**DAC Phase 3**

**Problem Statement:** Product Sales Analysis

**DATA CLEANING**

import numpy as np  
import pandas as pd  
import os  
for dirname, \_, filenames in os.walk('/content/statsfinal.csv'):  
 for filename in filenames:  
 print(os.path.join(dirname, filename))

#### Importing libraries

import pandas as pd  
import numpy as np  
import seaborn as sns  
import matplotlib.pyplot as plt  
pd.options.display.max\_columns=50  
sns.set(style="darkgrid")

#### Importing data

df=pd.read\_csv("/content/statsfinal.csv")  
df.head(5)

Unnamed: 0 Date Q-P1 Q-P2 Q-P3 Q-P4 S-P1 S-P2 \  
0 0 13-06-2010 5422 3725 576 907 17187.74 23616.50   
1 1 14-06-2010 7047 779 3578 1574 22338.99 4938.86   
2 2 15-06-2010 1572 2082 595 1145 4983.24 13199.88   
3 3 16-06-2010 5657 2399 3140 1672 17932.69 15209.66   
4 4 17-06-2010 3668 3207 2184 708 11627.56 20332.38   
  
 S-P3 S-P4   
0 3121.92 6466.91   
1 19392.76 11222.62   
2 3224.90 8163.85   
3 17018.80 11921.36   
4 11837.28 5048.04

df.sample(2)

Unnamed: 0 Date Q-P1 Q-P2 Q-P3 Q-P4 S-P1 S-P2 \  
3986 3986 29-05-2021 696 1160 846 1512 2206.32 7354.40   
2877 2877 11-05-2018 7990 3664 3091 526 25328.30 23229.76   
  
 S-P3 S-P4   
3986 4585.32 10780.56   
2877 16753.22 3750.38

# Changing dtype  
from datetime import datetime as dt  
df[df["Date"]=="31-9-2010"]

Unnamed: 0 Date Q-P1 Q-P2 Q-P3 Q-P4 S-P1 S-P2 \  
109 109 31-9-2010 4986 342 4978 558 15805.62 2168.28   
  
 S-P3 S-P4   
109 26980.76 3978.54

df['Date'] = pd.to\_datetime(df['Date'], errors='coerce')

<ipython-input-13-4ac1f8d99197>:1: UserWarning: Parsing dates in DD/MM/YYYY format when dayfirst=False (the default) was specified. This may lead to inconsistently parsed dates! Specify a format to ensure consistent parsing.  
 df['Date'] = pd.to\_datetime(df['Date'], errors='coerce')

df[df['Date'].isnull()]

Unnamed: 0 Date Q-P1 Q-P2 Q-P3 Q-P4 S-P1 S-P2 S-P3 \  
109 109 NaT 4986 342 4978 558 15805.62 2168.28 26980.76   
170 170 NaT 4632 3930 523 1581 14683.44 24916.20 2834.66   
473 473 NaT 2242 401 5926 789 7107.14 2542.34 32118.92   
534 534 NaT 325 3476 4588 1771 1030.25 22037.84 24866.96   
836 836 NaT 1003 256 1346 1449 3179.51 1623.04 7295.32   
897 897 NaT 2509 2666 4146 593 7953.53 16902.44 22471.32   
1200 1200 NaT 597 709 5470 1994 1892.49 4495.06 29647.40   
1261 1261 NaT 7681 1235 347 1087 24348.77 7829.90 1880.74   
1564 1564 NaT 5333 833 3494 618 16905.61 5281.22 18937.48   
1625 1625 NaT 3870 2779 3246 1290 12267.90 17618.86 17593.32   
1928 1928 NaT 3583 2111 4225 1401 11358.11 13383.74 22899.50   
1989 1989 NaT 7516 3423 3116 458 23825.72 21701.82 16888.72   
2291 2291 NaT 7891 741 2280 1068 25014.47 4697.94 12357.60   
2352 2352 NaT 2457 3144 533 1184 7788.69 19932.96 2888.86   
2655 2655 NaT 3512 2851 4072 1597 11133.04 18075.34 22070.24   
2716 2716 NaT 6094 3798 5849 881 19317.98 24079.32 31701.58   
3019 3019 NaT 1727 2645 5715 1295 5474.59 16769.30 30975.30   
3080 3080 NaT 7360 2974 2717 1127 23331.20 18855.16 14726.14   
3383 3383 NaT 3195 2525 5918 1003 10128.15 16008.50 32075.56   
3444 3444 NaT 2660 2674 2732 934 8432.20 16953.16 14807.44   
3746 3746 NaT 4713 1227 4065 403 14940.21 7779.18 22032.30   
3807 3807 NaT 870 3463 798 851 2757.90 21955.42 4325.16   
4110 4110 NaT 3511 2609 1543 853 11129.87 16541.06 8363.06   
4171 4171 NaT 506 3333 3897 574 1604.02 21131.22 21121.74   
4474 4474 NaT 6964 1873 5481 1336 22075.88 11874.82 29707.02   
4535 4535 NaT 4600 2006 3796 1426 14582.00 12718.04 20574.32   
  
 S-P4   
109 3978.54   
170 11272.53   
473 5625.57   
534 12627.23   
836 10331.37   
897 4228.09   
1200 14217.22   
1261 7750.31   
1564 4406.34   
1625 9197.70   
1928 9989.13   
1989 3265.54   
2291 7614.84   
2352 8441.92   
2655 11386.61   
2716 6281.53   
3019 9233.35   
3080 8035.51   
3383 7151.39   
3444 6659.42   
3746 2873.39   
3807 6067.63   
4110 6081.89   
4171 4092.62   
4474 9525.68   
4535 10167.38

## Filling the NaT values with average of time  
df["Date"].fillna(df["Date"].mean(),inplace=True)

df['Date'].isnull().sum()

0

df.dtypes

Unnamed: 0 int64  
Date datetime64[ns]  
Q-P1 int64  
Q-P2 int64  
Q-P3 int64  
Q-P4 int64  
S-P1 float64  
S-P2 float64  
S-P3 float64  
S-P4 float64  
dtype: object

#fetching month,day of week, weekday  
df["month"]=df["Date"].dt.month\_name()  
df["day"]=df["Date"].dt.day\_name()  
df["dayoftheweek"]=df["Date"].dt.weekday  
df["year"]=df["Date"].dt.year  
df.sample()

Unnamed: 0 Date Q-P1 Q-P2 Q-P3 Q-P4 S-P1 S-P2 \  
1855 1855 2015-07-19 6750 1269 1249 1242 21397.5 8045.46   
  
 S-P3 S-P4 month day dayoftheweek year   
1855 6769.58 8855.46 July Sunday 6 2015

## Droping column unnamed as it is not usefull for us  
df.drop(columns=["Unnamed: 0"],inplace=True)  
df.sample()

Date Q-P1 Q-P2 Q-P3 Q-P4 S-P1 S-P2 S-P3 S-P4 \  
2336 2016-11-15 5461 790 4408 1946 17311.37 5008.6 23891.36 13874.98   
  
 month day dayoftheweek year   
2336 November Tuesday 1 2016

df.corr().T

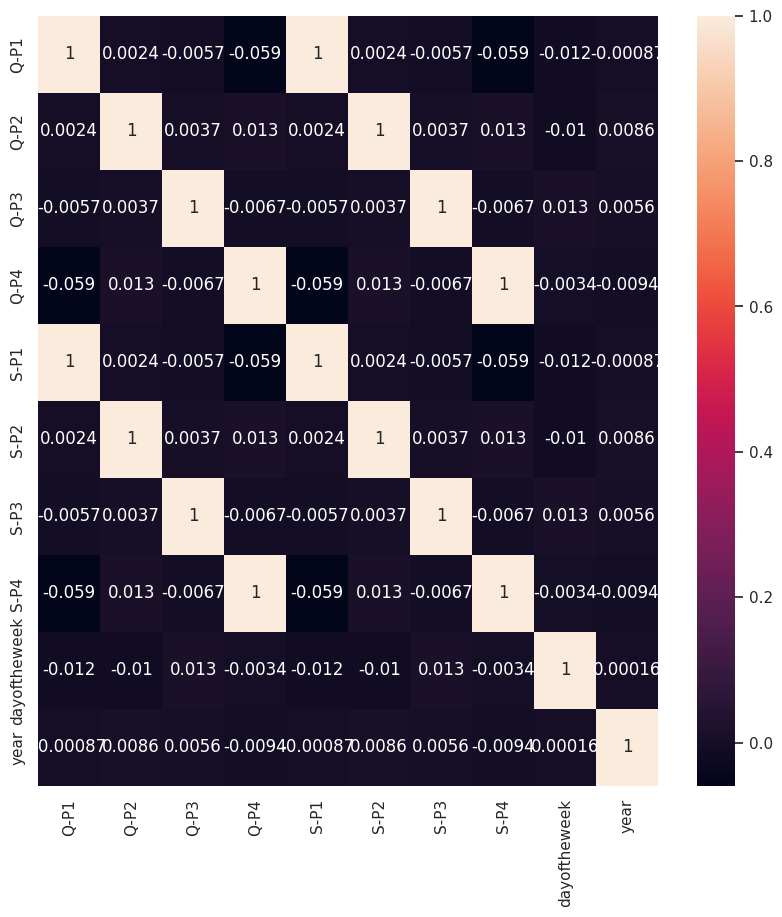
<ipython-input-20-50be2e233ef7>:1: FutureWarning: The default value of numeric\_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric\_only to silence this warning.  
 df.corr().T

Q-P1 Q-P2 Q-P3 Q-P4 S-P1 S-P2 \  
Q-P1 1.000000 0.002422 -0.005650 -0.059365 1.000000 0.002422   
Q-P2 0.002422 1.000000 0.003729 0.013082 0.002422 1.000000   
Q-P3 -0.005650 0.003729 1.000000 -0.006693 -0.005650 0.003729   
Q-P4 -0.059365 0.013082 -0.006693 1.000000 -0.059365 0.013082   
S-P1 1.000000 0.002422 -0.005650 -0.059365 1.000000 0.002422   
S-P2 0.002422 1.000000 0.003729 0.013082 0.002422 1.000000   
S-P3 -0.005650 0.003729 1.000000 -0.006693 -0.005650 0.003729   
S-P4 -0.059365 0.013082 -0.006693 1.000000 -0.059365 0.013082   
dayoftheweek -0.012221 -0.010037 0.012546 -0.003351 -0.012221 -0.010037   
year -0.000866 0.008556 0.005632 -0.009436 -0.000866 0.008556   
  
 S-P3 S-P4 dayoftheweek year   
Q-P1 -0.005650 -0.059365 -0.012221 -0.000866   
Q-P2 0.003729 0.013082 -0.010037 0.008556   
Q-P3 1.000000 -0.006693 0.012546 0.005632   
Q-P4 -0.006693 1.000000 -0.003351 -0.009436   
S-P1 -0.005650 -0.059365 -0.012221 -0.000866   
S-P2 0.003729 0.013082 -0.010037 0.008556   
S-P3 1.000000 -0.006693 0.012546 0.005632   
S-P4 -0.006693 1.000000 -0.003351 -0.009436   
dayoftheweek 0.012546 -0.003351 1.000000 0.000159   
year 0.005632 -0.009436 0.000159 1.000000

plt.figure(figsize=(10,10))  
sns.heatmap(df.corr(),annot=True)

<ipython-input-21-60c58fc9bbb1>:2: FutureWarning: The default value of numeric\_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric\_only to silence this warning.  
 sns.heatmap(df.corr(),annot=True)

<Axes: >

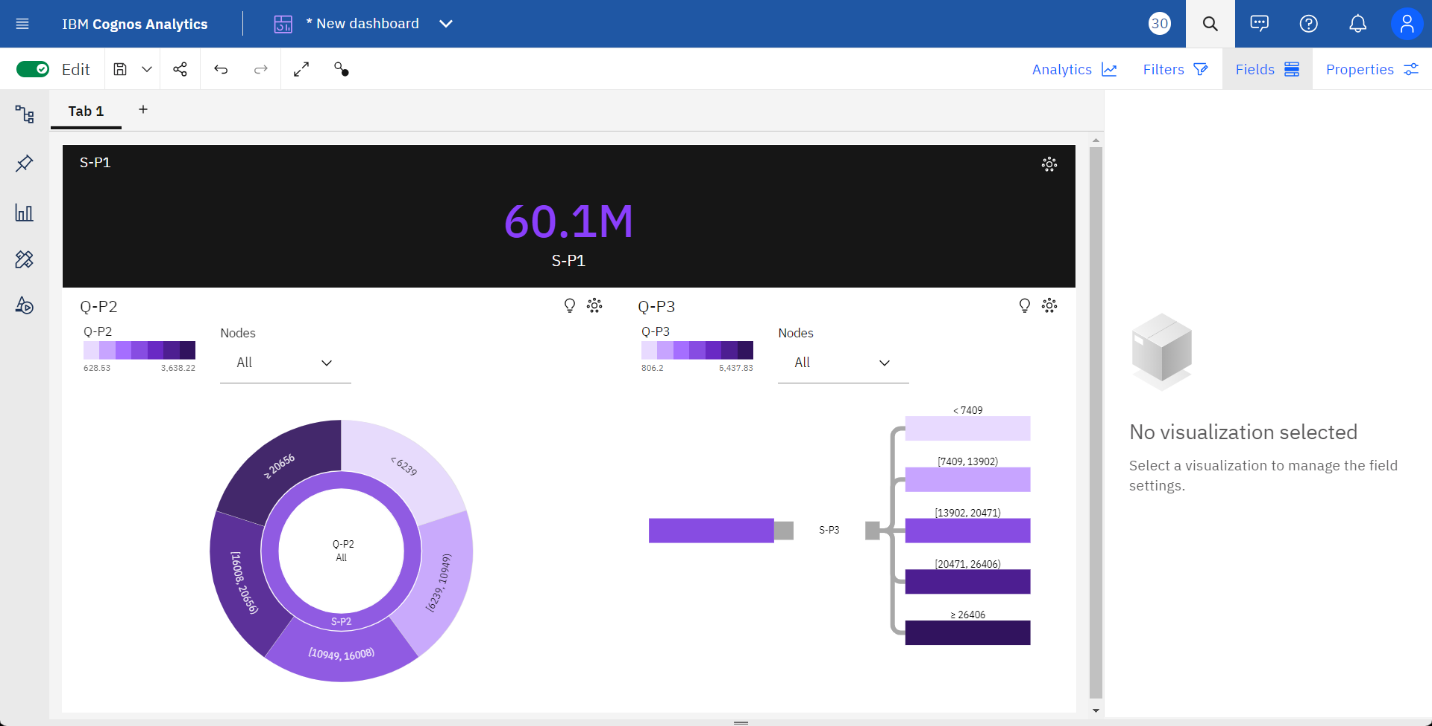


#### There is no strong correlation between the columns

for i in df.columns:  
 print(i,"---------",df[i].unique())

Date --------- ['2010-06-13T00:00:00.000000000' '2010-06-14T00:00:00.000000000'  
 '2010-06-15T00:00:00.000000000' ... '2023-01-02T00:00:00.000000000'  
 '2023-02-02T00:00:00.000000000' '2023-03-02T00:00:00.000000000']  
Q-P1 --------- [5422 7047 1572 ... 1227 3122 1234]  
Q-P2 --------- [3725 779 2082 ... 3404 841 3143]  
Q-P3 --------- [ 576 3578 595 ... 4825 3588 5899]  
Q-P4 --------- [ 907 1574 1145 ... 1161 1151 1112]  
S-P1 --------- [17187.74 22338.99 4983.24 ... 3889.59 9896.74 3911.78]  
S-P2 --------- [23616.5 4938.86 13199.88 ... 21581.36 5331.94 19926.62]  
S-P3 --------- [ 3121.92 19392.76 3224.9 ... 26151.5 19446.96 31972.58]  
S-P4 --------- [ 6466.91 11222.62 8163.85 ... 8277.93 8206.63 7928.56]  
month --------- ['June' 'January' 'February' 'March' 'April' 'May' 'July' 'August'  
 'September' 'October' 'November' 'December']  
day --------- ['Sunday' 'Monday' 'Tuesday' 'Wednesday' 'Thursday' 'Friday' 'Saturday']  
dayoftheweek --------- [6 0 1 2 3 4 5]  
year --------- [2010 2016 2011 2012 2013 2014 2015 2017 2018 2019 2020 2021 2022 2023]

**Visualization Using IBM Cognos Analytics:**

****