بسم الله الرحمن الرحيم

Tamkeen Insurance

Health Insurance Frauds

Exploratory Data Analysis (EDA)

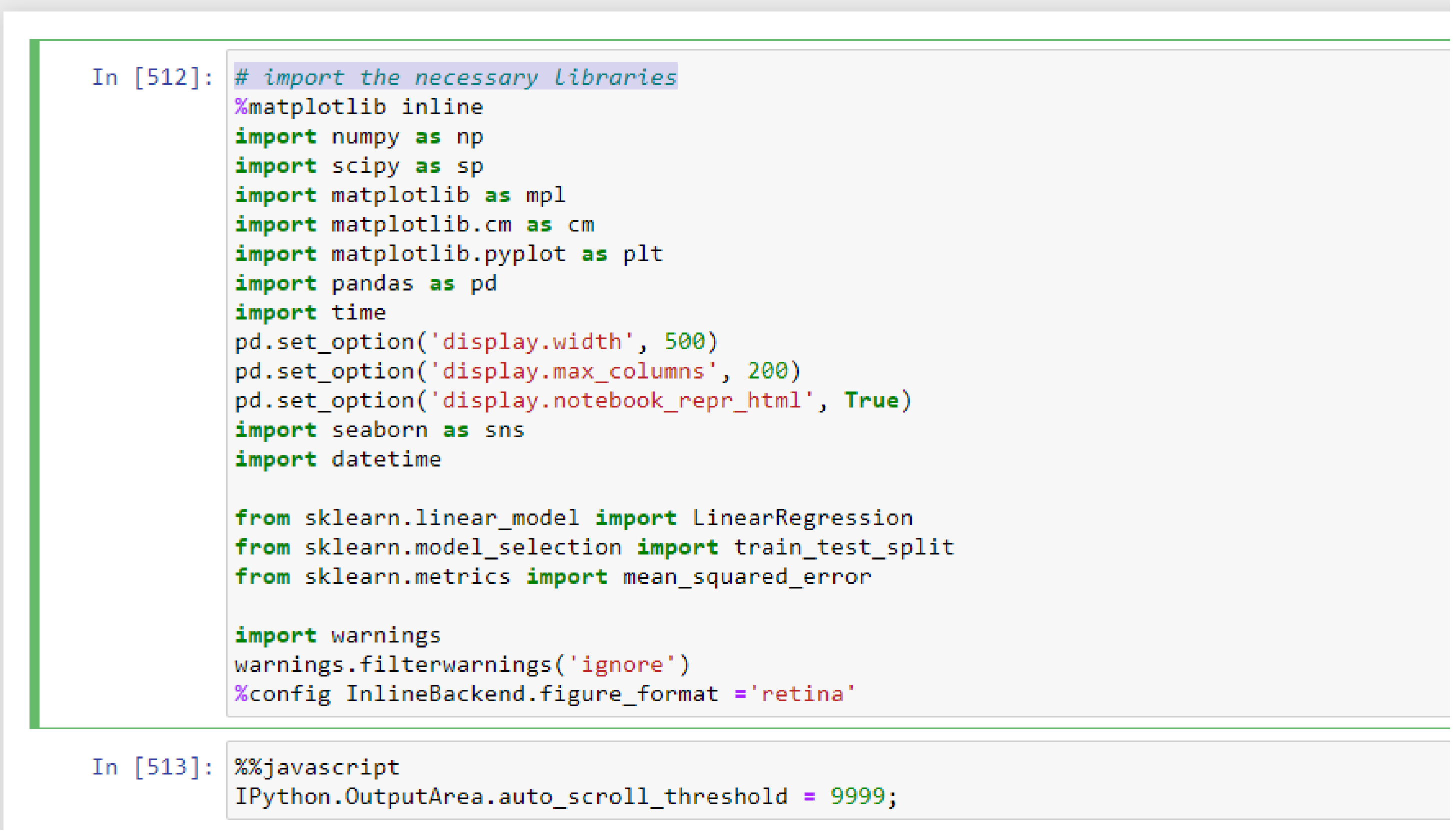
Prepared by : Date : 25/10/2021

Mohammad Dar Alsheikh

Raed Jaber

Abrar Mady

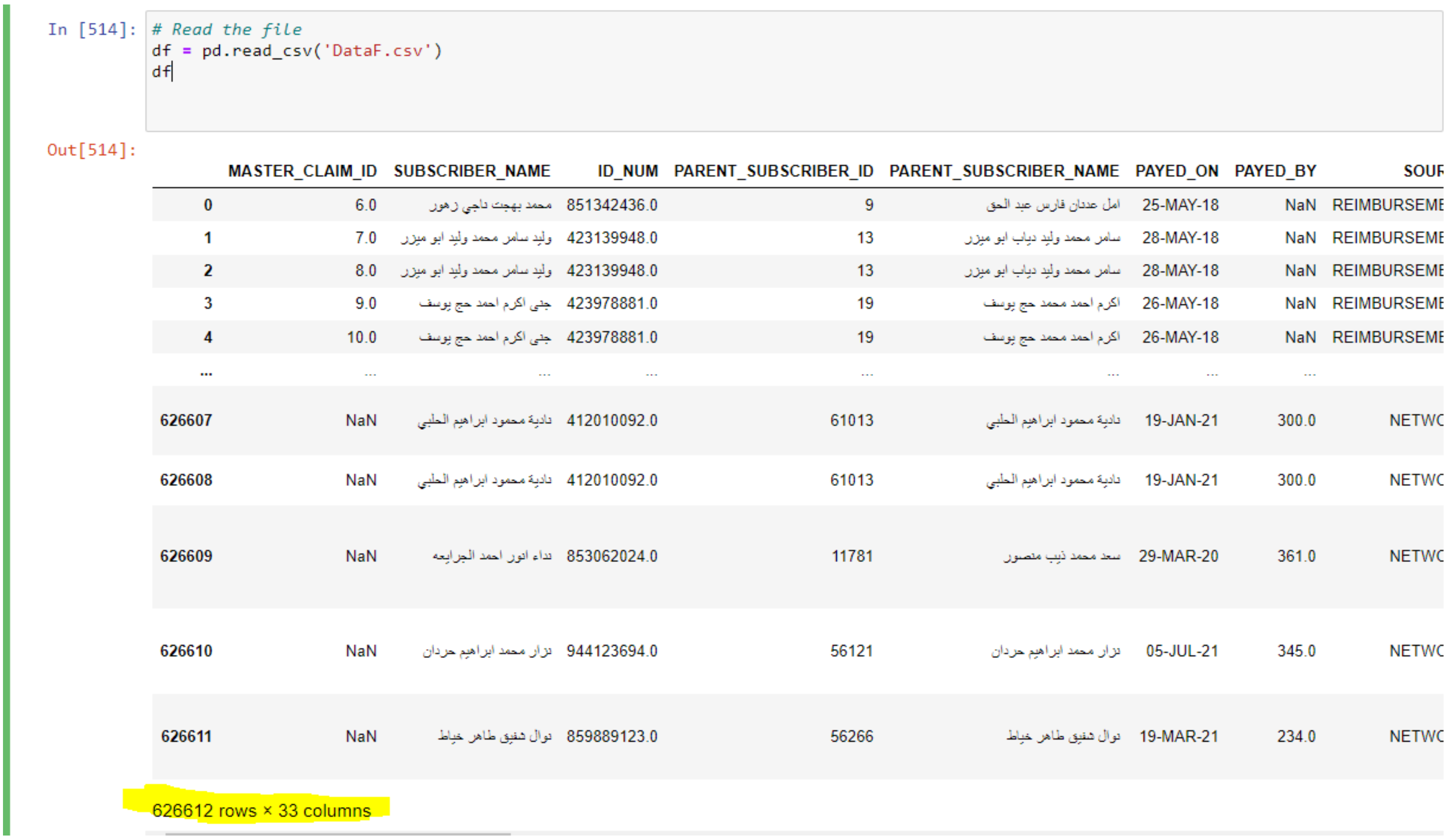
import the necessary libraries



```
In [512]: # import the necessary libraries
          %matplotlib inline
          import numpy as np
          import scipy as sp
          import matplotlib as mpl
          import matplotlib.cm as cm
          import matplotlib.pyplot as plt
          import pandas as pd
          import time
          pd.set_option('display.width', 500)
          pd.set_option('display.max_columns', 200)
          pd.set_option('display.notebook_repr_html', True)
          import seaborn as sns
          import datetime

          from sklearn.linear_model import LinearRegression
          from sklearn.model_selection import train_test_split
          from sklearn.metrics import mean_squared_error

          import warnings
          warnings.filterwarnings('ignore')
          %config InlineBackend.figure_format ='retina'
```

```
In [513]: %%javascript
          IPython.OutputArea.auto_scroll_threshold = 9999;
```

# Read the file and shown sahpe



```
In [514]: # Read the file
          df = pd.read_csv('DataF.csv')
          df
```

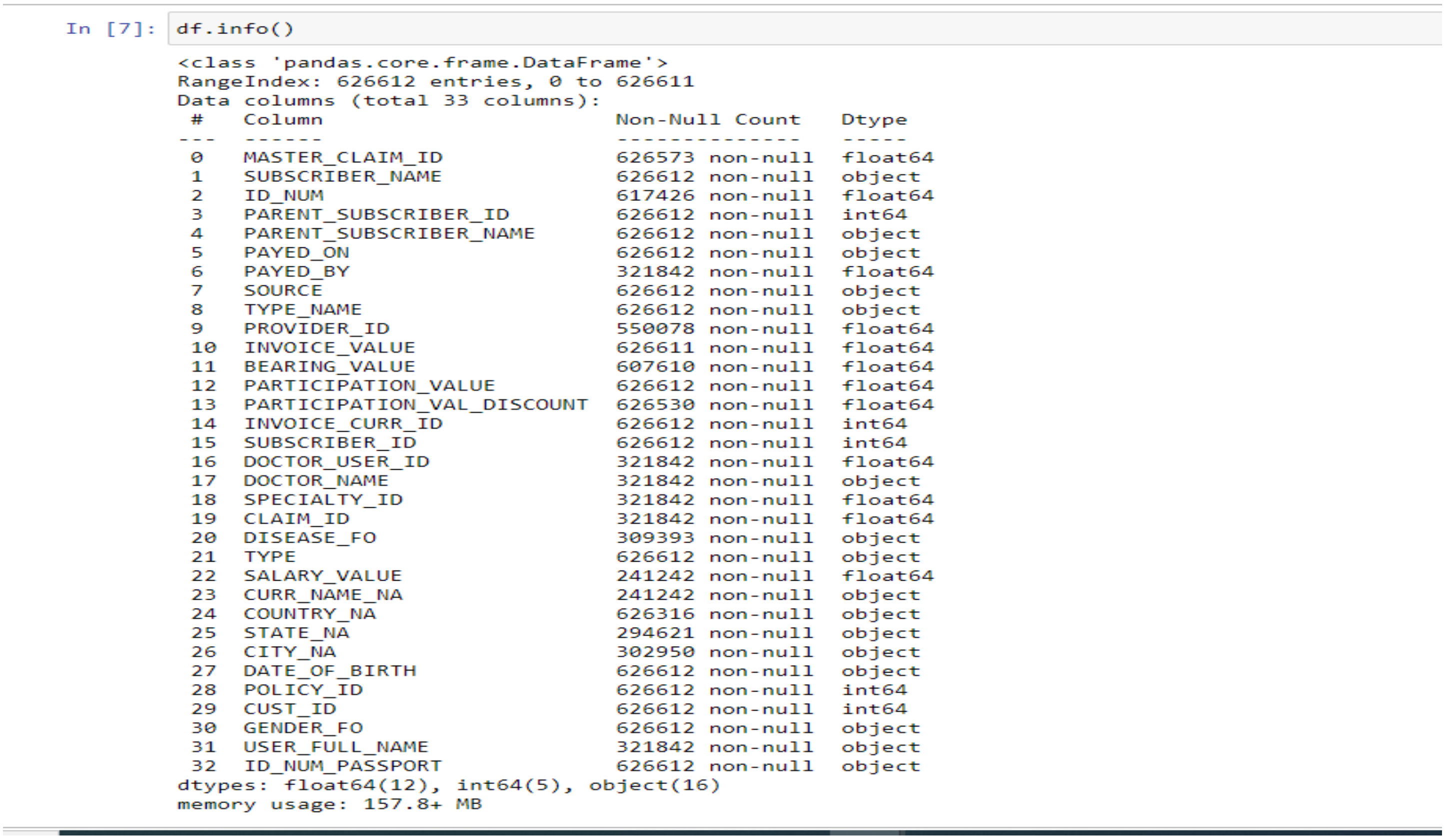
Out[514]:

	MASTER_CLAIM_ID	SUBSCRIBER_NAME	ID_NUM	PARENT_SUBSCRIBER_ID	PARENT_SUBSCRIBER_NAME	PAYED_ON	PAYED_BY	SOUR
0	6.0	محمد بهجت داجي زهور	851342436.0	9	امل عدنان فارس عبد الحق	25-MAY-18	NaN	REIMBURSEME
1	7.0	وليد سامر محمد وليد ابو ميزر	423139948.0	13	سامر محمد وليد دياب ابو ميزر	28-MAY-18	NaN	REIMBURSEME
2	8.0	وليد سامر محمد وليد ابو ميزر	423139948.0	13	سامر محمد وليد دياب ابو ميزر	28-MAY-18	NaN	REIMBURSEME
3	9.0	جنى اكرم احمد حج يوسف	423978881.0	19	اكرم احمد محمد حج يوسف	26-MAY-18	NaN	REIMBURSEME
4	10.0	جنى اكرم احمد حج يوسف	423978881.0	19	اكرم احمد محمد حج يوسف	26-MAY-18	NaN	REIMBURSEME
...	...	...	...	...	...	...	...	
626607	NaN	نادية محمود ابراهيم الحلبي	412010092.0	61013	نادية محمود ابراهيم الحلبي	19-JAN-21	300.0	NETWC
626608	NaN	نادية محمود ابراهيم الحلبي	412010092.0	61013	نادية محمود ابراهيم الحلبي	19-JAN-21	300.0	NETWC
626609	NaN	نداء انور احمد الجرايعه	853062024.0	11781	سعد محمد ذيب منصور	29-MAR-20	361.0	NETWC
626610	NaN	نزار محمد ابراهيم حردان	944123694.0	56121	نزار محمد ابراهيم حردان	05-JUL-21	345.0	NETWC
626611	NaN	نوال شفيق طاهر خياط	859889123.0	56266	نوال شفيق طاهر خياط	19-MAR-21	234.0	NETWC

626612 rows × 33 columns

The shape of data above is **626,612** rows and **33** column

# show the data information



```
In [7]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 626612 entries, 0 to 626611
Data columns (total 33 columns):
 #   Column                       Non-Null Count   Dtype  
---  ------                       --------------   -----  
 0   MASTER_CLAIM_ID              626573 non-null  float64
 1   SUBSCRIBER_NAME              626612 non-null  object 
 2   ID_NUM                       617426 non-null  float64
 3   PARENT_SUBSCRIBER_ID         626612 non-null  int64  
 4   PARENT_SUBSCRIBER_NAME       626612 non-null  object 
 5   PAYED_ON                     626612 non-null  object 
 6   PAYED_BY                     321842 non-null  float64
 7   SOURCE                       626612 non-null  object 
 8   TYPE_NAME                    626612 non-null  object 
 9   PROVIDER_ID                  550078 non-null  float64
 10  INVOICE_VALUE                626611 non-null  float64
 11  BEARING_VALUE                607610 non-null  float64
 12  PARTICIPATION_VALUE          626612 non-null  float64
 13  PARTICIPATION_VAL_DISCOUNT   626530 non-null  float64
 14  INVOICE_CURR_ID              626612 non-null  int64  
 15  SUBSCRIBER_ID                626612 non-null  int64  
 16  DOCTOR_USER_ID               321842 non-null  float64
 17  DOCTOR_NAME                  321842 non-null  object 
 18  SPECIALTY_ID                 321842 non-null  float64
 19  CLAIM_ID                     321842 non-null  float64
 20  DISEASE_FO                   309393 non-null  object 
 21  TYPE                         626612 non-null  object 
 22  SALARY_VALUE                 241242 non-null  float64
 23  CURR_NAME_NA                 241242 non-null  object 
 24  COUNTRY_NA                   626316 non-null  object 
 25  STATE_NA                     294621 non-null  object 
 26  CITY_NA                      302950 non-null  object 
 27  DATE_OF_BIRTH                626612 non-null  object 
 28  POLICY_ID                    626612 non-null  int64  
 29  CUST_ID                      626612 non-null  int64  
 30  GENDER_FO                    626612 non-null  object 
 31  USER_FULL_NAME               321842 non-null  object 
 32  ID_NUM_PASSPORT              626612 non-null  object 
dtypes: float64(12), int64(5), object(16)
memory usage: 157.8+ MB
```

Explain the data as following :

MASTER_CLAIM_ID : ...... رقم العيادة او الصيدلية او المختبر الخ

SUBSCRIBER_NAME : اسم المشترك

ID_NUM: رقم هوية المشترك

PARENT_SUBSCRIBER_ID : رقم المشترك الرئيسي (الموظف او الموظفة)

PARENT_SUBSCRIBER_NAME : اسم المشترك الرئيسي

PAYED_ON : تاريخ صرف المطالبة

PAYED_BY : تم الصرف بواسطة

SOURCE : مصدر المطالبة

TYPE_NAME : ... اسم الدواء او الاشعة او المختبر الخ

PROVIDER_ID : رقم المورد

INVOICE_VALUE : قيمة الفاتورة

BEARING_VALUE: التحمل على المريض

PARTICIPATION_VALUE : المبلغ المدفوع قبل الخصم

PARTICIPATION_VALUE_DISCOUNT: المبلغ المدفوع النهائي

INVOICE_CURR_ID : عملة الفاتورة

SUBSCRIBER_ID : رقم المشترك

DOCTOR_USER_ID : رقم الدكتور

DOCTOR_NAME : اسم الدكتور

SPECIALTY_ID : تخصص الطبيب

CLAIM_ID : رقم العيادة (الدكتور)

DISEASE_FO : التشخيص للمريض

TYPE : النوع (عيادة ، مختبر ، اشعة ، صيدلية)

SALARY_VALUE : الراتب

CURR_NAME _NA : عملة الراتب

COUNTRY_NA , STATE_NA , CITY_NA : االمنطقة الجغرافية (الدولة / المحافظة / المدينة)

DATE_OF_BIRTH: تاريخ الميلاد

POLICY_ID : رقم الوثيقة

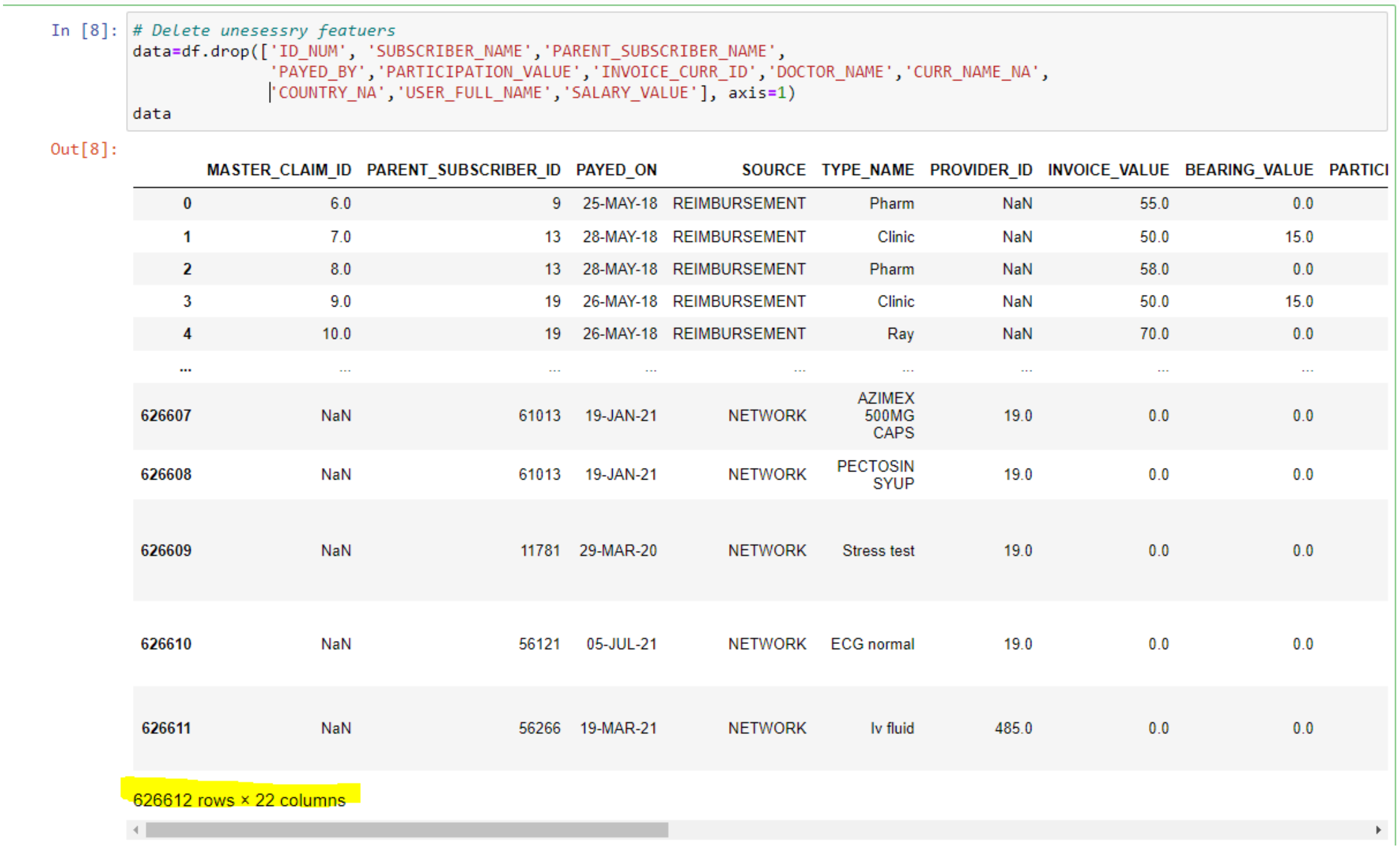
CUST_ID : رقم العميل (المؤمن)

GENDER_FO : الجنس

USER_FULL_NAME : اسم المورد

ID_NUM_PASSPORT: رقم الهوية او جواز السفر

# Drop unesessry column (fearuers) and show shape



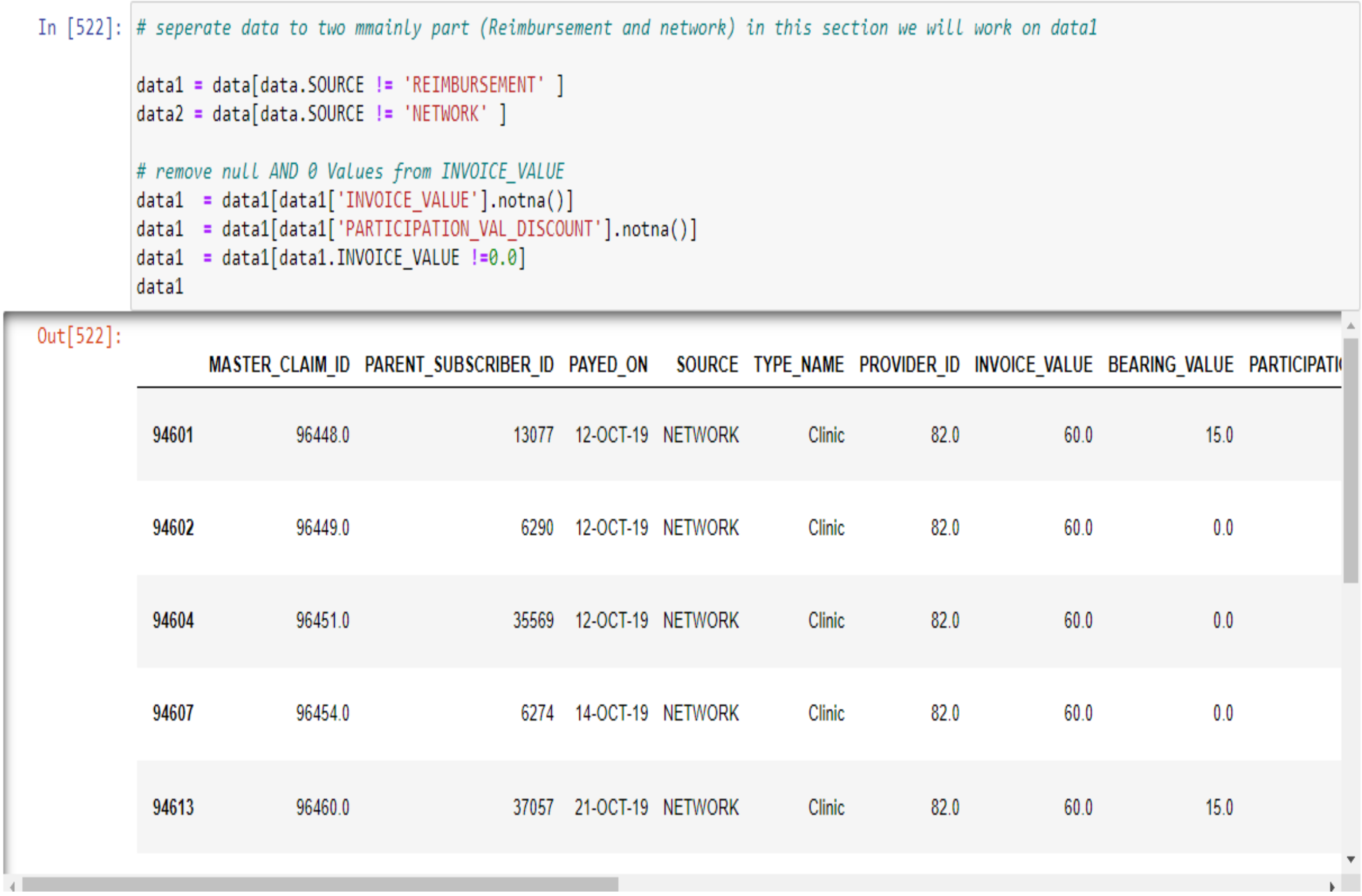
In [8]:
```
# Delete unesessry featuers
data=df.drop(['ID_NUM', 'SUBSCRIBER_NAME','PARENT_SUBSCRIBER_NAME',
            'PAYED_BY','PARTICIPATION_VALUE','INVOICE_CURR_ID','DOCTOR_NAME','CURR_NAME_NA',
            'COUNTRY_NA','USER_FULL_NAME','SALARY_VALUE'], axis=1)
data
```

Out[8]:

	MASTER_CLAIM_ID	PARENT_SUBSCRIBER_ID	PAYED_ON	SOURCE	TYPE_NAME	PROVIDER_ID	INVOICE_VALUE	BEARING_VALUE	PARTICI
0	6.0	9	25-MAY-18	REIMBURSEMENT	Pharm	NaN	55.0	0.0	
1	7.0	13	28-MAY-18	REIMBURSEMENT	Clinic	NaN	50.0	15.0	
2	8.0	13	28-MAY-18	REIMBURSEMENT	Pharm	NaN	58.0	0.0	
3	9.0	19	26-MAY-18	REIMBURSEMENT	Clinic	NaN	50.0	15.0	
4	10.0	19	26-MAY-18	REIMBURSEMENT	Ray	NaN	70.0	0.0	
...	...	...	...	...	...	...	...	...	
626607	NaN	61013	19-JAN-21	NETWORK	AZIMEX 500MG CAPS	19.0	0.0	0.0	
626608	NaN	61013	19-JAN-21	NETWORK	PECTOSIN SYUP	19.0	0.0	0.0	
626609	NaN	11781	29-MAR-20	NETWORK	Stress test	19.0	0.0	0.0	
626610	NaN	56121	05-JUL-21	NETWORK	ECG normal	19.0	0.0	0.0	
626611	NaN	56266	19-MAR-21	NETWORK	Iv fluid	485.0	0.0	0.0	

626612 rows × 22 columns

# separate data to two mainly part (Reimbursement and network) based on source column , in this section we will work on data1

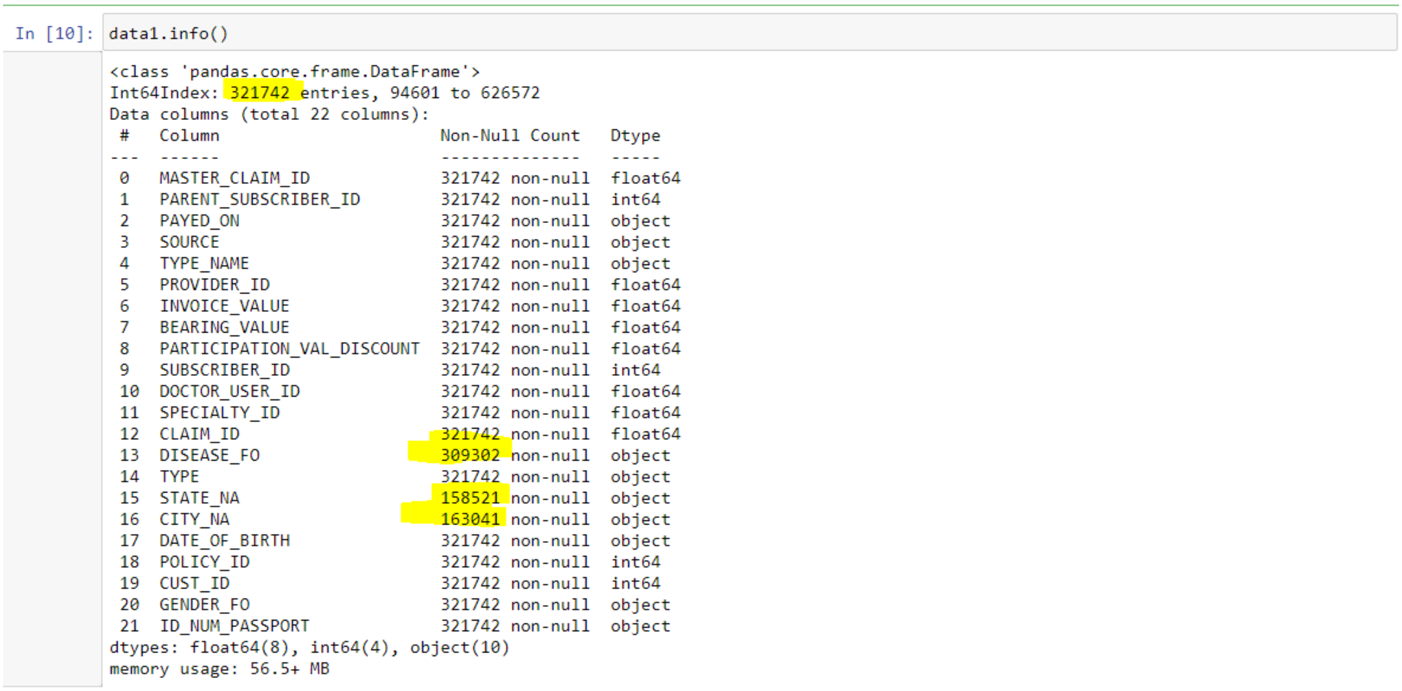


In [522]:
```
# seperate data to two mmainly part (Reimbursement and network) in this section we will work on data1

data1 = data[data.SOURCE != 'REIMBURSEMENT' ]
data2 = data[data.SOURCE != 'NETWORK' ]

# remove null AND 0 Values from INVOICE_VALUE
data1  = data1[data1['INVOICE_VALUE'].notna()]
data1  = data1[data1['PARTICIPATION_VAL_DISCOUNT'].notna()]
data1  = data1[data1.INVOICE_VALUE !=0.0]
data1
```

Out[522]:

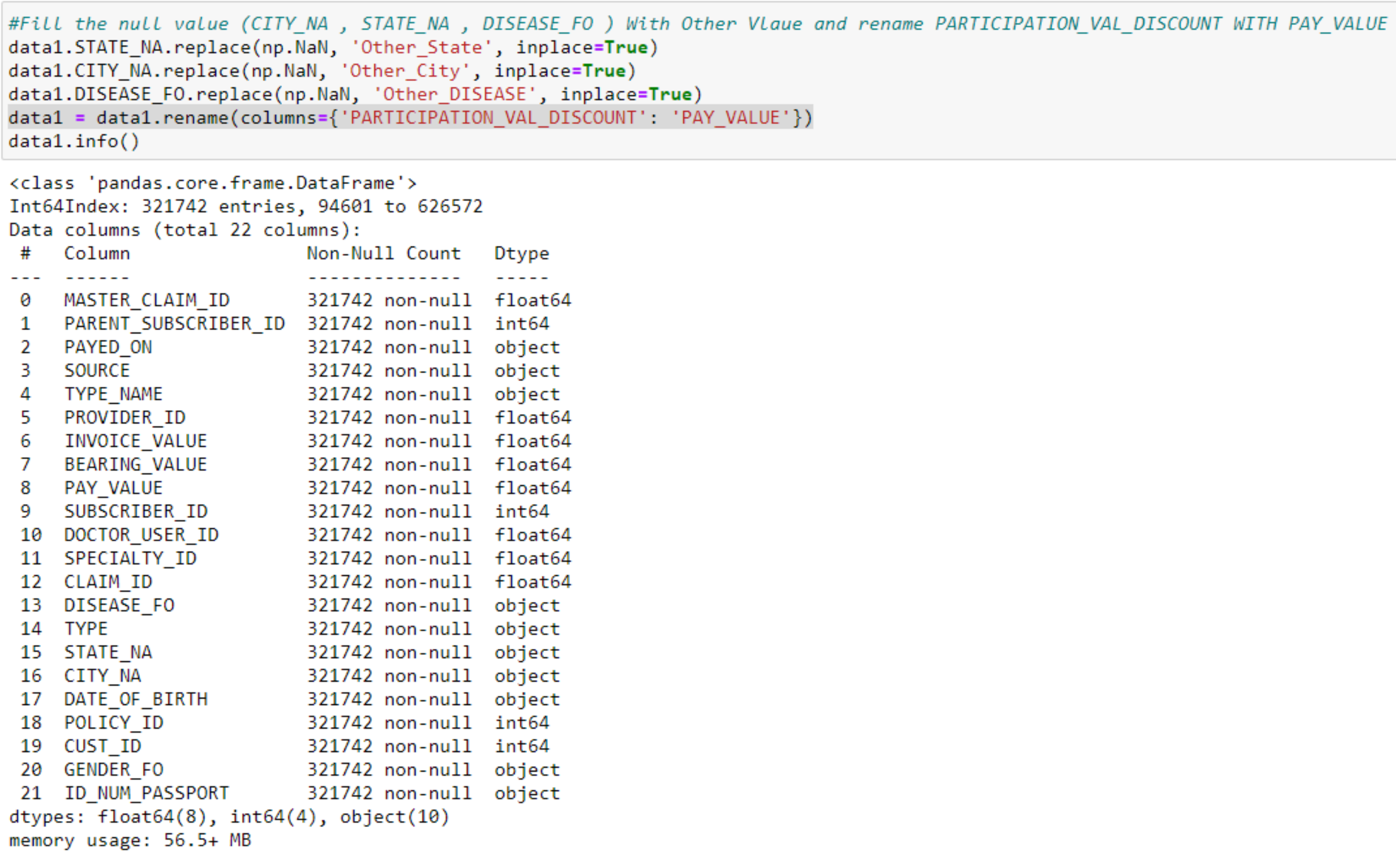
	MASTER_CLAIM_ID	PARENT_SUBSCRIBER_ID	PAYED_ON	SOURCE	TYPE_NAME	PROVIDER_ID	INVOICE_VALUE	BEARING_VALUE	PARTICIPATI
94601	96448.0	13077	12-OCT-19	NETWORK	Clinic	82.0	60.0	15.0	
94602	96449.0	6290	12-OCT-19	NETWORK	Clinic	82.0	60.0	0.0	
94604	96451.0	35569	12-OCT-19	NETWORK	Clinic	82.0	60.0	0.0	
94607	96454.0	6274	14-OCT-19	NETWORK	Clinic	82.0	60.0	0.0	
94613	96460.0	37057	21-OCT-19	NETWORK	Clinic	82.0	60.0	15.0	

# Read data information to shw null value and datatype



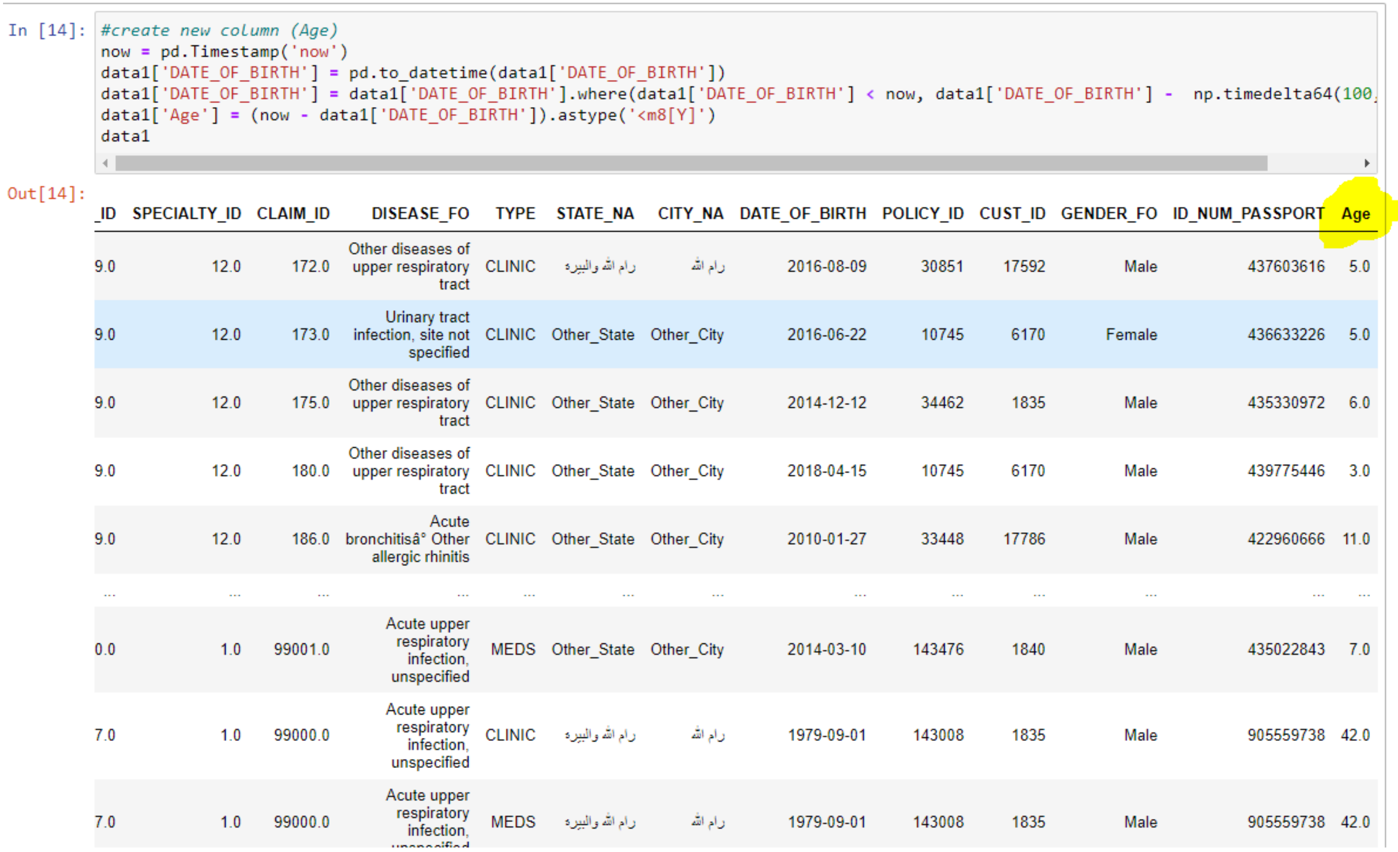
```
In [10]: data1.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 321742 entries, 94601 to 626572
Data columns (total 22 columns):
 #   Column                      Non-Null Count   Dtype  
---  ------                      --------------   -----  
 0   MASTER_CLAIM_ID             321742 non-null  float64
 1   PARENT_SUBSCRIBER_ID        321742 non-null  int64  
 2   PAYED_ON                    321742 non-null  object 
 3   SOURCE                      321742 non-null  object 
 4   TYPE_NAME                   321742 non-null  object 
 5   PROVIDER_ID                 321742 non-null  float64
 6   INVOICE_VALUE               321742 non-null  float64
 7   BEARING_VALUE               321742 non-null  float64
 8   PARTICIPATION_VAL_DISCOUNT  321742 non-null  float64
 9   SUBSCRIBER_ID               321742 non-null  int64  
 10  DOCTOR_USER_ID              321742 non-null  float64
 11  SPECIALTY_ID                321742 non-null  float64
 12  CLAIM_ID                    321742 non-null  float64
 13  DISEASE_FO                  309302 non-null  object 
 14  TYPE                        321742 non-null  object 
 15  STATE_NA                    158521 non-null  object 
 16  CITY_NA                     163041 non-null  object 
 17  DATE_OF_BIRTH               321742 non-null  object 
 18  POLICY_ID                   321742 non-null  int64  
 19  CUST_ID                     321742 non-null  int64  
 20  GENDER_FO                   321742 non-null  object 
 21  ID_NUM_PASSPORT             321742 non-null  object 
dtypes: float64(8), int64(4), object(10)
memory usage: 56.5+ MB
```

Fill the null value (CITY_NA , STATE_NA , DISEASE_FO) With Other Vlaue and rename PARTICIPATION_VAL_DISCOUNT WITH PAY_VALUE



```
#Fill the null value (CITY_NA , STATE_NA , DISEASE_FO ) With Other Vlaue and rename PARTICIPATION_VAL_DISCOUNT WITH PAY_VALUE
data1.STATE_NA.replace(np.NaN, 'Other_State', inplace=True)
data1.CITY_NA.replace(np.NaN, 'Other_City', inplace=True)
data1.DISEASE_FO.replace(np.NaN, 'Other_DISEASE', inplace=True)
data1 = data1.rename(columns={'PARTICIPATION_VAL_DISCOUNT': 'PAY_VALUE'})
data1.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 321742 entries, 94601 to 626572
Data columns (total 22 columns):
 #   Column                Non-Null Count   Dtype  
---  ------                --------------   -----  
 0   MASTER_CLAIM_ID       321742 non-null  float64
 1   PARENT_SUBSCRIBER_ID  321742 non-null  int64  
 2   PAYED_ON              321742 non-null  object 
 3   SOURCE                321742 non-null  object 
 4   TYPE_NAME             321742 non-null  object 
 5   PROVIDER_ID           321742 non-null  float64
 6   INVOICE_VALUE         321742 non-null  float64
 7   BEARING_VALUE         321742 non-null  float64
 8   PAY_VALUE             321742 non-null  float64
 9   SUBSCRIBER_ID         321742 non-null  int64  
 10  DOCTOR_USER_ID        321742 non-null  float64
 11  SPECIALTY_ID          321742 non-null  float64
 12  CLAIM_ID              321742 non-null  float64
 13  DISEASE_FO            321742 non-null  object 
 14  TYPE                  321742 non-null  object 
 15  STATE_NA              321742 non-null  object 
 16  CITY_NA               321742 non-null  object 
 17  DATE_OF_BIRTH         321742 non-null  object 
 18  POLICY_ID             321742 non-null  int64  
 19  CUST_ID               321742 non-null  int64  
 20  GENDER_FO             321742 non-null  object 
 21  ID_NUM_PASSPORT       321742 non-null  object 
dtypes: float64(8), int64(4), object(10)
memory usage: 56.5+ MB
```

# Add a new column Age

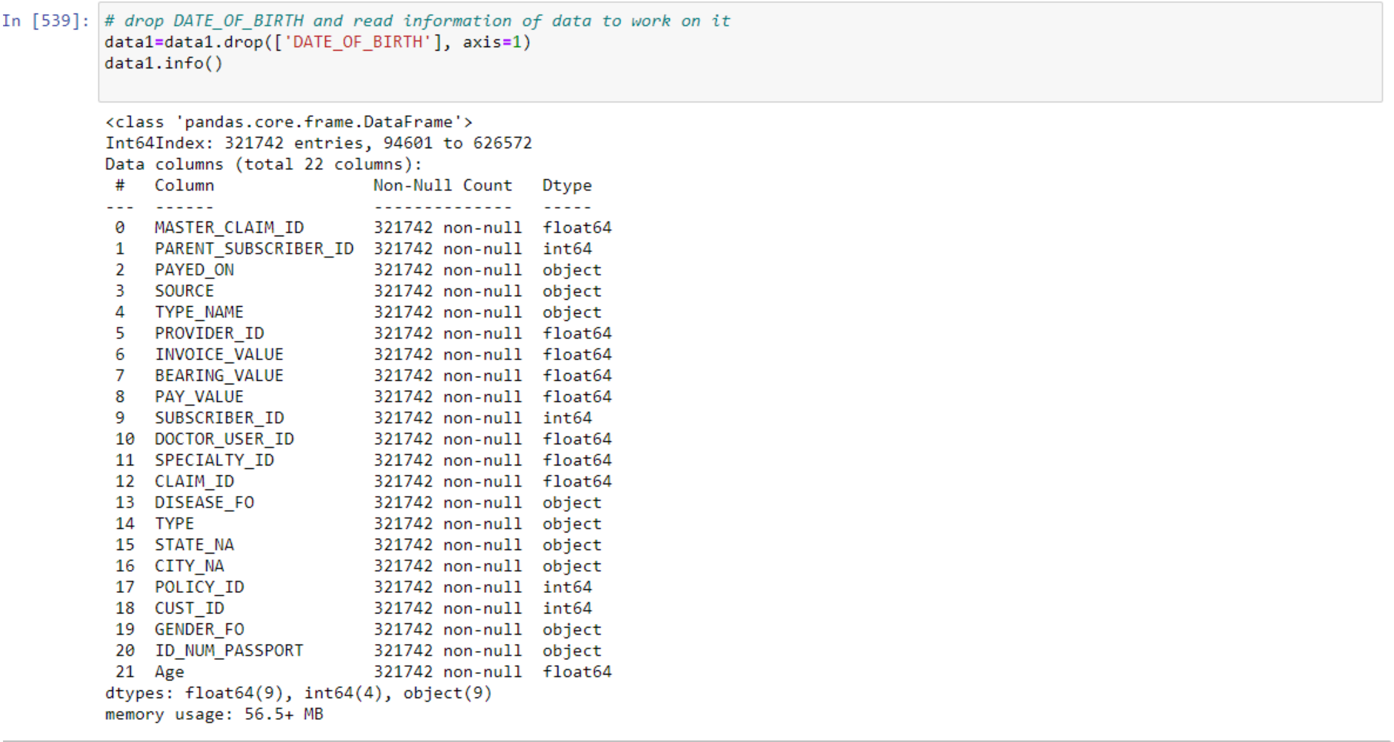


```
In [14]: #create new column (Age)
         now = pd.Timestamp('now')
         data1['DATE_OF_BIRTH'] = pd.to_datetime(data1['DATE_OF_BIRTH'])
         data1['DATE_OF_BIRTH'] = data1['DATE_OF_BIRTH'].where(data1['DATE_OF_BIRTH'] < now, data1['DATE_OF_BIRTH'] -  np.timedelta64(100,
         data1['Age'] = (now - data1['DATE_OF_BIRTH']).astype('<m8[Y]')
         data1
```

Out[14]:

_ID	SPECIALTY_ID	CLAIM_ID	DISEASE_FO	TYPE	STATE_NA	CITY_NA	DATE_OF_BIRTH	POLICY_ID	CUST_ID	GENDER_FO	ID_NUM_PASSPORT	Age
9.0	12.0	172.0	Other diseases of upper respiratory tract	CLINIC	رام الله والبيره	رام الله	2016-08-09	30851	17592	Male	437603616	5.0
9.0	12.0	173.0	Urinary tract infection, site not specified	CLINIC	Other_State	Other_City	2016-06-22	10745	6170	Female	436633226	5.0
9.0	12.0	175.0	Other diseases of upper respiratory tract	CLINIC	Other_State	Other_City	2014-12-12	34462	1835	Male	435330972	6.0
9.0	12.0	180.0	Other diseases of upper respiratory tract	CLINIC	Other_State	Other_City	2018-04-15	10745	6170	Male	439775446	3.0
9.0	12.0	186.0	Acute bronchitisâ° Other allergic rhinitis	CLINIC	Other_State	Other_City	2010-01-27	33448	17786	Male	422960666	11.0
...	...	...	...	...	...	...	...	...	...	...	...	...
0.0	1.0	99001.0	Acute upper respiratory infection, unspecified	MEDS	Other_State	Other_City	2014-03-10	143476	1840	Male	435022843	7.0
7.0	1.0	99000.0	Acute upper respiratory infection, unspecified	CLINIC	رام الله والبيره	رام الله	1979-09-01	143008	1835	Male	905559738	42.0
7.0	1.0	99000.0	Acute upper respiratory infection, unspecified	MEDS	رام الله والبيره	رام الله	1979-09-01	143008	1835	Male	905559738	42.0

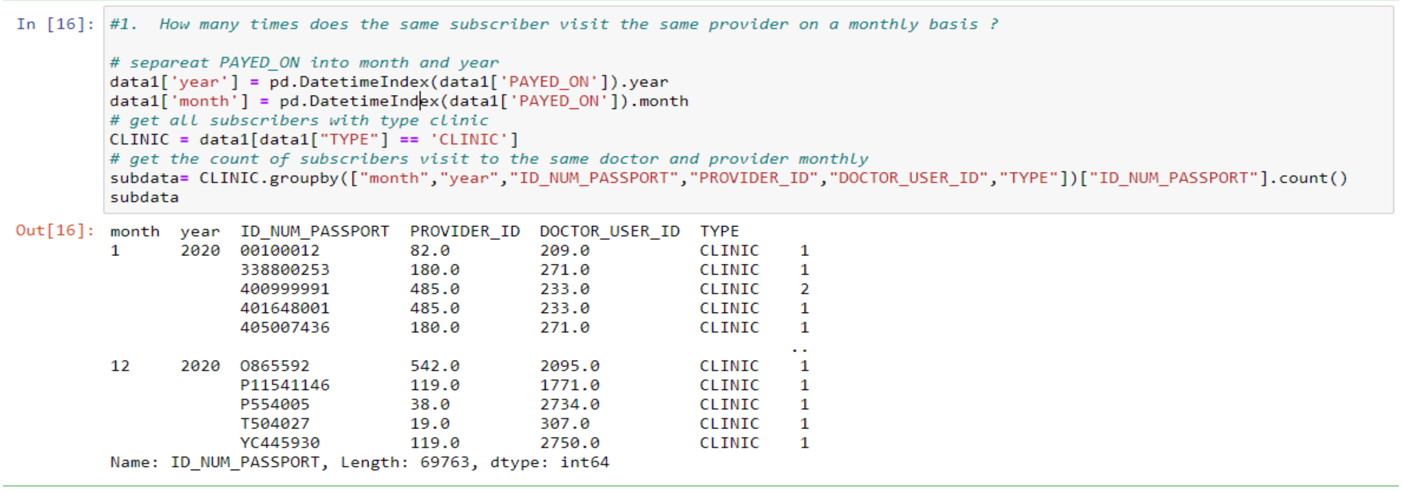
# Drop DATE_OF_BIRTH and read information of data to work on it



```
In [539]: # drop DATE_OF_BIRTH and read information of data to work on it
          data1=data1.drop(['DATE_OF_BIRTH'], axis=1)
          data1.info()
```

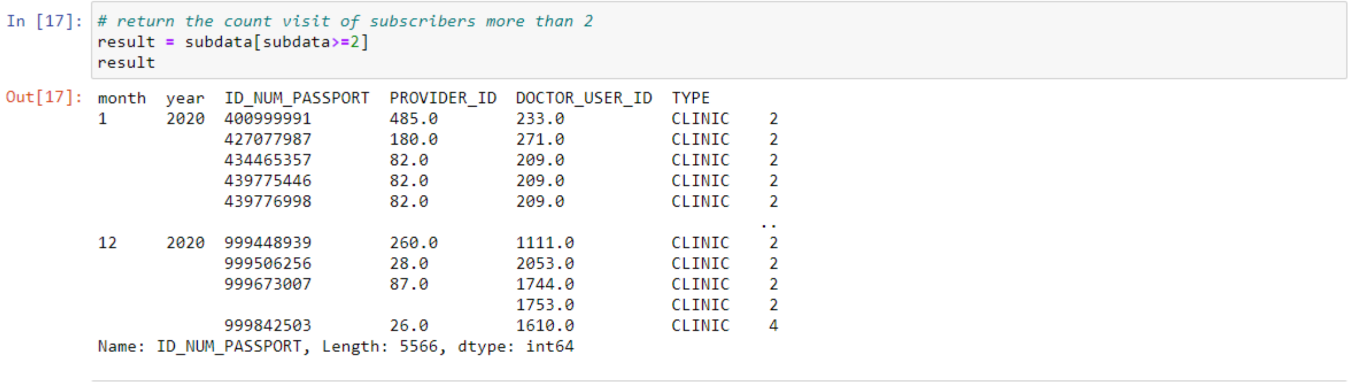
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 321742 entries, 94601 to 626572
Data columns (total 22 columns):
 #   Column                Non-Null Count   Dtype  
---  ------                --------------   -----  
 0   MASTER_CLAIM_ID       321742 non-null  float64
 1   PARENT_SUBSCRIBER_ID  321742 non-null  int64  
 2   PAYED_ON              321742 non-null  object 
 3   SOURCE                321742 non-null  object 
 4   TYPE_NAME             321742 non-null  object 
 5   PROVIDER_ID           321742 non-null  float64
 6   INVOICE_VALUE         321742 non-null  float64
 7   BEARING_VALUE         321742 non-null  float64
 8   PAY_VALUE             321742 non-null  float64
 9   SUBSCRIBER_ID         321742 non-null  int64  
 10  DOCTOR_USER_ID        321742 non-null  float64
 11  SPECIALTY_ID          321742 non-null  float64
 12  CLAIM_ID              321742 non-null  float64
 13  DISEASE_FO            321742 non-null  object 
 14  TYPE                  321742 non-null  object 
 15  STATE_NA              321742 non-null  object 
 16  CITY_NA               321742 non-null  object 
 17  POLICY_ID             321742 non-null  int64  
 18  CUST_ID               321742 non-null  int64  
 19  GENDER_FO             321742 non-null  object 
 20  ID_NUM_PASSPORT       321742 non-null  object 
 21  Age                   321742 non-null  float64
dtypes: float64(9), int64(4), object(9)
memory usage: 56.5+ MB
```

1. How many times does the same subscriber visit the same provider on a monthly basis ?



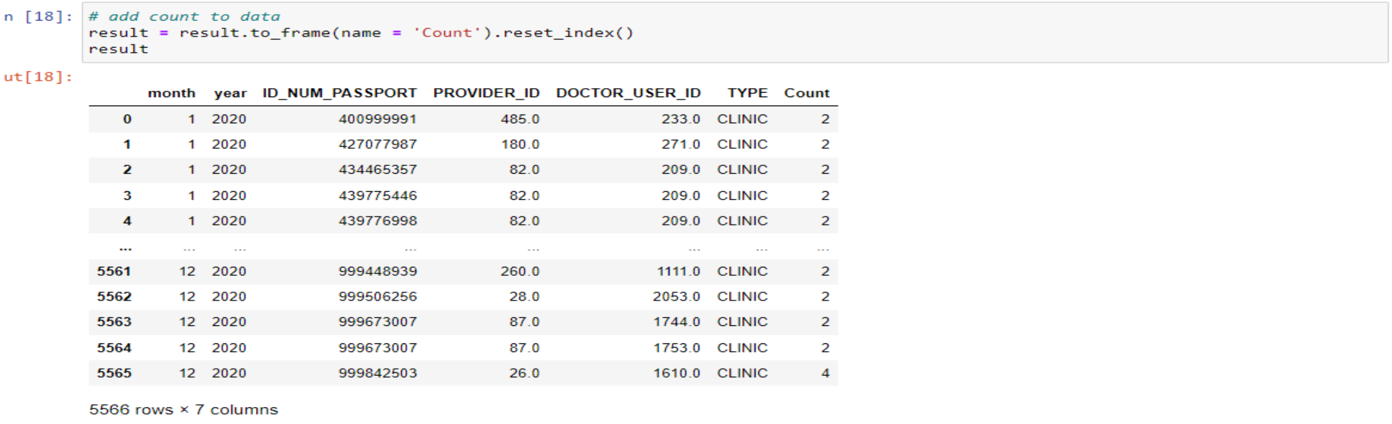
```
In [16]: #1.  How many times does the same subscriber visit the same provider on a monthly basis ?

         # separeat PAYED_ON into month and year
         data1['year'] = pd.DatetimeIndex(data1['PAYED_ON']).year
         data1['month'] = pd.DatetimeIndex(data1['PAYED_ON']).month
         # get all subscribers with type clinic
         CLINIC = data1[data1["TYPE"] == 'CLINIC']
         # get the count of subscribers visit to the same doctor and provider monthly
         subdata= CLINIC.groupby(["month","year","ID_NUM_PASSPORT","PROVIDER_ID","DOCTOR_USER_ID","TYPE"])["ID_NUM_PASSPORT"].count()
         subdata

Out[16]: month  year  ID_NUM_PASSPORT  PROVIDER_ID  DOCTOR_USER_ID  TYPE
         1      2020  00100012         82.0         209.0           CLINIC    1
                      338800253        180.0        271.0           CLINIC    1
                      400999991        485.0        233.0           CLINIC    2
                      401648001        485.0        233.0           CLINIC    1
                      405007436        180.0        271.0           CLINIC    1
                                                                                ..
         12     2020  0865592          542.0        2095.0          CLINIC    1
                      P11541146        119.0        1771.0          CLINIC    1
                      P554005          38.0         2734.0          CLINIC    1
                      T504027          19.0         307.0           CLINIC    1
                      YC445930         119.0        2750.0          CLINIC    1
         Name: ID_NUM_PASSPORT, Length: 69763, dtype: int64
```

return the count visit of subscribers more than 2



```
In [17]: # return the count visit of subscribers more than 2
         result = subdata[subdata>=2]
         result

Out[17]: month  year  ID_NUM_PASSPORT  PROVIDER_ID  DOCTOR_USER_ID  TYPE
         1      2020  400999991        485.0        233.0           CLINIC    2
                      427077987        180.0        271.0           CLINIC    2
                      434465357        82.0         209.0           CLINIC    2
                      439775446        82.0         209.0           CLINIC    2
                      439776998        82.0         209.0           CLINIC    2
                                                                                ..
         12     2020  999448939        260.0        1111.0          CLINIC    2
                      999506256        28.0         2053.0          CLINIC    2
                      999673007        87.0         1744.0          CLINIC    2
                                                    1753.0          CLINIC    2
                      999842503        26.0         1610.0          CLINIC    4
         Name: ID_NUM_PASSPORT, Length: 5566, dtype: int64
```

# Convert data to dataframe and add count on it



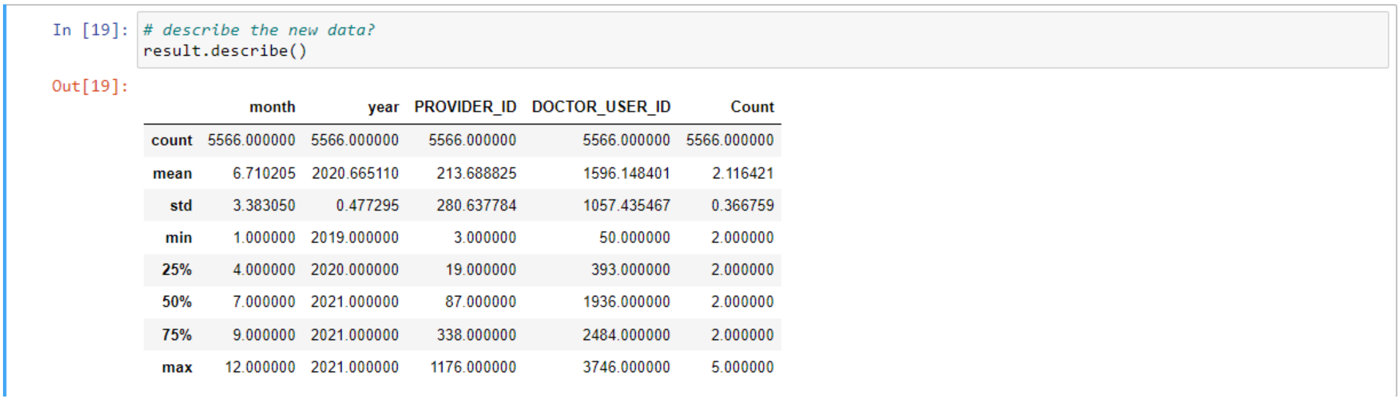
```
n [18]: # add count to data
        result = result.to_frame(name = 'Count').reset_index()
        result
```

ut[18]:

	month	year	ID_NUM_PASSPORT	PROVIDER_ID	DOCTOR_USER_ID	TYPE	Count
0	1	2020	400999991	485.0	233.0	CLINIC	2
1	1	2020	427077987	180.0	271.0	CLINIC	2
2	1	2020	434465357	82.0	209.0	CLINIC	2
3	1	2020	439775446	82.0	209.0	CLINIC	2
4	1	2020	439776998	82.0	209.0	CLINIC	2
...	...	...	...	...	...	...	...
5561	12	2020	999448939	260.0	1111.0	CLINIC	2
5562	12	2020	999506256	28.0	2053.0	CLINIC	2
5563	12	2020	999673007	87.0	1744.0	CLINIC	2
5564	12	2020	999673007	87.0	1753.0	CLINIC	2
5565	12	2020	999842503	26.0	1610.0	CLINIC	4

5566 rows × 7 columns

# Describe the new data

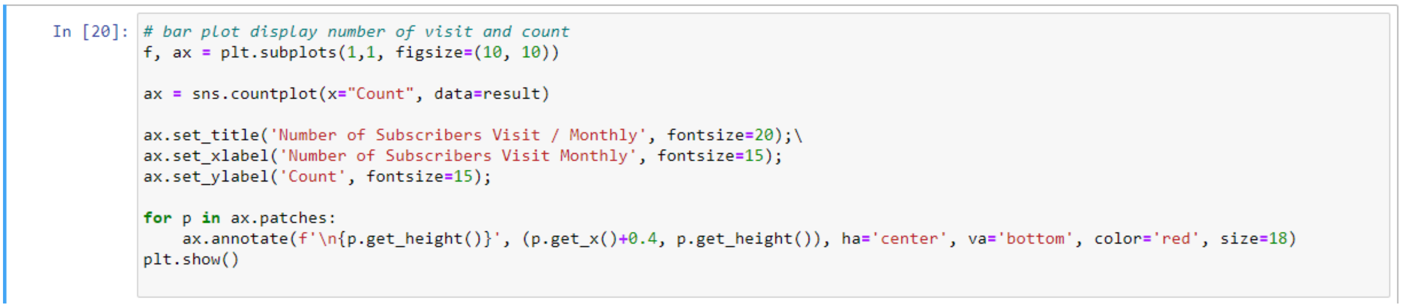


```
In [19]: # describe the new data?
         result.describe()
```

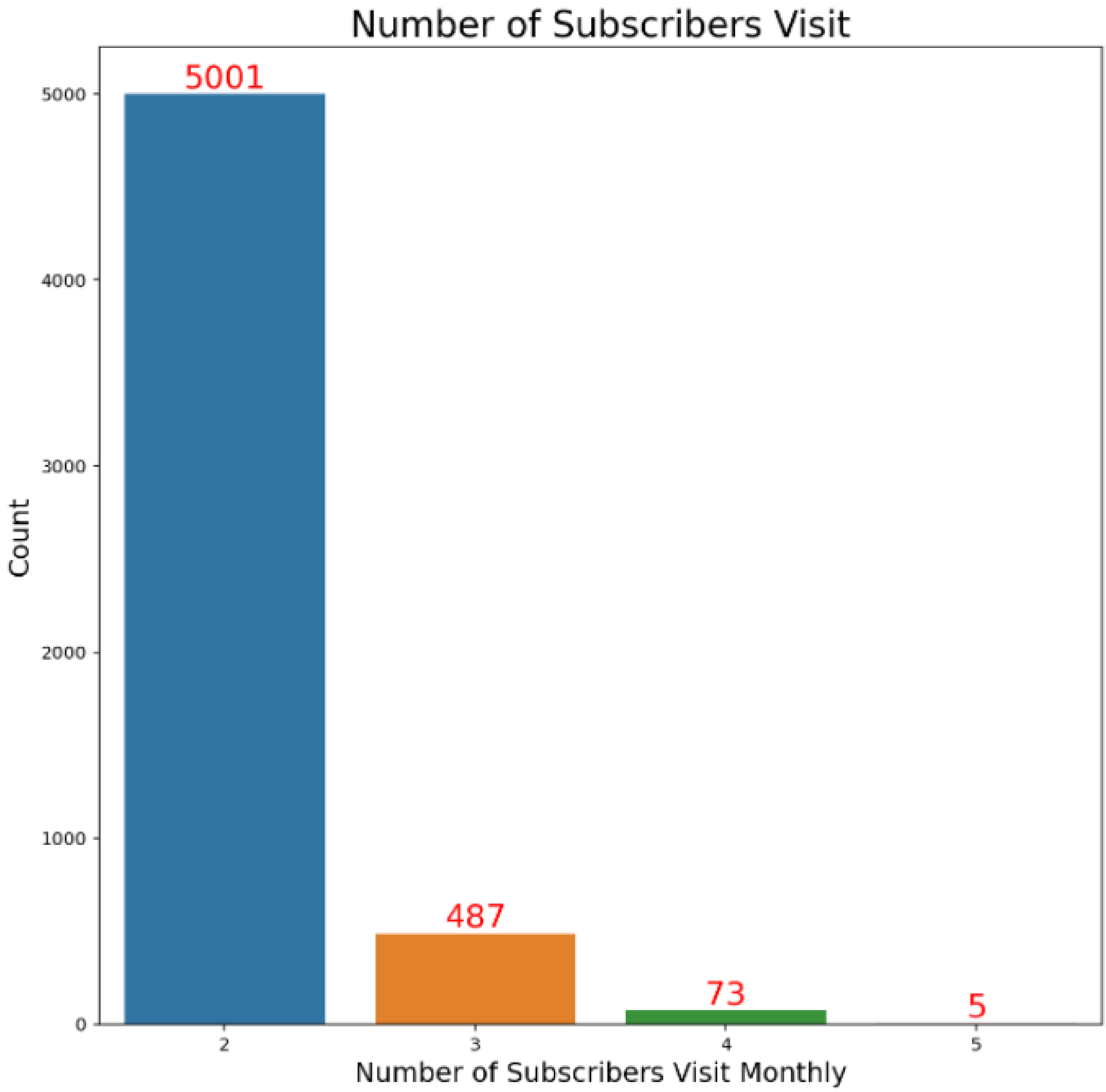
Out[19]:

	month	year	PROVIDER_ID	DOCTOR_USER_ID	Count
count	5566.000000	5566.000000	5566.000000	5566.000000	5566.000000
mean	6.710205	2020.665110	213.688825	1596.148401	2.116421
std	3.383050	0.477295	280.637784	1057.435467	0.366759
min	1.000000	2019.000000	3.000000	50.000000	2.000000
25%	4.000000	2020.000000	19.000000	393.000000	2.000000
50%	7.000000	2021.000000	87.000000	1936.000000	2.000000
75%	9.000000	2021.000000	338.000000	2484.000000	2.000000
max	12.000000	2021.000000	1176.000000	3746.000000	5.000000

bar plot display number of visit and count



```
In [20]: # bar plot display number of visit and count
         f, ax = plt.subplots(1,1, figsize=(10, 10))

         ax = sns.countplot(x="Count", data=result)

         ax.set_title('Number of Subscribers Visit / Monthly', fontsize=20);\
         ax.set_xlabel('Number of Subscribers Visit Monthly', fontsize=15);
         ax.set_ylabel('Count', fontsize=15);

         for p in ax.patches:
             ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.4, p.get_height()), ha='center', va='bottom', color='red', size=18)
         plt.show()
```



From chart above we note that :

The number of visit for the same subscriber at the same doctor monthly as following :

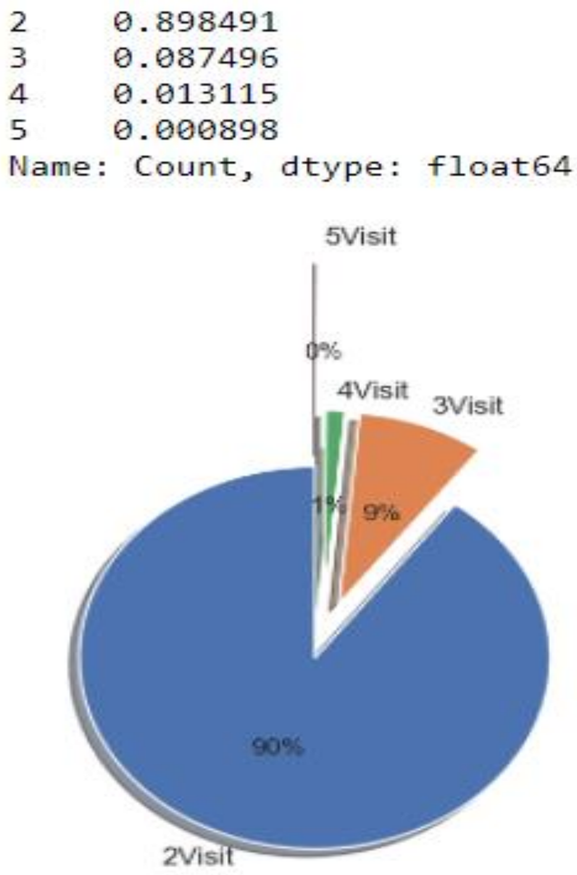
Two visit : 5001 subscriber
Three visit : 487 subscriber
Foure visit : 73 subscriber
Five visit : 5 subscriber

# pie chsrt to shwo percentage

In [21]:

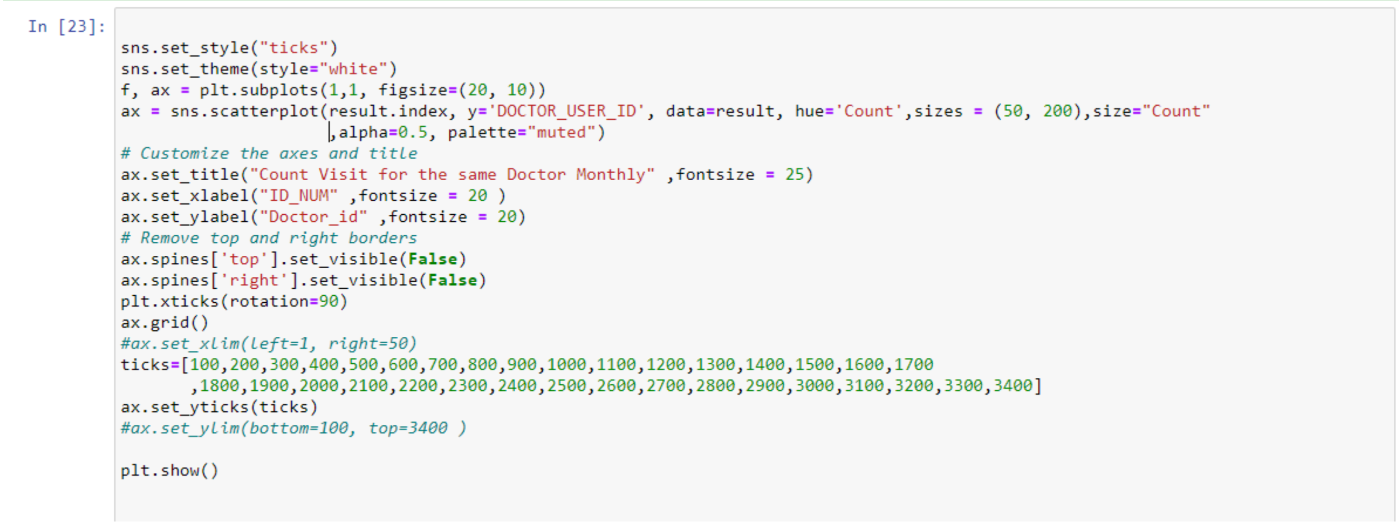
```
 # %load solutions/q04.py
# first find percentages

result
count=result['Count'].value_counts(normalize=True)

print (count)


# First and last time I will use a pie chart, let alone an exploding one!!
data = count
labels = ['2', '3','4','5']

explodeTuple = (0.1, 0.2, 0.2,1)


colors = sns.color_palette('bright')[0:5]

plt.pie(data,  autopct="%.f%%", labels=labels, pctdistance=0.5, startangle = 90 ,shadow=True,explode=explodeTuple)
plt.show()
```

```
2    0.898491
3    0.087496
4    0.013115
5    0.000898
Name: Count, dtype: float64
```



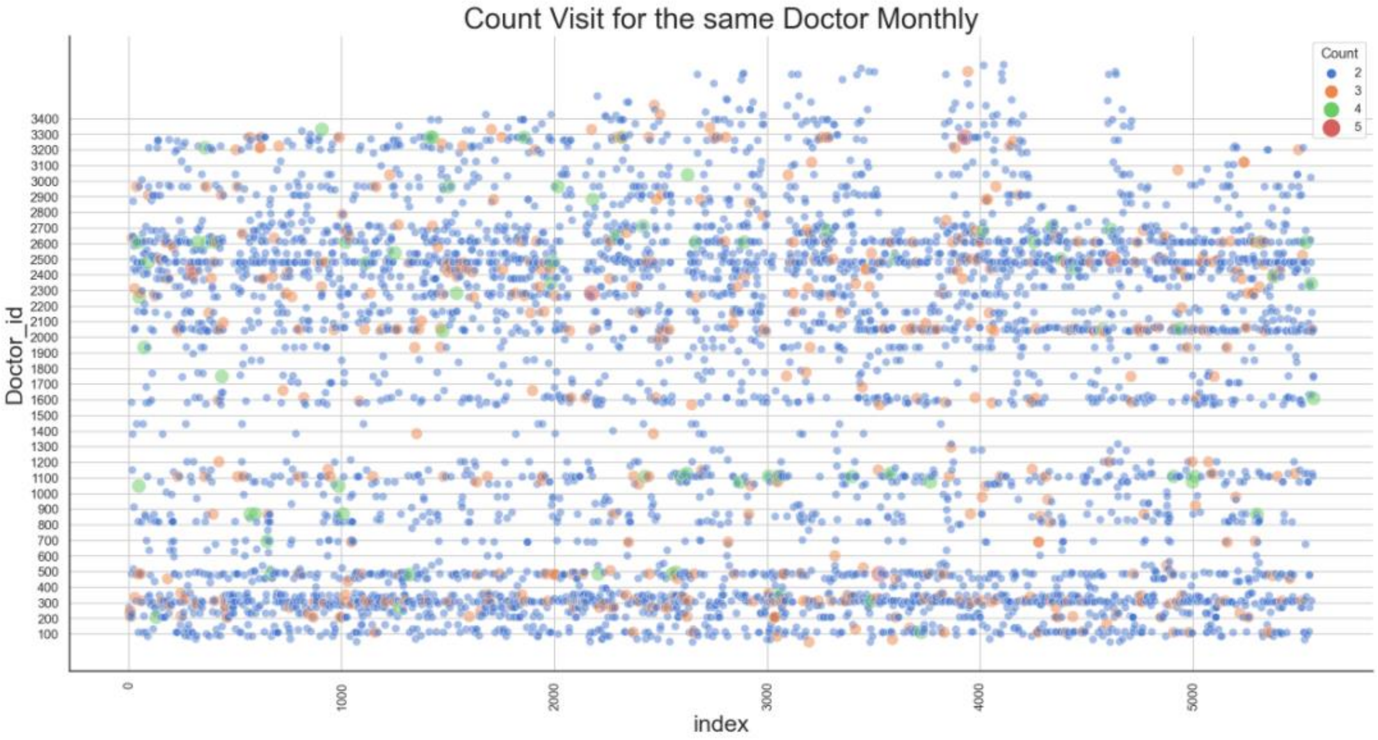
```
2     0.898491
3     0.087496
4     0.013115
5     0.000898
Name: Count, dtype: float64
```

# Scatter plot to detect subscriber with to doctor



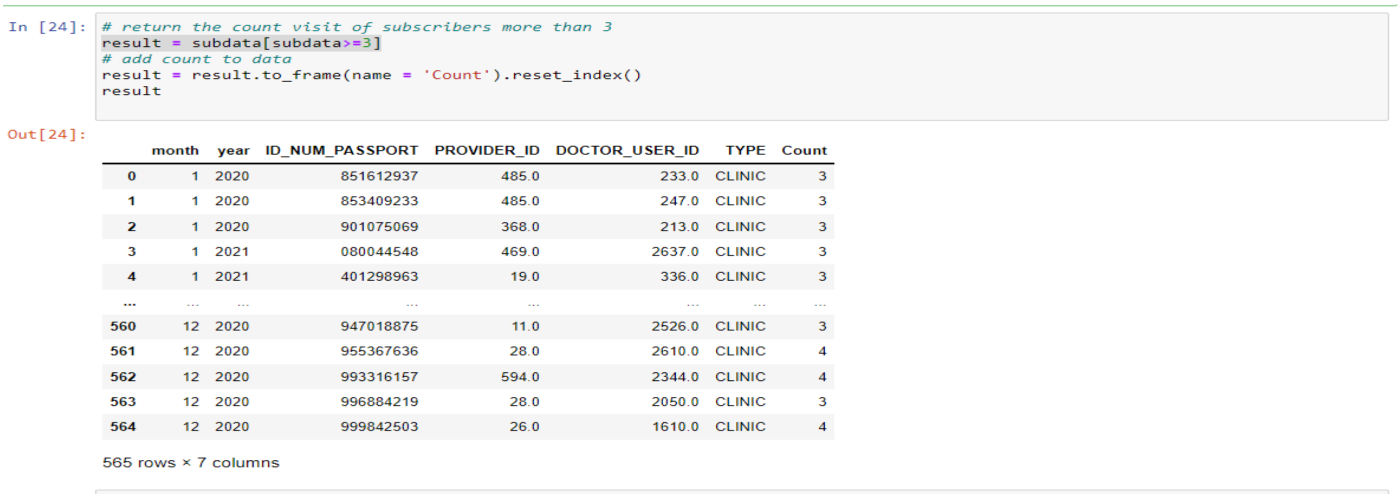
In [23]:

```
sns.set_style("ticks")
sns.set_theme(style="white")
f, ax = plt.subplots(1,1, figsize=(20, 10))
ax = sns.scatterplot(result.index, y='DOCTOR_USER_ID', data=result, hue='Count',sizes = (50, 200),size="Count"
                     ,alpha=0.5, palette="muted")
# Customize the axes and title
ax.set_title("Count Visit for the same Doctor Monthly" ,fontsize = 25)
ax.set_xlabel("ID_NUM" ,fontsize = 20 )
ax.set_ylabel("Doctor_id" ,fontsize = 20)
# Remove top and right borders
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
plt.xticks(rotation=90)
ax.grid()
#ax.set_xlim(left=1, right=50)
ticks=[100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,1400,1500,1600,1700
       ,1800,1900,2000,2100,2200,2300,2400,2500,2600,2700,2800,2900,3000,3100,3200,3300,3400]
ax.set_yticks(ticks)
#ax.set_ylim(bottom=100, top=3400 )

plt.show()
```



From pic above we can’t know where the subscriber went , so we decided to limitation number of vsist to 3 and above , that is mean remove blue color from the chart

return the count visit of subscribers more than 3



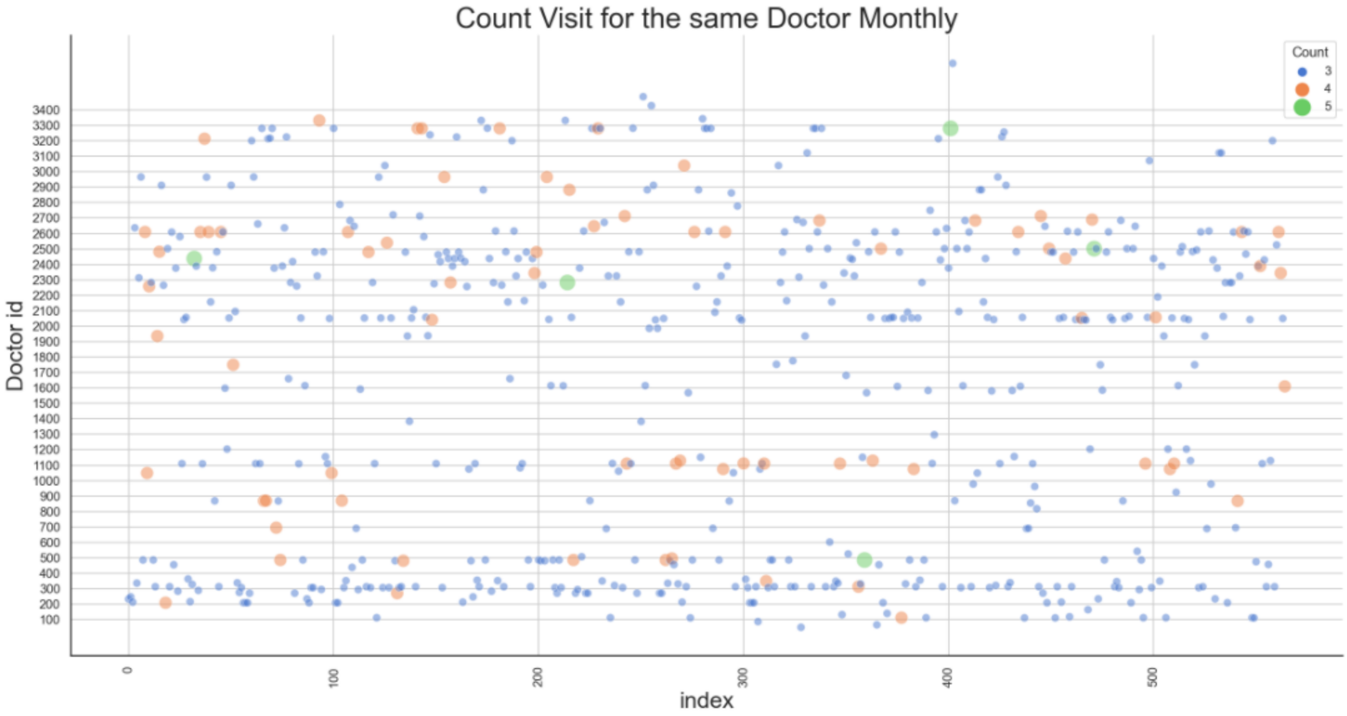
In [24]:

```
# return the count visit of subscribers more than 3
result = subdata[subdata>=3]
# add count to data
result = result.to_frame(name = 'Count').reset_index()
result
```

Out[24]:

	month	year	ID_NUM_PASSPORT	PROVIDER_ID	DOCTOR_USER_ID	TYPE	Count
0	1	2020	851612937	485.0	233.0	CLINIC	3
1	1	2020	853409233	485.0	247.0	CLINIC	3
2	1	2020	901075069	368.0	213.0	CLINIC	3
3	1	2021	080044548	469.0	2637.0	CLINIC	3
4	1	2021	401298963	19.0	336.0	CLINIC	3
...	...	...	...	...	...	...	...
560	12	2020	947018875	11.0	2526.0	CLINIC	3
561	12	2020	955367636	28.0	2610.0	CLINIC	4
562	12	2020	993316157	594.0	2344.0	CLINIC	4
563	12	2020	996884219	28.0	2050.0	CLINIC	3
564	12	2020	999842503	26.0	1610.0	CLINIC	4

565 rows × 7 columns



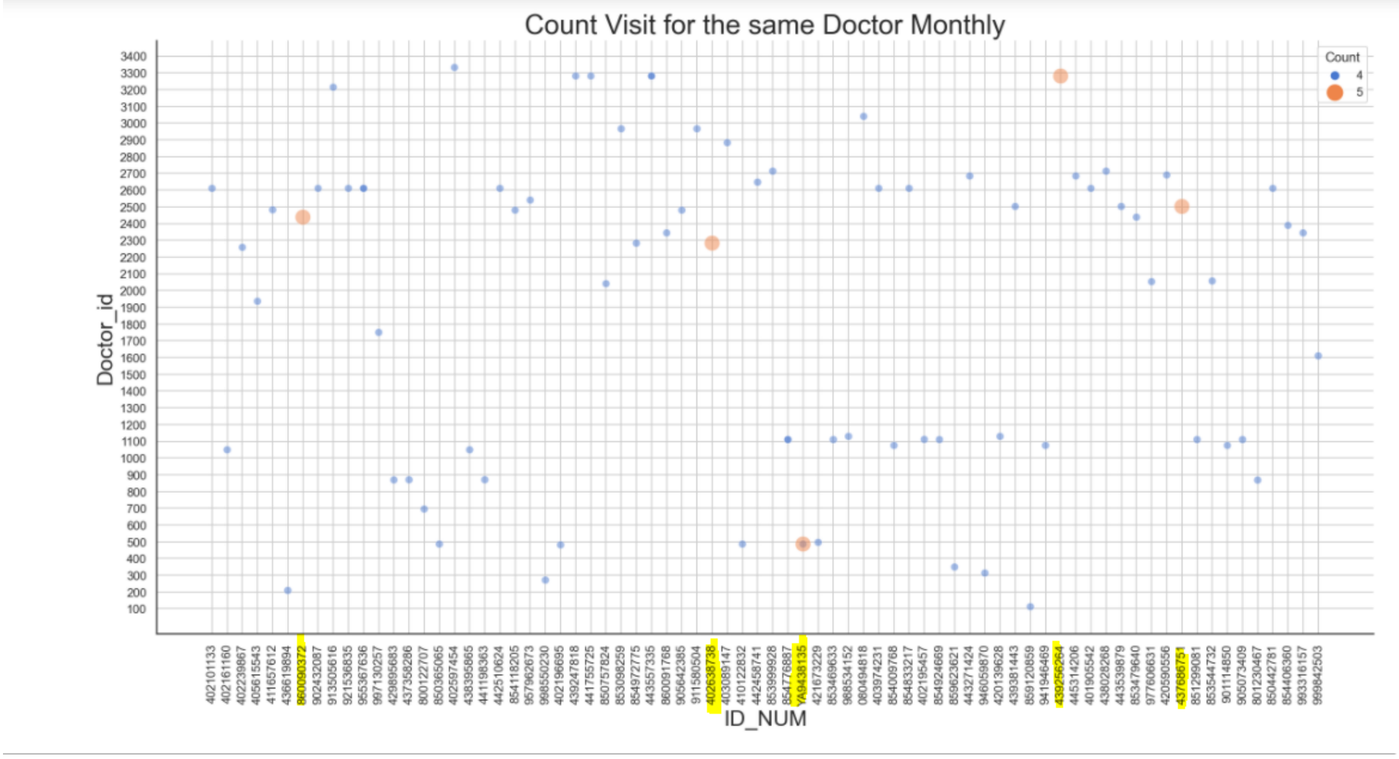
In [560]:

```
# return the count visit of subscribers more than 4
result = subdata[subdata>=4]
# add count to data
result = result.to_frame(name = 'Count').reset_index()
result
```

Out[560]:

	month	year	ID_NUM_PASSPORT	PROVIDER_ID	DOCTOR_USER_ID	TYPE	Count
0	1	2021	402101133	28.0	2610.0	CLINIC	4
1	1	2021	402161160	138.0	1049.0	CLINIC	4
2	1	2021	402239867	452.0	2258.0	CLINIC	4
3	1	2021	405615543	27.0	1936.0	CLINIC	4
4	1	2021	411657612	10.0	2482.0	CLINIC	4
...	...	...	...	...	...	...	...
73	12	2020	850442781	28.0	2610.0	CLINIC	4
74	12	2020	854406360	338.0	2389.0	CLINIC	4
75	12	2020	955367636	28.0	2610.0	CLINIC	4
76	12	2020	993316157	594.0	2344.0	CLINIC	4
77	12	2020	999842503	26.0	1610.0	CLINIC	4

78 rows × 7 columns



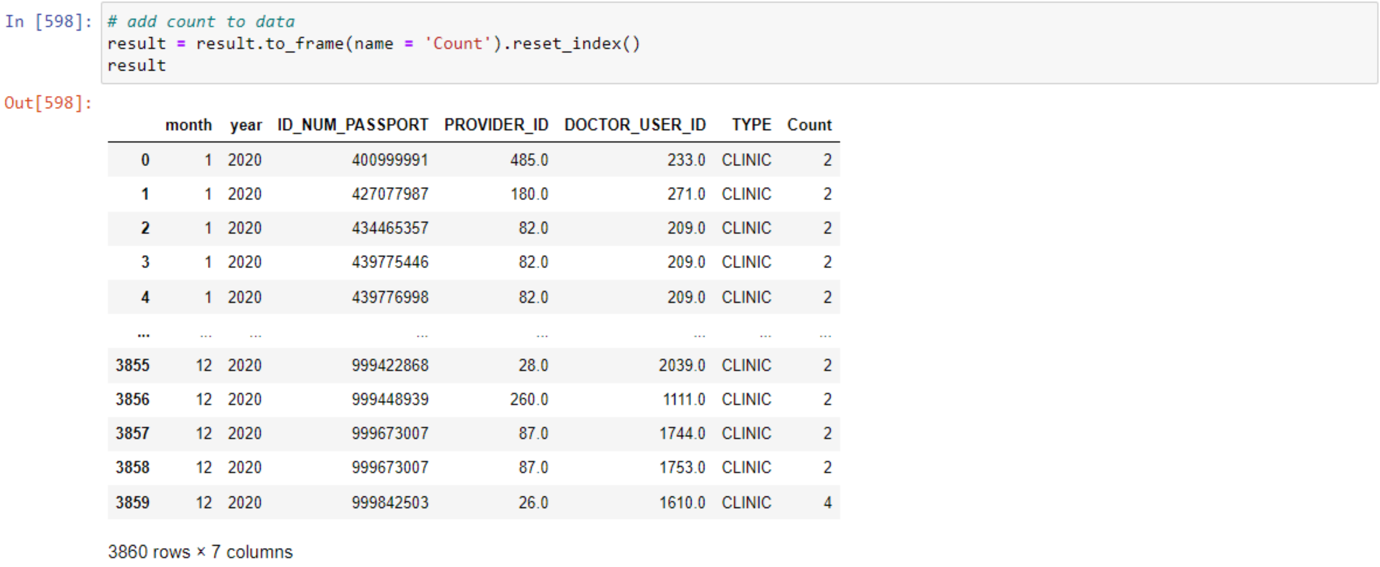
In [563]:

```
# return the count visit of subscribers more than 4
result = subdata[subdata>=5]
# add count to data
result = result.to_frame(name = 'Count').reset_index()
result
```

Out[563]:

	month	year	ID_NUM_PASSPORT	PROVIDER_ID	DOCTOR_USER_ID	TYPE	Count
0	1	2021	860090372	353.0	2438.0	CLINIC	5
1	6	2021	402638738	592.0	2283.0	CLINIC	5
2	8	2021	YA9438135	101.0	486.0	CLINIC	5
3	9	2021	439256264	1110.0	3281.0	CLINIC	5
4	10	2021	437686751	3.0	2502.0	CLINIC	5

The result below for subscribers visit to the same doctor at the same month than more one time , but without rejected claim , that is mean pay_value not equal **zero**

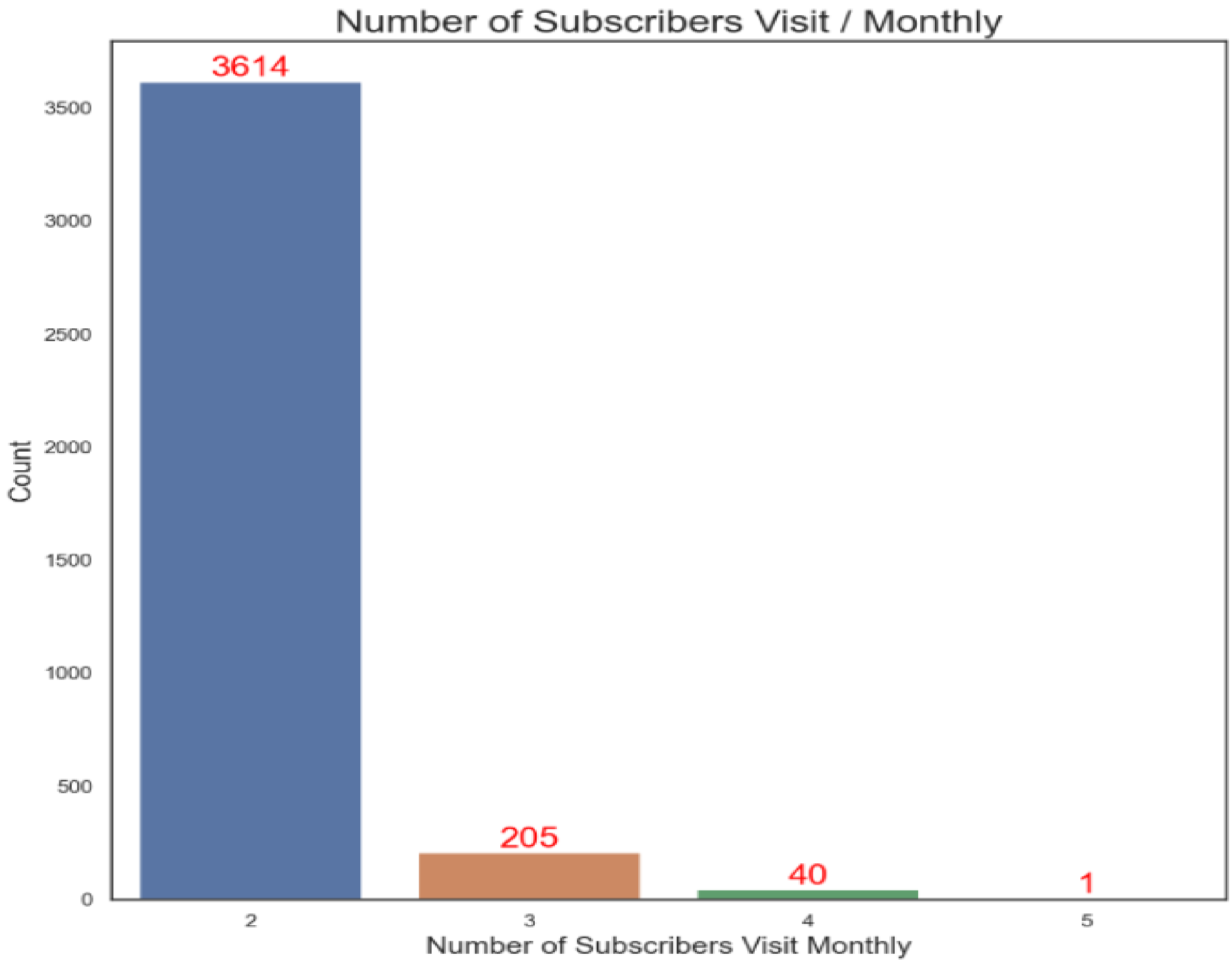


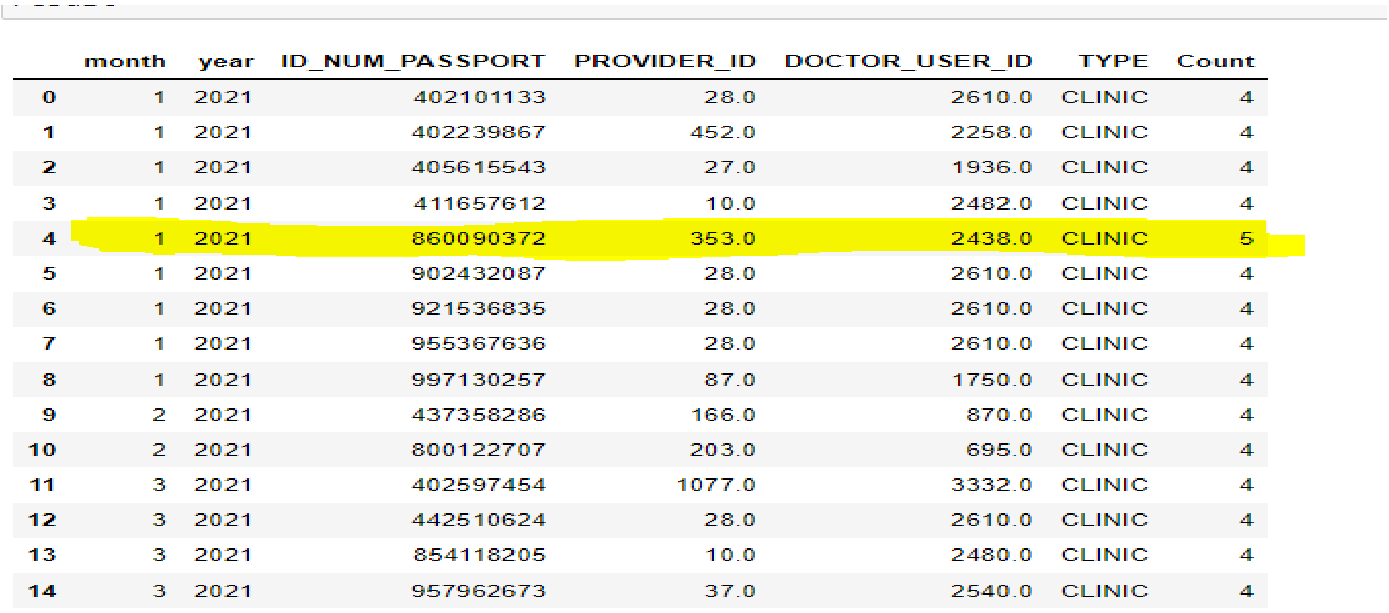
```
In [598]: # add count to data
          result = result.to_frame(name = 'Count').reset_index()
          result
```

Out[598]:

	month	year	ID_NUM_PASSPORT	PROVIDER_ID	DOCTOR_USER_ID	TYPE	Count
0	1	2020	400999991	485.0	233.0	CLINIC	2
1	1	2020	427077987	180.0	271.0	CLINIC	2
2	1	2020	434465357	82.0	209.0	CLINIC	2
3	1	2020	439775446	82.0	209.0	CLINIC	2
4	1	2020	439776998	82.0	209.0	CLINIC	2
...	...	...	...	...	...	...	...
3855	12	2020	999422868	28.0	2039.0	CLINIC	2
3856	12	2020	999448939	260.0	1111.0	CLINIC	2
3857	12	2020	999673007	87.0	1744.0	CLINIC	2
3858	12	2020	999673007	87.0	1753.0	CLINIC	2
3859	12	2020	999842503	26.0	1610.0	CLINIC	4

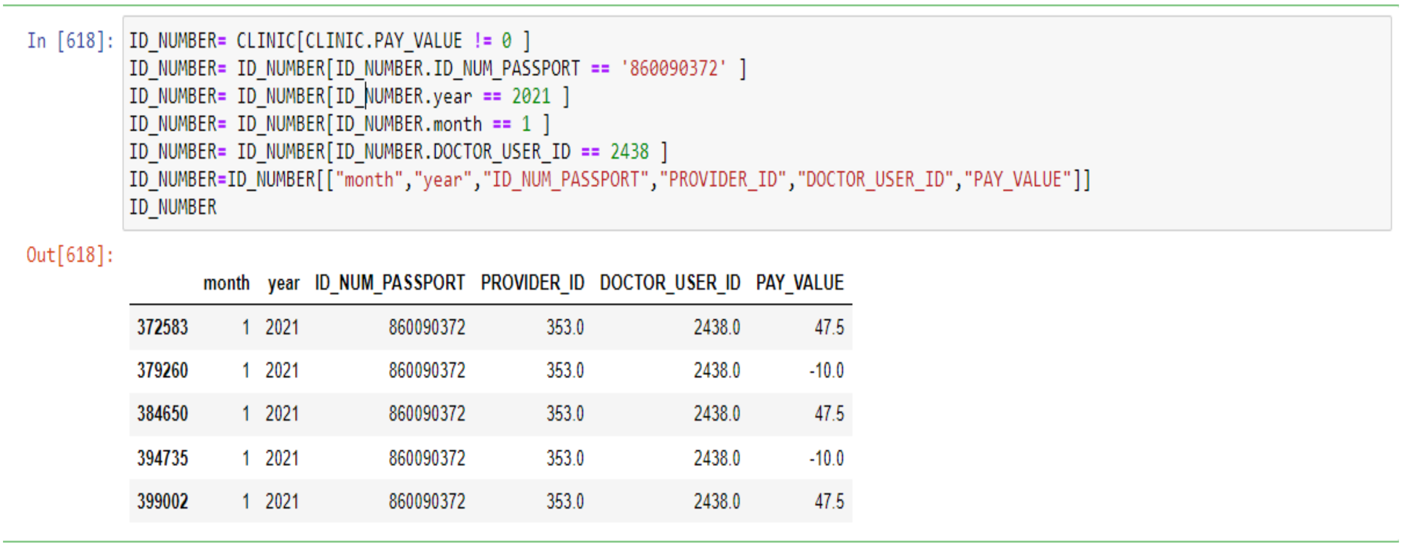
3860 rows × 7 columns





	month	year	ID_NUM_PASSPORT	PROVIDER_ID	DOCTOR_USER_ID	TYPE	Count
0	1	2021	402101133	28.0	2610.0	CLINIC	4
1	1	2021	402239867	452.0	2258.0	CLINIC	4
2	1	2021	405615543	27.0	1936.0	CLINIC	4
3	1	2021	411657612	10.0	2482.0	CLINIC	4
4	1	2021	860090372	353.0	2438.0	CLINIC	5
5	1	2021	902432087	28.0	2610.0	CLINIC	4
6	1	2021	921536835	28.0	2610.0	CLINIC	4
7	1	2021	955367636	28.0	2610.0	CLINIC	4
8	1	2021	997130257	87.0	1750.0	CLINIC	4
9	2	2021	437358286	166.0	870.0	CLINIC	4
10	2	2021	800122707	203.0	695.0	CLINIC	4
11	3	2021	402597454	1077.0	3332.0	CLINIC	4
12	3	2021	442510624	28.0	2610.0	CLINIC	4
13	3	2021	854118205	10.0	2480.0	CLINIC	4
14	3	2021	957962673	37.0	2540.0	CLINIC	4

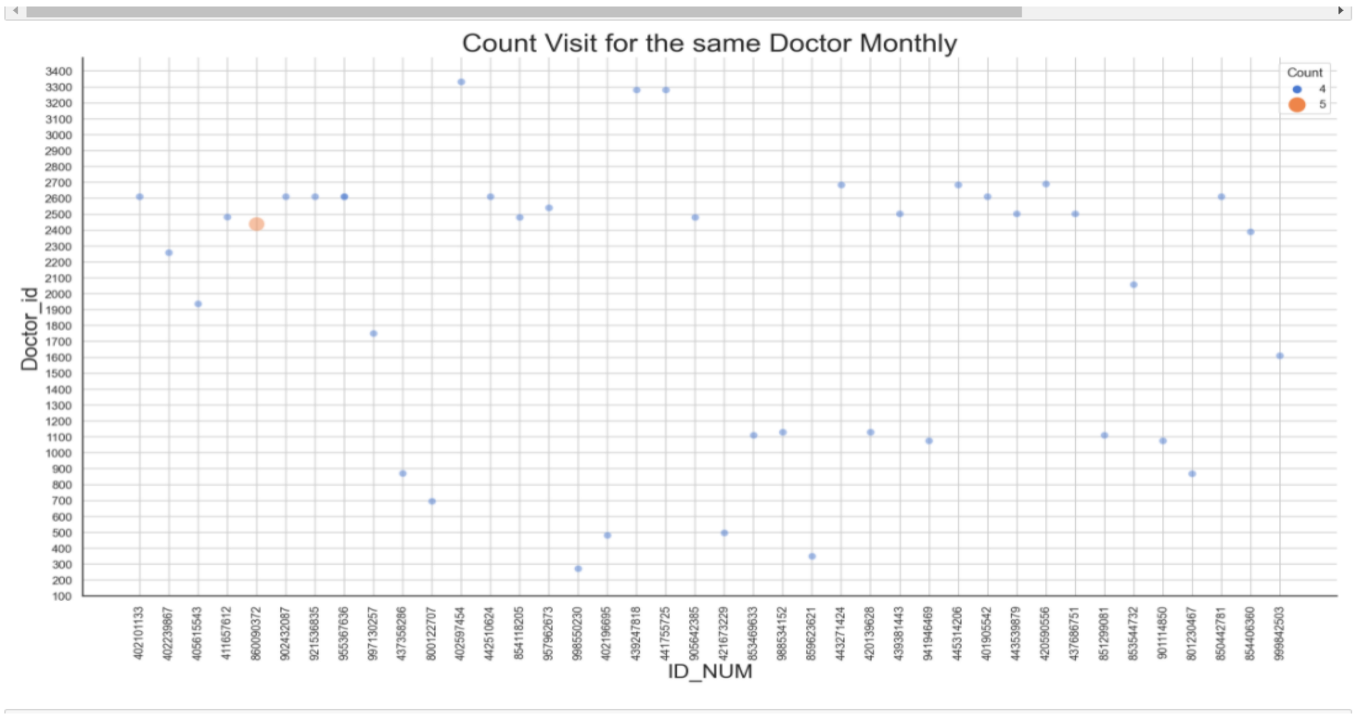
Example :



```
In [618]: ID_NUMBER= CLINIC[CLINIC.PAY_VALUE != 0 ]
          ID_NUMBER= ID_NUMBER[ID_NUMBER.ID_NUM_PASSPORT == '860090372' ]
          ID_NUMBER= ID_NUMBER[ID_NUMBER.year == 2021 ]
          ID_NUMBER= ID_NUMBER[ID_NUMBER.month == 1 ]
          ID_NUMBER= ID_NUMBER[ID_NUMBER.DOCTOR_USER_ID == 2438 ]
          ID_NUMBER=ID_NUMBER[["month","year","ID_NUM_PASSPORT","PROVIDER_ID","DOCTOR_USER_ID","PAY_VALUE"]]
          ID_NUMBER
```

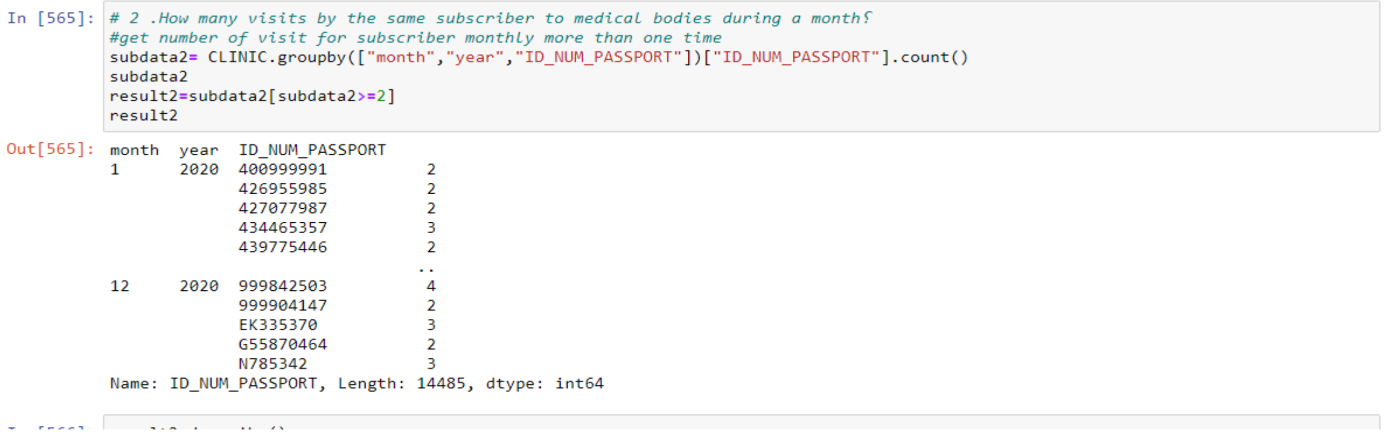
Out[618]:

	month	year	ID_NUM_PASSPORT	PROVIDER_ID	DOCTOR_USER_ID	PAY_VALUE
372583	1	2021	860090372	353.0	2438.0	47.5
379260	1	2021	860090372	353.0	2438.0	-10.0
384650	1	2021	860090372	353.0	2438.0	47.5
394735	1	2021	860090372	353.0	2438.0	-10.0
399002	1	2021	860090372	353.0	2438.0	47.5

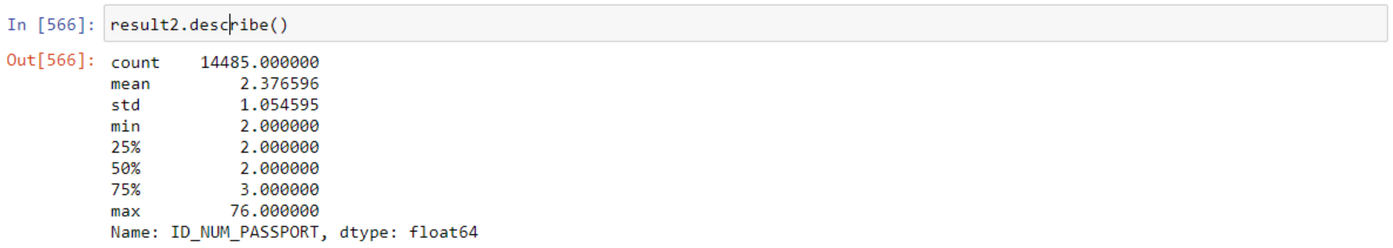


2 .How many visits by the same subscriber to medical bodies during a month

get number of visit for subscriber monthly more than one time



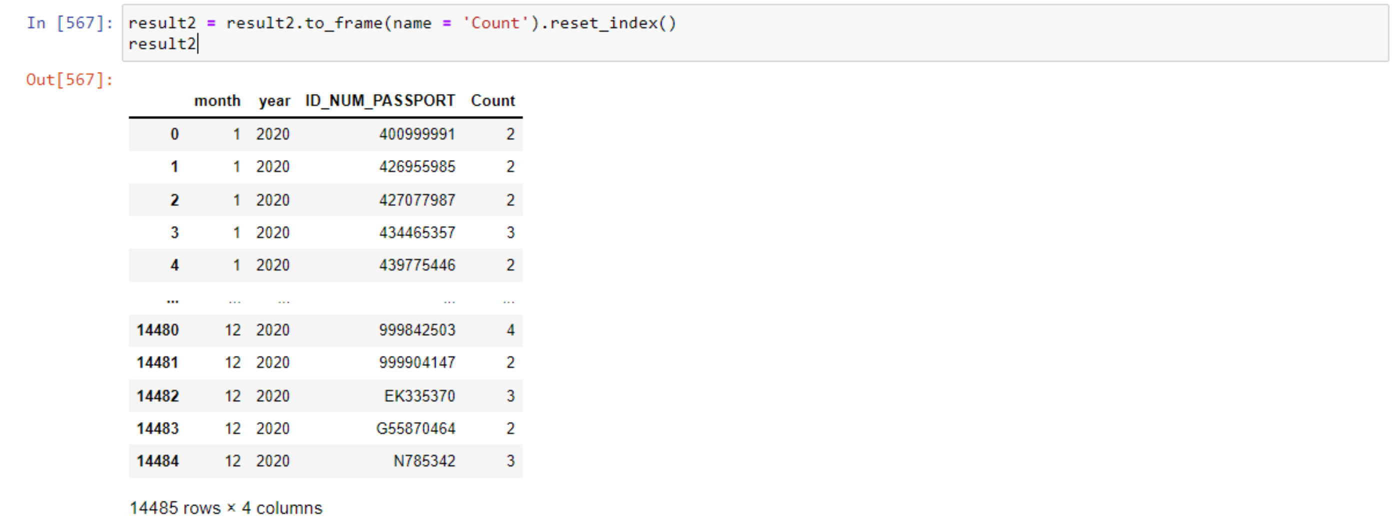
```
In [565]: # 2 .How many visits by the same subscriber to medical bodies during a month؟
          #get number of visit for subscriber monthly more than one time
          subdata2= CLINIC.groupby(["month","year","ID_NUM_PASSPORT"])["ID_NUM_PASSPORT"].count()
          subdata2
          result2=subdata2[subdata2>=2]
          result2

Out[565]: month  year  ID_NUM_PASSPORT
          1      2020  400999991          2
                       426955985          2
                       427077987          2
                       434465357          3
                       439775446          2
                                         ..
          12     2020  999842503          4
                       999904147          2
                       EK335370           3
                       G55870464          2
                       N785342            3
          Name: ID_NUM_PASSPORT, Length: 14485, dtype: int64
```



```
In [566]: result2.describe()

Out[566]: count    14485.000000
          mean         2.376596
          std          1.054595
          min          2.000000
          25%          2.000000
          50%          2.000000
          75%          3.000000
          max         76.000000
          Name: ID_NUM_PASSPORT, dtype: float64
```

From describe the data above that is mean 14485 subscribers have two or more than two in a month, the minimum visit is 2 and max 76

# insert count to data and covert it to datafram



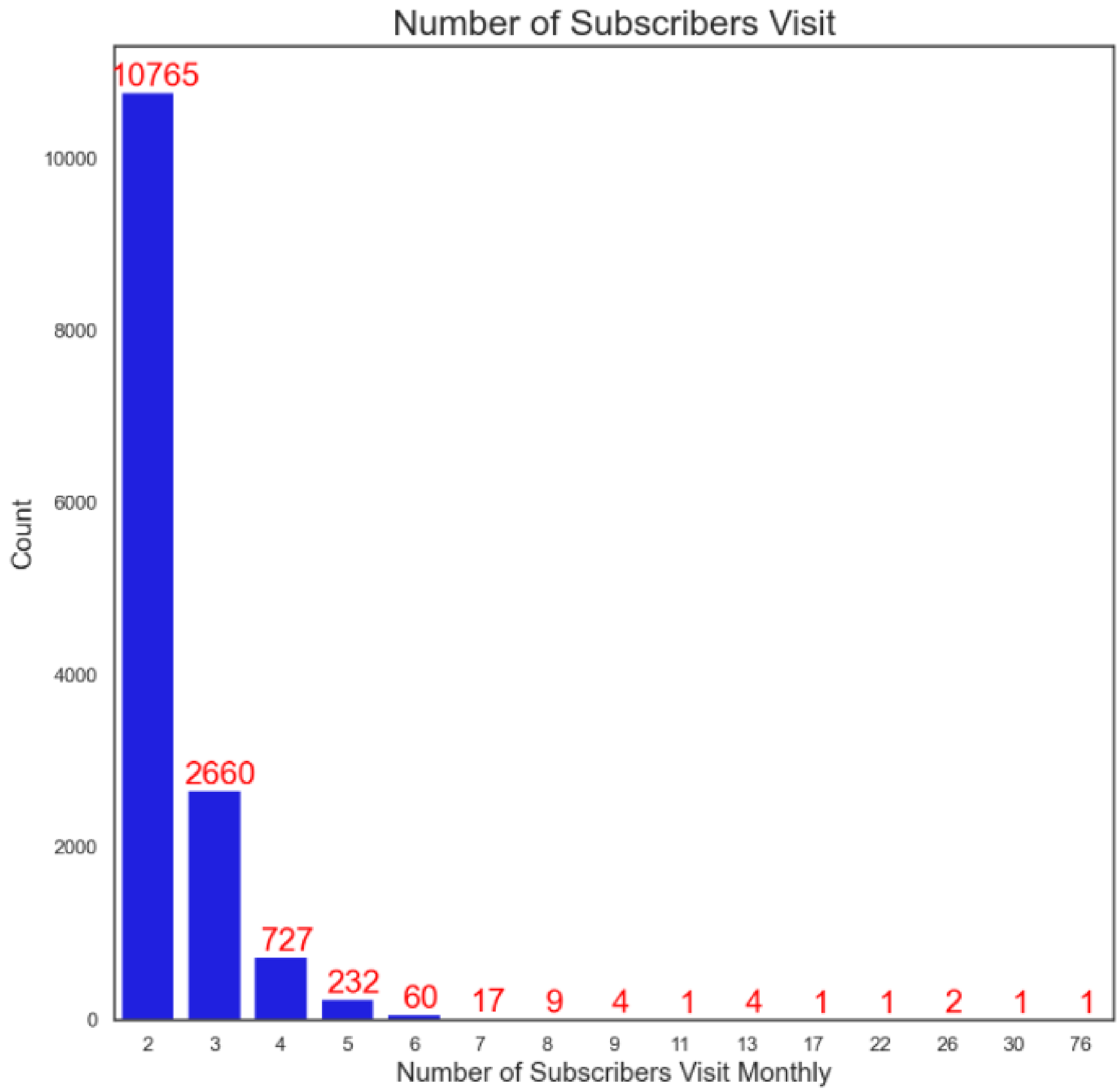
```
In [567]: result2 = result2.to_frame(name = 'Count').reset_index()
          result2
```

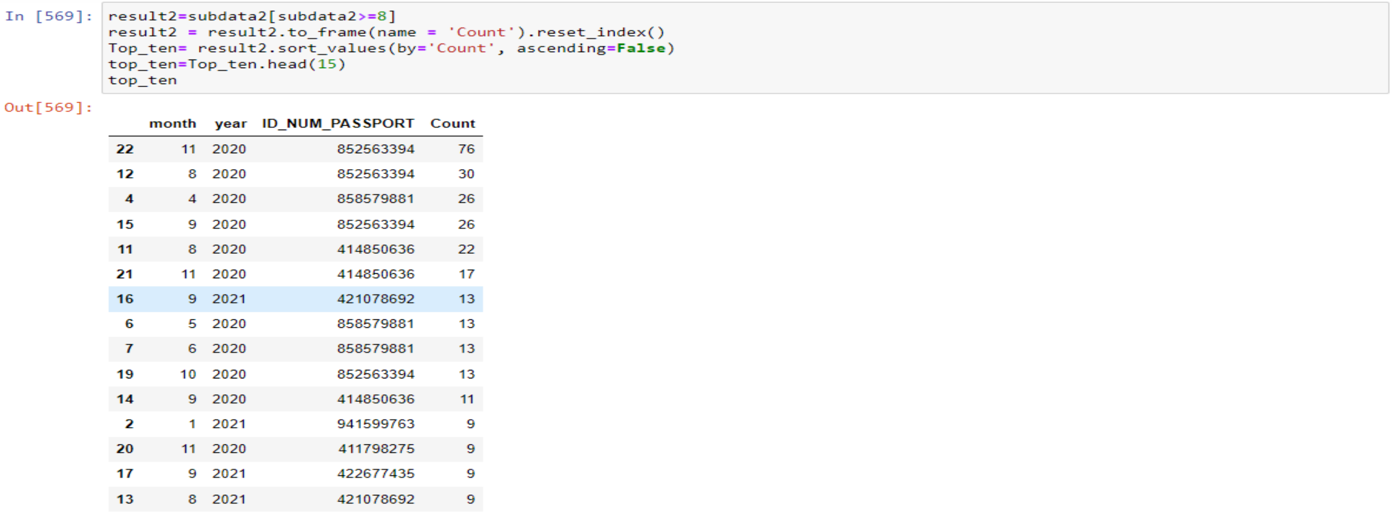
Out[567]:

	month	year	ID_NUM_PASSPORT	Count
0	1	2020	400999991	2
1	1	2020	426955985	2
2	1	2020	427077987	2
3	1	2020	434465357	3
4	1	2020	439775446	2
...	...	...	...	...
14480	12	2020	999842503	4
14481	12	2020	999904147	2
14482	12	2020	EK335370	3
14483	12	2020	G55870464	2
14484	12	2020	N785342	3

14485 rows × 4 columns

In [568]:
```
f, ax = plt.subplots(1,1, figsize=(10, 10))

ax = sns.countplot(x="Count", color='blue',data=result2)

ax.set_title('Number of Subscribers Visit', fontsize=20);\
ax.set_xlabel('Number of Subscribers Visit Monthly', fontsize=15);
ax.set_ylabel('Count', fontsize=15);

for p in ax.patches:
    ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.5, p.get_height()), ha='center', va='bottom', color='red', size=18)
plt.show()
```



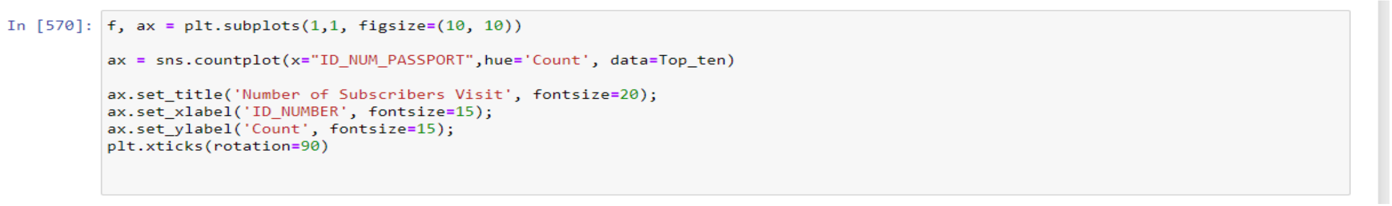


In [569]:

```
result2=subdata2[subdata2>=8]
result2 = result2.to_frame(name = 'Count').reset_index()
Top_ten= result2.sort_values(by='Count', ascending=False)
top_ten=Top_ten.head(15)
top_ten
```

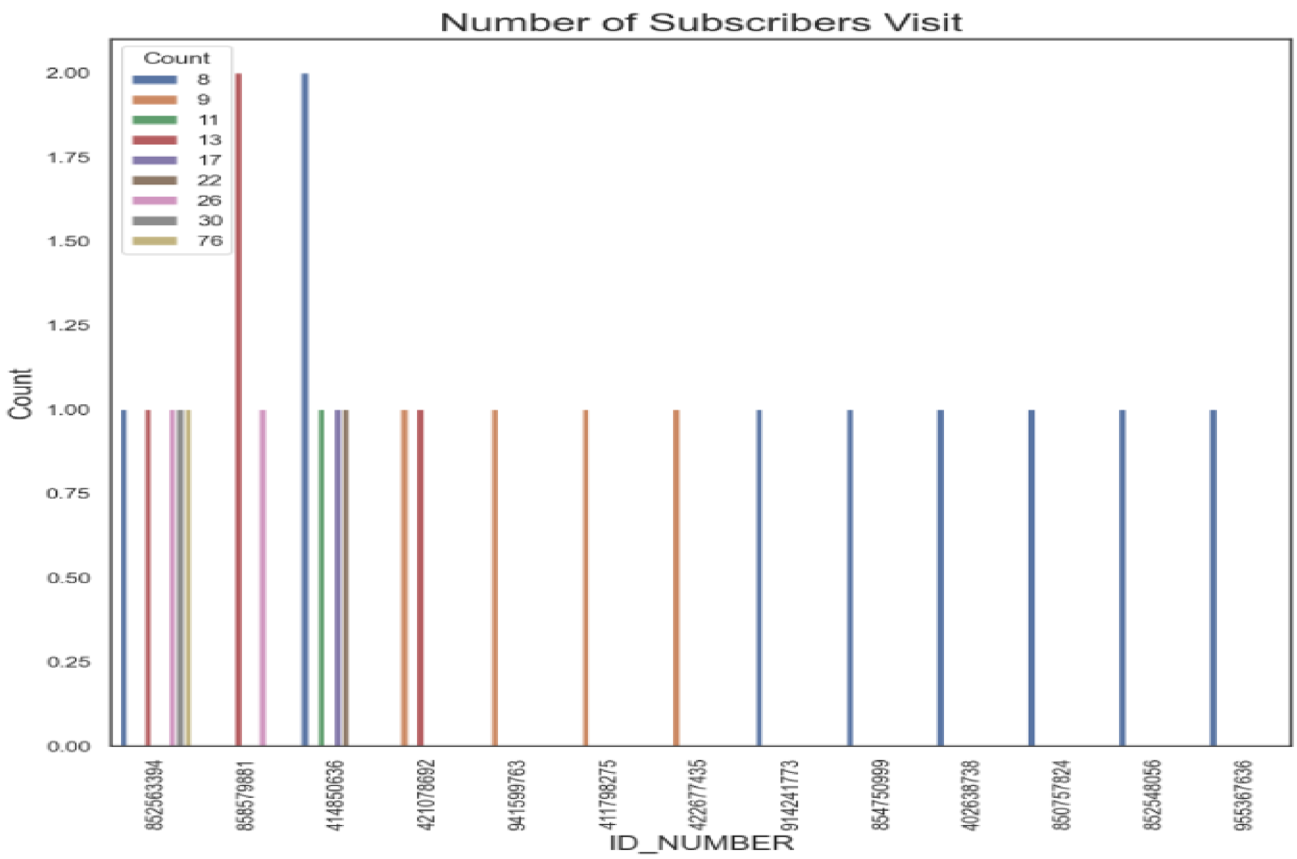
Out[569]:

	month	year	ID_NUM_PASSPORT	Count
22	11	2020	852563394	76
12	8	2020	852563394	30
4	4	2020	858579881	26
15	9	2020	852563394	26
11	8	2020	414850636	22
21	11	2020	414850636	17
16	9	2021	421078692	13
6	5	2020	858579881	13
7	6	2020	858579881	13
19	10	2020	852563394	13
14	9	2020	414850636	11
2	1	2021	941599763	9
20	11	2020	411798275	9
17	9	2021	422677435	9
13	8	2021	421078692	9

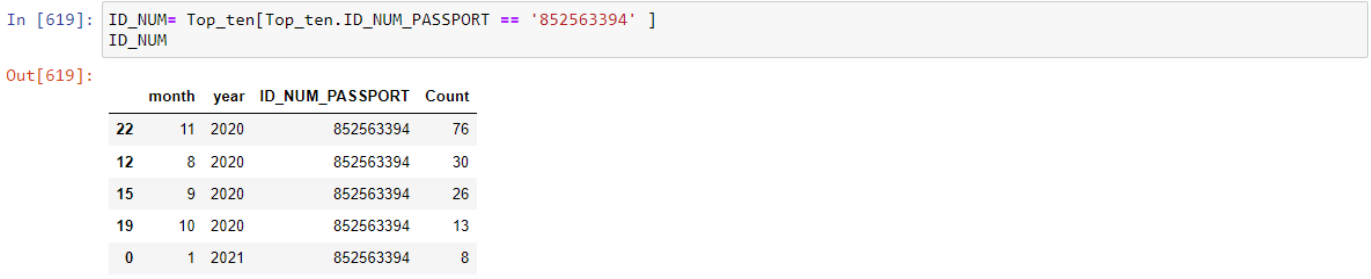


In [570]:

```
f, ax = plt.subplots(1,1, figsize=(10, 10))

ax = sns.countplot(x="ID_NUM_PASSPORT",hue='Count', data=Top_ten)

ax.set_title('Number of Subscribers Visit', fontsize=20);
ax.set_xlabel('ID_NUMBER', fontsize=15);
ax.set_ylabel('Count', fontsize=15);
plt.xticks(rotation=90)
```



From figer above the ID_Number 852563394 have a five type of visit : (8,13,26,30,76) in month



In [619]:

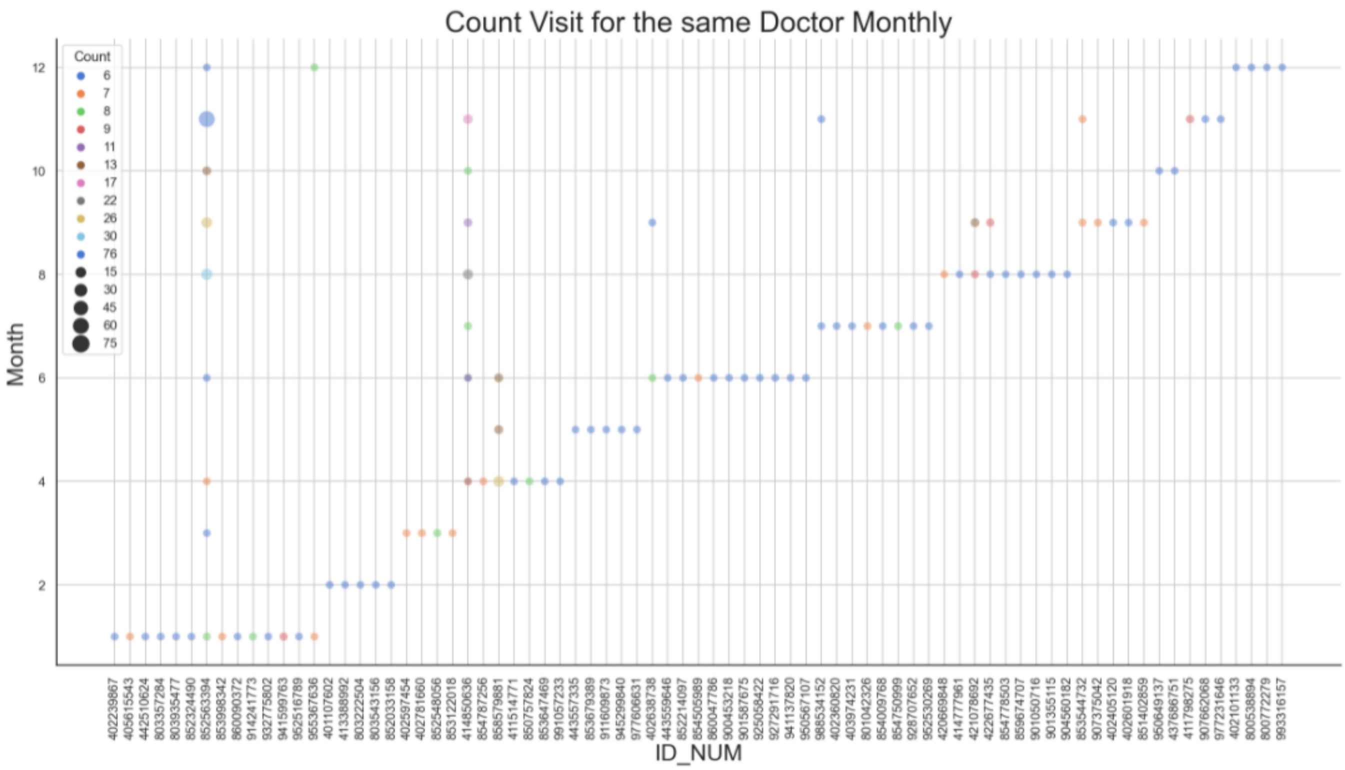
```
ID_NUM= Top_ten[Top_ten.ID_NUM_PASSPORT == '852563394' ]
ID_NUM
```

Out[619]:

	month	year	ID_NUM_PASSPORT	Count
22	11	2020	852563394	76
12	8	2020	852563394	30
15	9	2020	852563394	26
19	10	2020	852563394	13
0	1	2021	852563394	8

In [571]:
```
result2=subdata2[subdata2>=6]
result2 = result2.to_frame(name = 'Count').reset_index()
```

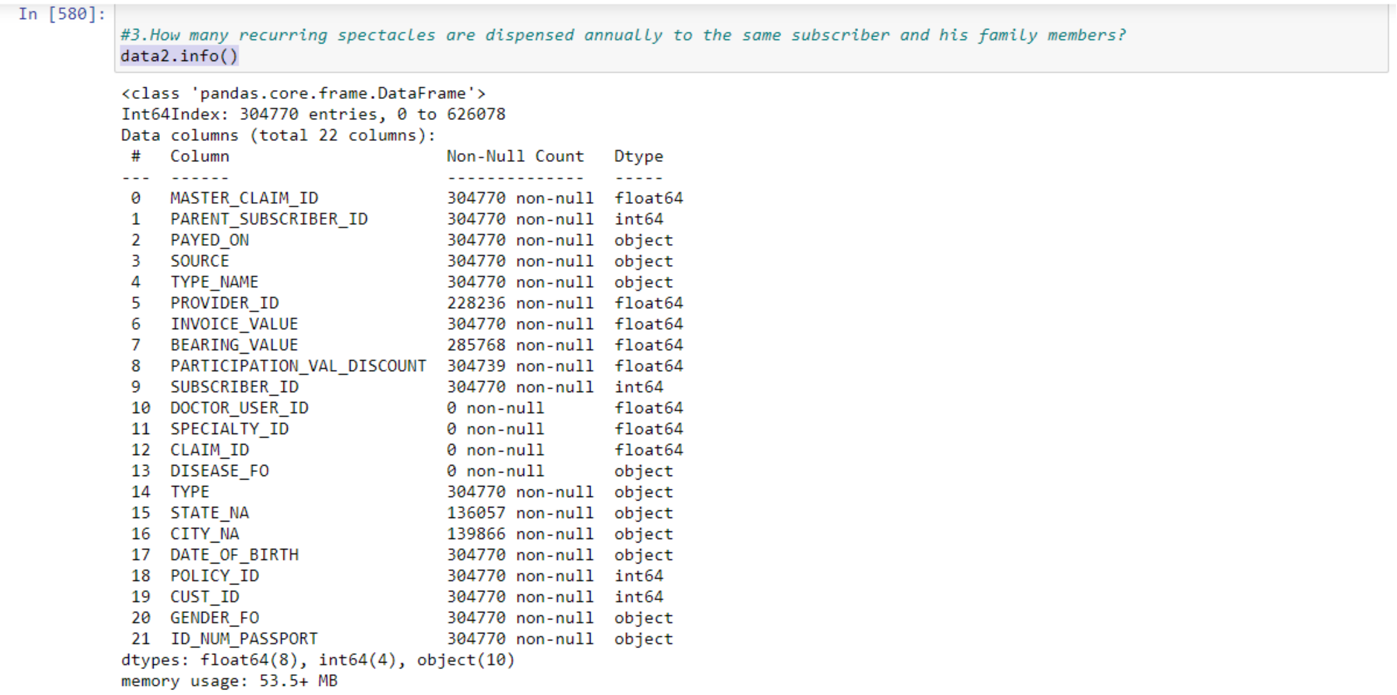
In [572]:
```
sns.set_style("ticks")
sns.set_theme(style="white")
f, ax = plt.subplots(1,1, figsize=(20, 10))
ax = sns.scatterplot(x='ID_NUM_PASSPORT', y='month', data=result2, hue='Count',sizes = (50, 200),size="Count" ,alpha=0.5, palette
# Customize the axes and title
ax.set_title("Count Visit for the same Doctor Monthly" ,fontsize = 25)
ax.set_xlabel("ID_NUM" ,fontsize = 20 )
ax.set_ylabel("Month" ,fontsize = 20)
# Remove top and right borders
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
plt.xticks(rotation=90)
ax.grid()
#ax.set_xlim(left=1, right=50)
#ticks=[100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,1400,1500,1600,1700,1800,1900,2000,2100,2200,2300,2400,2500,2600,
#ax.set_yticks(ticks)
#ax.set_ylim(bottom=100, top=3400 )

plt.show()
```



#3. How many recurring spectacles are dispensed annually to the same subscriber and his family members?

In this question we will work on the second part of data REIMBURSEMENT Because Spectacles on network

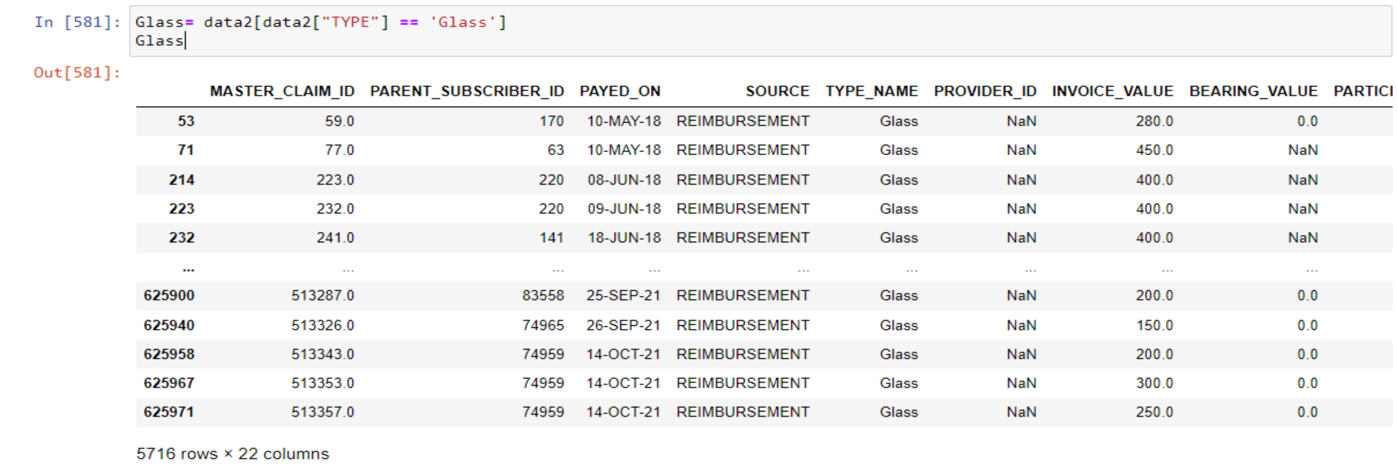
# Read information for data2



```
In [580]:
#3.How many recurring spectacles are dispensed annually to the same subscriber and his family members?
data2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 304770 entries, 0 to 626078
Data columns (total 22 columns):
 #   Column                      Non-Null Count   Dtype  
---  ------                      --------------   -----  
 0   MASTER_CLAIM_ID             304770 non-null  float64
 1   PARENT_SUBSCRIBER_ID        304770 non-null  int64  
 2   PAYED_ON                    304770 non-null  object 
 3   SOURCE                      304770 non-null  object 
 4   TYPE_NAME                   304770 non-null  object 
 5   PROVIDER_ID                 228236 non-null  float64
 6   INVOICE_VALUE               304770 non-null  float64
 7   BEARING_VALUE               285768 non-null  float64
 8   PARTICIPATION_VAL_DISCOUNT  304739 non-null  float64
 9   SUBSCRIBER_ID               304770 non-null  int64  
 10  DOCTOR_USER_ID              0 non-null       float64
 11  SPECIALTY_ID                0 non-null       float64
 12  CLAIM_ID                    0 non-null       float64
 13  DISEASE_FO                  0 non-null       object 
 14  TYPE                        304770 non-null  object 
 15  STATE_NA                    136057 non-null  object 
 16  CITY_NA                     139866 non-null  object 
 17  DATE_OF_BIRTH               304770 non-null  object 
 18  POLICY_ID                   304770 non-null  int64  
 19  CUST_ID                     304770 non-null  int64  
 20  GENDER_FO                   304770 non-null  object 
 21  ID_NUM_PASSPORT             304770 non-null  object 
dtypes: float64(8), int64(4), object(10)
memory usage: 53.5+ MB
```

# Return data with value type Glass

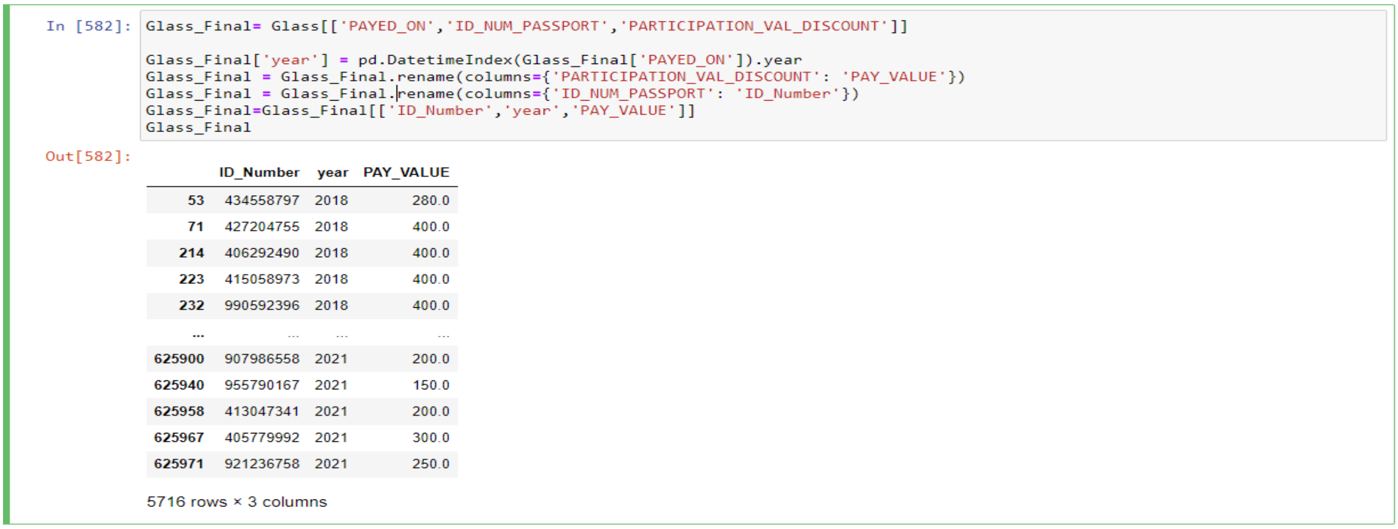


```
In [581]: Glass= data2[data2["TYPE"] == 'Glass']
Glass
```

Out[581]:

	MASTER_CLAIM_ID	PARENT_SUBSCRIBER_ID	PAYED_ON	SOURCE	TYPE_NAME	PROVIDER_ID	INVOICE_VALUE	BEARING_VALUE	PARTICI
53	59.0	170	10-MAY-18	REIMBURSEMENT	Glass	NaN	280.0	0.0	
71	77.0	63	10-MAY-18	REIMBURSEMENT	Glass	NaN	450.0	NaN	
214	223.0	220	08-JUN-18	REIMBURSEMENT	Glass	NaN	400.0	NaN	
223	232.0	220	09-JUN-18	REIMBURSEMENT	Glass	NaN	400.0	NaN	
232	241.0	141	18-JUN-18	REIMBURSEMENT	Glass	NaN	400.0	NaN	
...	...	...	...	...	...	...	...	...	
625900	513287.0	83558	25-SEP-21	REIMBURSEMENT	Glass	NaN	200.0	0.0	
625940	513326.0	74965	26-SEP-21	REIMBURSEMENT	Glass	NaN	150.0	0.0	
625958	513343.0	74959	14-OCT-21	REIMBURSEMENT	Glass	NaN	200.0	0.0	
625967	513353.0	74959	14-OCT-21	REIMBURSEMENT	Glass	NaN	300.0	0.0	
625971	513357.0	74959	14-OCT-21	REIMBURSEMENT	Glass	NaN	250.0	0.0	

5716 rows × 22 columns



```
In [582]: Glass_Final= Glass[['PAYED_ON','ID_NUM_PASSPORT','PARTICIPATION_VAL_DISCOUNT']]

Glass_Final['year'] = pd.DatetimeIndex(Glass_Final['PAYED_ON']).year
Glass_Final = Glass_Final.rename(columns={'PARTICIPATION_VAL_DISCOUNT': 'PAY_VALUE'})
Glass_Final = Glass_Final.rename(columns={'ID_NUM_PASSPORT': 'ID_Number'})
Glass_Final=Glass_Final[['ID_Number','year','PAY_VALUE']]
Glass_Final
```

Out[582]:

	ID_Number	year	PAY_VALUE
53	434558797	2018	280.0
71	427204755	2018	400.0
214	406292490	2018	400.0
223	415058973	2018	400.0
232	990592396	2018	400.0
...	...	...	...
625900	907986558	2021	200.0
625940	955790167	2021	150.0
625958	413047341	2021	200.0
625967	405779992	2021	300.0
625971	921236758	2021	250.0

5716 rows × 3 columns

In [632]:
```
# duplicated Rows in Data
Glass_Final.duplicated().sum()
```

Out[632]:
```
54
```

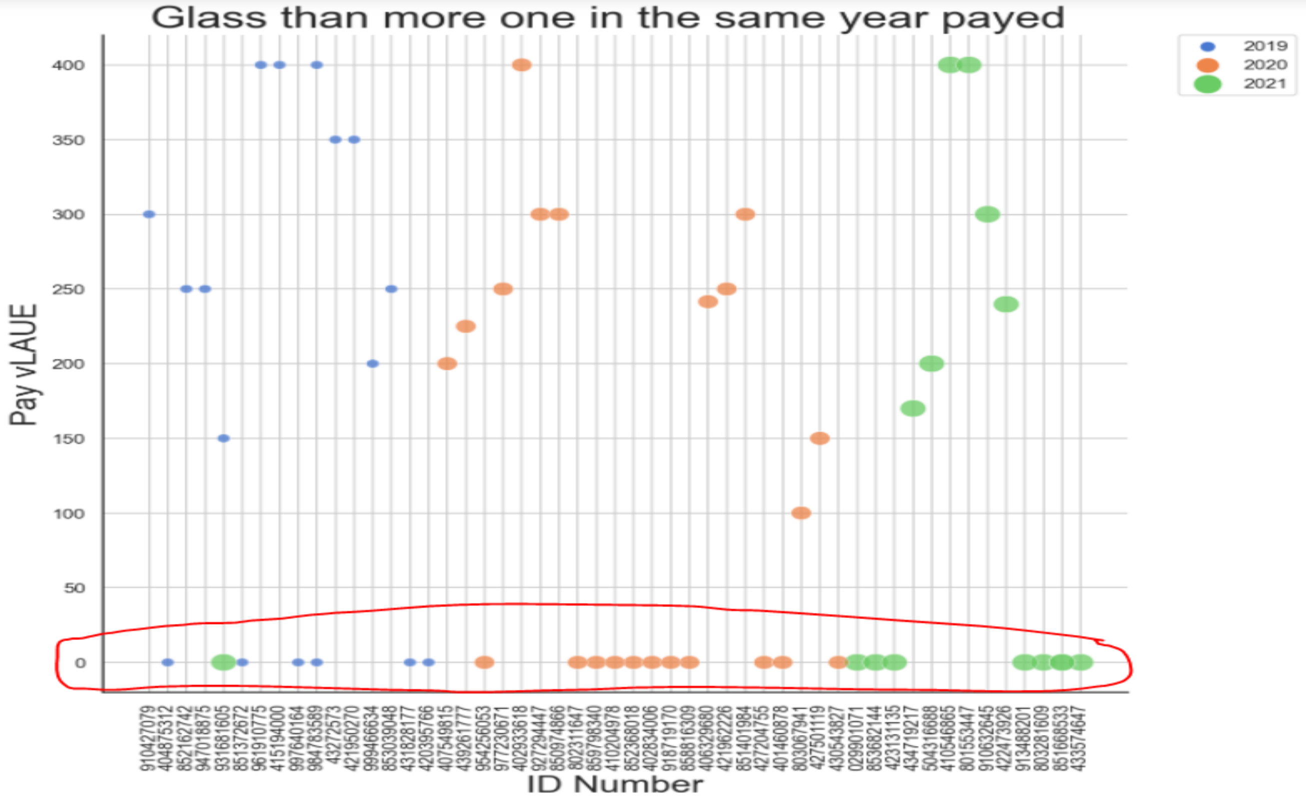
In [633]:
```
# subscribers have than one more Glass in a year
duplicated_rows=Glass_Final[Glass_Final.duplicated(keep=False)]
duplicated_rows
```

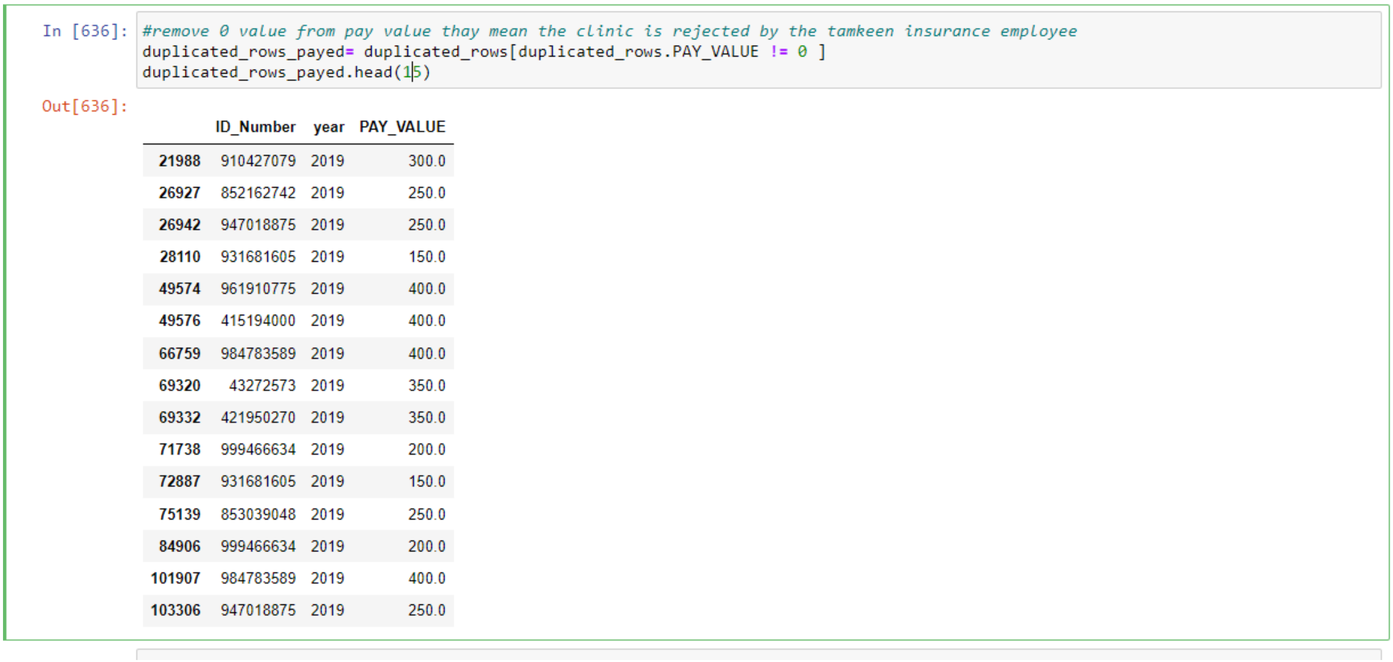
Out[633]:

	ID_Number	year	PAY_VALUE
21988	910427079	2019	300.00
22209	404875312	2019	0.00
25888	404875312	2019	0.00
26927	852162742	2019	250.00
26942	947018875	2019	250.00
...	...	...	...
605304	851668533	2021	0.00
609716	434719217	2021	170.00
615991	422473926	2021	239.75
618760	427501119	2020	150.00
622037	504316688	2021	200.00

107 rows × 3 columns

In [635]:
```
sns.set_style("ticks")
sns.set_theme(style="white")
f, ax = plt.subplots(1,1, figsize=(10, 10))
ax = sns.scatterplot(x="ID_Number", y='PAY_VALUE', data=duplicated_rows,
                     hue='year',sizes = (50, 200),size="year" , alpha=0.5, palette="muted")
# Customize the axes and title
ax.set_title("Glass than more one in the same year payed " ,fontsize = 25)
ax.set_xlabel("ID Number" ,fontsize = 20 )
ax.set_ylabel("Pay vLAUE" ,fontsize = 20)
# Remove top and right borders
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
plt.xticks(rotation=90)
ax.grid()
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
#ax.set_xlim(left=1, right=50)
#ticks=[100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,1400,1500,1600,1700,1800,1900,2000,2100,2200,2300,2400,2500,2600,
#ax.set_yticks(ticks)
#ax.set_ylim(bottom=100, top=3400 )

plt.show()
```



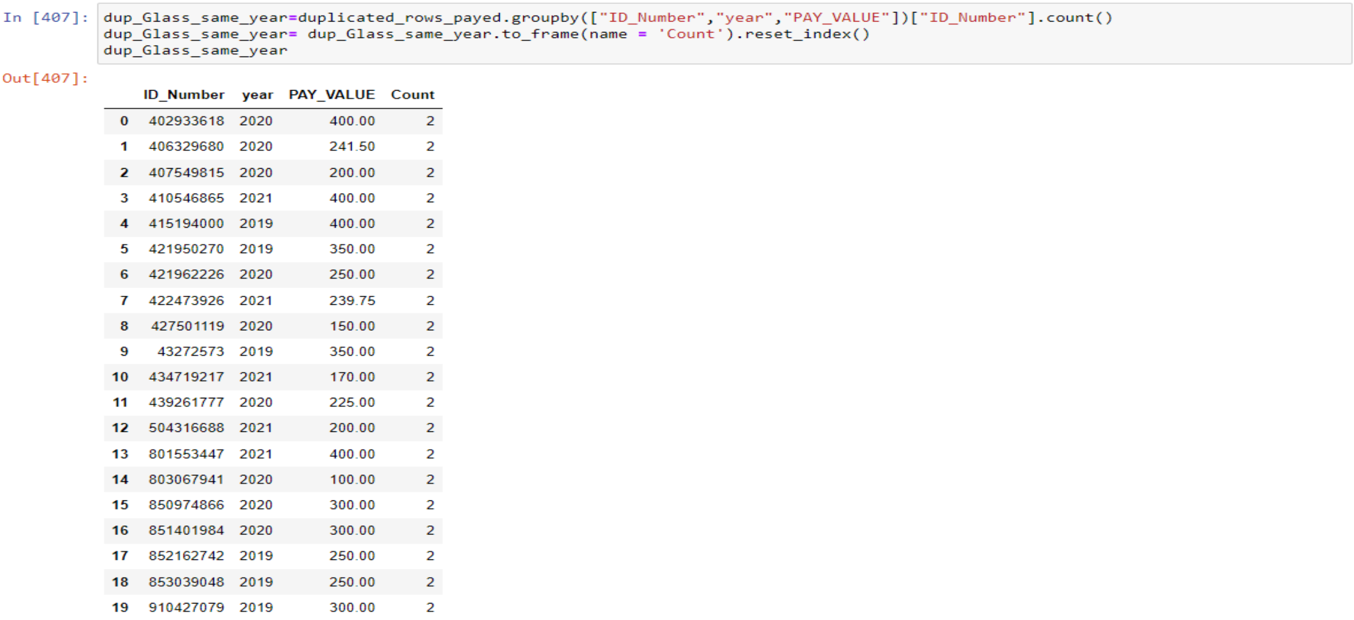


In [636]:

```
#remove 0 value from pay value thay mean the clinic is rejected by the tamkeen insurance employee
duplicated_rows_payed= duplicated_rows[duplicated_rows.PAY_VALUE != 0 ]
duplicated_rows_payed.head(15)
```

Out[636]:

	ID_Number	year	PAY_VALUE
21988	910427079	2019	300.0
26927	852162742	2019	250.0
26942	947018875	2019	250.0
28110	931681605	2019	150.0
49574	961910775	2019	400.0
49576	415194000	2019	400.0
66759	984783589	2019	400.0
69320	43272573	2019	350.0
69332	421950270	2019	350.0
71738	999466634	2019	200.0
72887	931681605	2019	150.0
75139	853039048	2019	250.0
84906	999466634	2019	200.0
101907	984783589	2019	400.0
103306	947018875	2019	250.0

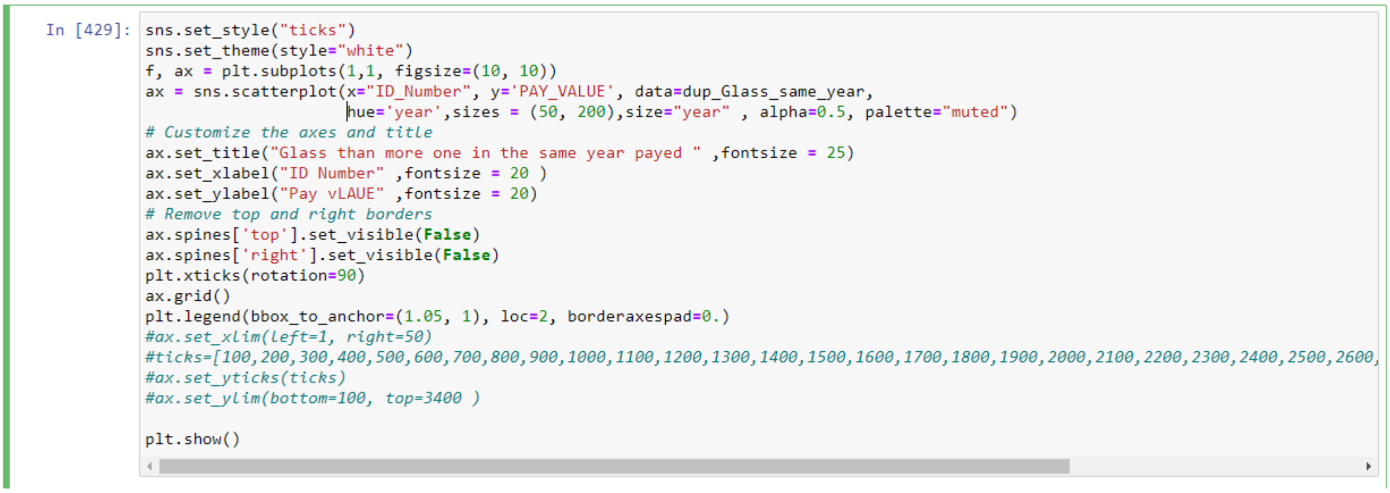


In [407]:

```
dup_Glass_same_year=duplicated_rows_payed.groupby(["ID_Number","year","PAY_VALUE"])["ID_Number"].count()
dup_Glass_same_year= dup_Glass_same_year.to_frame(name = 'Count').reset_index()
dup_Glass_same_year
```

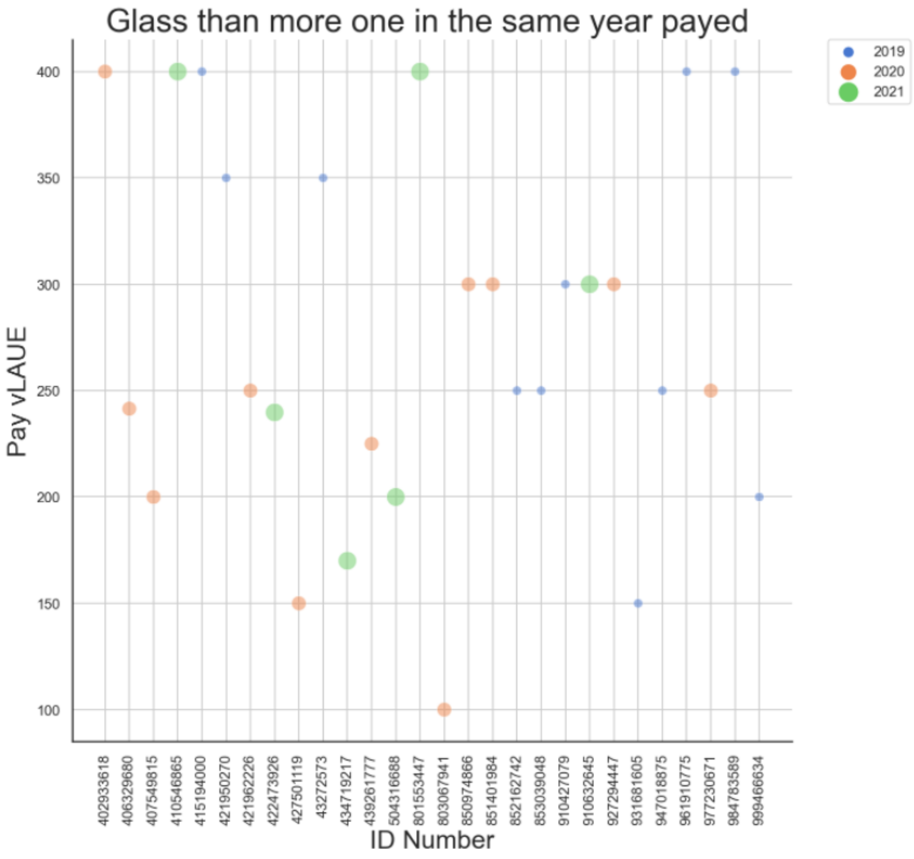
Out[407]:

	ID_Number	year	PAY_VALUE	Count
0	402933618	2020	400.00	2
1	406329680	2020	241.50	2
2	407549815	2020	200.00	2
3	410546865	2021	400.00	2
4	415194000	2019	400.00	2
5	421950270	2019	350.00	2
6	421962226	2020	250.00	2
7	422473926	2021	239.75	2
8	427501119	2020	150.00	2
9	43272573	2019	350.00	2
10	434719217	2021	170.00	2
11	439261777	2020	225.00	2
12	504316688	2021	200.00	2
13	801553447	2021	400.00	2
14	803067941	2020	100.00	2
15	850974866	2020	300.00	2
16	851401984	2020	300.00	2
17	852162742	2019	250.00	2
18	853039048	2019	250.00	2
19	910427079	2019	300.00	2



In [429]:

```
sns.set_style("ticks")
sns.set_theme(style="white")
f, ax = plt.subplots(1,1, figsize=(10, 10))
ax = sns.scatterplot(x="ID_Number", y='PAY_VALUE', data=dup_Glass_same_year,
                     hue='year',sizes = (50, 200),size="year" , alpha=0.5, palette="muted")
# Customize the axes and title
ax.set_title("Glass than more one in the same year payed " ,fontsize = 25)
ax.set_xlabel("ID Number" ,fontsize = 20 )
ax.set_ylabel("Pay vLAUE" ,fontsize = 20)
# Remove top and right borders
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
plt.xticks(rotation=90)
ax.grid()
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
#ax.set_xlim(left=1, right=50)
#ticks=[100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,1400,1500,1600,1700,1800,1900,2000,2100,2200,2300,2400,2500,2600,
#ax.set_yticks(ticks)
#ax.set_ylim(bottom=100, top=3400 )

plt.show()
```



In [391]:
```
# duplicated Id_Number in data fram
Glass_Final["ID_Number"].duplicated().sum()
```

Out[391]:
```
1900
```

In [502]:
```
duplicate=Glass_Final[Glass_Final["ID_Number"].duplicated(keep= False)]
duplicate
#duplicate = duplicate[duplicate.PAY_VALUE != 0 ]
duplicate
```

Out[502]:

	ID_Number	year	PAY_VALUE
53	434558797	2018	280.0
71	427204755	2018	400.0
214	406292490	2018	400.0
223	415058973	2018	400.0
232	990592396	2018	400.0
...	...	...	...
619340	405066622	2021	250.0
619654	852371376	2021	400.0
620248	410846356	2021	300.0
621979	431766534	2021	400.0
622037	504316688	2021	200.0

3176 rows × 3 columns

In [503]:

```
duplicate=duplicate.groupby(["ID_Number","year"])["ID_Number"].count()
duplicate= duplicate.to_frame(name = 'Count').reset_index()
duplicate= duplicate.sort_values(by='Count', ascending=False)
duplicate
```

Out[503]:

	ID_Number	year	Count
549	420395766	2019	4
959	431828177	2019	4
2470	984783589	2019	4
1268	802724195	2019	3
2256	948186309	2020	3
...	...	...	...
939	431605740	2018	1
938	431603166	2020	1
936	431523166	2020	1
935	431523166	2019	1
2535	999818933	2020	1

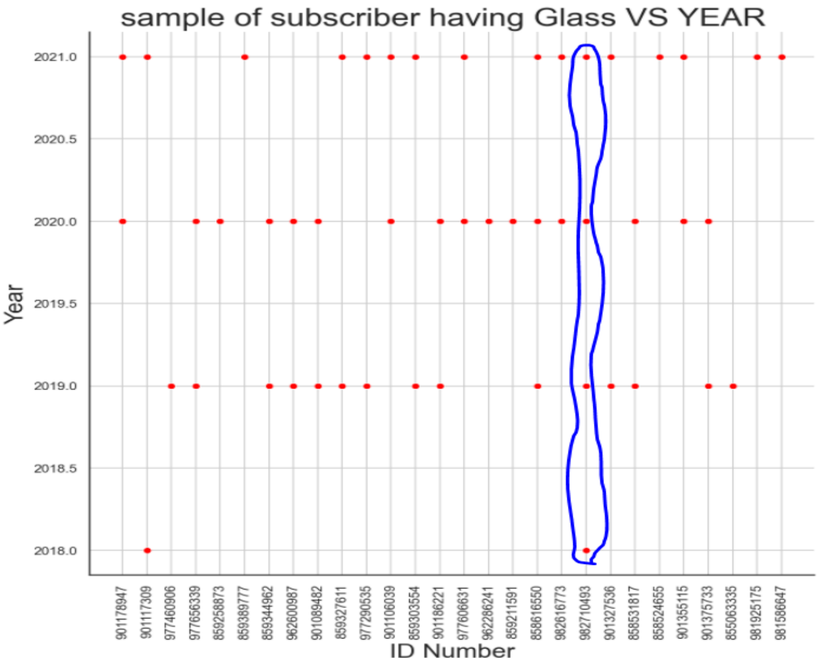
2536 rows × 3 columns

In [661]:

```
dup_one=duplicate.loc[(duplicate.Count == 1)]
y=dup_one.head(50)
```

In [666]:

```
sns.set_style("ticks")
sns.set_theme(style="white")
f, ax = plt.subplots(1,1, figsize=(10, 10))
ax = sns.scatterplot(x="ID_Number", y='year', data=y ,color='red')
# Customize the axes and title
ax.set_title("sample of subscriber having Glass VS YEAR  " ,fontsize = 25)
ax.set_xlabel("ID Number" ,fontsize = 20 )
ax.set_ylabel("Year" ,fontsize = 20)
# Remove top and right borders
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
plt.xticks(rotation=90)
ax.grid()
#ax.set_xlim(left=1, right=50)
#ticks=[100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,1400,1500,1600,1700,1800,1900,2000,2100,2200,2300,2400,2500,2600,
#ax.set_yticks(ticks)
#ax.set_ylim(bottom=100, top=3400 )
plt.show()
```



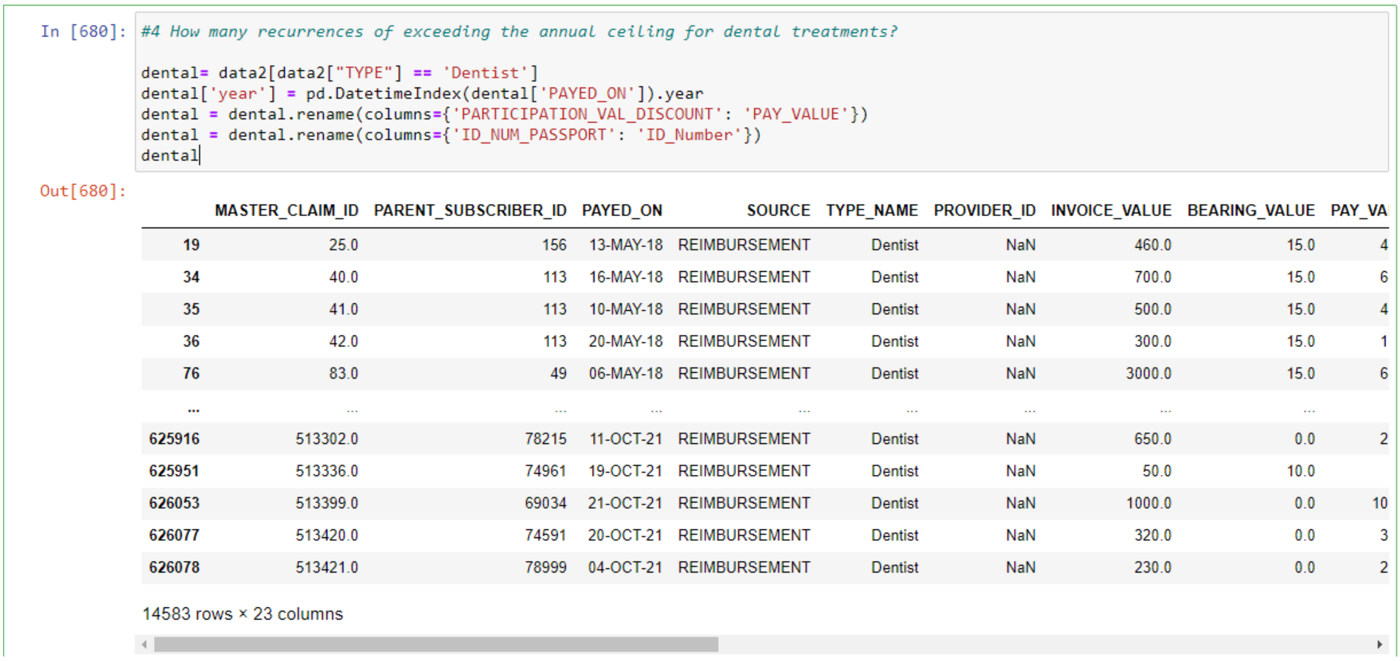
In [675]:

```
ID_NUM982= Glass_Final[Glass_Final.ID_Number == '982710493' ]
ID_NUM982
```

Out[675]:

	ID_Number	year	PAY_VALUE
18864	982710493	2018	250.0
81280	982710493	2019	190.0
340302	982710493	2020	0.0
556699	982710493	2021	250.0

4. How many recurrences of exceeding the annual ceiling for dental treatments?(more than 700)



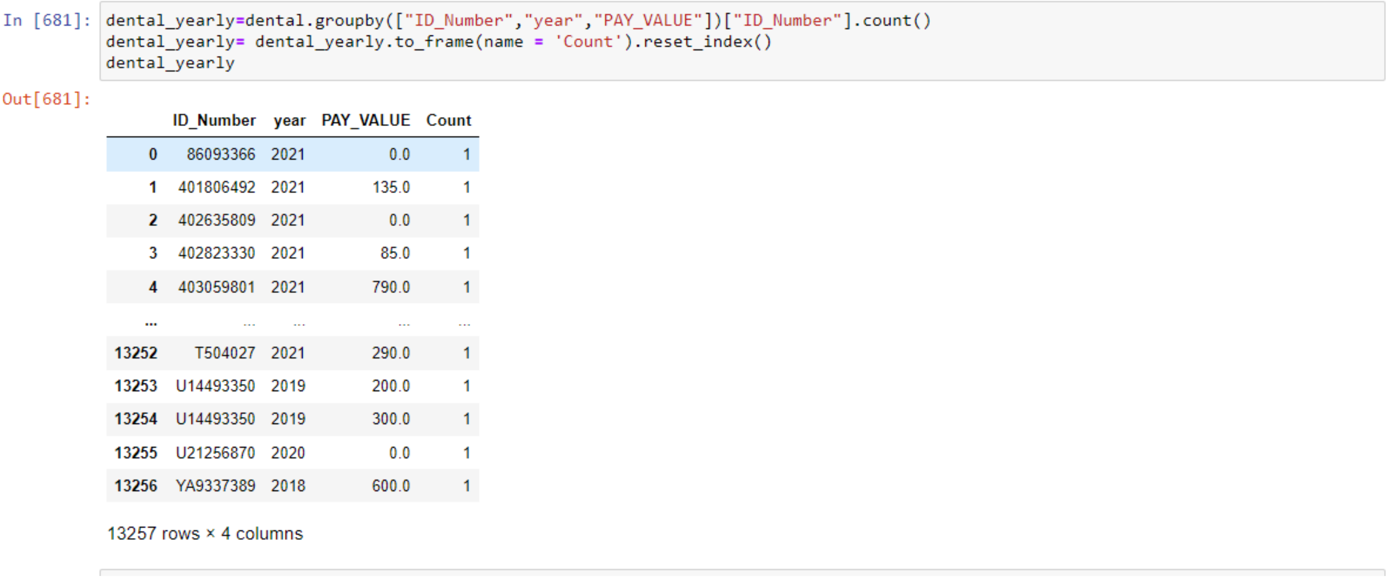
In [680]:

```
#4 How many recurrences of exceeding the annual ceiling for dental treatments?

dental= data2[data2["TYPE"] == 'Dentist']
dental['year'] = pd.DatetimeIndex(dental['PAYED_ON']).year
dental = dental.rename(columns={'PARTICIPATION_VAL_DISCOUNT': 'PAY_VALUE'})
dental = dental.rename(columns={'ID_NUM_PASSPORT': 'ID_Number'})
dental
```

Out[680]:

	MASTER_CLAIM_ID	PARENT_SUBSCRIBER_ID	PAYED_ON	SOURCE	TYPE_NAME	PROVIDER_ID	INVOICE_VALUE	BEARING_VALUE	PAY_VA
19	25.0	156	13-MAY-18	REIMBURSEMENT	Dentist	NaN	460.0	15.0	4
34	40.0	113	16-MAY-18	REIMBURSEMENT	Dentist	NaN	700.0	15.0	6
35	41.0	113	10-MAY-18	REIMBURSEMENT	Dentist	NaN	500.0	15.0	4
36	42.0	113	20-MAY-18	REIMBURSEMENT	Dentist	NaN	300.0	15.0	1
76	83.0	49	06-MAY-18	REIMBURSEMENT	Dentist	NaN	3000.0	15.0	6
...	...	...	...	...	...	...	...	...	
625916	513302.0	78215	11-OCT-21	REIMBURSEMENT	Dentist	NaN	650.0	0.0	2
625951	513336.0	74961	19-OCT-21	REIMBURSEMENT	Dentist	NaN	50.0	10.0	
626053	513399.0	69034	21-OCT-21	REIMBURSEMENT	Dentist	NaN	1000.0	0.0	10
626077	513420.0	74591	20-OCT-21	REIMBURSEMENT	Dentist	NaN	320.0	0.0	3
626078	513421.0	78999	04-OCT-21	REIMBURSEMENT	Dentist	NaN	230.0	0.0	2

14583 rows × 23 columns



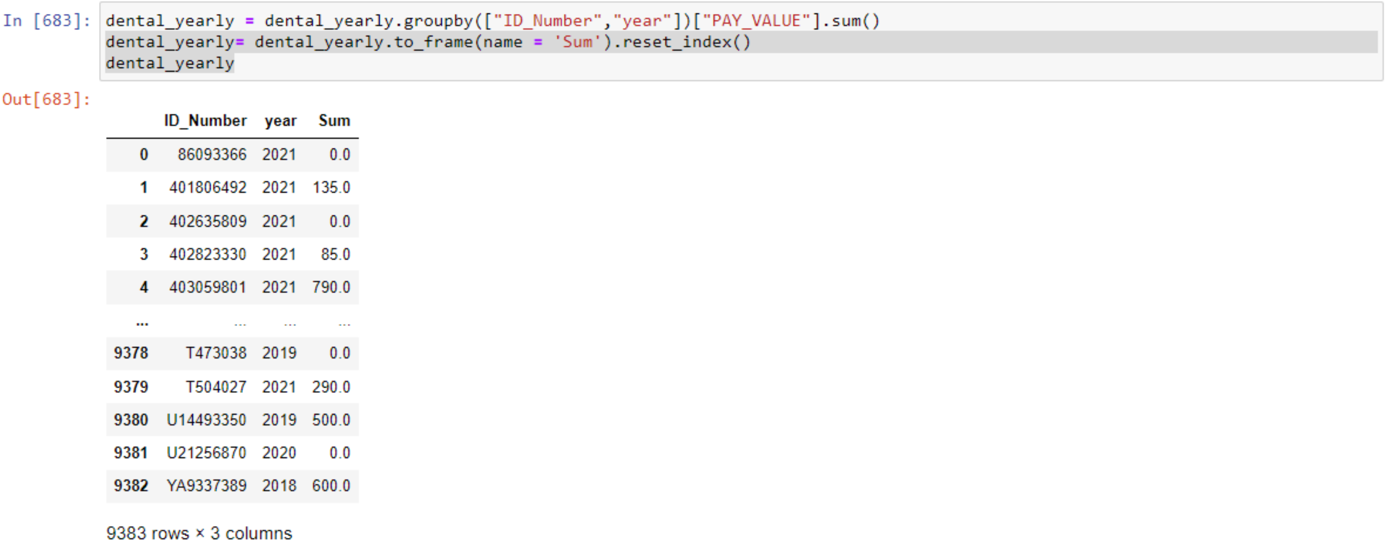
In [681]:

```
dental_yearly=dental.groupby(["ID_Number","year","PAY_VALUE"])["ID_Number"].count()
dental_yearly= dental_yearly.to_frame(name = 'Count').reset_index()
dental_yearly
```

Out[681]:

	ID_Number	year	PAY_VALUE	Count
0	86093366	2021	0.0	1
1	401806492	2021	135.0	1
2	402635809	2021	0.0	1
3	402823330	2021	85.0	1
4	403059801	2021	790.0	1
...	...	...	...	...
13252	T504027	2021	290.0	1
13253	U14493350	2019	200.0	1
13254	U14493350	2019	300.0	1
13255	U21256870	2020	0.0	1
13256	YA9337389	2018	600.0	1

13257 rows × 4 columns



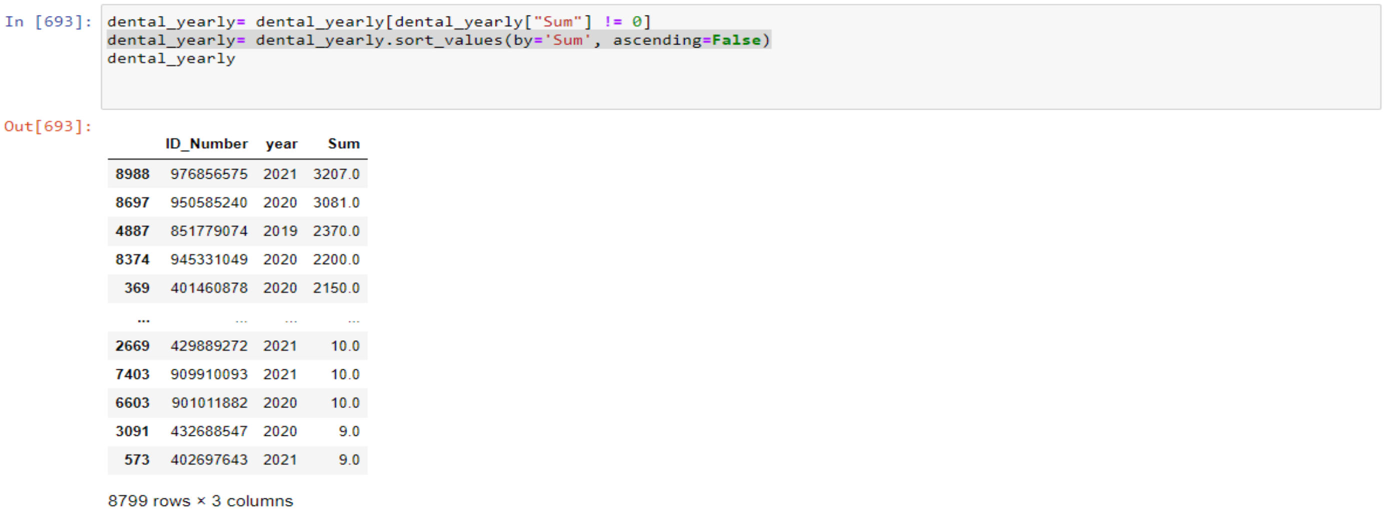
In [683]:

```
dental_yearly = dental_yearly.groupby(["ID_Number","year"])["PAY_VALUE"].sum()
dental_yearly= dental_yearly.to_frame(name = 'Sum').reset_index()
dental_yearly
```

Out[683]:

	ID_Number	year	Sum
0	86093366	2021	0.0
1	401806492	2021	135.0
2	402635809	2021	0.0
3	402823330	2021	85.0
4	403059801	2021	790.0
...	...	...	...
9378	T473038	2019	0.0
9379	T504027	2021	290.0
9380	U14493350	2019	500.0
9381	U21256870	2020	0.0
9382	YA9337389	2018	600.0

9383 rows × 3 columns

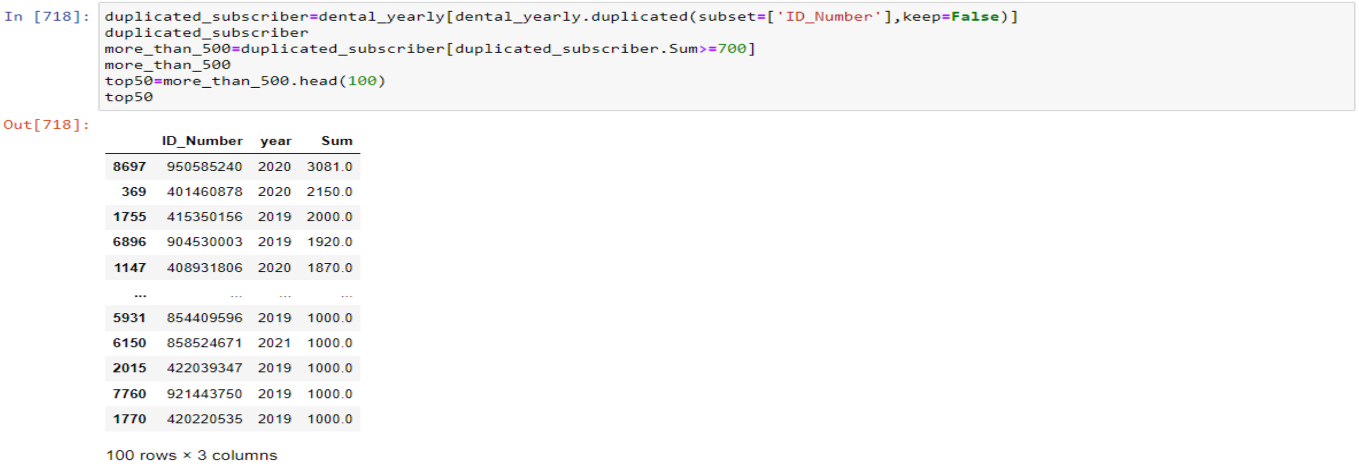


```
In [693]: dental_yearly= dental_yearly[dental_yearly["Sum"] != 0]
          dental_yearly= dental_yearly.sort_values(by='Sum', ascending=False)
          dental_yearly
```

Out[693]:

	ID_Number	year	Sum
8988	976856575	2021	3207.0
8697	950585240	2020	3081.0
4887	851779074	2019	2370.0
8374	945331049	2020	2200.0
369	401460878	2020	2150.0
...	...	...	...
2669	429889272	2021	10.0
7403	909910093	2021	10.0
6603	901011882	2020	10.0
3091	432688547	2020	9.0
573	402697643	2021	9.0

8799 rows × 3 columns

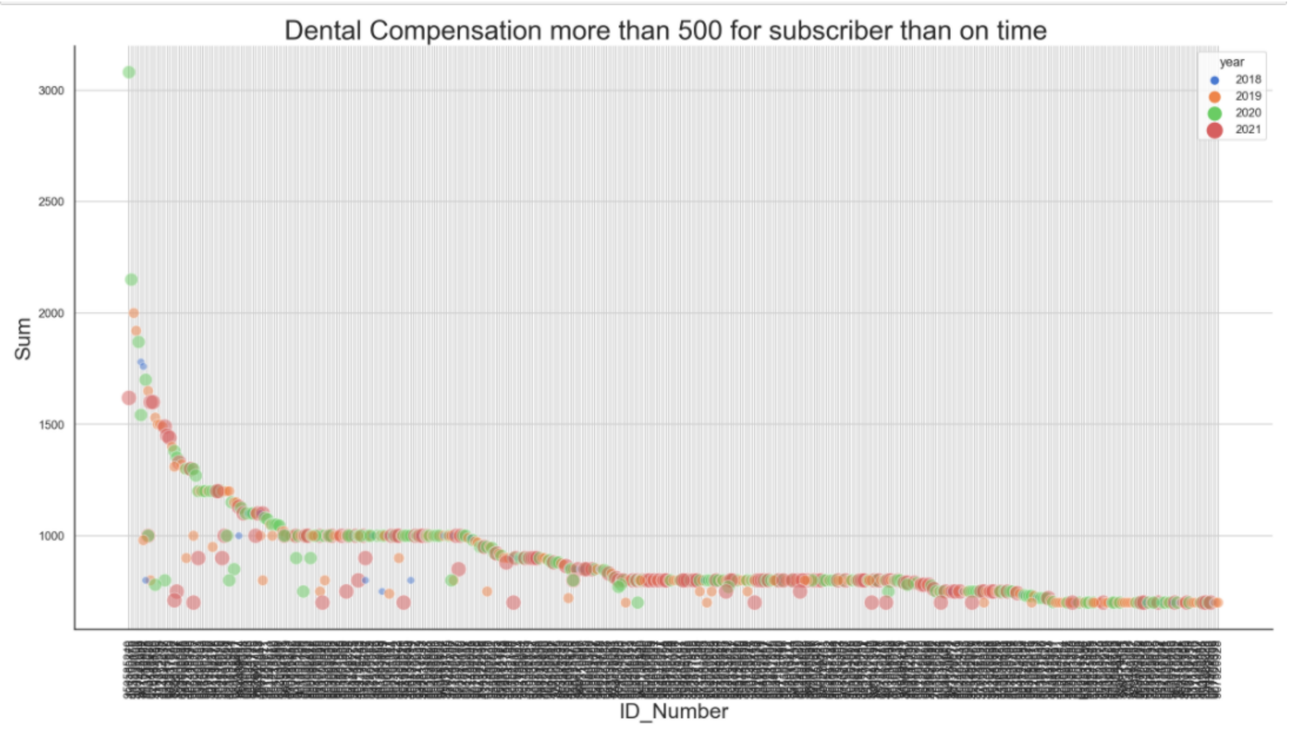


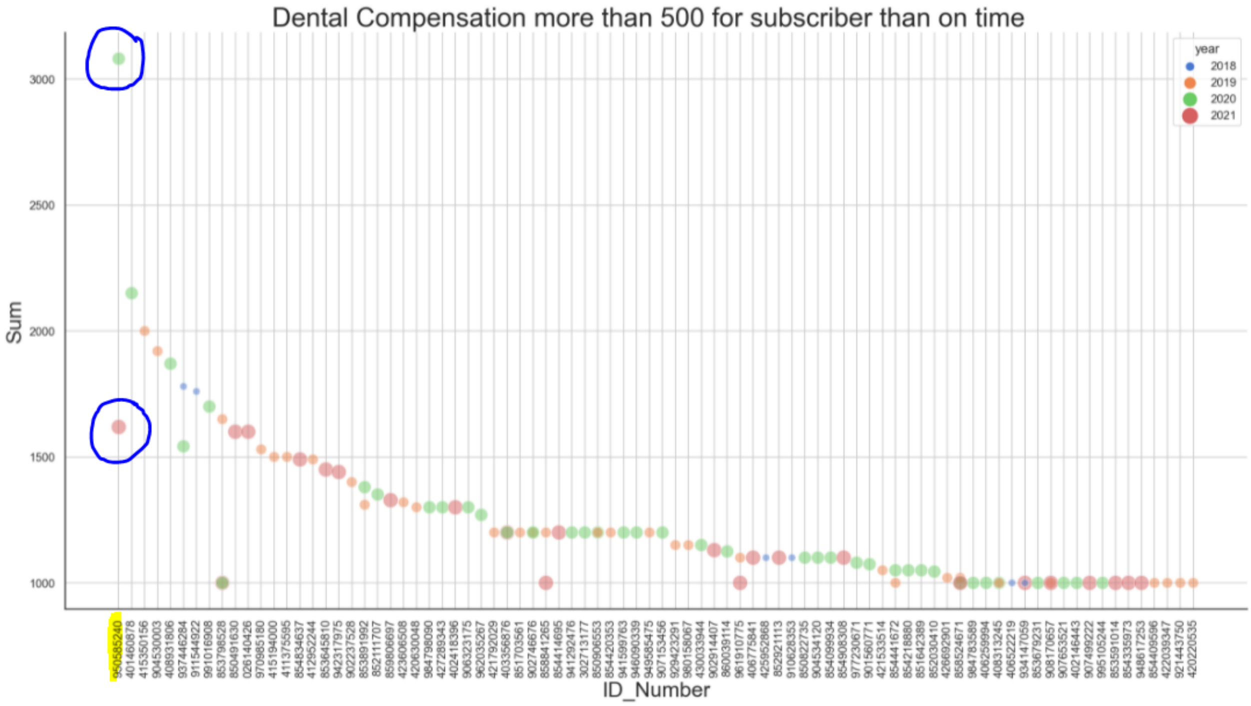
```
In [718]: duplicated_subscriber=dental_yearly[dental_yearly.duplicated(subset=['ID_Number'],keep=False)]
          duplicated_subscriber
          more_than_500=duplicated_subscriber[duplicated_subscriber.Sum>=700]
          more_than_500
          top50=more_than_500.head(100)
          top50
```

Out[718]:

	ID_Number	year	Sum
8697	950585240	2020	3081.0
369	401460878	2020	2150.0
1755	415350156	2019	2000.0
6896	904530003	2019	1920.0
1147	408931806	2020	1870.0
...	...	...	...
5931	854409596	2019	1000.0
6150	858524671	2021	1000.0
2015	422039347	2019	1000.0
7760	921443750	2019	1000.0
1770	420220535	2019	1000.0

100 rows × 3 columns



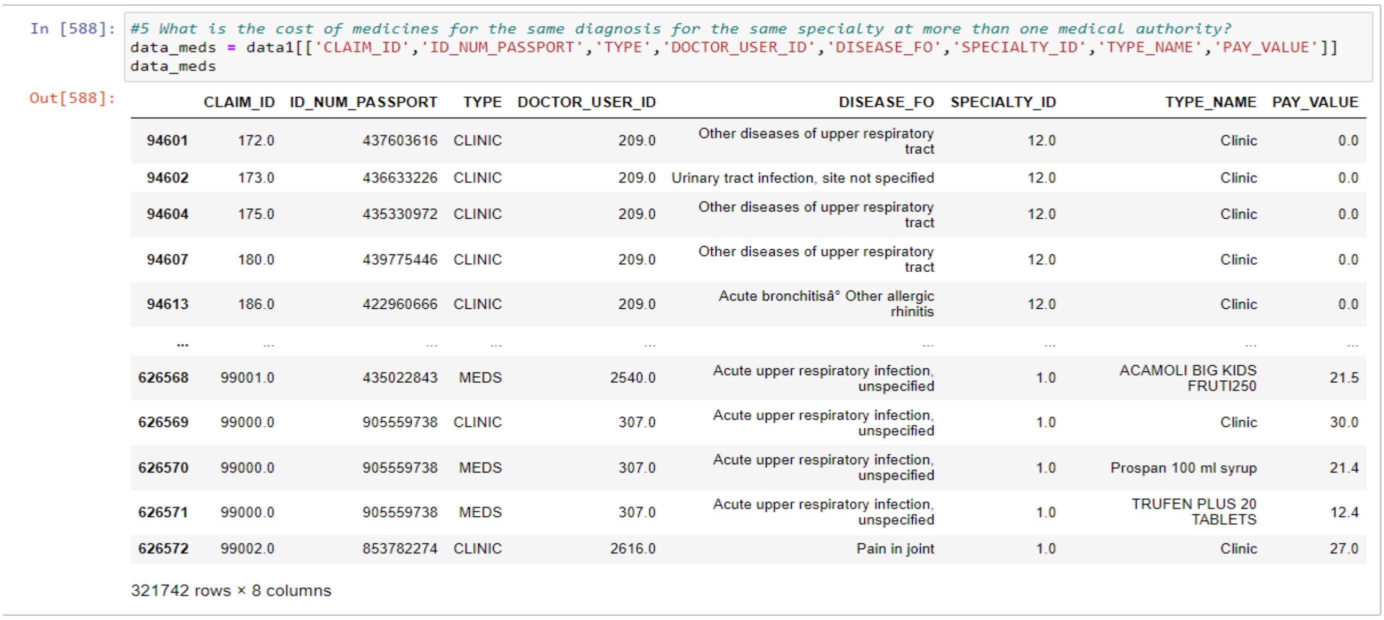


```
In [720]: ID_950585240= duplicated_subscriber[duplicated_subscriber.ID_Number=='950585240']
          ID_950585240
```

Out[720]:

	ID_Number	year	Sum
8697	950585240	2020	3081.0
8698	950585240	2021	1619.0
8696	950585240	2018	130.0

5. What is the cost of medicines for the same diagnosis for the same specialty at more than one medical authority?



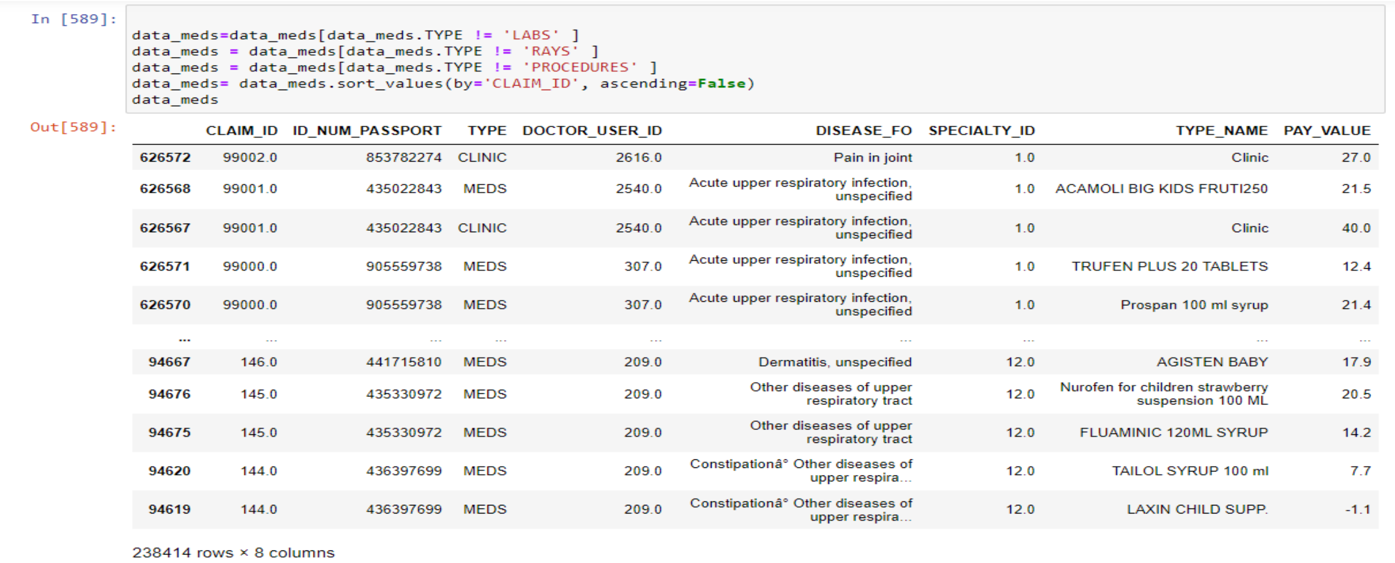
In [588]:

```
#5 What is the cost of medicines for the same diagnosis for the same specialty at more than one medical authority?
data_meds = data1[['CLAIM_ID','ID_NUM_PASSPORT','TYPE','DOCTOR_USER_ID','DISEASE_FO','SPECIALTY_ID','TYPE_NAME','PAY_VALUE']]
data_meds
```

Out[588]:

	CLAIM_ID	ID_NUM_PASSPORT	TYPE	DOCTOR_USER_ID	DISEASE_FO	SPECIALTY_ID	TYPE_NAME	PAY_VALUE
94601	172.0	437603616	CLINIC	209.0	Other diseases of upper respiratory tract	12.0	Clinic	0.0
94602	173.0	436633226	CLINIC	209.0	Urinary tract infection, site not specified	12.0	Clinic	0.0
94604	175.0	435330972	CLINIC	209.0	Other diseases of upper respiratory tract	12.0	Clinic	0.0
94607	180.0	439775446	CLINIC	209.0	Other diseases of upper respiratory tract	12.0	Clinic	0.0
94613	186.0	422960666	CLINIC	209.0	Acute bronchitisâ° Other allergic rhinitis	12.0	Clinic	0.0
...	...	...	...	...	...	...	...	...
626568	99001.0	435022843	MEDS	2540.0	Acute upper respiratory infection, unspecified	1.0	ACAMOLI BIG KIDS FRUTI250	21.5
626569	99000.0	905559738	CLINIC	307.0	Acute upper respiratory infection, unspecified	1.0	Clinic	30.0
626570	99000.0	905559738	MEDS	307.0	Acute upper respiratory infection, unspecified	1.0	Prospan 100 ml syrup	21.4
626571	99000.0	905559738	MEDS	307.0	Acute upper respiratory infection, unspecified	1.0	TRUFEN PLUS 20 TABLETS	12.4
626572	99002.0	853782274	CLINIC	2616.0	Pain in joint	1.0	Clinic	27.0

321742 rows × 8 columns



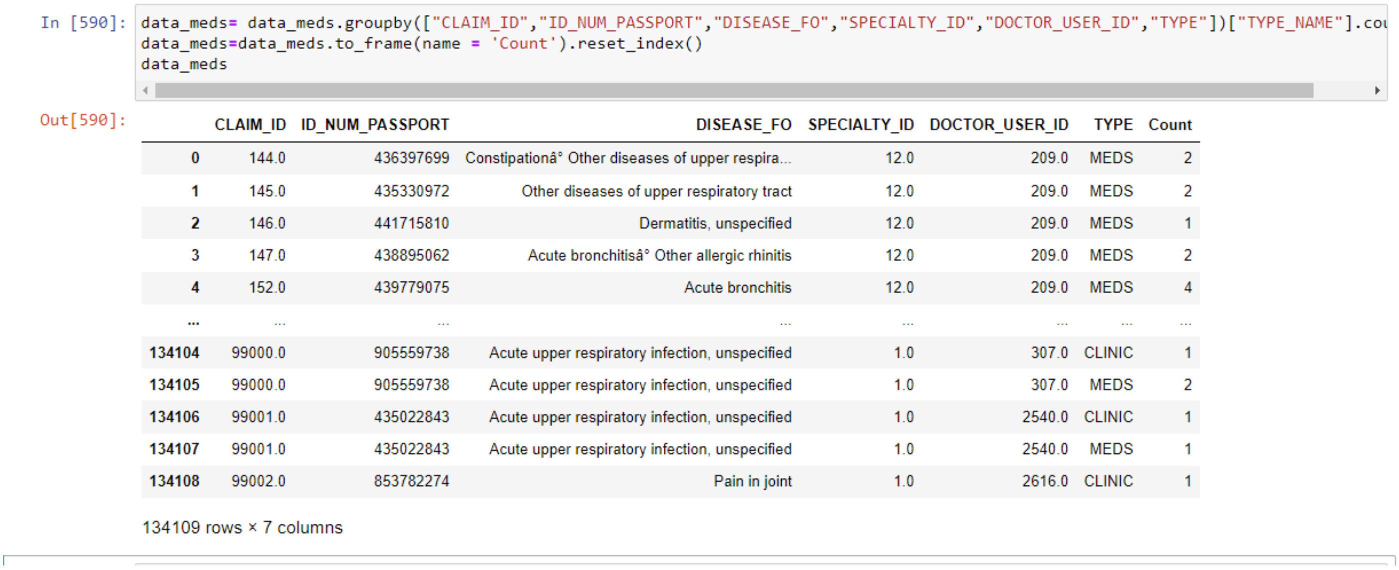
In [589]:

```
data_meds=data_meds[data_meds.TYPE != 'LABS' ]
data_meds = data_meds[data_meds.TYPE != 'RAYS' ]
data_meds = data_meds[data_meds.TYPE != 'PROCEDURES' ]
data_meds= data_meds.sort_values(by='CLAIM_ID', ascending=False)
data_meds
```

Out[589]:

	CLAIM_ID	ID_NUM_PASSPORT	TYPE	DOCTOR_USER_ID	DISEASE_FO	SPECIALTY_ID	TYPE_NAME	PAY_VALUE
626572	99002.0	853782274	CLINIC	2616.0	Pain in joint	1.0	Clinic	27.0
626568	99001.0	435022843	MEDS	2540.0	Acute upper respiratory infection, unspecified	1.0	ACAMOLI BIG KIDS FRUTI250	21.5
626567	99001.0	435022843	CLINIC	2540.0	Acute upper respiratory infection, unspecified	1.0	Clinic	40.0
626571	99000.0	905559738	MEDS	307.0	Acute upper respiratory infection, unspecified	1.0	TRUFEN PLUS 20 TABLETS	12.4
626570	99000.0	905559738	MEDS	307.0	Acute upper respiratory infection, unspecified	1.0	Prospan 100 ml syrup	21.4
...	...	...	...	...	...	...	...	...
94667	146.0	441715810	MEDS	209.0	Dermatitis, unspecified	12.0	AGISTEN BABY	17.9
94676	145.0	435330972	MEDS	209.0	Other diseases of upper respiratory tract	12.0	Nurofen for children strawberry suspension 100 ML	20.5
94675	145.0	435330972	MEDS	209.0	Other diseases of upper respiratory tract	12.0	FLUAMINIC 120ML SYRUP	14.2
94620	144.0	436397699	MEDS	209.0	Constipationâ° Other diseases of upper respira...	12.0	TAILOL SYRUP 100 ml	7.7
94619	144.0	436397699	MEDS	209.0	Constipationâ° Other diseases of upper respira...	12.0	LAXIN CHILD SUPP.	-1.1

238414 rows × 8 columns



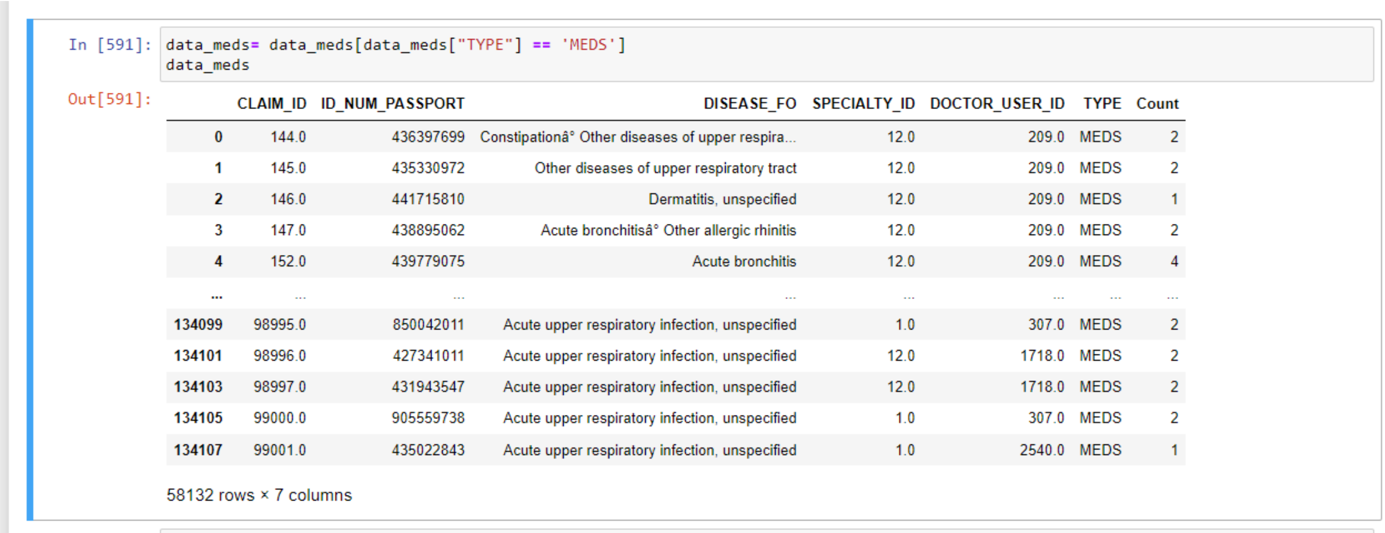
In [590]:

```
data_meds= data_meds.groupby(["CLAIM_ID","ID_NUM_PASSPORT","DISEASE_FO","SPECIALTY_ID","DOCTOR_USER_ID","TYPE"])["TYPE_NAME"].cou
data_meds=data_meds.to_frame(name = 'Count').reset_index()
data_meds
```

Out[590]:

	CLAIM_ID	ID_NUM_PASSPORT	DISEASE_FO	SPECIALTY_ID	DOCTOR_USER_ID	TYPE	Count
0	144.0	436397699	Constipationâ° Other diseases of upper respira...	12.0	209.0	MEDS	2
1	145.0	435330972	Other diseases of upper respiratory tract	12.0	209.0	MEDS	2
2	146.0	441715810	Dermatitis, unspecified	12.0	209.0	MEDS	1
3	147.0	438895062	Acute bronchitisâ° Other allergic rhinitis	12.0	209.0	MEDS	2
4	152.0	439779075	Acute bronchitis	12.0	209.0	MEDS	4
...	...	...	...	...	...	...	...
134104	99000.0	905559738	Acute upper respiratory infection, unspecified	1.0	307.0	CLINIC	1
134105	99000.0	905559738	Acute upper respiratory infection, unspecified	1.0	307.0	MEDS	2
134106	99001.0	435022843	Acute upper respiratory infection, unspecified	1.0	2540.0	CLINIC	1
134107	99001.0	435022843	Acute upper respiratory infection, unspecified	1.0	2540.0	MEDS	1
134108	99002.0	853782274	Pain in joint	1.0	2616.0	CLINIC	1

134109 rows × 7 columns

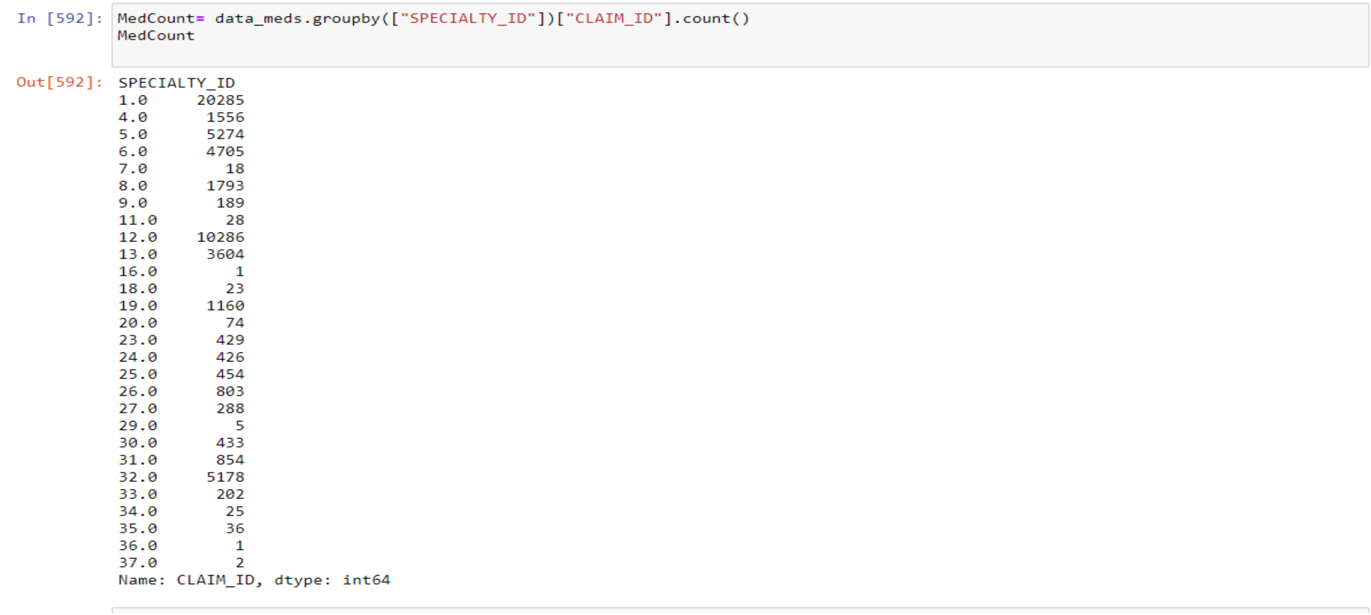


```
In [591]: data_meds= data_meds[data_meds["TYPE"] == 'MEDS']
          data_meds
```

Out[591]:

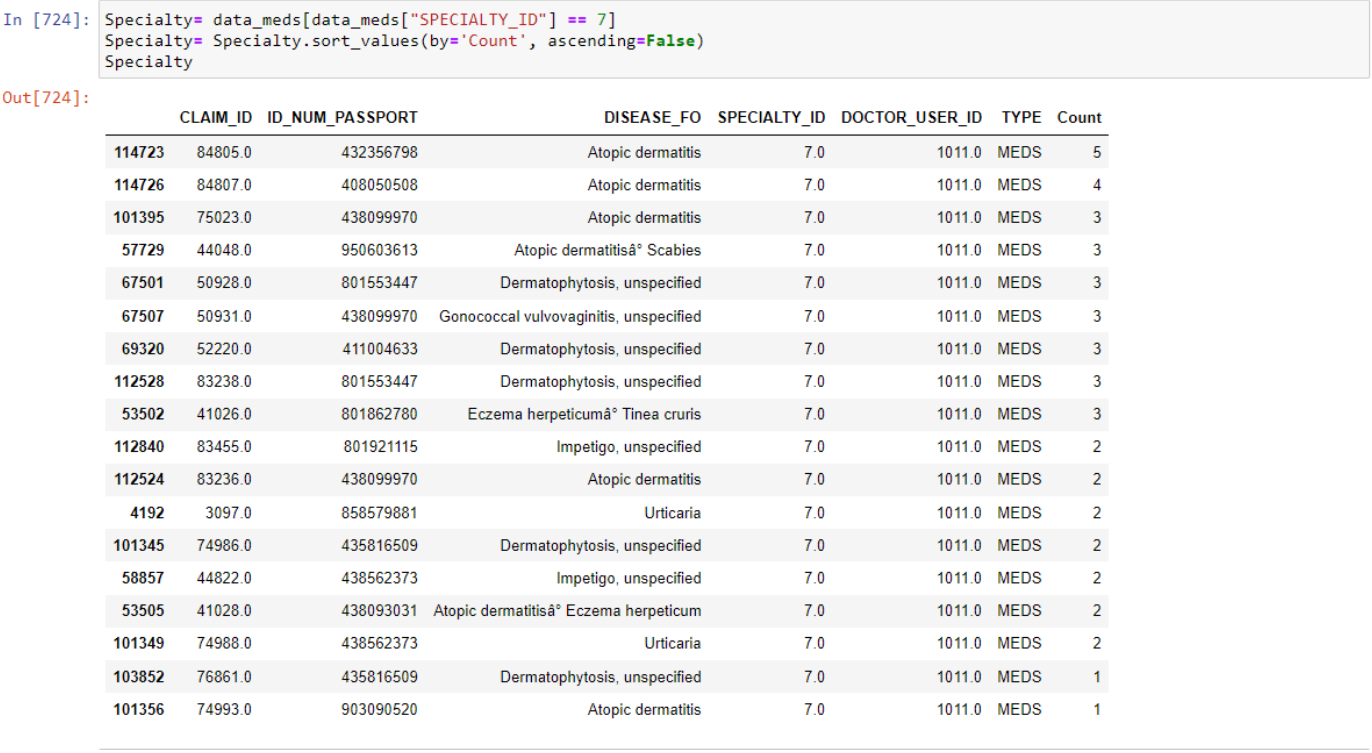
	CLAIM_ID	ID_NUM_PASSPORT	DISEASE_FO	SPECIALTY_ID	DOCTOR_USER_ID	TYPE	Count
0	144.0	436397699	Constipationâ° Other diseases of upper respira...	12.0	209.0	MEDS	2
1	145.0	435330972	Other diseases of upper respiratory tract	12.0	209.0	MEDS	2
2	146.0	441715810	Dermatitis, unspecified	12.0	209.0	MEDS	1
3	147.0	438895062	Acute bronchitisâ° Other allergic rhinitis	12.0	209.0	MEDS	2
4	152.0	439779075	Acute bronchitis	12.0	209.0	MEDS	4
...	...	...	...	...	...	...	...
134099	98995.0	850042011	Acute upper respiratory infection, unspecified	1.0	307.0	MEDS	2
134101	98996.0	427341011	Acute upper respiratory infection, unspecified	12.0	1718.0	MEDS	2
134103	98997.0	431943547	Acute upper respiratory infection, unspecified	12.0	1718.0	MEDS	2
134105	99000.0	905559738	Acute upper respiratory infection, unspecified	1.0	307.0	MEDS	2
134107	99001.0	435022843	Acute upper respiratory infection, unspecified	1.0	2540.0	MEDS	1

58132 rows × 7 columns



```
In [592]: MedCount= data_meds.groupby(["SPECIALTY_ID"])["CLAIM_ID"].count()
          MedCount
```

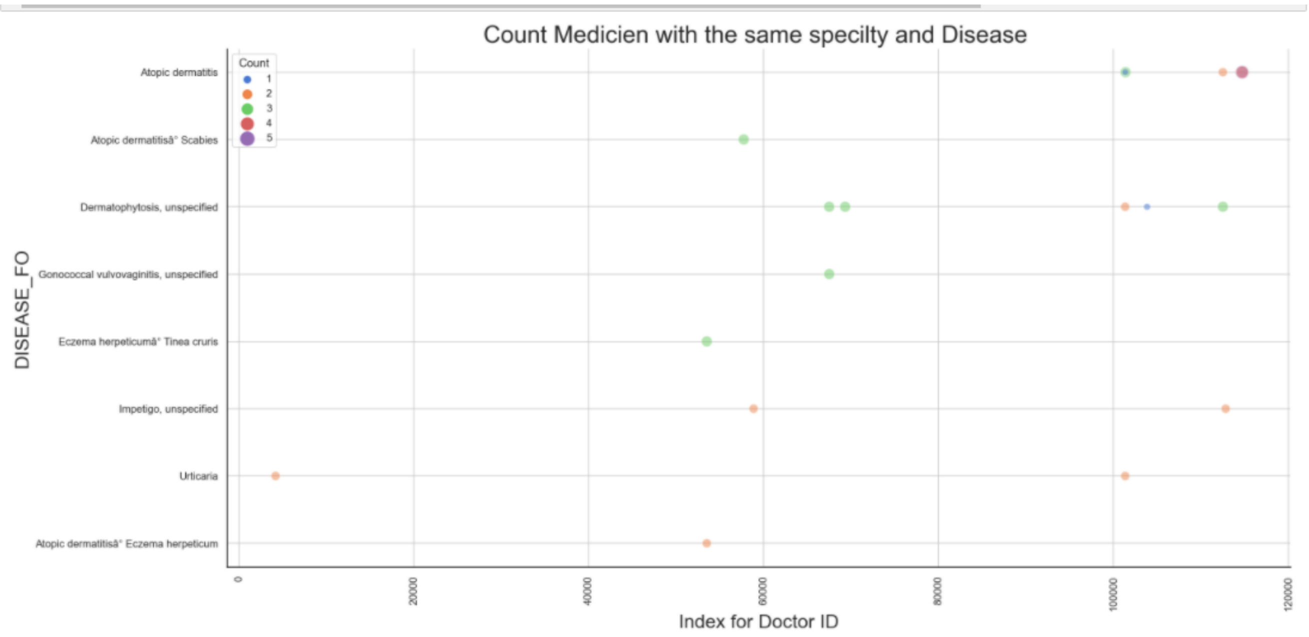
```
Out[592]: SPECIALTY_ID
          1.0     20285
          4.0      1556
          5.0      5274
          6.0      4705
          7.0        18
          8.0      1793
          9.0       189
          11.0       28
          12.0    10286
          13.0     3604
          16.0        1
          18.0       23
          19.0     1160
          20.0       74
          23.0      429
          24.0      426
          25.0      454
          26.0      803
          27.0      288
          29.0        5
          30.0      433
          31.0      854
          32.0     5178
          33.0      202
          34.0       25
          35.0       36
          36.0        1
          37.0        2
          Name: CLAIM_ID, dtype: int64
```



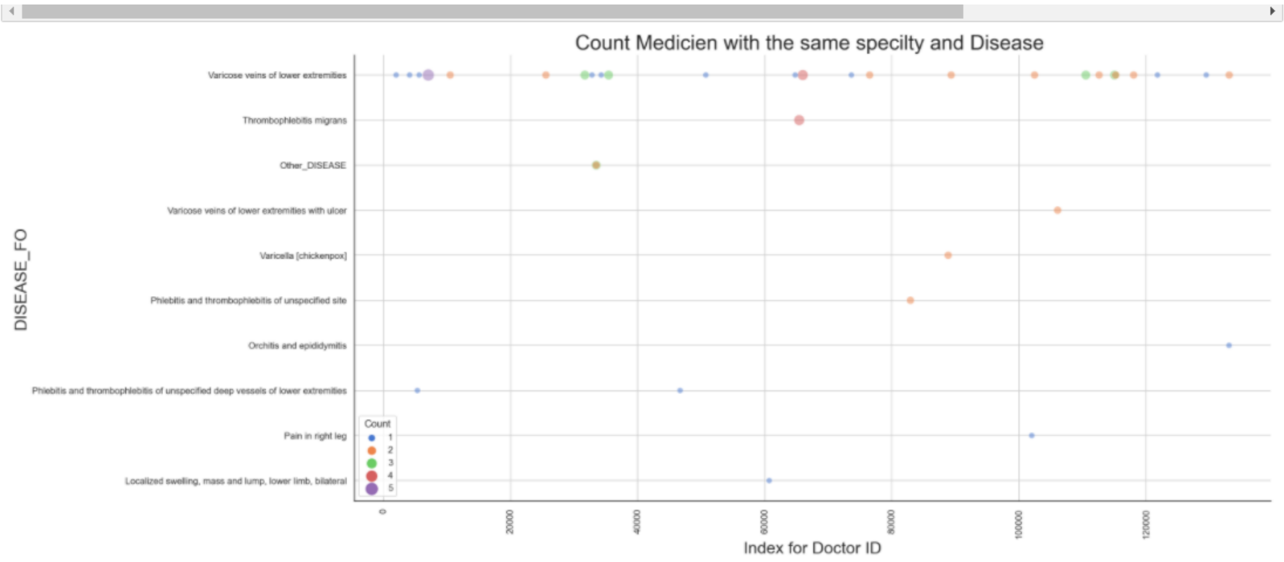
```
In [724]: Specialty= data_meds[data_meds["SPECIALTY_ID"] == 7]
          Specialty= Specialty.sort_values(by='Count', ascending=False)
          Specialty
```

Out[724]:

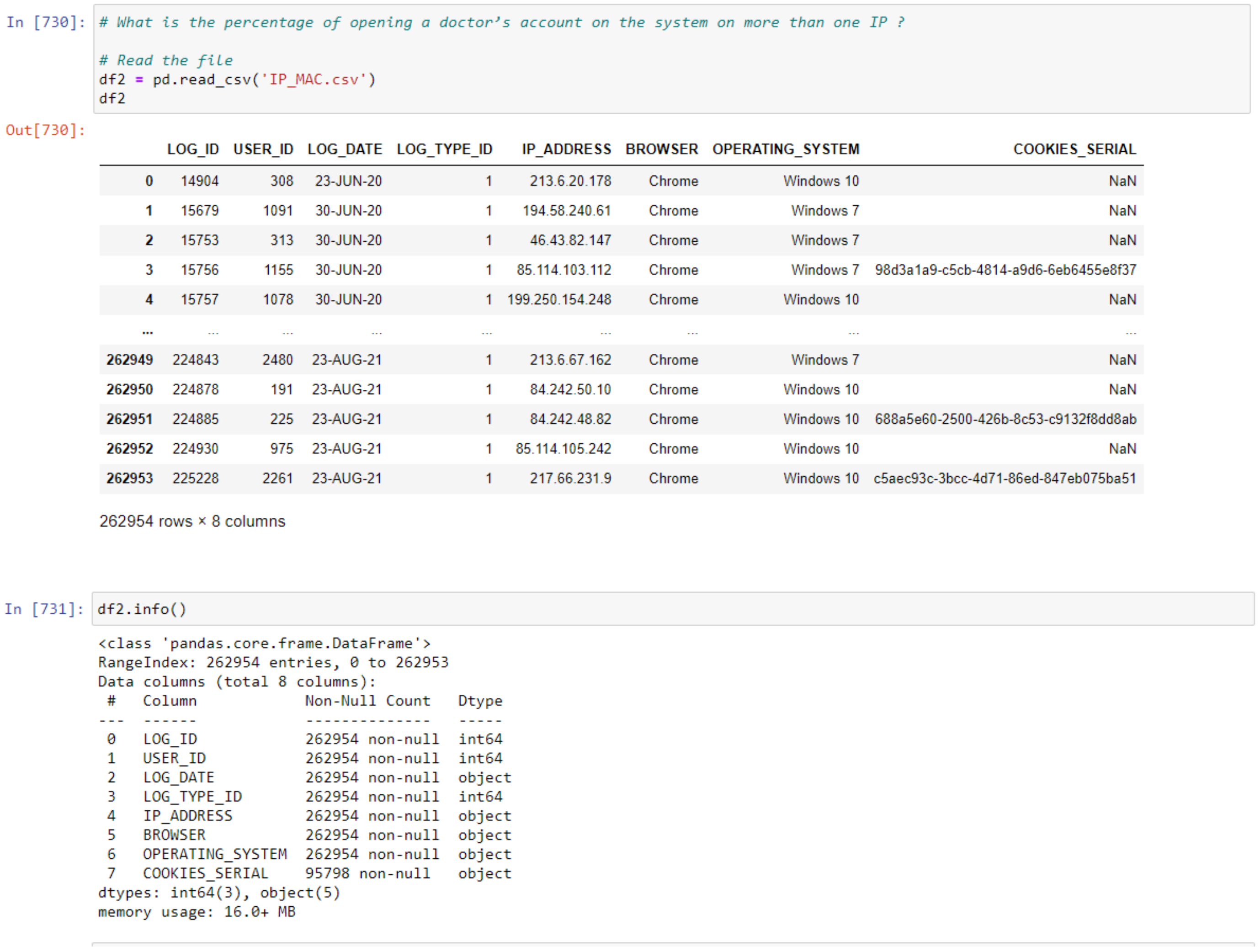
	CLAIM_ID	ID_NUM_PASSPORT	DISEASE_FO	SPECIALTY_ID	DOCTOR_USER_ID	TYPE	Count
114723	84805.0	432356798	Atopic dermatitis	7.0	1011.0	MEDS	5
114726	84807.0	408050508	Atopic dermatitis	7.0	1011.0	MEDS	4
101395	75023.0	438099970	Atopic dermatitis	7.0	1011.0	MEDS	3
57729	44048.0	950603613	Atopic dermatitisâ° Scabies	7.0	1011.0	MEDS	3
67501	50928.0	801553447	Dermatophytosis, unspecified	7.0	1011.0	MEDS	3
67507	50931.0	438099970	Gonococcal vulvovaginitis, unspecified	7.0	1011.0	MEDS	3
69320	52220.0	411004633	Dermatophytosis, unspecified	7.0	1011.0	MEDS	3
112528	83238.0	801553447	Dermatophytosis, unspecified	7.0	1011.0	MEDS	3
53502	41026.0	801862780	Eczema herpeticumâ° Tinea cruris	7.0	1011.0	MEDS	3
112840	83455.0	801921115	Impetigo, unspecified	7.0	1011.0	MEDS	2
112524	83236.0	438099970	Atopic dermatitis	7.0	1011.0	MEDS	2
4192	3097.0	858579881	Urticaria	7.0	1011.0	MEDS	2
101345	74986.0	435816509	Dermatophytosis, unspecified	7.0	1011.0	MEDS	2
58857	44822.0	438562373	Impetigo, unspecified	7.0	1011.0	MEDS	2
53505	41028.0	438093031	Atopic dermatitisâ° Eczema herpeticum	7.0	1011.0	MEDS	2
101349	74988.0	438562373	Urticaria	7.0	1011.0	MEDS	2
103852	76861.0	435816509	Dermatophytosis, unspecified	7.0	1011.0	MEDS	1
101356	74993.0	903090520	Atopic dermatitis	7.0	1011.0	MEDS	1



Specilty _id = 35



# 6 . What is the percentage of opening a doctor’s account on the system on more than one IP ?



In [730]:
```
# What is the percentage of opening a doctor’s account on the system on more than one IP ?

# Read the file
df2 = pd.read_csv('IP_MAC.csv')
df2
```

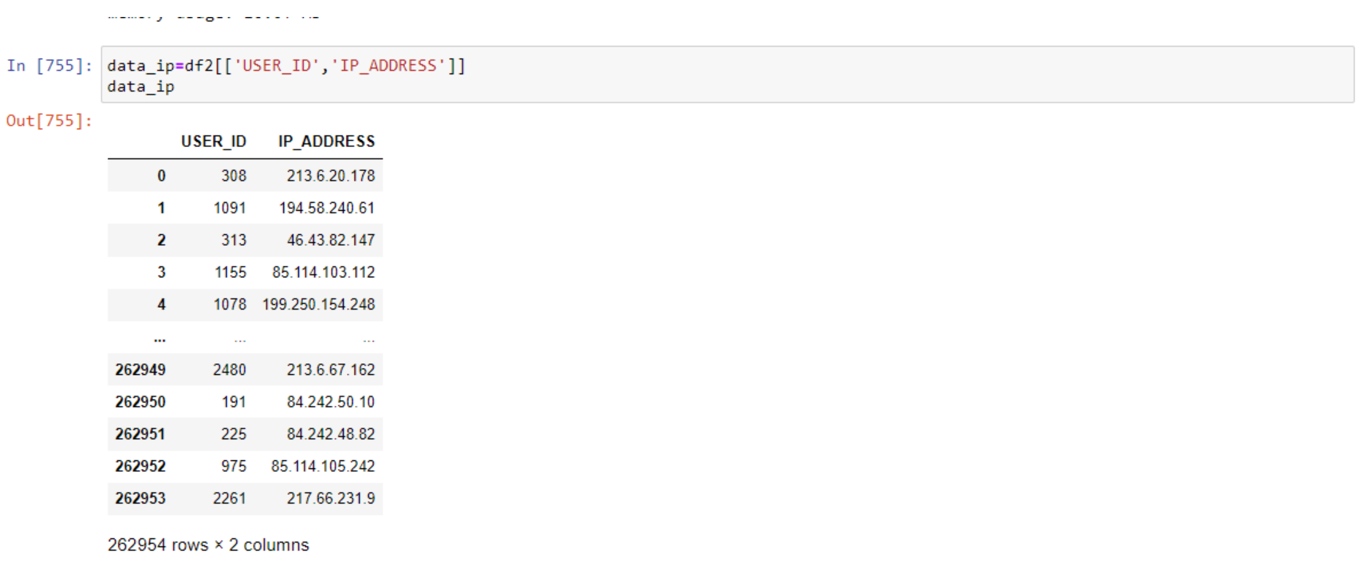
Out[730]:

	LOG_ID	USER_ID	LOG_DATE	LOG_TYPE_ID	IP_ADDRESS	BROWSER	OPERATING_SYSTEM	COOKIES_SERIAL
0	14904	308	23-JUN-20	1	213.6.20.178	Chrome	Windows 10	NaN
1	15679	1091	30-JUN-20	1	194.58.240.61	Chrome	Windows 7	NaN
2	15753	313	30-JUN-20	1	46.43.82.147	Chrome	Windows 7	NaN
3	15756	1155	30-JUN-20	1	85.114.103.112	Chrome	Windows 7	98d3a1a9-c5cb-4814-a9d6-6eb6455e8f37
4	15757	1078	30-JUN-20	1	199.250.154.248	Chrome	Windows 10	NaN
...	...	...	...	...	...	...	...	...
262949	224843	2480	23-AUG-21	1	213.6.67.162	Chrome	Windows 7	NaN
262950	224878	191	23-AUG-21	1	84.242.50.10	Chrome	Windows 10	NaN
262951	224885	225	23-AUG-21	1	84.242.48.82	Chrome	Windows 10	688a5e60-2500-426b-8c53-c9132f8dd8ab
262952	224930	975	23-AUG-21	1	85.114.105.242	Chrome	Windows 10	NaN
262953	225228	2261	23-AUG-21	1	217.66.231.9	Chrome	Windows 10	c5aec93c-3bcc-4d71-86ed-847eb075ba51

262954 rows × 8 columns

In [731]:
```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 262954 entries, 0 to 262953
Data columns (total 8 columns):
 #   Column            Non-Null Count   Dtype 
---  ------            --------------   ----- 
 0   LOG_ID            262954 non-null  int64 
 1   USER_ID           262954 non-null  int64 
 2   LOG_DATE          262954 non-null  object
 3   LOG_TYPE_ID       262954 non-null  int64 
 4   IP_ADDRESS        262954 non-null  object
 5   BROWSER           262954 non-null  object
 6   OPERATING_SYSTEM  262954 non-null  object
 7   COOKIES_SERIAL    95798 non-null   object
dtypes: int64(3), object(5)
memory usage: 16.0+ MB
```

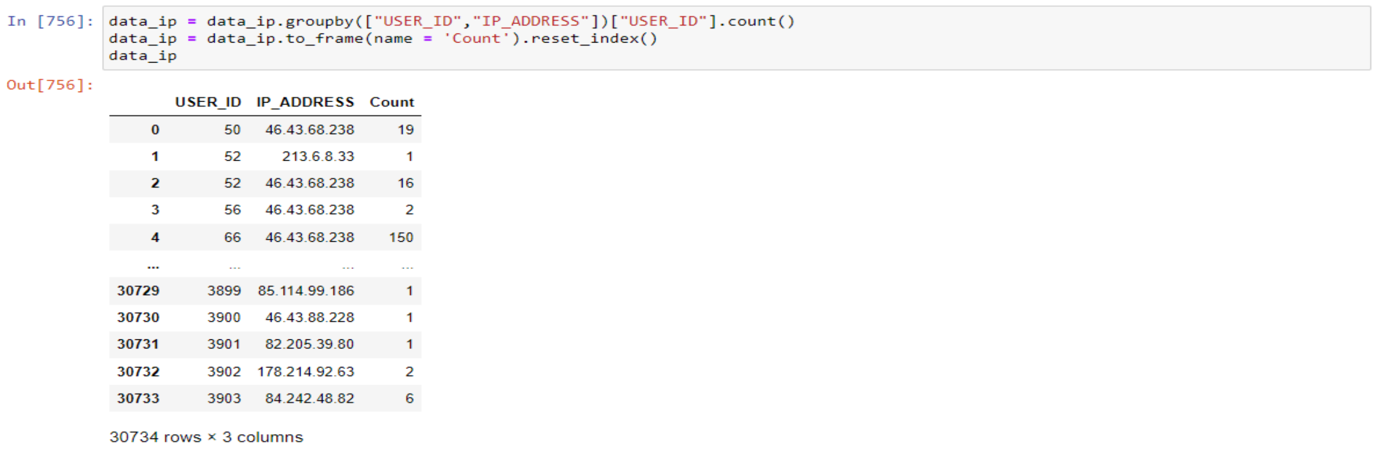


In [755]:
```
data_ip=df2[['USER_ID','IP_ADDRESS']]
data_ip
```

Out[755]:

	USER_ID	IP_ADDRESS
0	308	213.6.20.178
1	1091	194.58.240.61
2	313	46.43.82.147
3	1155	85.114.103.112
4	1078	199.250.154.248
...	...	...
262949	2480	213.6.67.162
262950	191	84.242.50.10
262951	225	84.242.48.82
262952	975	85.114.105.242
262953	2261	217.66.231.9

262954 rows × 2 columns

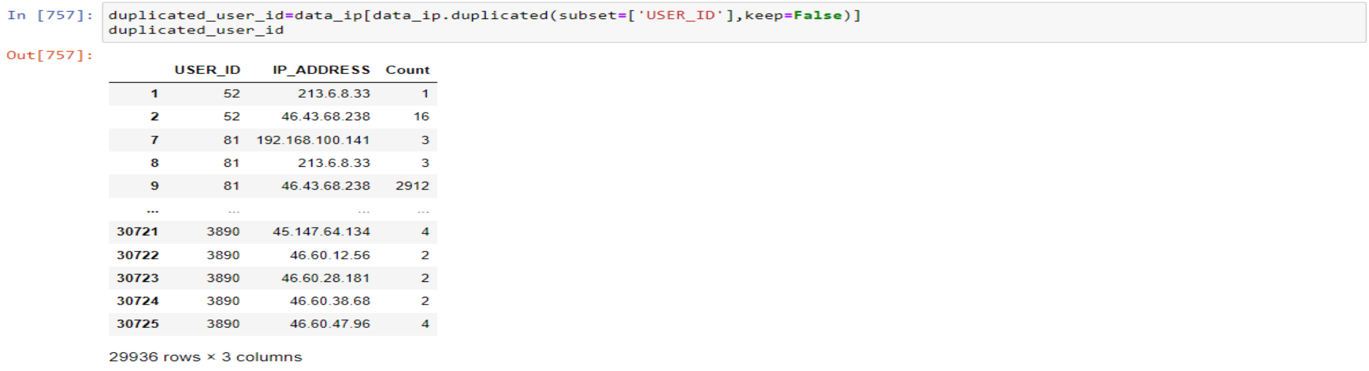


```
In [756]: data_ip = data_ip.groupby(["USER_ID","IP_ADDRESS"])["USER_ID"].count()
          data_ip = data_ip.to_frame(name = 'Count').reset_index()
          data_ip
```

Out[756]:

	USER_ID	IP_ADDRESS	Count
0	50	46.43.68.238	19
1	52	213.6.8.33	1
2	52	46.43.68.238	16
3	56	46.43.68.238	2
4	66	46.43.68.238	150
...	...	...	...
30729	3899	85.114.99.186	1
30730	3900	46.43.88.228	1
30731	3901	82.205.39.80	1
30732	3902	178.214.92.63	2
30733	3903	84.242.48.82	6

30734 rows × 3 columns

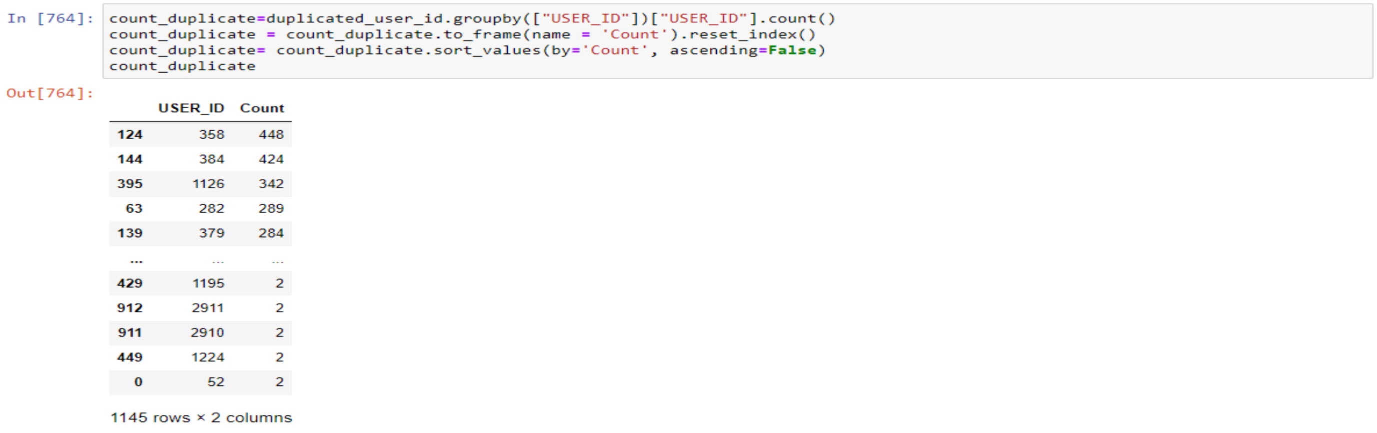


```
In [757]: duplicated_user_id=data_ip[data_ip.duplicated(subset=['USER_ID'],keep=False)]
          duplicated_user_id
```

Out[757]:

	USER_ID	IP_ADDRESS	Count
1	52	213.6.8.33	1
2	52	46.43.68.238	16
7	81	192.168.100.141	3
8	81	213.6.8.33	3
9	81	46.43.68.238	2912
...	...	...	...
30721	3890	45.147.64.134	4
30722	3890	46.60.12.56	2
30723	3890	46.60.28.181	2
30724	3890	46.60.38.68	2
30725	3890	46.60.47.96	4

29936 rows × 3 columns

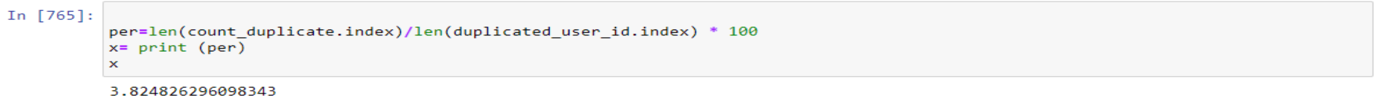


```
In [764]: count_duplicate=duplicated_user_id.groupby(["USER_ID"])["USER_ID"].count()
          count_duplicate = count_duplicate.to_frame(name = 'Count').reset_index()
          count_duplicate= count_duplicate.sort_values(by='Count', ascending=False)
          count_duplicate
```

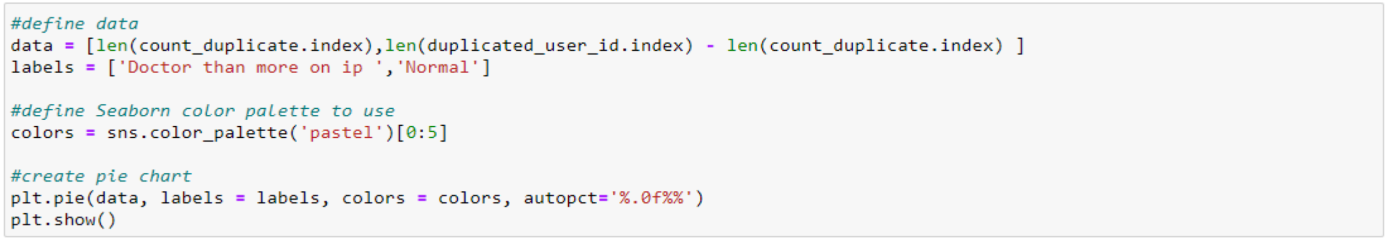
Out[764]:

	USER_ID	Count
124	358	448
144	384	424
395	1126	342
63	282	289
139	379	284
...	...	...
429	1195	2
912	2911	2
911	2910	2
449	1224	2
0	52	2

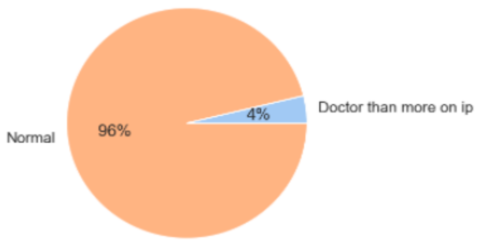
1145 rows × 2 columns



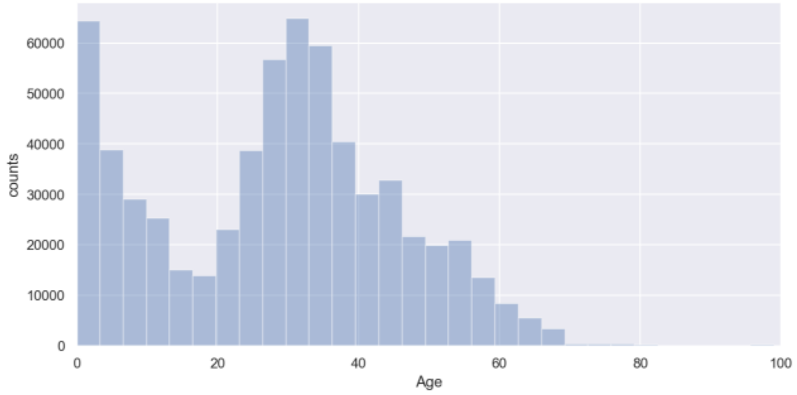
```
In [765]: 
          per=len(count_duplicate.index)/len(duplicated_user_id.index) * 100
          x= print (per)
          x

          3.824826296098343
```



```
#define data
data = [len(count_duplicate.index),len(duplicated_user_id.index) - len(count_duplicate.index) ]
labels = ['Doctor than more on ip ','Normal']

#define Seaborn color palette to use
colors = sns.color_palette('pastel')[0:5]

#create pie chart
plt.pie(data, labels = labels, colors = colors, autopct='%.0f%%')
plt.show()
```

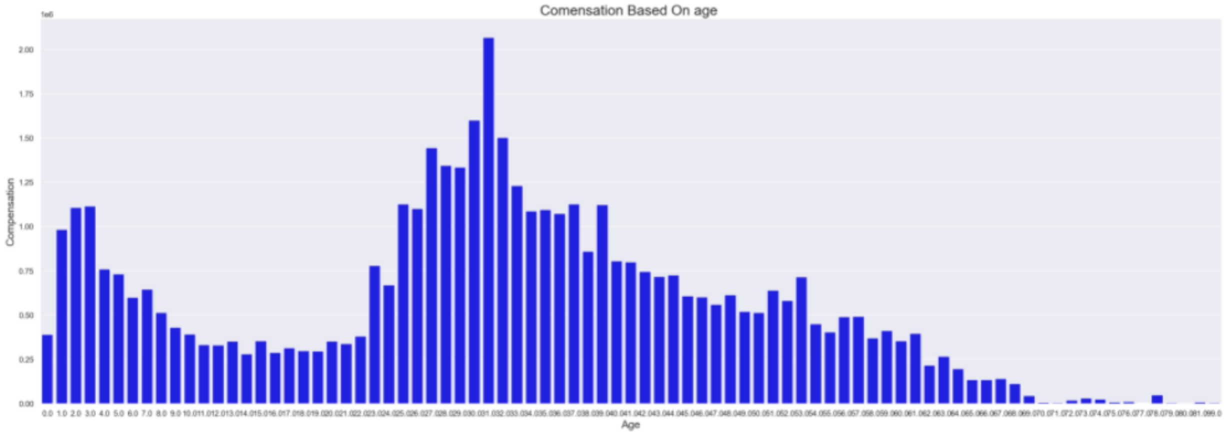


```
In [785]: # more analysis


          # What was the age distribution among passengers in the Titanic?

          import seaborn as sns
          sns.set(color_codes=True)

          f, ax = plt.subplots(1,1, figsize=(10, 5));
          ax = sns.distplot(distribution_age.Age, kde=False, bins=30)

          # bug
          #ax = sns.distplot(titanic.age, kde=False, bins=20).set(xlim=(0, 90));
          ax.set(xlim=(0, 100));
          ax.set_ylabel('counts');
```



PAY VALUE Based on Age

```
In [806]: # compensation Based On Age

          fig, ax = plt.subplots(figsize=(30, 10))
          sns.barplot(x='Age', y='sum', data=distribution_age, color='blue',ax=ax)
          sns.despine(fig)


          ax.set_title('Comensation Based On age ',fontsize=20);\
          ax.set_xlabel('Age', fontsize=15);
          ax.set_ylabel('Compensation', fontsize=15);
          plt.show()
```



In [813]:

```
Male_Femal=df[['GENDER_FO','PARTICIPATION_VAL_DISCOUNT']]
Male_Femal
```

Out[813]:

```
        GENDER_FO  PARTICIPATION_VAL_DISCOUNT
     0       Male                        55.0
     1       Male                        35.0
     2       Male                        58.0
     3     Female                        35.0
     4     Female                        70.0
   ...        ...                         ...
626607     Female                         0.0
626608     Female                         0.0
626609     Female                         0.0
626610       Male                         0.0
626611     Female                         0.0

626612 rows × 2 columns
```

In [814]:

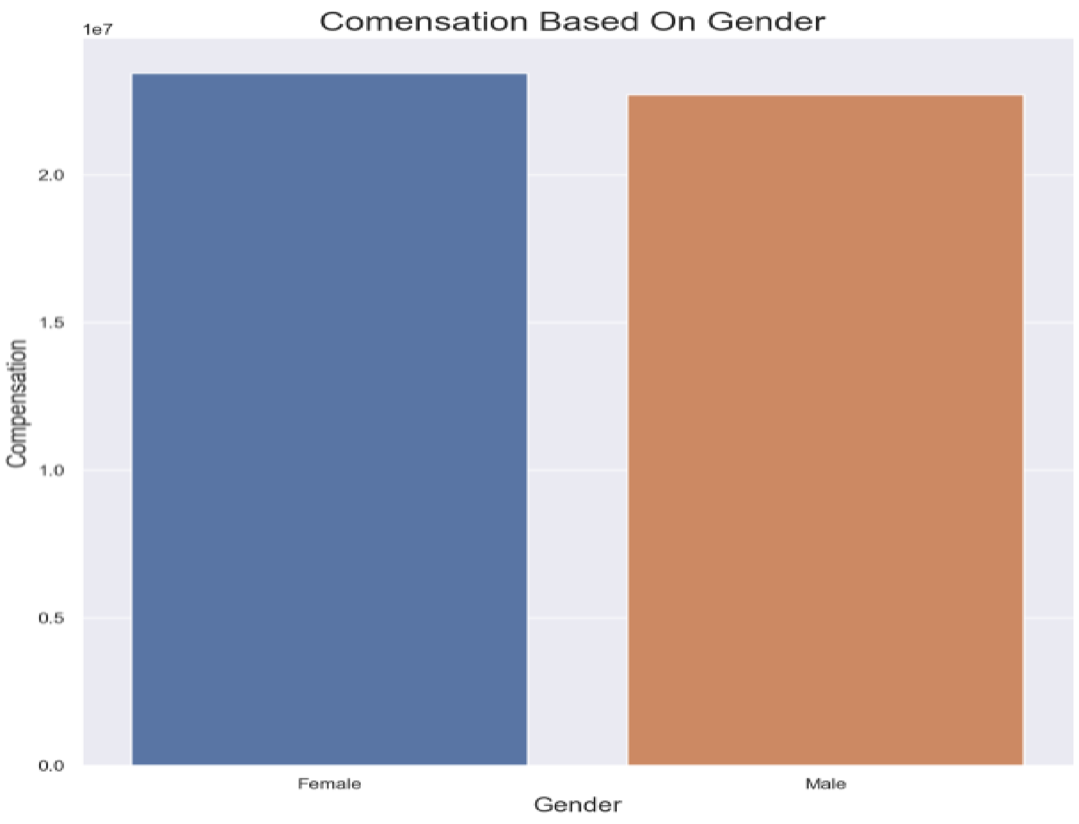
```
Male_Femal=Male_Femal.groupby(["GENDER_FO"])["PARTICIPATION_VAL_DISCOUNT"].sum()
Male_Femal = Male_Femal.to_frame(name = 'sum').reset_index()
Male_Femal= Male_Femal.sort_values(by='sum', ascending=False)
Male_Femal
```

Out[814]:

```
   GENDER_FO           sum
0     Female  2.342558e+07
1       Male  2.270185e+07
```

In [816]:

```
# compensation Based On Age

fig, ax = plt.subplots(figsize=(10, 10))
sns.barplot(x='GENDER_FO', y='sum', data=Male_Femal ,ax=ax)
sns.despine(fig)


ax.set_title('Comensation Based On Gender ',fontsize=20);\
ax.set_xlabel('Gender', fontsize=15);
ax.set_ylabel('Compensation', fontsize=15);
plt.show()
```

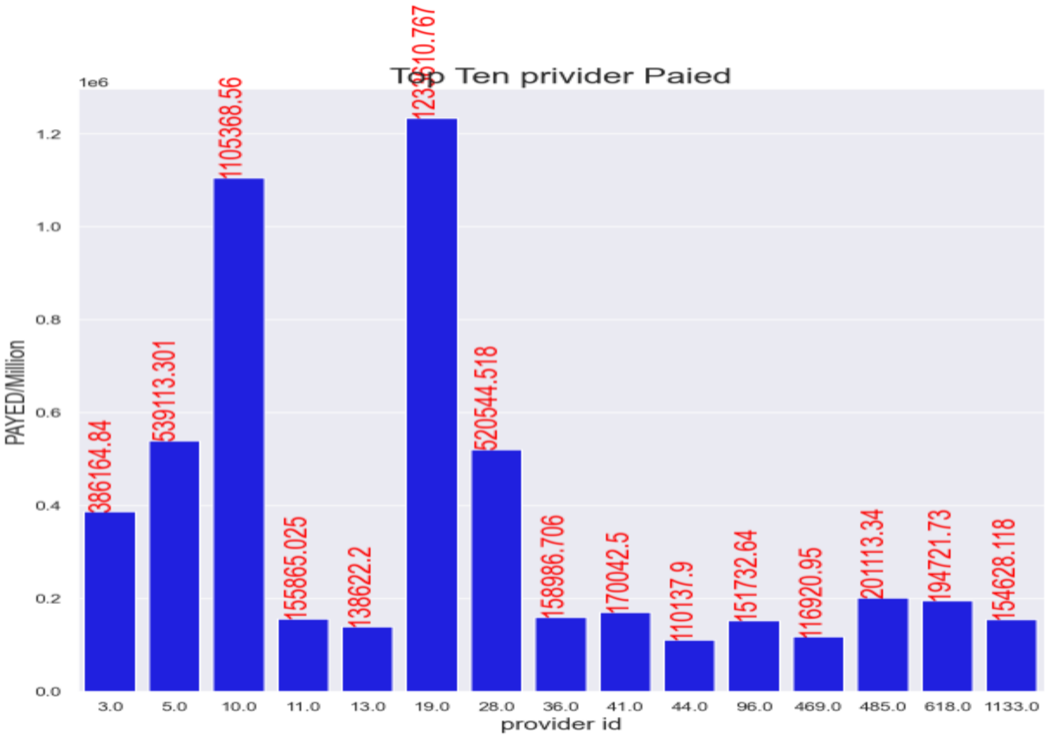


In [574]:
```
#Top Ten Provider Compensation
maximum_payed= data1.groupby(["PROVIDER_ID"])["PAY_VALUE"].sum()
maximum_payed = maximum_payed.to_frame(name = 'SUM').reset_index()
Top_ten_provider= maximum_payed.sort_values(by='SUM', ascending=False)
top_ten=Top_ten_provider.head(15)
top_ten
```

Out[574]:

	PROVIDER_ID	SUM
12	19.0	1233610.767
6	10.0	1105368.560
2	5.0	539113.301
19	28.0	520544.518
0	3.0	386164.840
321	485.0	201113.340
409	618.0	194721.730
30	41.0	170042.500
25	36.0	158986.706
7	11.0	155865.025
613	1133.0	154628.118
66	96.0	151732.640
9	13.0	138622.200
313	469.0	116920.950
32	44.0	110137.900

In [576]:
```


fig, ax = plt.subplots(figsize=(10, 10))
sns.barplot(x='PROVIDER_ID', y='SUM', data=top_ten, color='blue',ax=ax)
sns.despine(fig)


ax.set_title('Top Ten privider Paied',fontsize=20);\
ax.set_xlabel('provider id', fontsize=15);
ax.set_ylabel('PAYED/Million', fontsize=15);
for p in ax.patches:
    ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.5, p.get_height()), ha='right', va='bottom',
                color='red', size=18, rotation=90)

plt.show()
```



In [577]:

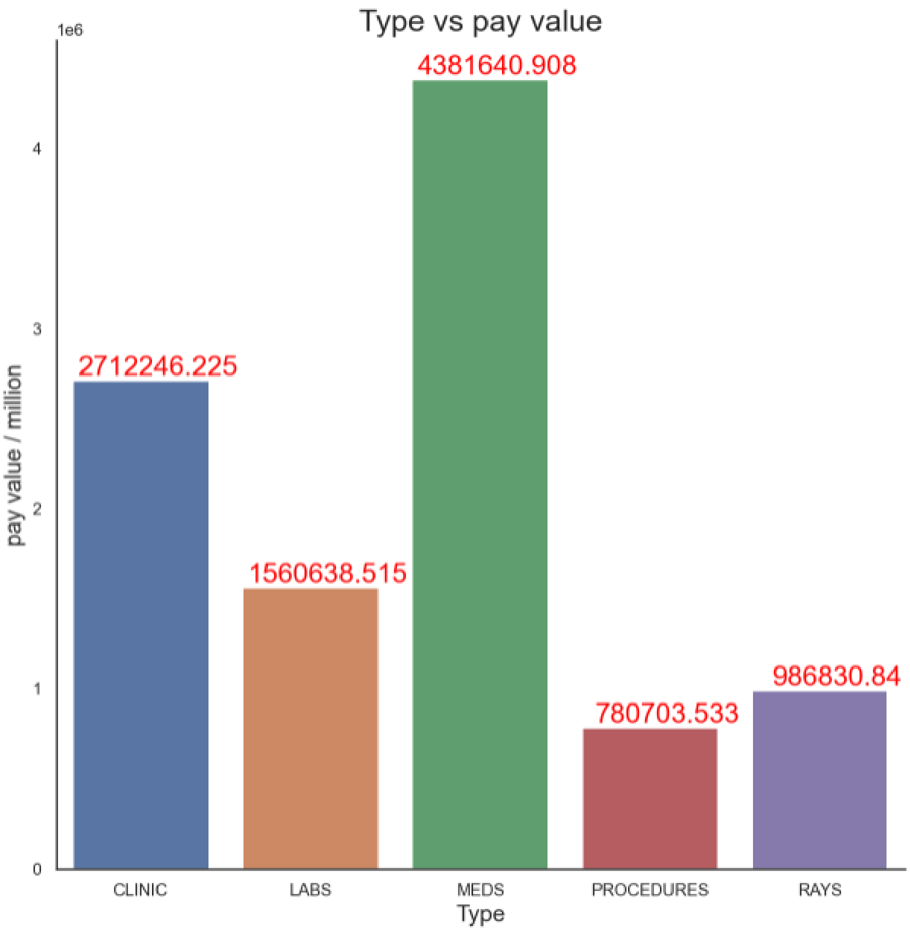
```
# Type vs pay value
TYPE= data1.groupby(["TYPE"])["PAY_VALUE"].sum()
TYPE= TYPE.to_frame(name = 'SUM').reset_index()
TYPE
```

Out[577]:

	TYPE	SUM
0	CLINIC	2712246.225
1	LABS	1560638.515
2	MEDS	4381640.908
3	PROCEDURES	780703.533
4	RAYS	986830.840

In [578]:

```
fig, ax = plt.subplots(figsize=(10, 10))
sns.barplot(x='TYPE', y='SUM', data=TYPE, ax=ax)
sns.despine(fig)

ax.set_title('Type vs pay value', fontsize=20);\
ax.set_xlabel('Type', fontsize=15);
ax.set_ylabel('pay value / million', fontsize=15);

for p in ax.patches:
    ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.5, p.get_height()), ha='center', va='bottom', color='red', size=18)

plt.show()
```


Type vs pay value
1e6
pay value / million
CLINIC
LABS
MEDS
PROCEDURES
RAYS
Type