CARE
COLLEGE OF ENGINEERING
Approved by AICTE, New Delhi | Affiliated to Anna University, Chennai
Accredited by NAAC with 'A' Grade
#27, Thayanur, Tiruchirappalli - 620009

COLLEGE CODE: 8107

COURSE: DATA ANALYTICS WITH COGNOS

PHASE III: IMPORTING AND CLEANING THE DATA

TITLE: PRODUCT SALES ANALYSIS

TEAM MEMBERS:

- ARUL PALANIAPPA S
  8107243009
  arulpalaniappa.s@care.ac.in

- AZARUDEEN B
  810721243011
  azarudeen.b@care.ac.in

- BALAJI C
  810721243012
  balaji.c@care.ac.in

- SARAVANAN V
  810721243045
  saravanan.v@care.ac.in

- VIMAL M
  810721243057
  vimal.m@care.ac.in

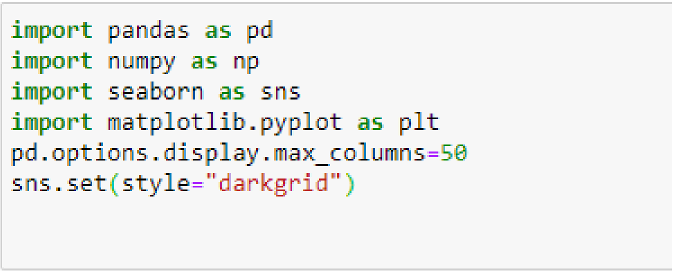
About Dataset :

Greetings , fellow analyst ! REC corp LTD. is small-scaled business venture established in India. They have been selling FOUR PRODUCTS for OVER TEN YEARS. The products are P1, P2, P3 and P4. They have collected data from their retail centers and organized it into a small csv file , which has been given to you.

The excel file contains about 8 numerical parameters :

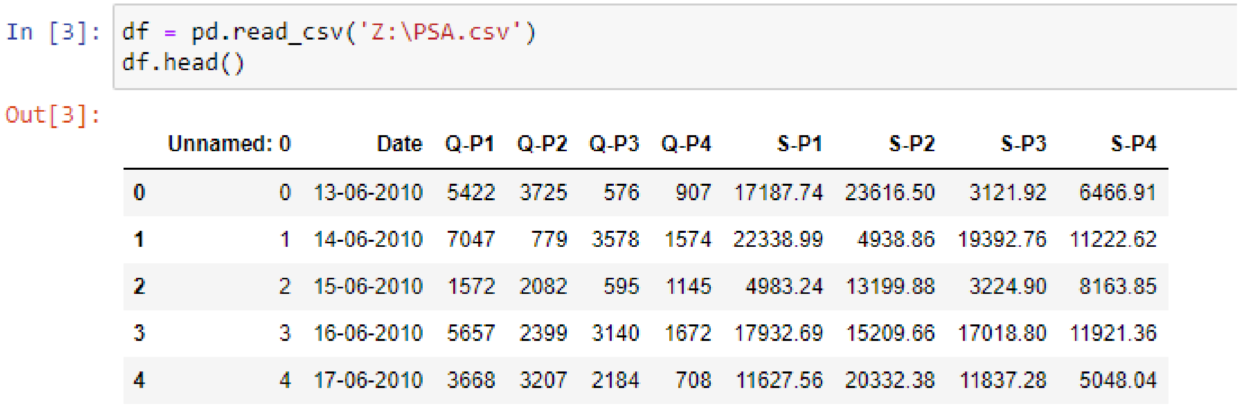
- Q1- Total unit sales of product 1
- Q2- Total unit sales of product 2
- Q3- Total unit sales of product 3
- Q4- Total unit sales of product 4
- S1- Total revenue from product 1
- S2- Total revenue from product 2
- S3- Total revenue from product 3
- S4- Total revenue from product 4

Import **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")
```

Import Data:



```
In [3]: df = pd.read_csv('Z:\PSA.csv')
        df.head()
```

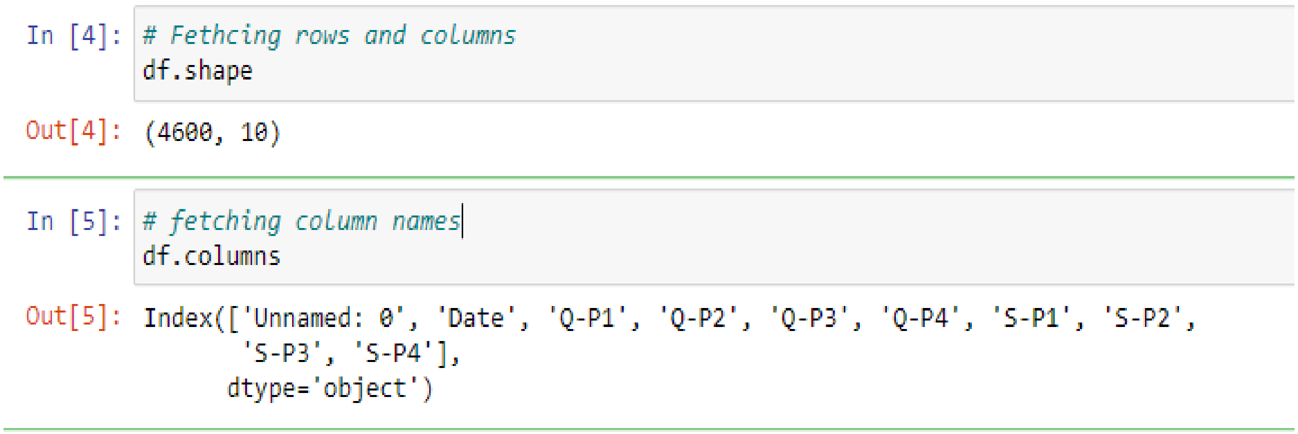
Out[3]:

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

Workflow :

- Understanding the data
- Data cleaning
- Exploratory Data Analysis
- Insights

Understanding the data

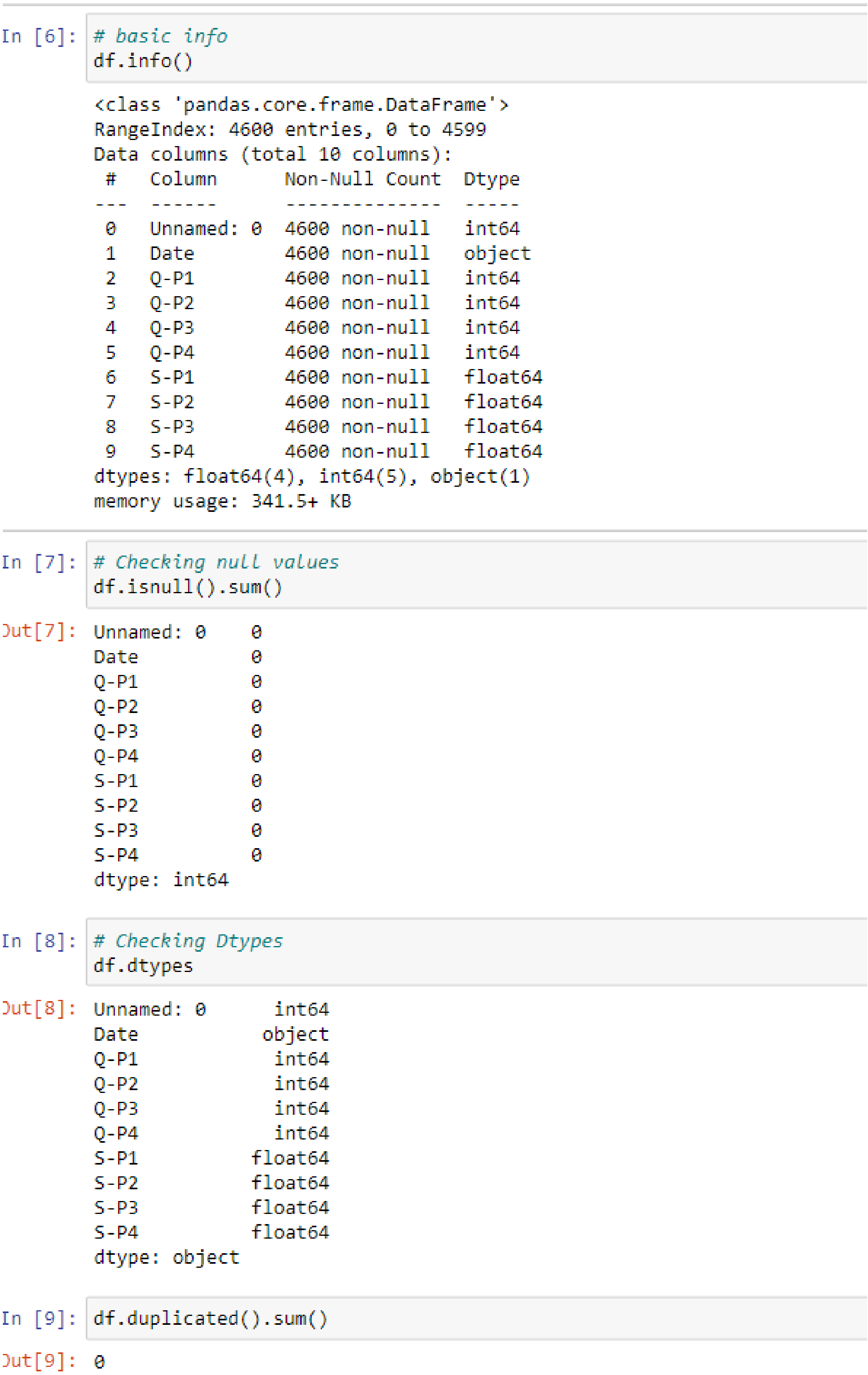


```
In [4]: # Fethcing rows and columns
        df.shape

Out[4]: (4600, 10)
```

```
In [5]: # fetching column names
        df.columns

Out[5]: Index(['Unnamed: 0', 'Date', 'Q-P1', 'Q-P2', 'Q-P3', 'Q-P4', 'S-P1', 'S-P2',
               'S-P3', 'S-P4'],
              dtype='object')
```



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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4600 entries, 0 to 4599
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Unnamed: 0  4600 non-null   int64  
 1   Date        4600 non-null   object 
 2   Q-P1        4600 non-null   int64  
 3   Q-P2        4600 non-null   int64  
 4   Q-P3        4600 non-null   int64  
 5   Q-P4        4600 non-null   int64  
 6   S-P1        4600 non-null   float64
 7   S-P2        4600 non-null   float64
 8   S-P3        4600 non-null   float64
 9   S-P4        4600 non-null   float64
dtypes: float64(4), int64(5), object(1)
memory usage: 341.5+ KB
```

```
In [7]: # Checking null values
        df.isnull().sum()
```

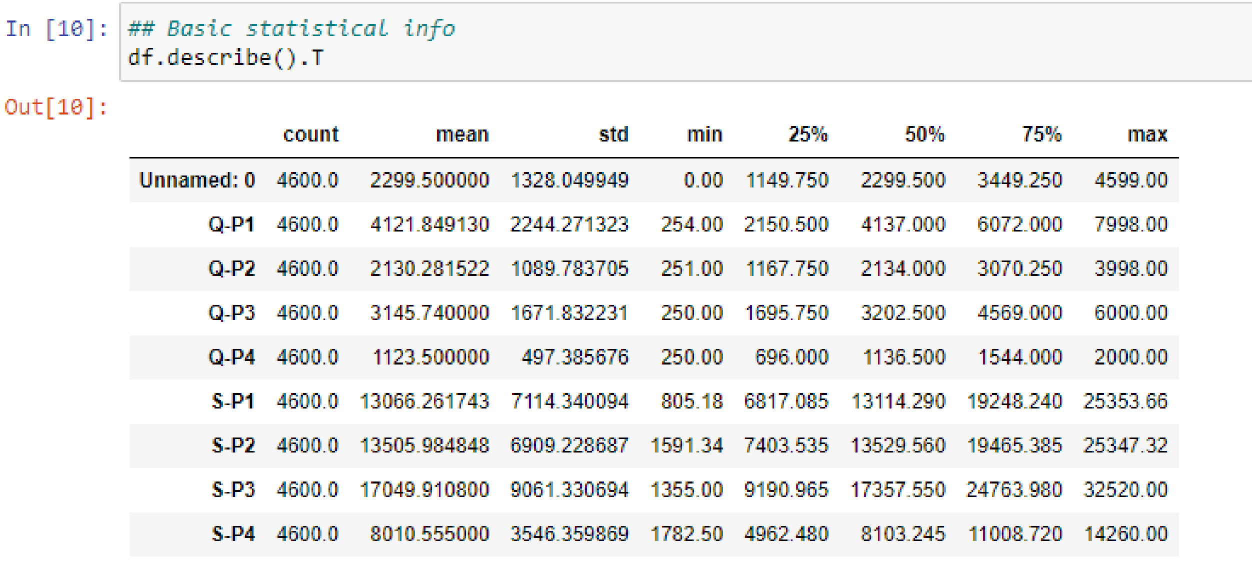
```
)ut[7]: Unnamed: 0    0
        Date          0
        Q-P1          0
        Q-P2          0
        Q-P3          0
        Q-P4          0
        S-P1          0
        S-P2          0
        S-P3          0
        S-P4          0
        dtype: int64
```

```
In [8]: # Checking Dtypes
        df.dtypes
```

```
)ut[8]: Unnamed: 0      int64
        Date           object
        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
```

```
In [9]: df.duplicated().sum()
```

```
)ut[9]: 0
```



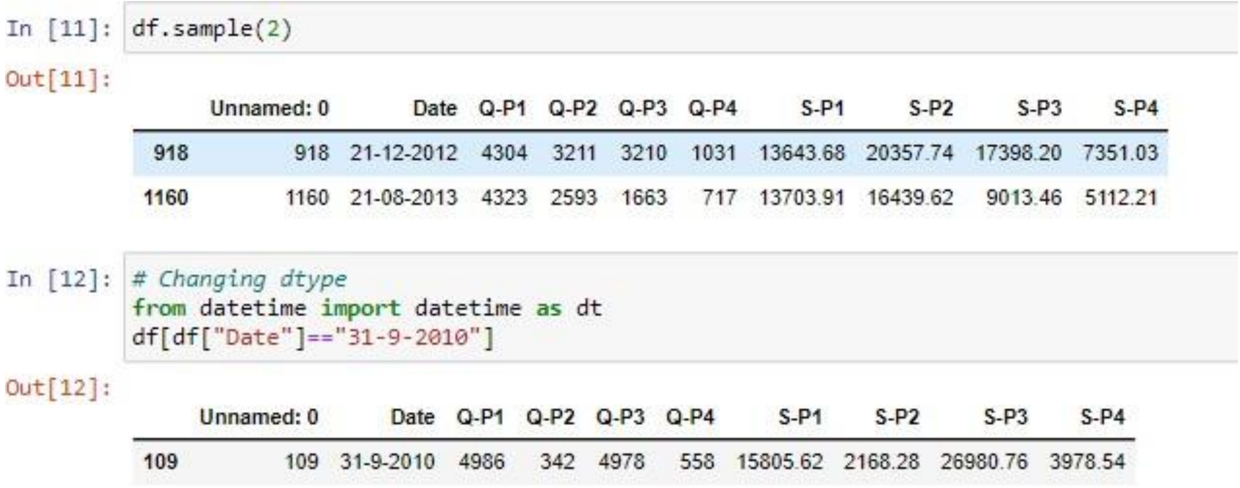
In [10]:

```
## Basic statistical info
df.describe().T
```

Out[10]:

	count	mean	std	min	25%	50%	75%	max
Unnamed: 0	4600.0	2299.500000	1328.049949	0.00	1149.750	2299.500	3449.250	4599.00
Q-P1	4600.0	4121.849130	2244.271323	254.00	2150.500	4137.000	6072.000	7998.00
Q-P2	4600.0	2130.281522	1089.783705	251.00	1167.750	2134.000	3070.250	3998.00
Q-P3	4600.0	3145.740000	1671.832231	250.00	1695.750	3202.500	4569.000	6000.00
Q-P4	4600.0	1123.500000	497.385676	250.00	696.000	1136.500	1544.000	2000.00
S-P1	4600.0	13066.261743	7114.340094	805.18	6817.085	13114.290	19248.240	25353.66
S-P2	4600.0	13505.984848	6909.228687	1591.34	7403.535	13529.560	19465.385	25347.32
S-P3	4600.0	17049.910800	9061.330694	1355.00	9190.965	17357.550	24763.980	32520.00
S-P4	4600.0	8010.555000	3546.359869	1782.50	4962.480	8103.245	11008.720	14260.00

Data Cleaning



In [11]:

```
df.sample(2)
```

Out[11]:

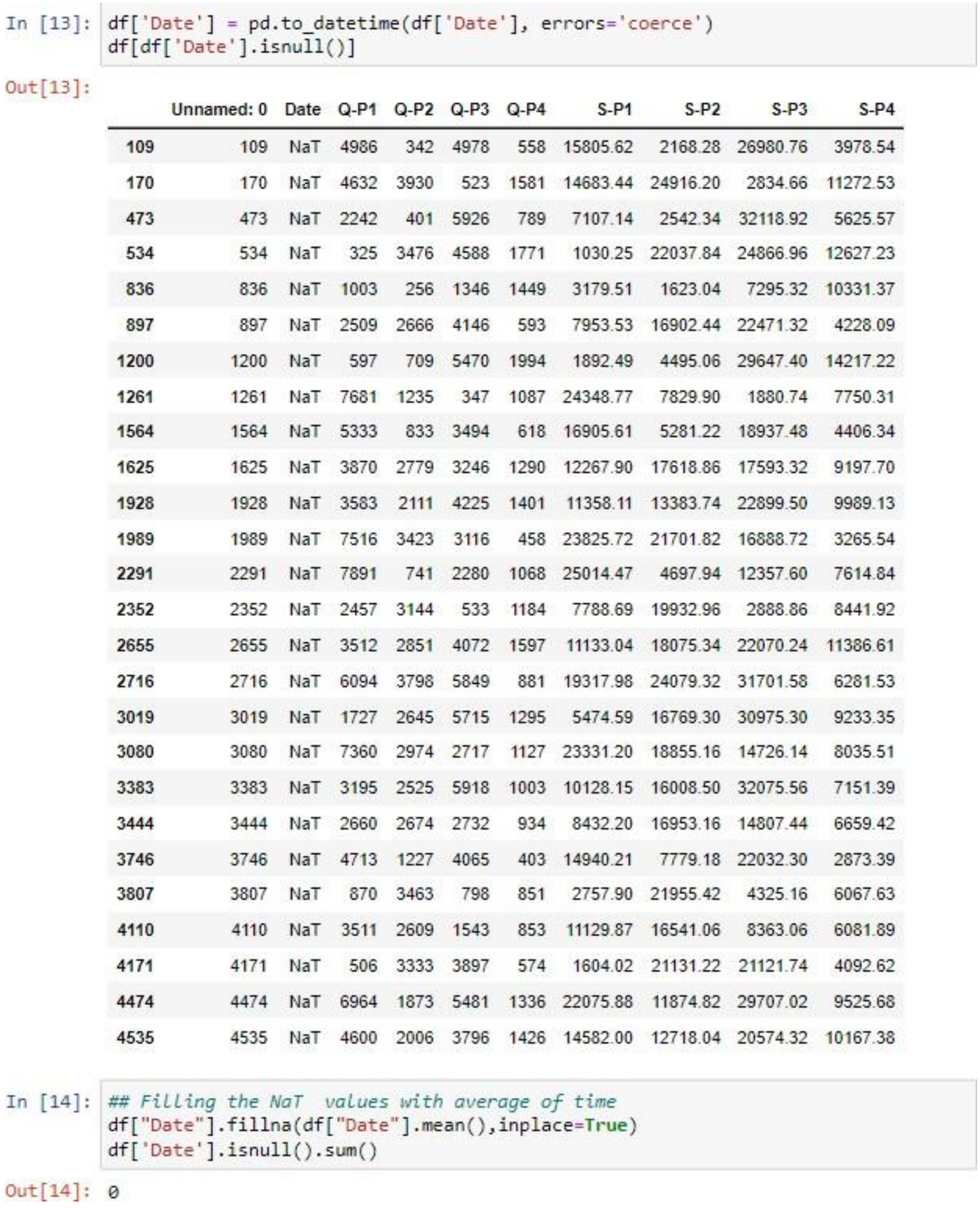
	Unnamed: 0	Date	Q-P1	Q-P2	Q-P3	Q-P4	S-P1	S-P2	S-P3	S-P4
918	918	21-12-2012	4304	3211	3210	1031	13643.68	20357.74	17398.20	7351.03
1160	1160	21-08-2013	4323	2593	1663	717	13703.91	16439.62	9013.46	5112.21

In [12]:

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

Out[12]:

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



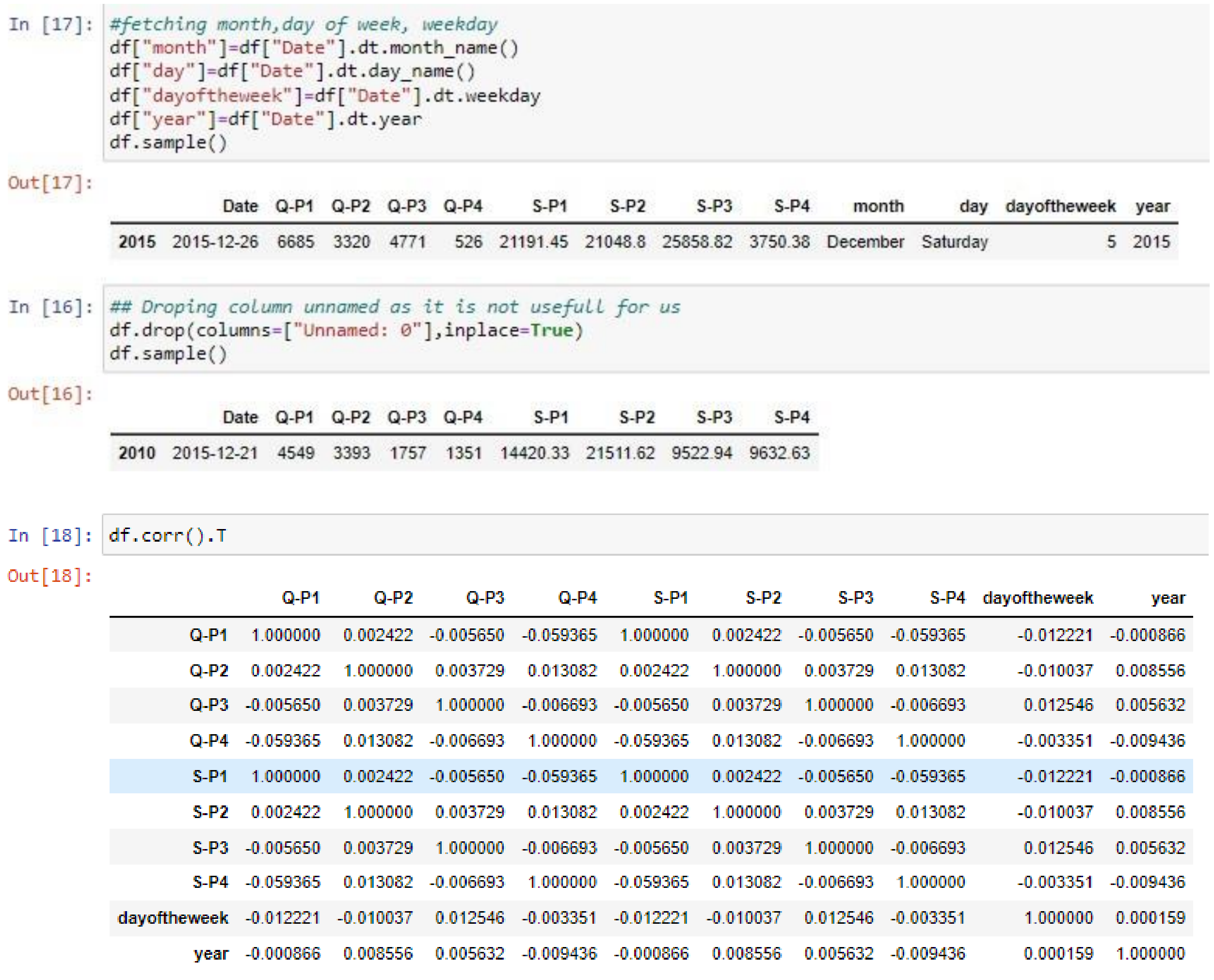
```
In [13]: df['Date'] = pd.to_datetime(df['Date'], errors='coerce')
         df[df['Date'].isnull()]
```

Out[13]:

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

```
In [14]: ## Filling the NaT  values with average of time
         df["Date"].fillna(df["Date"].mean(),inplace=True)
         df['Date'].isnull().sum()

Out[14]: 0
```



```
In [17]: #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()
```

Out[17]:

	Date	Q-P1	Q-P2	Q-P3	Q-P4	S-P1	S-P2	S-P3	S-P4	month	day	dayoftheweek	year
2015	2015-12-26	6685	3320	4771	526	21191.45	21048.8	25858.82	3750.38	December	Saturday	5	2015

```
In [16]: ## Droping column unnamed as it is not usefull for us
         df.drop(columns=["Unnamed: 0"],inplace=True)
         df.sample()
```

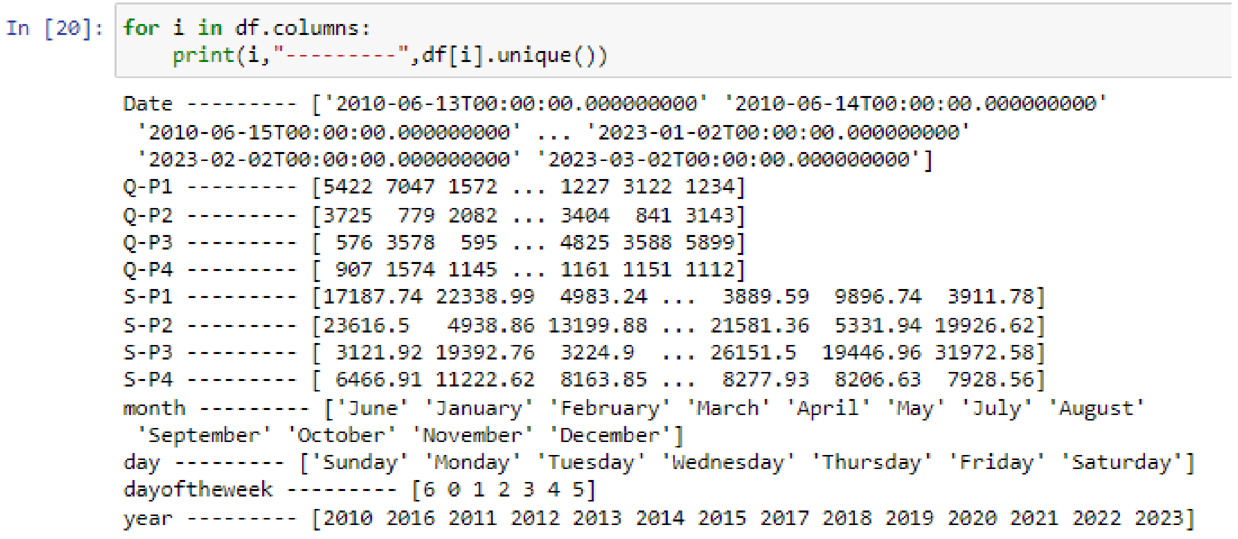
Out[16]:

	Date	Q-P1	Q-P2	Q-P3	Q-P4	S-P1	S-P2	S-P3	S-P4
2010	2015-12-21	4549	3393	1757	1351	14420.33	21511.62	9522.94	9632.63

```
In [18]: df.corr().T
```

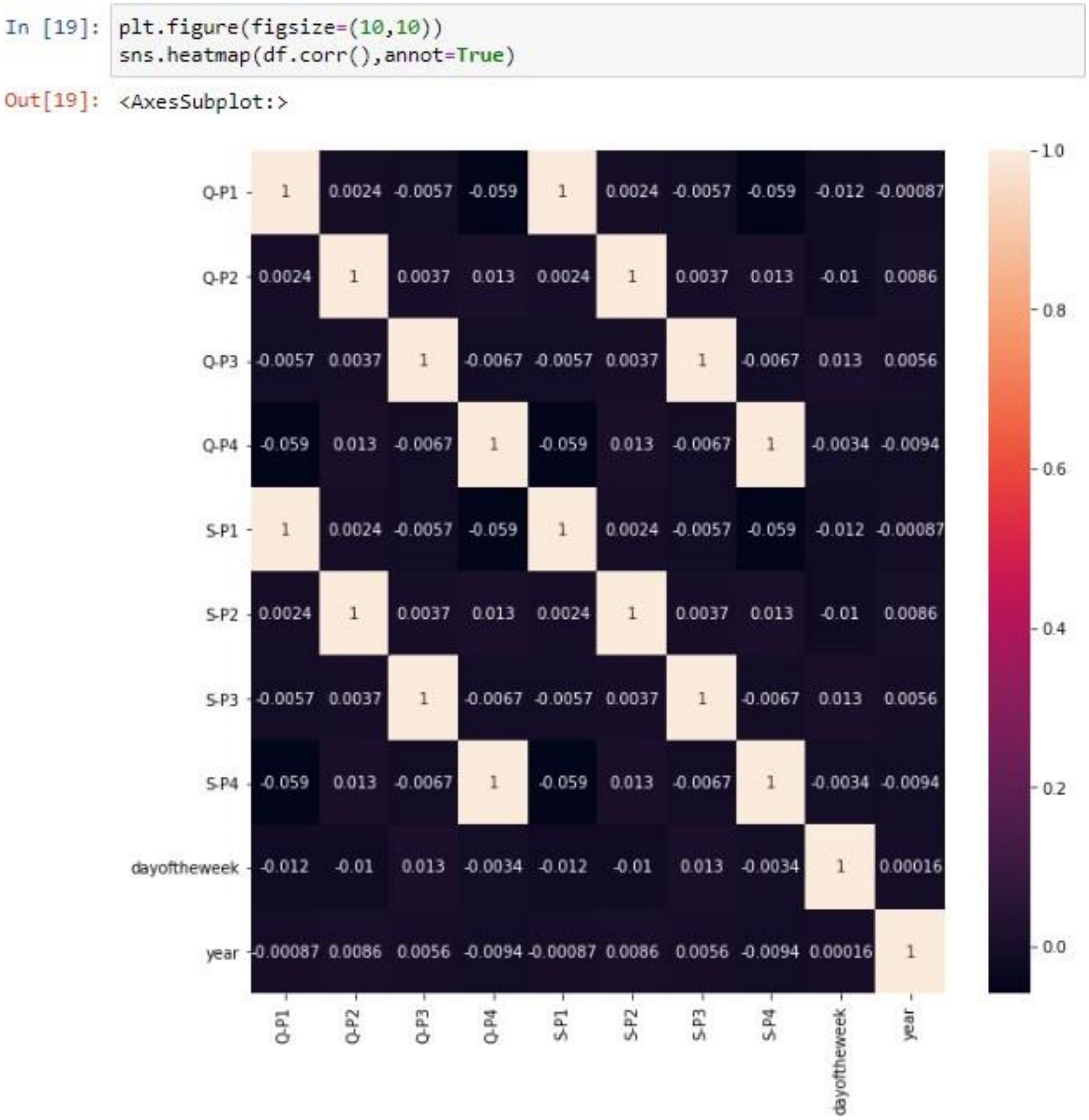
Out[18]:

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



```
In [20]: 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]
```



```
In [19]: plt.figure(figsize=(10,10))
         sns.heatmap(df.corr(),annot=True)

Out[19]: <AxesSubplot:>
```

	Q-P1	Q-P2	Q-P3	Q-P4	S-P1	S-P2	S-P3	S-P4	dayoftheweek	year
Q-P1	1	0.0024	-0.0057	-0.059	1	0.0024	-0.0057	-0.059	-0.012	-0.00087
Q-P2	0.0024	1	0.0037	0.013	0.0024	1	0.0037	0.013	-0.01	0.0086
Q-P3	-0.0057	0.0037	1	-0.0067	-0.0057	0.0037	1	-0.0067	0.013	0.0056
Q-P4	-0.059	0.013	-0.0067	1	-0.059	0.013	-0.0067	1	-0.0034	-0.0094
S-P1	1	0.0024	-0.0057	-0.059	1	0.0024	-0.0057	-0.059	-0.012	-0.00087
S-P2	0.0024	1	0.0037	0.013	0.0024	1	0.0037	0.013	-0.01	0.0086
S-P3	-0.0057	0.0037	1	-0.0067	-0.0057	0.0037	1	-0.0067	0.013	0.0056
S-P4	-0.059	0.013	-0.0067	1	-0.059	0.013	-0.0067	1	-0.0034	-0.0094
dayoftheweek	-0.012	-0.01	0.013	-0.0034	-0.012	-0.01	0.013	-0.0034	1	0.00016
year	-0.00087	0.0086	0.0056	-0.0094	-0.00087	0.0086	0.0056	-0.0094	0.00016	1

Conclusion :

the product sales analysis workflow is a vital component of any business's success. By meticulously tracking and analyzing sales data, companies can make informed decisions, identify trends, and adapt their strategies to maximize revenue and customer satisfaction. This process enables businesses to understand their products' performance, customer preferences, and market dynamics, ultimately leading to improved profitability and competitiveness.