Experiment 07:

Aim : To write a Pandas program to create a Pivot table and find the maximum and minimum sale value of the items.

Source code:

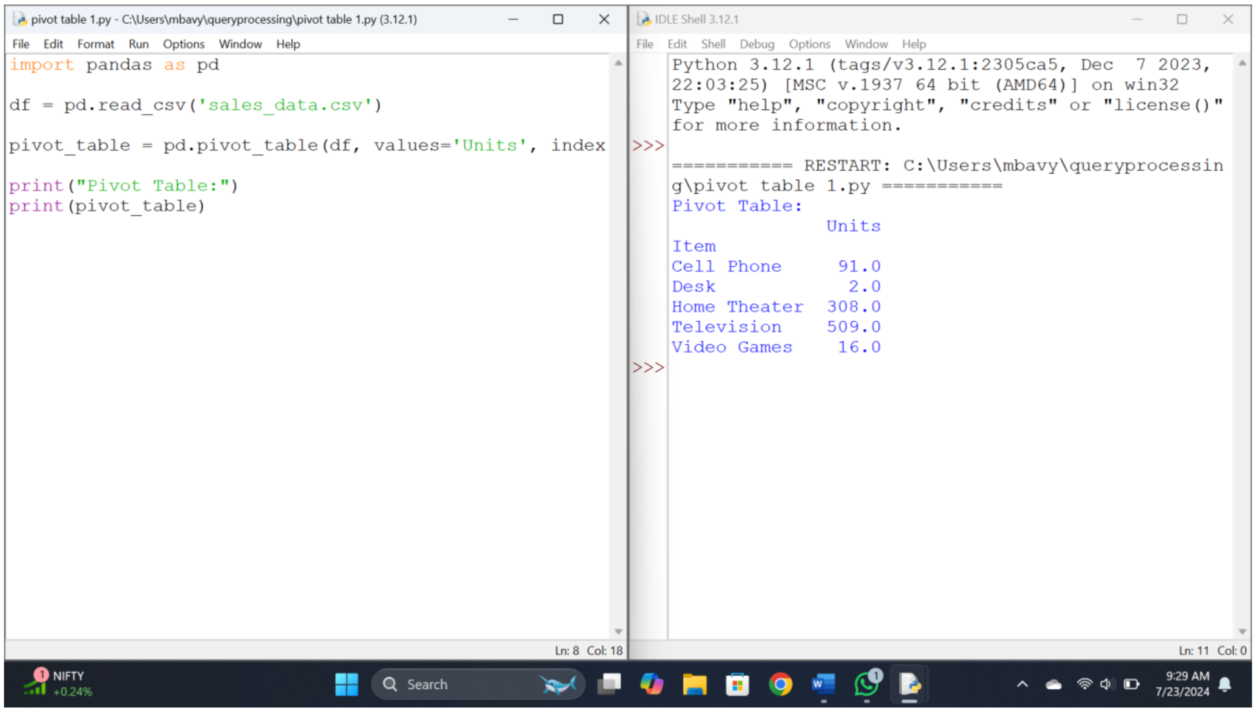
```
import pandas as pd

df = pd.read_csv('sales_data.csv')

pivot_table = pd.pivot_table(df, values='Units', index='Item', aggfunc='sum')

print("Pivot Table:")

print(pivot_table)
```

Input:

OrderDate	Region	Manager	SalesMan	Item	Units	Unit_price	Sale_amt
1-6-18	East	Martha	Alexander	Television	95	1,198.00	1,13,810.00
1-23-18	Central	Hermann	Shelli	Home Theater	50	500.00	25,000.00
2-9-18	Central	Hermann	Luis	Television	36	1,198.00	43,128.00
2-26-18	Central	Timothy	David	Cell Phone	27	225.00	6,075.00
3-15-18	West	Timothy	Stephen	Television	56	1,198.00	67,088.00
4-1-18	East	Martha	Alexander	Home Theater	60	500.00	30,000.00
4-18-18	Central	Martha	Steven	Television	75	1,198.00	89,850.00
5-5-18	Central	Hermann	Luis	Television	90	1,198.00	1,07,820.00
5-22-18	West	Douglas	Michael	Television	32	1,198.00	38,336.00
6-8-18	East	Martha	Alexander	Home Theater	60	500.00	30,000.00
6-25-18	Central	Hermann	Sigal	Television	90	1,198.00	1,07,820.00
7-12-18	East	Martha	Diana	Home Theater	29	500.00	14,500.00
7-29-18	East	Douglas	Karen	Home Theater	81	500.00	40,500.00
8-15-18	East	Martha	Alexander	Television	35	1,198.00	41,930.00
9-1-18	Central	Douglas	John	Desk	2	125.00	250.00
9-18-18	East	Martha	Alexander	Video Games	16	58.50	936.00

10-5-18	Central	Hermann	Sigal	Home Theater	28	500.00	14,000.00
10-22-18	East	Martha	Alexander	Cell Phone	64	225.00	14,400.00

Output:



```
import pandas as pd

df = pd.read_csv('sales_data.csv')

pivot_table = pd.pivot_table(df, values='Units', index

print("Pivot Table:")
print(pivot_table)
```

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 
=========== RESTART: C:\Users\mbavy\queryprocessing\pivot table 1.py ===========
Pivot Table:
              Units
Item               
Cell Phone     91.0
Desk            2.0
Home Theater  308.0
Television    509.0
Video Games    16.0
>>> 
```

Results: Thus a Pandas program to create a Pivot table and find the maximum and minimum sale value of the items is done.

Experiment 8 :

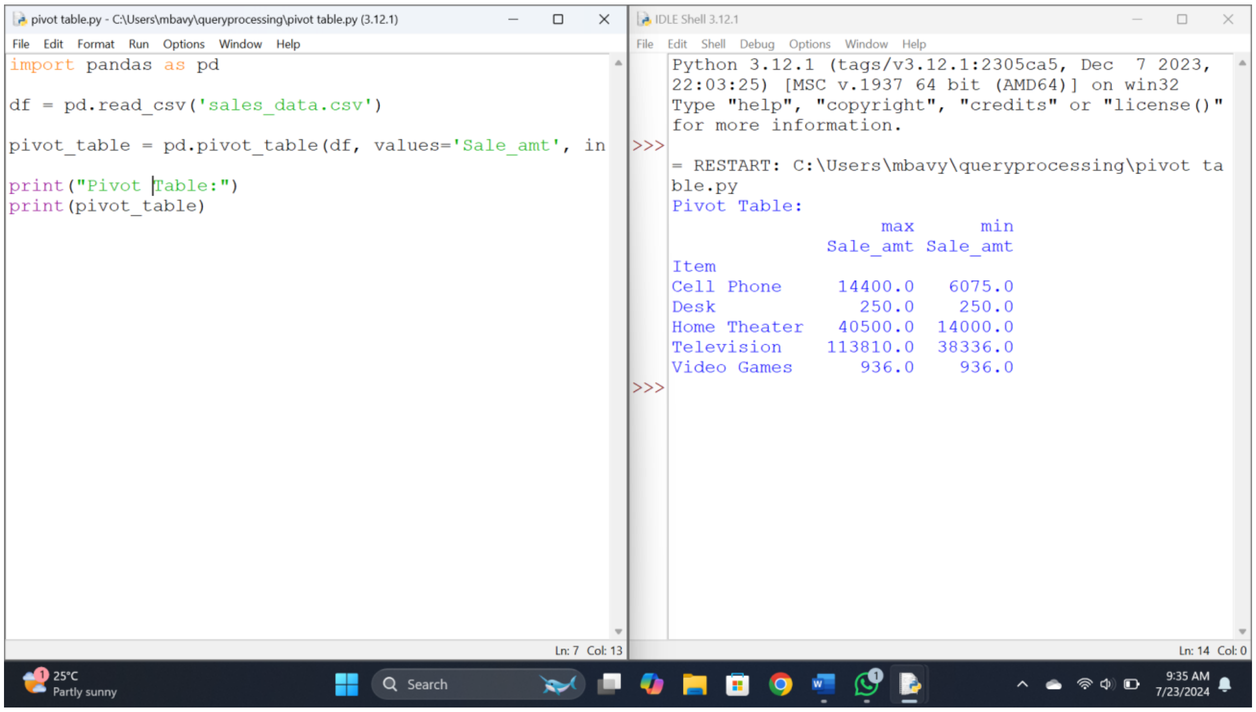
Aim :

To write a Pandas program to create a Pivot table and find the item wise unit sold.

Code:

```
import pandas as pd

df = pd.read_csv('sales_data.csv')

pivot_table = pd.pivot_table(df, values='Sale_amt', index='Item', aggfunc=['max', 'min'])

print("Pivot Table:")

print(pivot_table)
```

Output:



```
import pandas as pd

df = pd.read_csv('sales_data.csv')

pivot_table = pd.pivot_table(df, values='Sale_amt', in

print("Pivot Table:")
print(pivot_table)
```

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023,
22:03:25) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()"
for more information.
>>>
= RESTART: C:\Users\mbavy\queryprocessing\pivot ta
ble.py
Pivot Table:
                   max       min
              Sale_amt  Sale_amt
Item
Cell Phone     14400.0    6075.0
Desk             250.0     250.0
Home Theater   40500.0   14000.0
Television    113810.0   38336.0
Video Games      936.0     936.0
>>>
```

Results:

Thus a Pandas program to create a Pivot table and find the item wise unit sold is done.

Experiment 9:

Aim:

To Write a Pandas program to create a Pivot table and find the total sale amount region wise, manager wise, sales man wise.

Code:

import pandas as pd

df = pd.read_csv('sales_data.csv')

pivot_region = pd.pivot_table(df, values='Sale_amt', index='Region', aggfunc='sum')

pivot_manager = pd.pivot_table(df, values='Sale_amt', index='Manager', aggfunc='sum')

pivot_salesman = pd.pivot_table(df, values='Sale_amt', index='SalesMan', aggfunc='sum')

print("Total Sale Amount Region-wise:")

print(pivot_region)

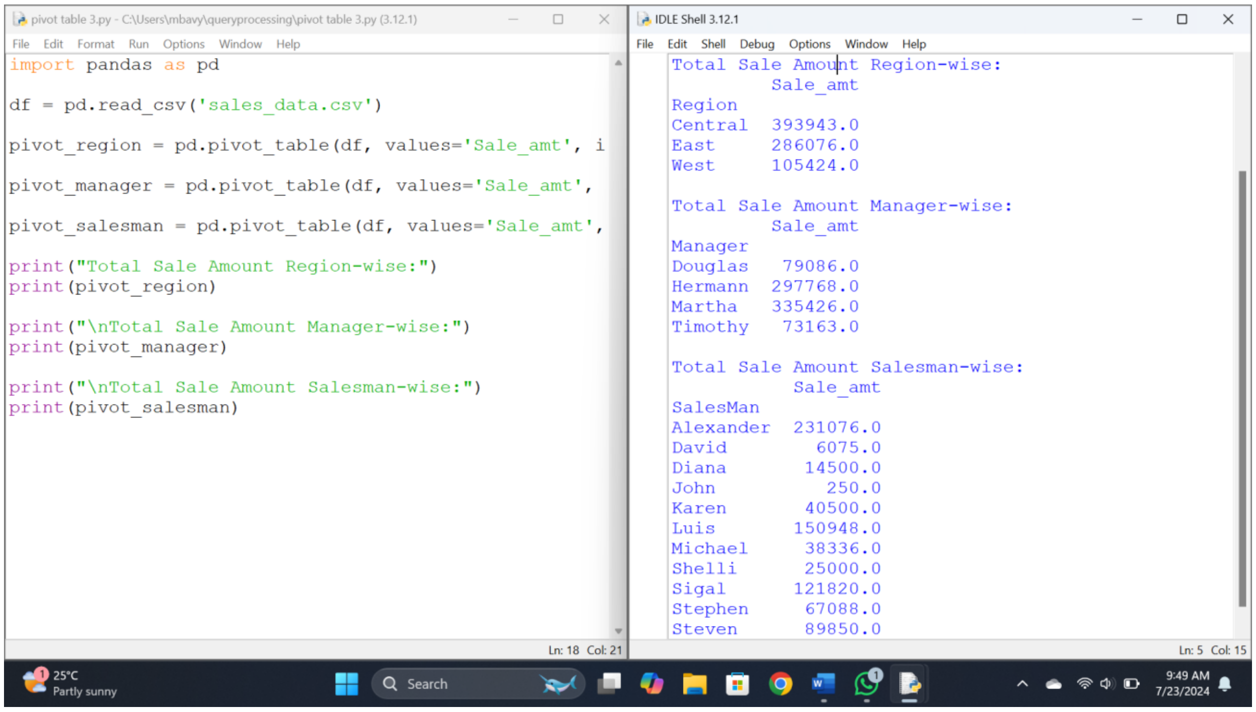
print("\nTotal Sale Amount Manager-wise:")

print(pivot_manager)

print("\nTotal Sale Amount Salesman-wise:")

print(pivot_salesman)

Output:



```
import pandas as pd

df = pd.read_csv('sales_data.csv')

pivot_region = pd.pivot_table(df, values='Sale_amt', i

pivot_manager = pd.pivot_table(df, values='Sale_amt',

pivot_salesman = pd.pivot_table(df, values='Sale_amt',

print("Total Sale Amount Region-wise:")
print(pivot_region)

print("\nTotal Sale Amount Manager-wise:")
print(pivot_manager)

print("\nTotal Sale Amount Salesman-wise:")
print(pivot_salesman)
```

```
Total Sale Amount Region-wise:
         Sale_amt
Region
Central  393943.0
East     286076.0
West     105424.0

Total Sale Amount Manager-wise:
         Sale_amt
Manager
Douglas   79086.0
Hermann  297768.0
Martha   335426.0
Timothy   73163.0

Total Sale Amount Salesman-wise:
           Sale_amt
SalesMan
Alexander  231076.0
David        6075.0
Diana       14500.0
John          250.0
Karen       40500.0
Luis       150948.0
Michael     38336.0
Shelli      25000.0
Sigal      121820.0
Stephen     67088.0
Steven      89850.0
```

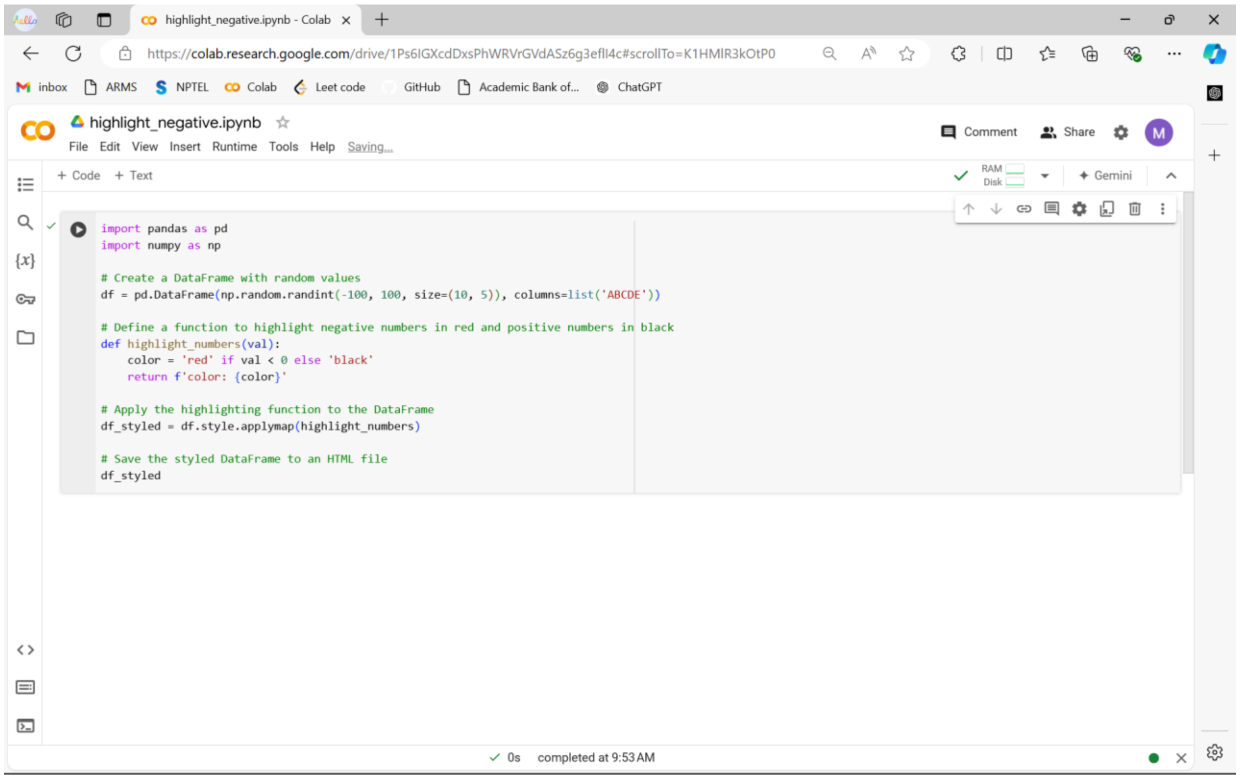
Results:

Thus a Pandas program to create a Pivot table and find the total sale amount region wise, manager wise, sales man wise is done.

Experiment 10 :

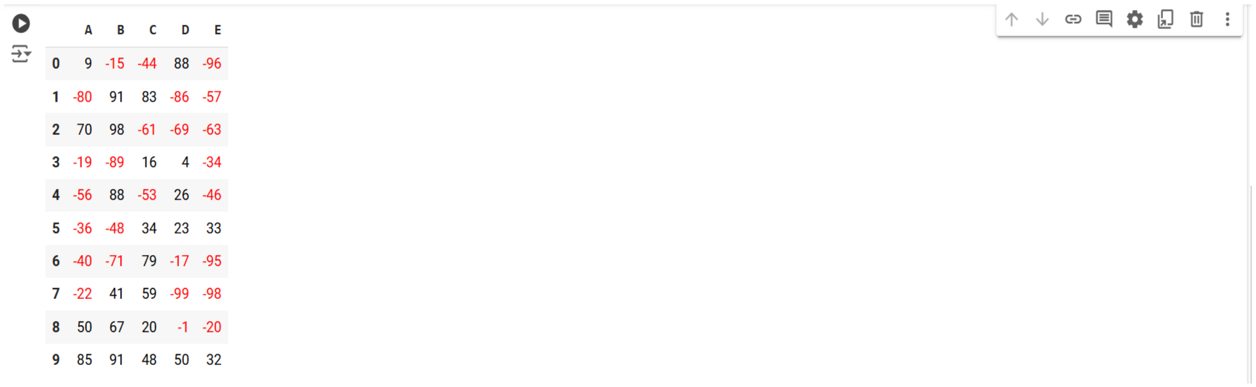
Aim: To create a data frame of ten rows, four columns with random values. Write a Pandas program to highlight the negative numbers red and positive numbers black.

Code:



```python
import pandas as pd
import numpy as np

# Create a DataFrame with random values
df = pd.DataFrame(np.random.randint(-100, 100, size=(10, 5)), columns=list('ABCDE'))

# Define a function to highlight negative numbers in red and positive numbers in black
def highlight_numbers(val):
    color = 'red' if val < 0 else 'black'
    return f'color: {color}'

# Apply the highlighting function to the DataFrame
df_styled = df.style.applymap(highlight_numbers)

# Save the styled DataFrame to an HTML file
df_styled
```

Output:



```
     A    B    C    D    E
0    9  -15  -44   88  -96
1  -80   91   83  -86  -57
2   70   98  -61  -69  -63
3  -19  -89   16    4  -34
4  -56   88  -53   26  -46
5  -36  -48   34   23   33
6  -40  -71   79  -17  -95
7  -22   41   59  -99  -98
8   50   67   20   -1  -20
9   85   91   48   50   32
```

Results :

Thus a data frame of ten rows, four columns with random values. Write a Pandas program to highlight the negative numbers red and positive numbers black.

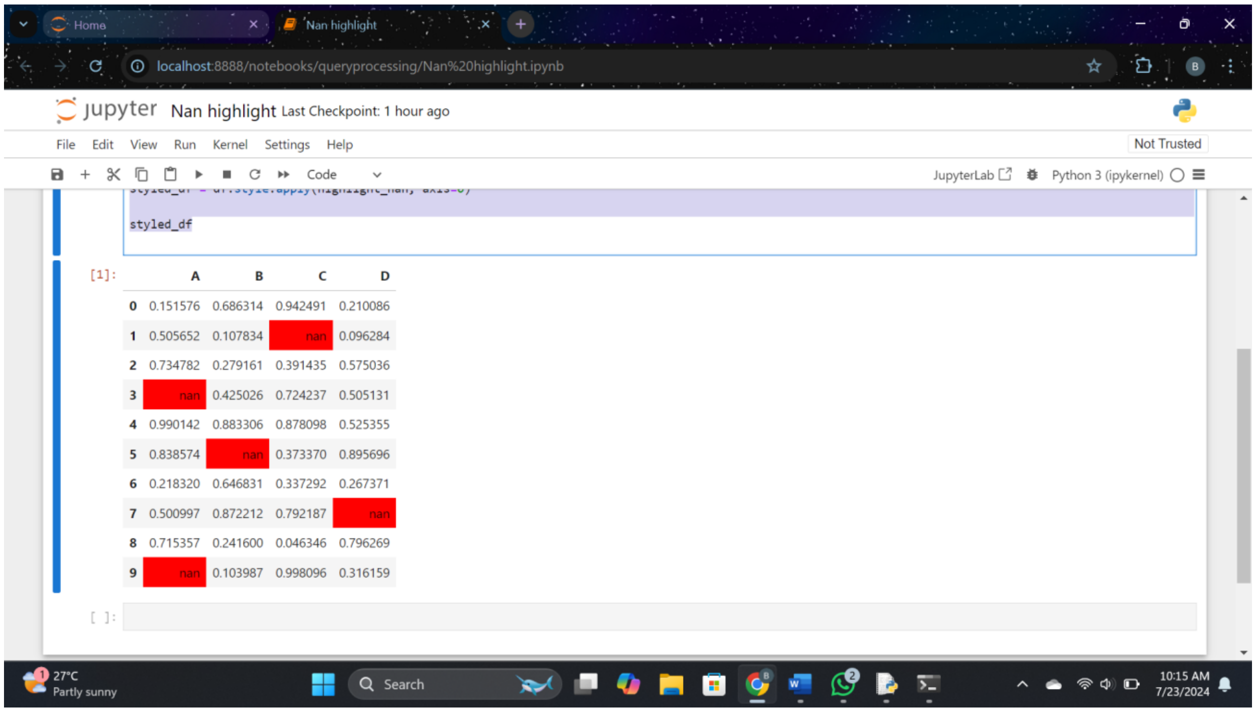
Experiment 11:

Aim: To create a data frame of ten rows, four columns with random values. Convert some values to nan values. Write a Pandas program which will highlight the nan values.

Code:

```
import pandas as pd

import numpy as np

df = pd.DataFrame(np.random.rand(10, 4), columns=['A', 'B', 'C', 'D'])

nan_indices = [(1, 2), (3, 0), (5, 1), (7, 3), (9, 0)]

for index in nan_indices:

    df.iat[index] = np.nan

def highlight_nan(s):

    is_nan = pd.isna(s)

    return ['background-color: red' if v else '' for v in is_nan]

styled_df = df.style.apply(highlight_nan, axis=0)

styled_df
```

Output:


Home
Nan highlight
localhost:8888/notebooks/queryprocessing/Nan%20highlight.ipynb
jupyter Nan highlight Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Not Trusted
Code
JupyterLab
Python 3 (ipykernel)
27°C
Partly sunny
Search
10:15 AM
7/23/2024

```
styled_df
```

[1]:

	A	B	C	D
0	0.151576	0.686314	0.942491	0.210086
1	0.505652	0.107834	nan	0.096284
2	0.734782	0.279161	0.391435	0.575036
3	nan	0.425026	0.724237	0.505131
4	0.990142	0.883306	0.878098	0.525355
5	0.838574	nan	0.373370	0.895696
6	0.218320	0.646831	0.337292	0.267371
7	0.500997	0.872212	0.792187	nan
8	0.715357	0.241600	0.046346	0.796269
9	nan	0.103987	0.998096	0.316159

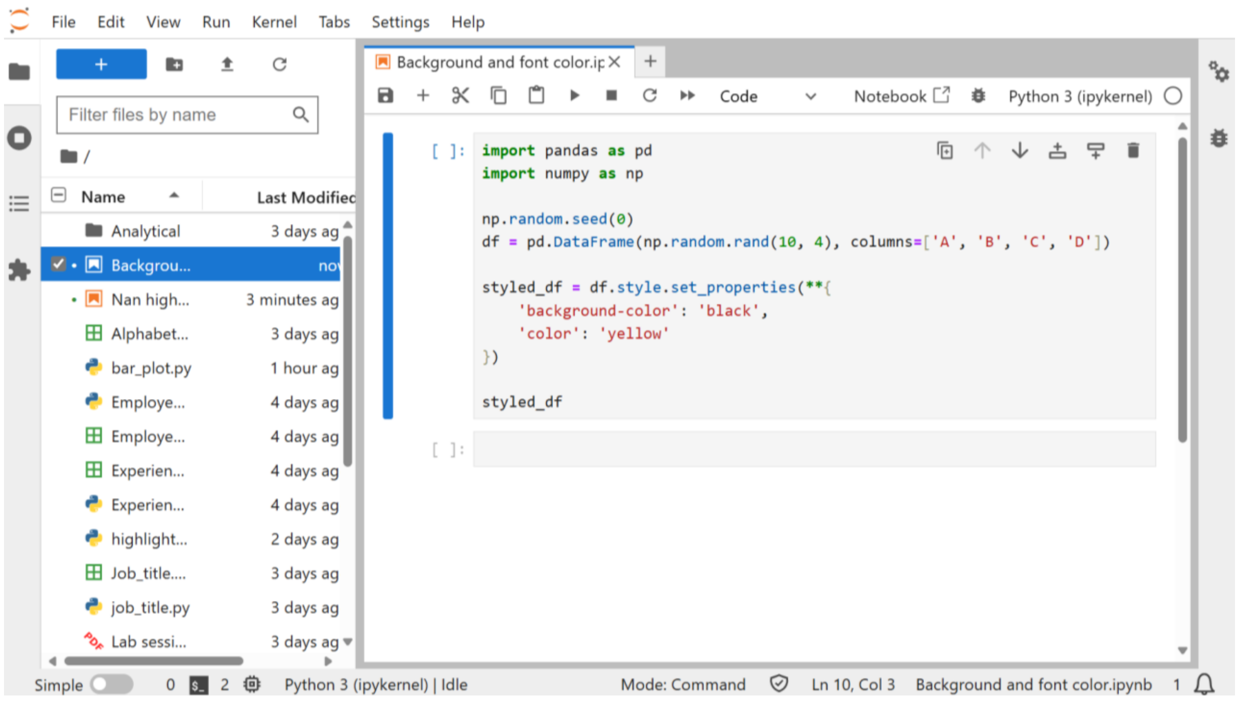
[ ]:

Results :Thus a data frame of ten rows, four columns with random values. Convert some values to nan values. Write a Pandas program which will highlight the nan values.

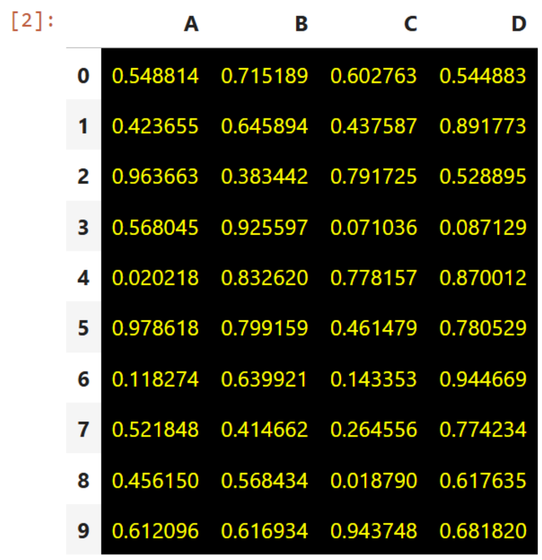
Experiment 12 :

Aim: To Create a dataframe of ten rows, four columns with random values. Write a Pandas program to set dataframe background Color black and font color yellow

Code:



```
[ ]: import pandas as pd
     import numpy as np

     np.random.seed(0)
     df = pd.DataFrame(np.random.rand(10, 4), columns=['A', 'B', 'C', 'D'])

     styled_df = df.style.set_properties(**{
         'background-color': 'black',
         'color': 'yellow'
     })

     styled_df
```


[2]:

	A	B	C	D
0	0.548814	0.715189	0.602763	0.544883
1	0.423655	0.645894	0.437587	0.891773
2	0.963663	0.383442	0.791725	0.528895
3	0.568045	0.925597	0.071036	0.087129
4	0.020218	0.832620	0.778157	0.870012
5	0.978618	0.799159	0.461479	0.780529
6	0.118274	0.639921	0.143353	0.944669
7	0.521848	0.414662	0.264556	0.774234
8	0.456150	0.568434	0.018790	0.617635
9	0.612096	0.616934	0.943748	0.681820

Output:

Results :

Thus a dataframe of ten rows, four columns with random values. Write a Pandas program to set dataframe background Color black and font color yellow.

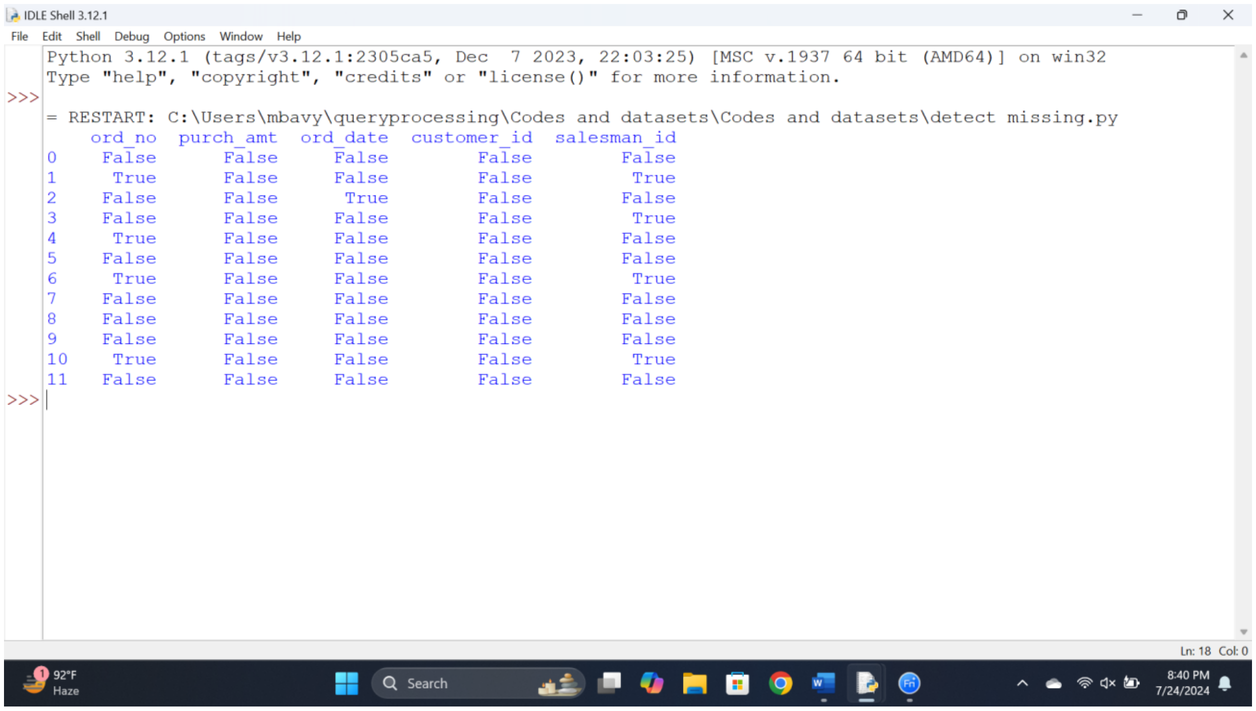
Experiment 13 :

Aim: Write a Pandas program to detect missing values of a given DataFrame. Display True or False.

Code:

```
import pandas as pd

df = pd.read_csv('sales.csv')

missing_values = df.isnull()

print(missing_values)
```

Output:



```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\mbavy\queryprocessing\Codes and datasets\Codes and datasets\detect missing.py
    ord_no  purch_amt  ord_date  customer_id  salesman_id
0    False      False     False        False        False
1     True      False     False        False         True
2    False      False      True        False        False
3    False      False     False        False         True
4     True      False     False        False        False
5    False      False     False        False        False
6     True      False     False        False         True
7    False      False     False        False        False
8    False      False     False        False        False
9    False      False     False        False        False
10    True      False     False        False         True
11   False      False     False        False        False
>>>
```

Results: Write a Pandas program to detect missing values of a given DataFrame. Display True or False.

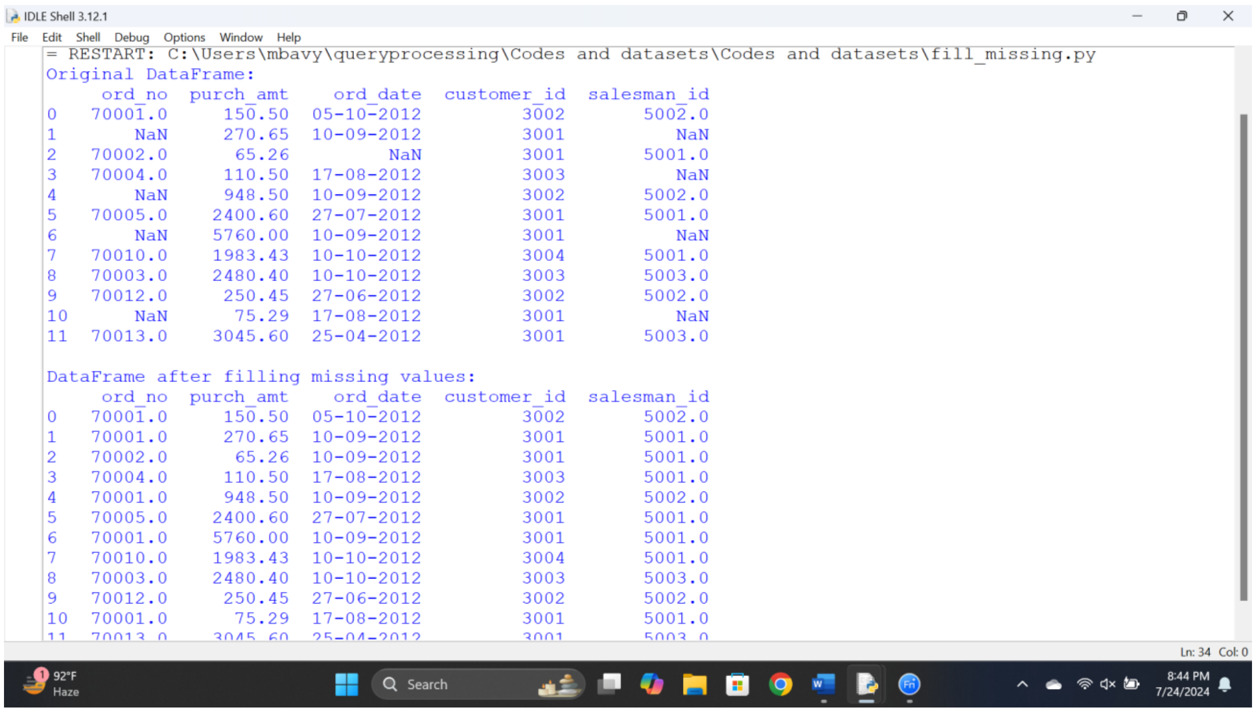
Experiment 14

Aim: To Write a Pandas program to find and replace the missing values in a given DataFrame which do not have any valuable information.

Code:

```
import pandas as pd
df = pd.read_csv('sales.csv')
print("Original DataFrame:")
print(df)
numerical_columns = ['purch_amt']
df[numerical_columns] = df[numerical_columns].fillna(df[numerical_columns].mean())
categorical_columns = ['ord_no', 'ord_date', 'customer_id', 'salesman_id']
for column in categorical_columns:
   df[column] = df[column].fillna(df[column].mode()[0])
print("\nDataFrame after filling missing values:")
print(df)
```

Output:



```
= RESTART: C:\Users\mbavy\queryprocessing\Codes and datasets\Codes and datasets\fill_missing.py
Original DataFrame:
     ord_no  purch_amt    ord_date  customer_id  salesman_id
0   70001.0     150.50  05-10-2012         3002       5002.0
1       NaN     270.65  10-09-2012         3001          NaN
2   70002.0      65.26         NaN         3001       5001.0
3   70004.0     110.50  17-08-2012         3003          NaN
4       NaN     948.50  10-09-2012         3002       5002.0
5   70005.0    2400.60  27-07-2012         3001       5001.0
6       NaN    5760.00  10-09-2012         3001          NaN
7   70010.0    1983.43  10-10-2012         3004       5001.0
8   70003.0    2480.40  10-10-2012         3003       5003.0
9   70012.0     250.45  27-06-2012         3002       5002.0
10      NaN      75.29  17-08-2012         3001          NaN
11  70013.0    3045.60  25-04-2012         3001       5003.0

DataFrame after filling missing values:
     ord_no  purch_amt    ord_date  customer_id  salesman_id
0   70001.0     150.50  05-10-2012         3002       5002.0
1   70001.0     270.65  10-09-2012         3001       5001.0
2   70002.0      65.26  10-09-2012         3001       5001.0
3   70004.0     110.50  17-08-2012         3003       5001.0
4   70001.0     948.50  10-09-2012         3002       5002.0
5   70005.0    2400.60  27-07-2012         3001       5001.0
6   70001.0    5760.00  10-09-2012         3001       5001.0
7   70010.0    1983.43  10-10-2012         3004       5001.0
8   70003.0    2480.40  10-10-2012         3003       5003.0
9   70012.0     250.45  27-06-2012         3002       5002.0
10  70001.0      75.29  17-08-2012         3001       5001.0
11  70013.0    3045.60  25-04-2012         3001       5003.0
```

Results :

Thus a Pandas program to find and replace the missing values in a given DataFrame which do not have any valuable information.

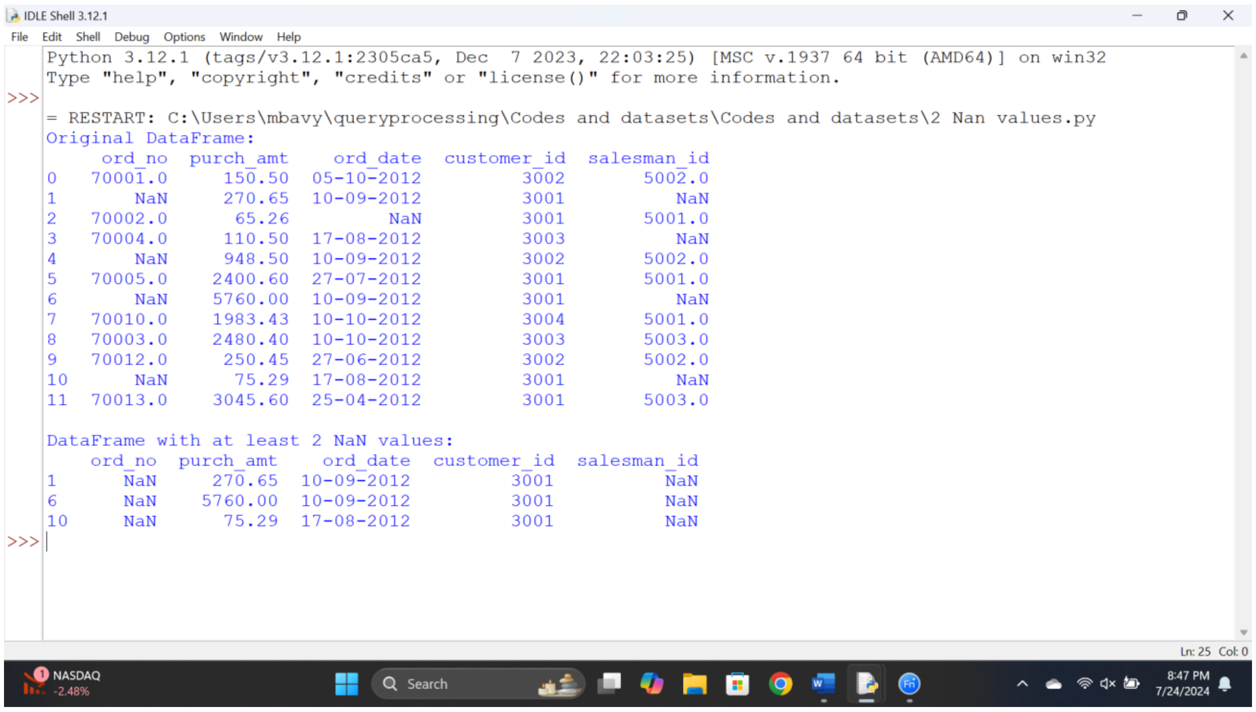
Experiment 15

Aim: To Write a Pandas program to keep the rows with at least 2 NaN values in a given DataFrame.

Code :

```
import pandas as pd

df = pd.read_csv('sales.csv')

print("Original DataFrame:")

print(df)

df_filtered = df[df.isna().sum(axis=1) >= 2]

print("\nDataFrame with at least 2 NaN values:")

print(df_filtered)

df_filtered.to_csv('filtered_sales.csv', index=False)
```

Output :



```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\mbavy\queryprocessing\Codes and datasets\Codes and datasets\2 Nan values.py
Original DataFrame:
     ord_no  purch_amt    ord_date  customer_id  salesman_id
0   70001.0     150.50  05-10-2012         3002       5002.0
1       NaN     270.65  10-09-2012         3001          NaN
2   70002.0      65.26         NaN         3001       5001.0
3   70004.0     110.50  17-08-2012         3003          NaN
4       NaN     948.50  10-09-2012         3002       5002.0
5   70005.0    2400.60  27-07-2012         3001       5001.0
6       NaN    5760.00  10-09-2012         3001          NaN
7   70010.0    1983.43  10-10-2012         3004       5001.0
8   70003.0    2480.40  10-10-2012         3003       5003.0
9   70012.0     250.45  27-06-2012         3002       5002.0
10      NaN      75.29  17-08-2012         3001          NaN
11  70013.0    3045.60  25-04-2012         3001       5003.0

DataFrame with at least 2 NaN values:
    ord_no  purch_amt    ord_date  customer_id  salesman_id
1      NaN     270.65  10-09-2012         3001          NaN
6      NaN    5760.00  10-09-2012         3001          NaN
10     NaN      75.29  17-08-2012         3001          NaN
>>>
```

Results :

Thus a Pandas program to keep the rows with at least 2 NaN values in a given DataFrame.

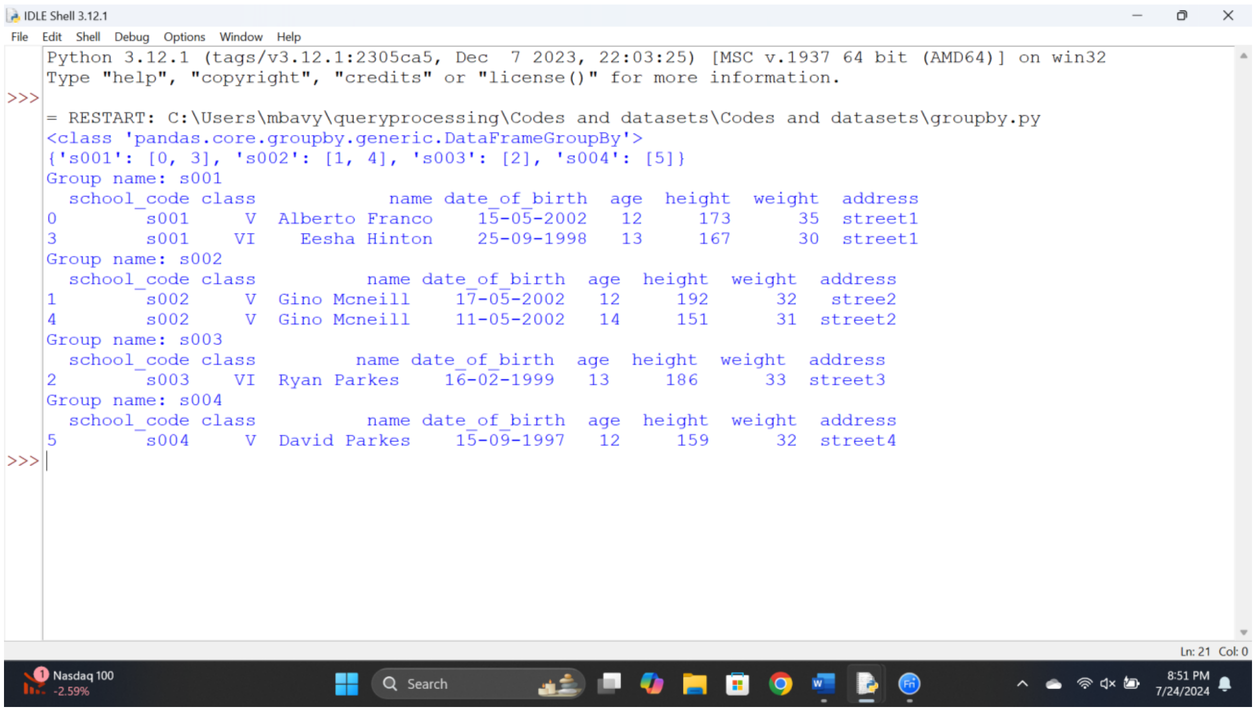
Experiment 16:

Aim: To Write a Pandas program to split the following dataframe into groups based on school code. Also check the type of GroupBy object.

Code:

```
import pandas as pd

df = pd.read_csv('school.csv')

grouped = df.groupby('school_code')

print(type(grouped))

print(grouped.groups)

for name, group in grouped:

    print(f"Group name: {name}")

    print(group)
```

Output:



```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\mbavy\queryprocessing\Codes and datasets\Codes and datasets\groupby.py
<class 'pandas.core.groupby.generic.DataFrameGroupBy'>
{'s001': [0, 3], 's002': [1, 4], 's003': [2], 's004': [5]}
Group name: s001
  school_code class             name date_of_birth  age  height  weight  address
0        s001     V  Alberto Franco    15-05-2002   12     173      35  street1
3        s001    VI    Eesha Hinton    25-09-1998   13     167      30  street1
Group name: s002
  school_code class           name date_of_birth  age  height  weight  address
1        s002     V  Gino Mcneill    17-05-2002   12     192      32   stree2
4        s002     V  Gino Mcneill    11-05-2002   14     151      31  street2
Group name: s003
  school_code class          name date_of_birth  age  height  weight  address
2        s003    VI  Ryan Parkes    16-02-1999   13     186      33  street3
Group name: s004
  school_code class           name date_of_birth  age  height  weight  address
5        s004     V  David Parkes    15-09-1997   12     159      32  street4
>>>
```

Results :

Thus a Pandas program to split the following dataframe into groups based on school code. Also check the type of GroupBy object.