excel('EnergyRating.xlsx')
Ratings = Ratings.drop(columns=Ratings.columns[((Ratings==0).mean()>0.9)])
Ratings = Ratings.fillna(Ratings.mean())
Ratings = Ratings.dropna(axis = 1, how = 'all')
Ratings
```

	Global Company Key	S&P Domestic Long Term Issuer Credit Rating	S&P Domestic Short Term Issuer Credit Rating	Data Date	Address Line 1	Ticker Symbol
0	1380	BBB-	NaN	20100131	1185 Avenue of the Americas, 40th Floor	HES
1	1380	BBB-	NaN	20100228	1185 Avenue of the Americas, 40th Floor	HES
2	1380	BBB-	NaN	20100331	1185 Avenue of the Americas, 40th Floor	HES
3	1380	BBB-	NaN	20100430	1185 Avenue of the Americas, 40th Floor	HES
4	1380	BBB-	NaN	20100531	1185 Avenue of the Americas, 40th Floor	HES
...	...	...	...	...	...	...
2517	186989	BBB	A-2	20161031	539 South Main Street	MPC
2518	186989	BBB	A-2	20161130	539 South Main Street	MPC
2519	186989	BBB	A-2	20161231	539 South Main Street	MPC
2520	186989	BBB	A-2	20170131	539 South Main Street	MPC
2521	186989	BBB	A-2	20170228	539 South Main Street	MPC

2522 rows × 6 columns

```
#4. Normalize the table
BalanceSheet_cols = BalanceSheet.select_dtypes('float') #Normalization to applied only on Numerical Dat
Ratings_cols = Ratings.select_dtypes(np.number)
```

```
#4. Normalize the table
def x_new(x):
    a = (x-x.min())/(x.max()-x.min())
    return a
Ratings_Normalized = Ratings_cols.apply(x_new)
print(Ratings_Normalized)
```

	Global Company Key	Data Date
0	0.0	0.000000
1	0.0	0.001384
2	0.0	0.002853
3	0.0	0.004266
4	0.0	0.005706
...	...	...
2517	1.0	0.868796
2518	1.0	0.870208
2519	1.0	0.871649
2520	1.0	0.998616
2521	1.0	1.000000

2522 rows × 2 columns

```
#4. Normalize the table
BalanceSheet_Normalized = BalanceSheet_cols.apply(x_new)
(BalanceSheet_Normalized.dropna(axis = 1, how = 'all'))
```

	Accumulated Other Comprehensive Income (Loss)	Current Assets - Other - Total	Current Assets - Total	Other Long-term Assets	Non-Current Assets - Total	Assets Netting & Other Adjustments	Accum Other Comp Inc - Derivatives Unrealized Gain/Loss	Accum Other Comp Inc - Other Adjustments	Accum Other Comp Inc - Min Pension Liab Adj	Assets Level2 (Observable)
0	0.741506	0.158268	0.113598	0.063524	0.061304	0.954930	0.000000	0.504673	0.972251	0.006726
1	0.746299	0.151090	0.107231	0.063996	0.061422	0.959571	0.154973	0.325545	0.972730	0.007952
2	0.753718	0.091027	0.114375	0.067850	0.073071	0.962252	0.213545	0.514019	0.973347	0.005881
3	0.760875	0.105489	0.113598	0.065264	0.079738	0.932446	0.320927	0.573209	0.973621	0.006554
4	0.770264	0.079838	0.122228	0.064369	0.081718	0.904394	0.380720	0.855140	0.974101	0.006226
...	...	...	...	...	...	...	...	...	...	...
839	0.788484	0.019035	0.122689	0.002634	0.103523	0.979373	0.802929	0.556075	0.977869	0.000012
840	0.788484	0.021252	0.104613	0.021704	0.103580	0.978857	0.802929	0.556075	0.977869	0.000006
841	0.788385	0.014708	0.133647	0.021430	0.104149	0.970710	0.802929	0.556075	0.977664	0.000006
842	0.788188	0.018297	0.123662	0.020759	0.103875	0.957096	0.802929	0.556075	0.977252	0.000006
843	0.791241	0.024630	0.134923	0.002335	0.104768	0.914810	0.802929	0.556075	0.983625	0.000000

844 rows × 292 columns

```
'''

5. Define an apply function to return the statistical information for variables =
['Current Assets - Other - Total', 'Current Assets - Total', 'Other Long-termAssets',
'Assets Netting & Other Adjustments'], you need to return a dataframe
which has exactly same format with pandas method .describe().

'''
cor = pd.read_excel('Energy.xlsx')
cor1 = cor[['Current Assets - Other - Total', 'Current Assets - Total', 'Other Long-term Assets','Asset
print(cor1.describe())
#df_cor = df_cor[['Current Assets - Other - Total', 'Current Assets - Total', 'Other Long-term
#Assets', 'Assets Netting  Other Adjustments']
```

	Current Assets - Other - Total	Current Assets - Total	Other Long-term Assets	Assets Netting & Other Adjustments
count	830.000000	830.000000	746.000000	695.000000
mean	1037.255108	9735.614198	1486.818614	-166.147714
std	1592.111892	13682.311837	2597.436070	618.162480
min	2.671000	144.786000	13.072000	-9558.000000
25%	177.487000	1477.057000	155.750000	-45.500000
50%	431.500000	4612.000000	629.053000	0.000000
75%	1017.500000	11739.000000	1733.500000	0.000000
max	9476.000000	76160.000000	40233.000000	138.000000

```
'''

6. Calculate the correlation matrix for variables = ['Current Assets - Other - Total',
'Current Assets - Total', 'Other Long-term Assets', 'Assets Netting & Other
Adjustments'].

'''
(cor1.corr())
```

	Current Assets - Other - Total	Current Assets - Total	Other Long-term Assets	Assets Netting & Other Adjustments
Current Assets - Other - Total	1.000000	0.790047	0.637424	0.047226
Current Assets - Total	0.790047	1.000000	0.677142	-0.081643
Other Long-term Assets	0.637424	0.677142	1.000000	-0.030717
Assets Netting & Other Adjustments	0.047226	-0.081643	-0.030717	1.000000

```
'''
7. If you look at column ('Company Name'), you will find some company name
end with 'CORP', 'CO' or 'INC'. Create a new column (Name: 'CO') to store
the last word of company name. (For example: 'CORP' or, 'CO' or 'INC') (Hint:
using map function)

'''

BalanceSheet['CO'] = BalanceSheet['Company Name'].str.split().str[-1]
(BalanceSheet[['CO']])
```

	CO
0	CORP
1	CORP
2	CORP
3	CORP
4	CORP
...	...
839	CORP
840	CORP
841	CORP
842	CORP
843	CORP

844 rows × 1 columns

```
'''
8. Merge (inner) Ratings and BalanceSheet based on ’datadate’ and ’Global Company
Key’, and name merged dataset ’Matched’.

'''
matched = pd.merge(Ratings,BalanceSheet, on = ['Data Date', 'Global Company Key'],how = 'inner' )#on =  'Global Company Key')
(matched)
```

	Global Company Key	S&P Domestic Long Term Issuer Credit Rating	S&P Domestic Short Term Issuer Credit Rating	Data Date	Address Line 1	Ticker Symbol_x	Fiscal Year	Fiscal Quarter	Fiscal Year-end Month	Industry Format	Level of Consolidation - Company Interim Descriptor	Population Source	Data Format	Ticker Symbol_y	CUS
0	1380	BBB-	NaN	20100331	1185 Avenue of the Americas, 40th Floor	HES	2010	1	12	INDL	C	D	STD	HES	42809H1
1	1380	BBB-	NaN	20100630	1185 Avenue of the Americas, 40th Floor	HES	2010	2	12	INDL	C	D	STD	HES	42809H1
2	1380	BBB	NaN	20100930	1185 Avenue of the Americas, 40th Floor	HES	2010	3	12	INDL	C	D	STD	HES	42809H1
3	1380	BBB	NaN	20101231	1185 Avenue of the Americas, 40th Floor	HES	2010	4	12	INDL	C	D	STD	HES	42809H1
4	1380	BBB	NaN	20110331	1185 Avenue of the Americas, 40th Floor	HES	2011	1	12	INDL	C	D	STD	HES	42809H1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
817	186989	BBB	A-2	20151231	539 South Main	MPC	2015	4	12	INDL	C	D	STD	MPC	56585A1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
817	186989	BBB	A-2	20151231	539 South Main Street	MPC	2015	4	12	INDL	C	D	STD	MPC	56585A1
818	186989	BBB	A-2	20160331	539 South Main Street	MPC	2016	1	12	INDL	C	D	STD	MPC	56585A1
819	186989	BBB	A-2	20160630	539 South Main Street	MPC	2016	2	12	INDL	C	D	STD	MPC	56585A1
820	186989	BBB	A-2	20160930	539 South Main Street	MPC	2016	3	12	INDL	C	D	STD	MPC	56585A1
821	186989	BBB	A-2	20161231	539 South Main Street	MPC	2016	4	12	INDL	C	D	STD	MPC	56585A1

822 rows × 349 columns

```
'''
9. Mapping
For dataset 'Matched', we have following mapping:
AAA = 0
AA+ = 1
AA = 2
AA- =3
A+ = 4
A = 5
A- = 6
BBB+ = 7
BBB = 8
BBB- = 9
BB+ = 10
BB = 11
others = 12
Using map function to create a new varible = 'Rate', which maps ratings to
numerical ratings.

'''



number_rating = {
    'AAA' : 0,
    'AA+' : 1,
    'AA' : 2,
    'AA-' :3,
    'A+' : 4,
    'A' : 5,
    'A-' : 6,
    'BBB+' :7,
    'BBB' : 8,
    'BBB-' : 9,
    'BB+' : 10,
    'BB' : 11,
    'others' :12,
    '':12
}

matched['Rate'] = matched['S&P Domestic Long Term Issuer Credit Rating'].map(number_rating)
print(matched['Rate'])
```

	Global Company Key	S&P Domestic Long Term Issuer Credit Rating	S&P Domestic Short Term Issuer Credit Rating	Data Date	Address Line 1	Ticker Symbol_x	Fiscal Year	Fiscal Quarter	Fiscal Year-end Month	Industry Format	Level of Consolidation - Company Interim Descriptor	Population Source	Data Format	Ticker Symbol_y	CUS
0	1380	BBB-	NaN	20100331	1185 Avenue of the Americas, 40th Floor	HES	2010	1	12	INDL	C	D	STD	HES	42809H1
1	1380	BBB-	NaN	20100630	1185 Avenue of the Americas, 40th Floor	HES	2010	2	12	INDL	C	D	STD	HES	42809H1
2	1380	BBB	NaN	20100930	1185 Avenue of the Americas, 40th Floor	HES	2010	3	12	INDL	C	D	STD	HES	42809H1
3	1380	BBB	NaN	20101231	1185 Avenue of the Americas, 40th Floor	HES	2010	4	12	INDL	C	D	STD	HES	42809H1
4	1380	BBB	NaN	20110331	1185 Avenue of the Americas, 40th Floor	HES	2011	1	12	INDL	C	D	STD	HES	42809H1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
817	186989	BBB	A-2	20151231	539 South Main Street	MPC	2015	4	12	INDL	C	D	STD	MPC	56585A1
818	186989	BBB	A-2	20160331	539 South Main Street	MPC	2016	1	12	INDL	C	D	STD	MPC	56585A1
819	186989	BBB	A-2	20160630	539 South Main Street	MPC	2016	2	12	INDL	C	D	STD	MPC	56585A1
820	186989	BBB	A-2	20160930	539 South Main Street	MPC	2016	3	12	INDL	C	D	STD	MPC	56585A1
821	186989	BBB	A-2	20161231	539 South Main Street	MPC	2016	4	12	INDL	C	D	STD	MPC	56585A1

822 rows × 350 columns

```
'''
10. Calculate the rating frequency of company whose name end with 'CO'. (Calculate
the distribution of rating given the company name ending with 'CO', Hint,
use map function)

'''
CO = matched[matched['CO'] == 'CO']
RatingFrequency = CO['Rate'].mean()
print("Rating Frequency of Companies ending with CO is {ratings}".format(ratings = RatingFrequency))
```

```
Rating Frequency of Companies ending with CO is 8.392857142857142
```