In [1]:
```
!pip install matplotlib
```

```
Requirement already satisfied: matplotlib in c:\users\sayal\anaconda3\lib\site-packag
es (3.7.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\sayal\anaconda3\lib\site-
packages (from matplotlib) (1.0.5)
Requirement already satisfied: cycler>=0.10 in c:\users\sayal\anaconda3\lib\site-pack
ages (from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\sayal\anaconda3\lib\site
-packages (from matplotlib) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\sayal\anaconda3\lib\site
-packages (from matplotlib) (1.4.4)
Requirement already satisfied: numpy>=1.20 in c:\users\sayal\anaconda3\lib\site-packa
ges (from matplotlib) (1.24.3)
Requirement already satisfied: packaging>=20.0 in c:\users\sayal\anaconda3\lib\site-p
ackages (from matplotlib) (23.1)
Requirement already satisfied: pillow>=6.2.0 in c:\users\sayal\anaconda3\lib\site-pac
kages (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing<3.1,>=2.3.1 in c:\users\sayal\anaconda3\lib
\site-packages (from matplotlib) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\sayal\anaconda3\lib\s
ite-packages (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\sayal\anaconda3\lib\site-packages
(from python-dateutil>=2.7->matplotlib) (1.16.0)
```

In [4]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [5]:
```
# for question 5
from io import StringIO
```

In [6]:
```
# Loading data
df_mdata = pd.read_csv("C:\\Users\\Sayal\\OneDrive\\Desktop\\6600_NU\\Assignment 1\\Me
```

In [7]:
```
# inspecting dataframe
df_mdata
```

Out[7]:

	Brnd_Name	Gnrc_Name	Tot_Mftr	Mftr_Name	Tot_Spndng_2017	Tot_Dsg_Unts_2017	T
0	8hr Arthritis Pain	Acetaminophen	1	Overall	NaN	NaN	
1	8hr Arthritis Pain	Acetaminophen	1	Topco	NaN	NaN	
2	A & D	Vitamins A And D	1	Overall	NaN	NaN	
3	A & D	Vitamins A And D	1	Schering-Plough	NaN	NaN	
4	A And D	Vits A And D/White Pet/Lanolin	1	Overall	406.33	3502.50	
...	...	...	...	...	...	...	
16141	Zyvox	Linezolid	1	Pharmaci/Pfizer	881336.15	52500.00	
16142	Zyvox	Linezolid In Dextrose 5%	3	Overall	516194.24	2811493.99	
16143	Zyvox	Linezolid In Dextrose 5%	1	Pharmaci/Pfizer	400349.22	2252392.99	
16144	Zyvox	Linezolid In Dextrose 5%	1	Phar-Prep/Pfize	63972.27	373810.00	
16145	Zyvox	Linezolid In Dextrose 5%	1	Phar-Nov/Pfizer	51872.75	185291.00	

16146 rows × 36 columns

In [8]:
```
df_mdata.head
```

Out[8]:
```
<bound method NDFrame.head of                Brnd_Name                        Gnrc_Na
me  Tot_Mftr  \
0      8hr Arthritis Pain                  Acetaminophen         1
1      8hr Arthritis Pain                  Acetaminophen         1
2                   A & D               Vitamins A And D         1
3                   A & D               Vitamins A And D         1
4                 A And D  Vits A And D/White Pet/Lanolin         1
...                   ...                             ...       ...
16141               Zyvox                       Linezolid         1
16142               Zyvox       Linezolid In Dextrose 5%         3
16143               Zyvox       Linezolid In Dextrose 5%         1
16144               Zyvox       Linezolid In Dextrose 5%         1
16145               Zyvox       Linezolid In Dextrose 5%         1

             Mftr_Name  Tot_Spndng_2017  Tot_Dsg_Unts_2017  Tot_Clms_2017  \
0              Overall             NaN                NaN            NaN
1                Topco             NaN                NaN            NaN
2              Overall             NaN                NaN            NaN
3      Schering-Plough             NaN                NaN            NaN
4              Overall          406.33            3502.50           47.0
...                ...             ...                ...            ...
16141  Pharmaci/Pfizer       881336.15           52500.00          314.0
16142          Overall       516194.24         2811493.99         1308.0
16143  Pharmaci/Pfizer       400349.22         2252392.99          987.0
16144  Phar-Prep/Pfize        63972.27          373810.00          169.0
16145  Phar-Nov/Pfizer        51872.75          185291.00          152.0

       Avg_Spnd_Per_Dsg_Unt_Wghtd_2017  Avg_Spnd_Per_Clm_2017  \
0                                  NaN                    NaN
1                                  NaN                    NaN
2                                  NaN                    NaN
3                                  NaN                    NaN
4                             0.116011               8.645319
...                                ...                    ...
16141                        90.123927            2806.803025
16142                         0.195819             394.643914
16143                         0.187088             405.622310
16144                         0.171136             378.534142
16145                         0.279953             341.268092

       Outlier_Flag_2017  ...  Avg_Spnd_Per_Clm_2020  Outlier_Flag_2020  \
0                    NaN  ...               9.281739                1.0
1                    NaN  ...               9.281739                1.0
2                    NaN  ...               7.866098                1.0
3                    NaN  ...               7.866098                1.0
4                    1.0  ...               7.288947                1.0
...                  ...  ...                    ...                ...
16141                1.0  ...            1544.316444                1.0
16142                0.0  ...             175.597041                0.0
16143                0.0  ...             315.766394                0.0
16144                0.0  ...              78.177544                0.0
16145                1.0  ...              79.955617                0.0

       Tot_Spndng_2021  Tot_Dsg_Unts_2021  Tot_Clms_2021  \
0               331.69           4642.000             54
1               331.69           4642.000             54
2               253.76           9136.000             26
3               253.76           9136.000             26
4               106.45           1032.708             14
...                ...                ...            ...
```

```
16141         22141.27           2133.000             19
16142        108270.21         555519.000            613
16143         67237.88         364907.000            245
16144         17779.03         107844.000            238
16145         23253.30          82768.000            130

       Avg_Spnd_Per_Dsg_Unt_Wghtd_2021  Avg_Spnd_Per_Clm_2021  \
0                             0.071454               6.142407
1                             0.071454               6.142407
2                             0.027776               9.760000
3                             0.027776               9.760000
4                             0.103079               7.603571
...                                ...                    ...
16141                        45.248481            1165.330000
16142                         0.202670             176.623507
16143                         0.197868             274.440327
16144                         0.164859              74.701807
16145                         0.280946             178.871538

       Outlier_Flag_2021  Chg_Avg_Spnd_Per_Dsg_Unt_20_21  \
0                      1                        0.469539
1                      1                       -0.253762
2                      1                        0.364446
3                      1                        0.356607
4                      1                        0.296316
...                  ...                             ...
16141                  1                        1.177785
16142                  0                        0.451434
16143                  0                        0.256644
16144                  0                        0.548867
16145                  0                        0.792195

       CAGR_Avg_Spnd_Per_Dsg_Unt_17_21
0                             0.047199
1                             0.047199
2                             0.594980
3                             0.594980
4                            -0.029117
...                                ...
16141                        -0.158235
16142                         0.008635
16143                         0.014104
16144                        -0.009299
16145                         0.000885

[16146 rows x 36 columns]>
```

In [9]:
```
# selecting top 10 rows
df_mdata.head(10)
```

Out[9]:

	Brnd_Name	Gnrc_Name	Tot_Mftr	Mftr_Name	Tot_Spndng_2017	Tot_Dsg_Unts_2017	Tot_Clm
0	8hr Arthritis Pain	Acetaminophen	1	Overall	NaN	NaN	
1	8hr Arthritis Pain	Acetaminophen	1	Topco	NaN	NaN	
2	A & D	Vitamins A And D	1	Overall	NaN	NaN	
3	A & D	Vitamins A And D	1	Schering-Plough	NaN	NaN	
4	A And D	Vits A And D/White Pet/Lanolin	1	Overall	406.33	3502.5	
5	A And D	Vits A And D/White Pet/Lanolin	1	Bayer Healthcar	406.33	3502.5	
6	A And D Diaper Rash	Dimethic/Zinc Ox/Vits A,D/Aloe	1	Overall	NaN	NaN	
7	A And D Diaper Rash	Dimethic/Zinc Ox/Vits A,D/Aloe	1	Bayer Healthcar	NaN	NaN	
8	A-Hydrocort	Hydrocortisone Sod Succinate	1	Overall	22317.83	108.0	
9	A-Hydrocort	Hydrocortisone Sod Succinate	1	Hospira/Pfizer	22317.83	108.0	

10 rows × 36 columns

In [75]:
```
# selecting bottom rows
df_mdata.tail(10)
```

Out[75]:

	Brnd_Name	Gnrc_Name	Tot_Mftr	Mftr_Name	Tot_Spndng_2017	Tot_Dsg_Unts_2017	Tot_(
16136	Zyprexa*	Olanzapine	1	Eli Lilly & Co.*	8240818.50	432253.20	
16137	Zytiga	Abiraterone Acetate	2	Overall	49115337.85	601210.00	
16138	Zytiga	Abiraterone Acetate	1	Janssen Pharm.	4861214.50	30982.00	
16139	Zytiga	Abiraterone Acetate	1	Janssen Biotech	44254123.35	570228.00	
16140	Zyvox	Linezolid	1	Overall	881336.15	52500.00	
16141	Zyvox	Linezolid	1	Pharmaci/Pfizer	881336.15	52500.00	
16142	Zyvox	Linezolid In Dextrose 5%	3	Overall	516194.24	2811493.99	
16143	Zyvox	Linezolid In Dextrose 5%	1	Pharmaci/Pfizer	400349.22	2252392.99	
16144	Zyvox	Linezolid In Dextrose 5%	1	Phar-Prep/Pfize	63972.27	373810.00	
16145	Zyvox	Linezolid In Dextrose 5%	1	Phar-Nov/Pfizer	51872.75	185291.00	

10 rows × 36 columns

In [10]:
```
# Inspecting dimensions: retrieveing the number of rows and columns
df_mdata.shape
```

Out[10]:
```
(16146, 36)
```

In [11]:
```
# Checking all column names
print(df_mdata.columns.tolist())
```

```
['Brnd_Name', 'Gnrc_Name', 'Tot_Mftr', 'Mftr_Name', 'Tot_Spndng_2017', 'Tot_Dsg_Unts_
2017', 'Tot_Clms_2017', 'Avg_Spnd_Per_Dsg_Unt_Wghtd_2017', 'Avg_Spnd_Per_Clm_2017',
'Outlier_Flag_2017', 'Tot_Spndng_2018', 'Tot_Dsg_Unts_2018', 'Tot_Clms_2018', 'Avg_Sp
nd_Per_Dsg_Unt_Wghtd_2018', 'Avg_Spnd_Per_Clm_2018', 'Outlier_Flag_2018', 'Tot_Spndng
_2019', 'Tot_Dsg_Unts_2019', 'Tot_Clms_2019', 'Avg_Spnd_Per_Dsg_Unt_Wghtd_2019', 'Avg
_Spnd_Per_Clm_2019', 'Outlier_Flag_2019', 'Tot_Spndng_2020', 'Tot_Dsg_Unts_2020', 'To
t_Clms_2020', 'Avg_Spnd_Per_Dsg_Unt_Wghtd_2020', 'Avg_Spnd_Per_Clm_2020', 'Outlier_Fl
ag_2020', 'Tot_Spndng_2021', 'Tot_Dsg_Unts_2021', 'Tot_Clms_2021', 'Avg_Spnd_Per_Dsg_
Unt_Wghtd_2021', 'Avg_Spnd_Per_Clm_2021', 'Outlier_Flag_2021', 'Chg_Avg_Spnd_Per_Dsg_
Unt_20_21', 'CAGR_Avg_Spnd_Per_Dsg_Unt_17_21']
```

In [12]:
```
list(df_mdata)
```

Out[12]:
```
['Brnd_Name',
 'Gnrc_Name',
 'Tot_Mftr',
 'Mftr_Name',
 'Tot_Spndng_2017',
 'Tot_Dsg_Unts_2017',
 'Tot_Clms_2017',
 'Avg_Spnd_Per_Dsg_Unt_Wghtd_2017',
 'Avg_Spnd_Per_Clm_2017',
 'Outlier_Flag_2017',
 'Tot_Spndng_2018',
 'Tot_Dsg_Unts_2018',
 'Tot_Clms_2018',
 'Avg_Spnd_Per_Dsg_Unt_Wghtd_2018',
 'Avg_Spnd_Per_Clm_2018',
 'Outlier_Flag_2018',
 'Tot_Spndng_2019',
 'Tot_Dsg_Unts_2019',
 'Tot_Clms_2019',
 'Avg_Spnd_Per_Dsg_Unt_Wghtd_2019',
 'Avg_Spnd_Per_Clm_2019',
 'Outlier_Flag_2019',
 'Tot_Spndng_2020',
 'Tot_Dsg_Unts_2020',
 'Tot_Clms_2020',
 'Avg_Spnd_Per_Dsg_Unt_Wghtd_2020',
 'Avg_Spnd_Per_Clm_2020',
 'Outlier_Flag_2020',
 'Tot_Spndng_2021',
 'Tot_Dsg_Unts_2021',
 'Tot_Clms_2021',
 'Avg_Spnd_Per_Dsg_Unt_Wghtd_2021',
 'Avg_Spnd_Per_Clm_2021',
 'Outlier_Flag_2021',
 'Chg_Avg_Spnd_Per_Dsg_Unt_20_21',
 'CAGR_Avg_Spnd_Per_Dsg_Unt_17_21']
```

In [13]:
```
df_mdata.columns
```

Out[13]:
```
Index(['Brnd_Name', 'Gnrc_Name', 'Tot_Mftr', 'Mftr_Name', 'Tot_Spndng_2017',
       'Tot_Dsg_Unts_2017', 'Tot_Clms_2017', 'Avg_Spnd_Per_Dsg_Unt_Wghtd_2017',
       'Avg_Spnd_Per_Clm_2017', 'Outlier_Flag_2017', 'Tot_Spndng_2018',
       'Tot_Dsg_Unts_2018', 'Tot_Clms_2018', 'Avg_Spnd_Per_Dsg_Unt_Wghtd_2018',
       'Avg_Spnd_Per_Clm_2018', 'Outlier_Flag_2018', 'Tot_Spndng_2019',
       'Tot_Dsg_Unts_2019', 'Tot_Clms_2019', 'Avg_Spnd_Per_Dsg_Unt_Wghtd_2019',
       'Avg_Spnd_Per_Clm_2019', 'Outlier_Flag_2019', 'Tot_Spndng_2020',
       'Tot_Dsg_Unts_2020', 'Tot_Clms_2020', 'Avg_Spnd_Per_Dsg_Unt_Wghtd_2020',
       'Avg_Spnd_Per_Clm_2020', 'Outlier_Flag_2020', 'Tot_Spndng_2021',
       'Tot_Dsg_Unts_2021', 'Tot_Clms_2021', 'Avg_Spnd_Per_Dsg_Unt_Wghtd_2021',
       'Avg_Spnd_Per_Clm_2021', 'Outlier_Flag_2021',
       'Chg_Avg_Spnd_Per_Dsg_Unt_20_21', 'CAGR_Avg_Spnd_Per_Dsg_Unt_17_21'],
      dtype='object')
```

In [86]:
```
# Checking datatypes for all the columns
column_datatypes = df_mdata.dtypes
print(column_datatypes)
```

```
Brnd_Name                          object
Gnrc_Name                          object
Tot_Mftr                            int64
Mftr_Name                          object
Tot_Spndng_2017                   float64
Tot_Dsg_Unts_2017                 float64
Tot_Clms_2017                     float64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2017   float64
Avg_Spnd_Per_Clm_2017             float64
Outlier_Flag_2017                 float64
Tot_Spndng_2018                   float64
Tot_Dsg_Unts_2018                 float64
Tot_Clms_2018                     float64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2018   float64
Avg_Spnd_Per_Clm_2018             float64
Outlier_Flag_2018                 float64
Tot_Spndng_2019                   float64
Tot_Dsg_Unts_2019                 float64
Tot_Clms_2019                     float64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2019   float64
Avg_Spnd_Per_Clm_2019             float64
Outlier_Flag_2019                 float64
Tot_Spndng_2020                   float64
Tot_Dsg_Unts_2020                 float64
Tot_Clms_2020                     float64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2020   float64
Avg_Spnd_Per_Clm_2020             float64
Outlier_Flag_2020                 float64
Tot_Spndng_2021                   float64
Tot_Dsg_Unts_2021                 float64
Tot_Clms_2021                       int64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2021   float64
Avg_Spnd_Per_Clm_2021             float64
Outlier_Flag_2021                   int64
Chg_Avg_Spnd_Per_Dsg_Unt_20_21    float64
CAGR_Avg_Spnd_Per_Dsg_Unt_17_21   float64
dtype: object
```

In [14]:
```
# below is the summary of the dataset
df_mdata.describe()
```

Out[14]:

	Tot_Mftr	Tot_Spndng_2017	Tot_Dsg_Unts_2017	Tot_Clms_2017	Avg_Spnd_Per_Dsg_Unt_Wgl
count	16146.000000	1.113600e+04	1.113600e+04	1.113600e+04	1113
mean	1.461910	1.190001e+07	7.850797e+06	1.232592e+05	13
std	2.289365	5.800085e+07	5.595186e+07	5.952454e+05	106
min	1.000000	0.000000e+00	8.400000e-02	1.100000e+01	
25%	1.000000	6.250336e+04	1.370190e+04	6.060000e+02	
50%	1.000000	5.597136e+05	1.537005e+05	4.382500e+03	
75%	1.000000	3.503468e+06	1.586952e+06	3.574625e+04	1
max	44.000000	1.298196e+09	2.165630e+09	1.508415e+07	3380

8 rows × 33 columns

In [18]:
```
# checking for missing
df_mdata.isnull()
```

Out[18]:

	Brnd_Name	Gnrc_Name	Tot_Mftr	Mftr_Name	Tot_Spndng_2017	Tot_Dsg_Unts_2017	Tot_Clm
0	False	False	False	False	True	True	
1	False	False	False	False	True	True	
2	False	False	False	False	True	True	
3	False	False	False	False	True	True	
4	False	False	False	False	False	False	
...	...	...	...	...	...	...	
16141	False	False	False	False	False	False	
16142	False	False	False	False	False	False	
16143	False	False	False	False	False	False	
16144	False	False	False	False	False	False	
16145	False	False	False	False	False	False	

16146 rows × 36 columns

In [20]:
```
# Counting the number of missing values for all the columns
df_mdata.isnull().sum()
```

Out[20]:

```
Brnd_Name                          0
Gnrc_Name                          0
Tot_Mftr                           0
Mftr_Name                          0
Tot_Spndng_2017                 5010
Tot_Dsg_Unts_2017               5010
Tot_Clms_2017                   5010
Avg_Spnd_Per_Dsg_Unt_Wghtd_2017 5010
Avg_Spnd_Per_Clm_2017           5010
Outlier_Flag_2017               5010
Tot_Spndng_2018                 3851
Tot_Dsg_Unts_2018               3851
Tot_Clms_2018                   3851
Avg_Spnd_Per_Dsg_Unt_Wghtd_2018 3851
Avg_Spnd_Per_Clm_2018           3851
Outlier_Flag_2018               3851
Tot_Spndng_2019                 2615
Tot_Dsg_Unts_2019               2615
Tot_Clms_2019                   2615
Avg_Spnd_Per_Dsg_Unt_Wghtd_2019 2615
Avg_Spnd_Per_Clm_2019           2615
Outlier_Flag_2019               2615
Tot_Spndng_2020                 1447
Tot_Dsg_Unts_2020               1447
Tot_Clms_2020                   1447
Avg_Spnd_Per_Dsg_Unt_Wghtd_2020 1447
Avg_Spnd_Per_Clm_2020           1447
Outlier_Flag_2020               1447
Tot_Spndng_2021                    0
Tot_Dsg_Unts_2021                  0
Tot_Clms_2021                      0
Avg_Spnd_Per_Dsg_Unt_Wghtd_2021    0
Avg_Spnd_Per_Clm_2021              0
Outlier_Flag_2021                  0
Chg_Avg_Spnd_Per_Dsg_Unt_20_21  1451
CAGR_Avg_Spnd_Per_Dsg_Unt_17_21   25
dtype: int64
```

In [307…

```
# Getting the total number of elements
tot_elements = np.product(df_mdata.shape)
# Counting the number of missing values
no_missing_values = df_mdata.isnull().sum().sum()
# Calculating the percentage of missing values
missing_values_percentage = (no_missing_values / tot_elements) * 100

df_mdata
```

Out[307]:

	Bmd_Name	Gnrc_Name	Tot_Mftr	Mftr_Name	Tot_Spndng_2017	Tot_Dsg_Unts_2017	T
0	8hr Arthritis Pain	Acetaminophen	1	Overall	1.190001e+07	7.850797e+06	1
1	8hr Arthritis Pain	Acetaminophen	1	Topco	1.190001e+07	7.850797e+06	1
2	A & D	Vitamins A And D	1	Overall	1.190001e+07	7.850797e+06	1
3	A & D	Vitamins A And D	1	Schering-Plough	1.190001e+07	7.850797e+06	1
4	A And D	Vits A And D/White Pet/Lanolin	1	Overall	4.063300e+02	3.502500e+03	
...	...	...	...	...	...	...	...
16141	Zyvox	Linezolid	1	Pharmaci/Pfizer	8.813362e+05	5.250000e+04	
16142	Zyvox	Linezolid In Dextrose 5%	3	Overall	5.161942e+05	2.811494e+06	
16143	Zyvox	Linezolid In Dextrose 5%	1	Pharmaci/Pfizer	4.003492e+05	2.252393e+06	
16144	Zyvox	Linezolid In Dextrose 5%	1	Phar-Prep/Pfize	6.397227e+04	3.738100e+05	
16145	Zyvox	Linezolid In Dextrose 5%	1	Phar-Nov/Pfizer	5.187275e+04	1.852910e+05	

16146 rows × 36 columns

In [464…

```
# Filling missing values with 0


df_mdata['Tot_Spndng_2017'] = df_mdata['Tot_Spndng_2017'].fillna(0)
df_mdata['Tot_Dsg_Unts_2017'] = df_mdata['Tot_Dsg_Unts_2017'].fillna(0)
df_mdata['Tot_Clms_2017'] = df_mdata['Tot_Clms_2017'].fillna(0)
df_mdata['Avg_Spnd_Per_Dsg_Unt_Wghtd_2017'] = df_mdata['Avg_Spnd_Per_Dsg_Unt_Wghtd_201
df_mdata['Avg_Spnd_Per_Clm_2017'] = df_mdata['Avg_Spnd_Per_Clm_2017'].fillna(0)

# Filling missing values for the year of 2018 with mean values
df_mdata['Tot_Spndng_2018'] = df_mdata['Tot_Spndng_2018'].fillna(0)
df_mdata['Tot_Dsg_Unts_2018'] = df_mdata['Tot_Dsg_Unts_2018'].fillna(0)
df_mdata['Tot_Clms_2018'] = df_mdata['Tot_Clms_2018'].fillna(0)
df_mdata['Avg_Spnd_Per_Dsg_Unt_Wghtd_2018'] = df_mdata['Avg_Spnd_Per_Dsg_Unt_Wghtd_201
df_mdata['Avg_Spnd_Per_Clm_2018'] = df_mdata['Avg_Spnd_Per_Clm_2018'].fillna(0)

# Filling missing values for the year of 2019 with mean values
df_mdata['Tot_Spndng_2019'] = df_mdata['Tot_Spndng_2019'].fillna(0)
df_mdata['Tot_Dsg_Unts_2019'] = df_mdata['Tot_Dsg_Unts_2019'].fillna(0)
df_mdata['Tot_Clms_2019'] = df_mdata['Tot_Clms_2019'].fillna(0)
df_mdata['Avg_Spnd_Per_Dsg_Unt_Wghtd_2019'] = df_mdata['Avg_Spnd_Per_Dsg_Unt_Wghtd_201
df_mdata['Avg_Spnd_Per_Clm_2019'] = df_mdata['Avg_Spnd_Per_Clm_2019'].fillna(0)

# Filling missing values for the year of 2020 with mean values
```

```
df_mdata['Tot_Spndng_2020'] = df_mdata['Tot_Spndng_2020'].fillna(0)
df_mdata['Tot_Dsg_Unts_2020'] = df_mdata['Tot_Dsg_Unts_2020'].fillna(0)
df_mdata['Tot_Clms_2020'] = df_mdata['Tot_Clms_2020'].fillna(0)
df_mdata['Avg_Spnd_Per_Dsg_Unt_Wghtd_2020'] = df_mdata['Avg_Spnd_Per_Dsg_Unt_Wghtd_202
df_mdata['Avg_Spnd_Per_Clm_2020'] = df_mdata['Avg_Spnd_Per_Clm_2020'].fillna(0)


df_mdata.isnull().sum()

#df_mdata
```

Out[464]:

```
Brnd_Name                          0
Gnrc_Name                          0
Tot_Mftr                           0
Mftr_Name                          0
Tot_Spndng_2017                    0
Tot_Dsg_Unts_2017                  0
Tot_Clms_2017                      0
Avg_Spnd_Per_Dsg_Unt_Wghtd_2017    0
Avg_Spnd_Per_Clm_2017              0
Outlier_Flag_2017                  0
Tot_Spndng_2018                    0
Tot_Dsg_Unts_2018                  0
Tot_Clms_2018                      0
Avg_Spnd_Per_Dsg_Unt_Wghtd_2018    0
Avg_Spnd_Per_Clm_2018              0
Outlier_Flag_2018                  0
Tot_Spndng_2019                    0
Tot_Dsg_Unts_2019                  0
Tot_Clms_2019                      0
Avg_Spnd_Per_Dsg_Unt_Wghtd_2019    0
Avg_Spnd_Per_Clm_2019              0
Outlier_Flag_2019                  0
Tot_Spndng_2020                    0
Tot_Dsg_Unts_2020                  0
Tot_Clms_2020                      0
Avg_Spnd_Per_Dsg_Unt_Wghtd_2020    0
Avg_Spnd_Per_Clm_2020              0
Outlier_Flag_2020                  0
Tot_Spndng_2021                    0
Tot_Dsg_Unts_2021                  0
Tot_Clms_2021                      0
Avg_Spnd_Per_Dsg_Unt_Wghtd_2021    0
Avg_Spnd_Per_Clm_2021              0
Outlier_Flag_2021                  0
Chg_Avg_Spnd_Per_Dsg_Unt_20_21     0
CAGR_Avg_Spnd_Per_Dsg_Unt_17_21    0
dtype: int64
```

In [462…

```
# Outlier flag: filling missing values with zero

df_mdata['Outlier_Flag_2017'] = df_mdata['Outlier_Flag_2017'].fillna(0)
df_mdata['Outlier_Flag_2018'] = df_mdata['Outlier_Flag_2017'].fillna(0)
df_mdata['Outlier_Flag_2019'] = df_mdata['Outlier_Flag_2017'].fillna(0)
df_mdata['Outlier_Flag_2020'] = df_mdata['Outlier_Flag_2017'].fillna(0)
```

In [463…

```
df_mdata['Chg_Avg_Spnd_Per_Dsg_Unt_20_21'] = df_mdata['Chg_Avg_Spnd_Per_Dsg_Unt_20_21'
df_mdata.isnull().sum()
```

```
df_mdata['CAGR_Avg_Spnd_Per_Dsg_Unt_17_21'] = df_mdata['Chg_Avg_Spnd_Per_Dsg_Unt_20_21
df_mdata.isnull().sum()
```

Out[463]:

```
Brnd_Name                          0
Gnrc_Name                          0
Tot_Mftr                           0
Mftr_Name                          0
Tot_Spndng_2017                 5010
Tot_Dsg_Unts_2017               5010
Tot_Clms_2017                   5010
Avg_Spnd_Per_Dsg_Unt_Wghtd_2017 5010
Avg_Spnd_Per_Clm_2017           5010
Outlier_Flag_2017                  0
Tot_Spndng_2018                 3851
Tot_Dsg_Unts_2018               3851
Tot_Clms_2018                   3851
Avg_Spnd_Per_Dsg_Unt_Wghtd_2018 3851
Avg_Spnd_Per_Clm_2018           3851
Outlier_Flag_2018                  0
Tot_Spndng_2019                 2615
Tot_Dsg_Unts_2019               2615
Tot_Clms_2019                   2615
Avg_Spnd_Per_Dsg_Unt_Wghtd_2019 2615
Avg_Spnd_Per_Clm_2019           2615
Outlier_Flag_2019                  0
Tot_Spndng_2020                 1447
Tot_Dsg_Unts_2020               1447
Tot_Clms_2020                   1447
Avg_Spnd_Per_Dsg_Unt_Wghtd_2020 1447
Avg_Spnd_Per_Clm_2020           1447
Outlier_Flag_2020                  0
Tot_Spndng_2021                    0
Tot_Dsg_Unts_2021                  0
Tot_Clms_2021                      0
Avg_Spnd_Per_Dsg_Unt_Wghtd_2021    0
Avg_Spnd_Per_Clm_2021              0
Outlier_Flag_2021                  0
Chg_Avg_Spnd_Per_Dsg_Unt_20_21     0
CAGR_Avg_Spnd_Per_Dsg_Unt_17_21    0
dtype: int64
```

In [24]:

```
display(df_mdata.dtypes)
```

```
Brnd_Name                          object
Gnrc_Name                          object
Tot_Mftr                            int64
Mftr_Name                          object
Tot_Spndng_2017                   float64
Tot_Dsg_Unts_2017                 float64
Tot_Clms_2017                     float64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2017   float64
Avg_Spnd_Per_Clm_2017             float64
Outlier_Flag_2017                 float64
Tot_Spndng_2018                   float64
Tot_Dsg_Unts_2018                 float64
Tot_Clms_2018                     float64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2018   float64
Avg_Spnd_Per_Clm_2018             float64
Outlier_Flag_2018                 float64
Tot_Spndng_2019                   float64
Tot_Dsg_Unts_2019                 float64
Tot_Clms_2019                     float64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2019   float64
Avg_Spnd_Per_Clm_2019             float64
Outlier_Flag_2019                 float64
Tot_Spndng_2020                   float64
Tot_Dsg_Unts_2020                 float64
Tot_Clms_2020                     float64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2020   float64
Avg_Spnd_Per_Clm_2020             float64
Outlier_Flag_2020                 float64
Tot_Spndng_2021                   float64
Tot_Dsg_Unts_2021                 float64
Tot_Clms_2021                       int64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2021   float64
Avg_Spnd_Per_Clm_2021             float64
Outlier_Flag_2021                   int64
Chg_Avg_Spnd_Per_Dsg_Unt_20_21    float64
CAGR_Avg_Spnd_Per_Dsg_Unt_17_21   float64
dtype: object
```

In [25]:

```
#data cleaning

print(df_mdata.dtypes)
df_mdata['Tot_Clms_2017'] = df_mdata['Tot_Clms_2017'].apply(np.int64)
df_mdata['Tot_Clms_2018'] = df_mdata['Tot_Clms_2018'].apply(np.int64)
df_mdata['Tot_Clms_2019'] = df_mdata['Tot_Clms_2019'].apply(np.int64)
df_mdata['Tot_Clms_2020'] = df_mdata['Tot_Clms_2020'].apply(np.int64)
```

```
Brnd_Name                          object
Gnrc_Name                          object
Tot_Mftr                            int64
Mftr_Name                          object
Tot_Spndng_2017                   float64
Tot_Dsg_Unts_2017                 float64
Tot_Clms_2017                     float64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2017   float64
Avg_Spnd_Per_Clm_2017             float64
Outlier_Flag_2017                 float64
Tot_Spndng_2018                   float64
Tot_Dsg_Unts_2018                 float64
Tot_Clms_2018                     float64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2018   float64
Avg_Spnd_Per_Clm_2018             float64
Outlier_Flag_2018                 float64
Tot_Spndng_2019                   float64
Tot_Dsg_Unts_2019                 float64
Tot_Clms_2019                     float64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2019   float64
Avg_Spnd_Per_Clm_2019             float64
Outlier_Flag_2019                 float64
Tot_Spndng_2020                   float64
Tot_Dsg_Unts_2020                 float64
Tot_Clms_2020                     float64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2020   float64
Avg_Spnd_Per_Clm_2020             float64
Outlier_Flag_2020                 float64
Tot_Spndng_2021                   float64
Tot_Dsg_Unts_2021                 float64
Tot_Clms_2021                       int64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2021   float64
Avg_Spnd_Per_Clm_2021             float64
Outlier_Flag_2021                   int64
Chg_Avg_Spnd_Per_Dsg_Unt_20_21    float64
CAGR_Avg_Spnd_Per_Dsg_Unt_17_21   float64
dtype: object
```

In [26]:

```
# Outlier flag: Changing data type of Outlier flag to int

df_mdata['Outlier_Flag_2017'] = df_mdata['Outlier_Flag_2017'].astype(int)
df_mdata['Outlier_Flag_2018'] = df_mdata['Outlier_Flag_2018'].astype(int)
df_mdata['Outlier_Flag_2019'] = df_mdata['Outlier_Flag_2019'].astype(int)
df_mdata['Outlier_Flag_2020'] = df_mdata['Outlier_Flag_2020'].astype(int)

display(df_mdata.dtypes)
```

```
Brnd_Name                          object
Gnrc_Name                          object
Tot_Mftr                            int64
Mftr_Name                          object
Tot_Spndng_2017                   float64
Tot_Dsg_Unts_2017                 float64
Tot_Clms_2017                       int64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2017   float64
Avg_Spnd_Per_Clm_2017             float64
Outlier_Flag_2017                   int32
Tot_Spndng_2018                   float64
Tot_Dsg_Unts_2018                 float64
Tot_Clms_2018                       int64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2018   float64
Avg_Spnd_Per_Clm_2018             float64
Outlier_Flag_2018                   int32
Tot_Spndng_2019                   float64
Tot_Dsg_Unts_2019                 float64
Tot_Clms_2019                       int64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2019   float64
Avg_Spnd_Per_Clm_2019             float64
Outlier_Flag_2019                   int32
Tot_Spndng_2020                   float64
Tot_Dsg_Unts_2020                 float64
Tot_Clms_2020                       int64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2020   float64
Avg_Spnd_Per_Clm_2020             float64
Outlier_Flag_2020                   int32
Tot_Spndng_2021                   float64
Tot_Dsg_Unts_2021                 float64
Tot_Clms_2021                       int64
Avg_Spnd_Per_Dsg_Unt_Wghtd_2021   float64
Avg_Spnd_Per_Clm_2021             float64
Outlier_Flag_2021                   int64
Chg_Avg_Spnd_Per_Dsg_Unt_20_21    float64
CAGR_Avg_Spnd_Per_Dsg_Unt_17_21   float64
dtype: object
```

In [129…

```
#Filling missing values of Outlier flag with zero

df_mdata['Outlier_Flag_2017'] = df_mdata['Outlier_Flag_2017'].fillna(0)
df_mdata['Outlier_Flag_2018'] = df_mdata['Outlier_Flag_2017'].fillna(0)
df_mdata['Outlier_Flag_2019'] = df_mdata['Outlier_Flag_2017'].fillna(0)
df_mdata['Outlier_Flag_2020'] = df_mdata['Outlier_Flag_2017'].fillna(0)

df_mdata.isnull().sum()
```

Out[129]:
```
Brnd_Name                          0
Gnrc_Name                          0
Tot_Mftr                           0
Mftr_Name                          0
Tot_Spndng_2017                    0
Tot_Dsg_Unts_2017                  0
Tot_Clms_2017                      0
Avg_Spnd_Per_Dsg_Unt_Wghtd_2017    0
Avg_Spnd_Per_Clm_2017              0
Outlier_Flag_2017                  0
Tot_Spndng_2018                    0
Tot_Dsg_Unts_2018                  0
Tot_Clms_2018                      0
Avg_Spnd_Per_Dsg_Unt_Wghtd_2018    0
Avg_Spnd_Per_Clm_2018              0
Outlier_Flag_2018                  0
Tot_Spndng_2019                    0
Tot_Dsg_Unts_2019                  0
Tot_Clms_2019                      0
Avg_Spnd_Per_Dsg_Unt_Wghtd_2019    0
Avg_Spnd_Per_Clm_2019              0
Outlier_Flag_2019                  0
Tot_Spndng_2020                    0
Tot_Dsg_Unts_2020                  0
Tot_Clms_2020                      0
Avg_Spnd_Per_Dsg_Unt_Wghtd_2020    0
Avg_Spnd_Per_Clm_2020              0
Outlier_Flag_2020                  0
Tot_Spndng_2021                    0
Tot_Dsg_Unts_2021                  0
Tot_Clms_2021                      0
Avg_Spnd_Per_Dsg_Unt_Wghtd_2021    0
Avg_Spnd_Per_Clm_2021              0
Outlier_Flag_2021                  0
Chg_Avg_Spnd_Per_Dsg_Unt_20_21     0
CAGR_Avg_Spnd_Per_Dsg_Unt_17_21    0
dtype: int64
```

In [27]:
```
# checking for duplicate values
duplicate_values = df_mdata.duplicated()
print(duplicate_values)
```

```
0        False
1        False
2        False
3        False
4        False
         ...
16141    False
16142    False
16143    False
16144    False
16145    False
Length: 16146, dtype: bool
```

In [460…
```
# Checking for any duplicate values
actual_file_path = "C:\\Users\\Sayal\\OneDrive\\Desktop\\6600_NU\\Assignment 1\\Medica
df_mdata = pd.read_csv("C:\\Users\\Sayal\\OneDrive\\Desktop\\6600_NU\\Assignment 1\\Me
duplicate_rows_present = df_mdata.duplicated().any()
if duplicate_rows_present:
```

```
    print("Yes! The file contains duplicate values")
else:
    print("No! The file does not contain duplicate values")
```

```
No! The file does not contain duplicate values
```

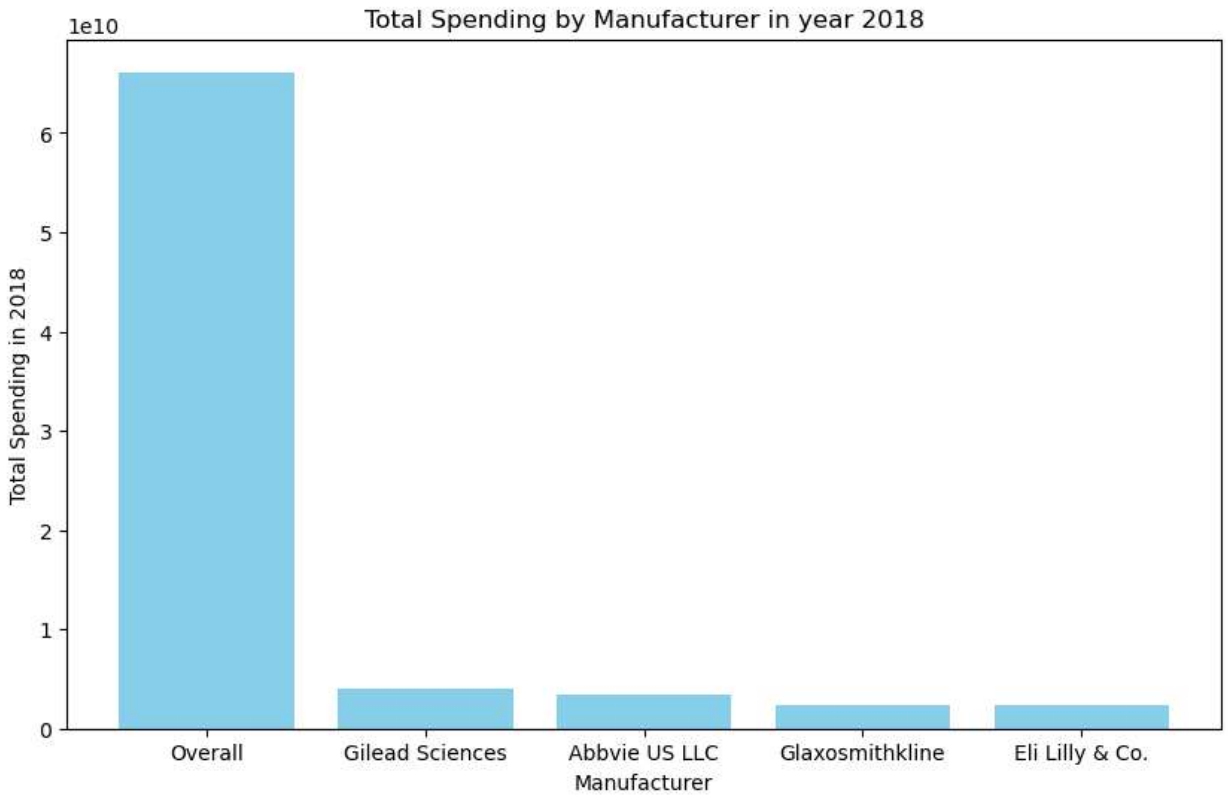
In [62]:
```
# Checking for inconsistent and incomplete data

#removing * from manufacturer column
df_mdata['Mftr_Name'] = df_mdata['Mftr_Name'].str.rstrip('*')
df_mdata['Mftr_Name']
```

Out[62]:
```
0                Overall
1                  Topco
2                Overall
3        Schering-Plough
4                Overall
              ...
16141    Pharmaci/Pfizer
16142            Overall
16143    Pharmaci/Pfizer
16144    Phar-Prep/Pfize
16145    Phar-Nov/Pfizer
Name: Mftr_Name, Length: 16146, dtype: object
```

In [454…
```
# Question 2: Identify the brand with the highest total spending in 2019

index_of_highest_total_spending = df_mdata['Tot_Spndng_2019'].idxmax()
brand_name_highest_total_spending_2019 = df_mdata.loc[index_of_highest_total_spending,
print(brand_name_highest_total_spending_2019)
index_of_highest_total_spending = df_mdata['Tot_Spndng_2019'].idxmax()
Highest_total_spending_2019 = df_mdata.loc[index_of_highest_total_spending, 'Tot_Spndr
print(Highest_total_spending_2019)
```

```
Latuda
1346188758.6
```

In [453…
```
# Question 3: Generate the table showing the top 5 manufacturers with the hihgest tota

mftr = df_mdata.groupby(['Mftr_Name'])
total_spending_mftr_2018 = mftr['Tot_Spndng_2018'].sum()
mftr_sort = total_spending_mftr_2018.sort_values(ascending=False)
top_five_mftr_hihest_tot_spending_2018 = mftr_sort.head(5)

# Bar chart
plt.figure(figsize=(10,6))
plt.bar(top_five_mftr_hihest_tot_spending_2018.index,top_five_mftr_hihest_tot_spending
plt.xlabel('Manufacturer')
plt.ylabel('Total Spending in 2018')
plt.title('Total Spending by Manufacturer in year 2018')
plt.show()

# creating a dataframe to generate a table
table = pd.DataFrame({'Manufacturer':top_five_mftr_hihest_tot_spending_2018.index, 'To
print("Below is the table showing top 5 manufacturers with the highest total spending
table
```



Below is the table showing top 5 manufacturers with the highest total spending in the year 2018

Out[453]:

	Manufacturer	Total Spending in 2018
0	Overall	6.612381e+10
1	Gilead Sciences	4.041785e+09
2	Abbvie US LLC	3.441056e+09
3	Glaxosmithkline	2.327969e+09
4	Eli Lilly & Co.	2.313548e+09

In [452…

```
# Question 4: List the generic names with the highest average spending per claim in 20

names_generic = df_mdata.groupby(['Gnrc_Name'])
tot_avg_spend_per_claim_2017 = names_generic['Avg_Spnd_Per_Clm_2017'].sum()
sorting_of_generic_names = tot_avg_spend_per_claim_2017.sort_values(ascending=False)
top_generic_names_highest_tot_avg_spend_per_claim_2017 = sorting_of_generic_names.head

# creating a dataframe to generate a table
table = pd.DataFrame({'Generic Names':top_generic_names_highest_tot_avg_spend_per_clai
print("Below is the table showing top generic names with the highest average spending
table
```

Below is the table showing top generic names with the highest average spending per cl
aim in 2017

Out[452]:

	Generic Names	Highest Average Spending Claim in 2017
0	Nusinersen Sodium/PF	268609.739260
1	Coagulation Factor VIIA,Recomb	201692.086640
2	C1 Esterase Inhibitor	184313.731938
3	Antihemophil.FVIII,Full Length	141039.561698
4	Antihemophilic Factor/VWF	137687.955198

In [134…

```
# Question 5: Create a table displaying the brands with the highest total claims in 20


brand_name = df_mdata.groupby(['Brnd_Name'])
tot_claim_2018 = brand_name['Tot_Clms_2018'].sum()
sorting_of_brand_names = tot_claim_2018.sort_values(ascending=False)
top_brands_highest_tot_claim_2018 = sorting_of_brand_names.head(5)

# creating a dataframe to generate a table
table = pd.DataFrame({'Brand Names':top_brands_highest_tot_claim_2018.index, 'Highest
print("Below is the table showing top brands with highest total claims in 2018")
table
```

```
Below is the table showing top brands with highest total claims in 2018
```

Out[134]:

	Brand Names	Highest Total CLaims in 2018
0	Amoxicillin	28170752.0
1	Gabapentin	26688688.0
2	Ibuprofen	25762050.0
3	Atorvastatin Calcium	25615284.0
4	Lisinopril	23698512.0

In [101…

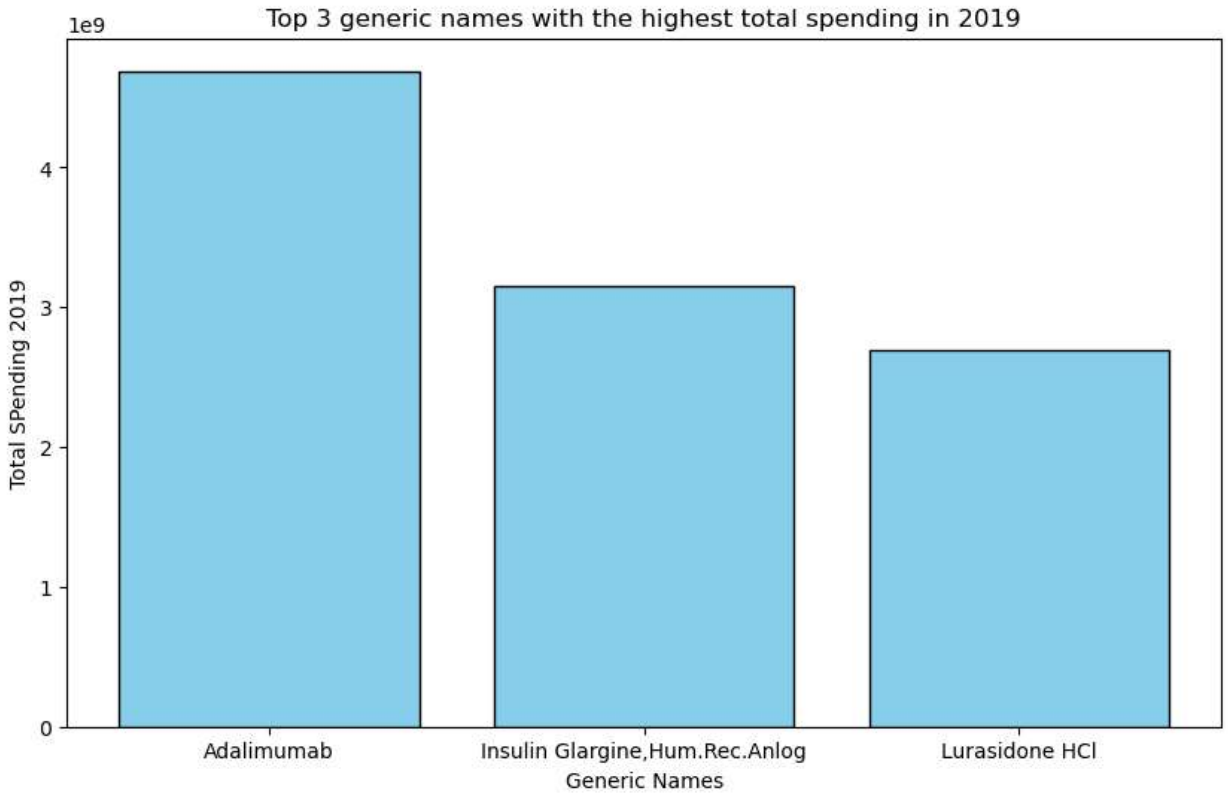
```
# Question 6: Identify the brand with the highest average spending per dosage unit in

index_of_highest_avg_spending_per_dosage_unit_2018 = df_mdata['Avg_Spnd_Per_Dsg_Unt_Wg
brand_name_highest_avg_spending_per_dosage_unit_2018 = df_mdata.loc[index_of_highest_a
print(brand_name_highest_avg_spending_per_dosage_unit_2018)
```

```
Kymriah
```

In [451…

```
# Question 7: Create a table listing the top 3 generic names with the highest total sp

names_generic = df_mdata.groupby(['Gnrc_Name'])
total_spending_2019 = names_generic['Tot_Spndng_2019'].sum()
sorting_of_generic_names = total_spending_2019.sort_values(ascending=False)
top_three_generic_names_highest_tot_spending_2019 = sorting_of_generic_names.head(3)

plt.figure(figsize=(10,6))

plt.bar(top_three_generic_names_highest_tot_spending_2019.index, top_three_generic_nam
plt.xlabel('Generic Names')
plt.ylabel('Total SPending 2019')
plt.title('Top 3 generic names with the highest total spending in 2019')
plt.show()
```

```
# creating a dataframe to generate a table
table = pd.DataFrame({'Generic Names':top_three_generic_names_highest_tot_spending_201
print("Below is the table showing top 3 generic names with the highest total spending
table
```



Below is the table showing top 3 generic names with the highest total spending in the year 2019

Out[451]:

	Generic Names	Highest Total Spending in 2019
0	Adalimumab	4.690149e+09
1	Insulin Glargine,Hum.Rec.Anlog	3.155721e+09
2	Lurasidone HCl	2.692378e+09

In [137…

```
# Question 8: Build a table displaying the brands with the highest average spending pe

brand_names = df_mdata.groupby(['Brnd_Name'])
total_avg_spending_per_claim_2019 = brand_names['Avg_Spnd_Per_Clm_2019'].sum()
sorting_of_brand_names = total_avg_spending_per_claim_2019.sort_values(ascending=False
top_brand_names_highest_avg_spending_per_claim_2019 = sorting_of_brand_names.head(10)

# creating a dataframe to generate a table
table = pd.DataFrame({'Brand Names':top_brand_names_highest_avg_spending_per_claim_201
print("Below is the table showing top brand names with the highest average spending pe
table
```

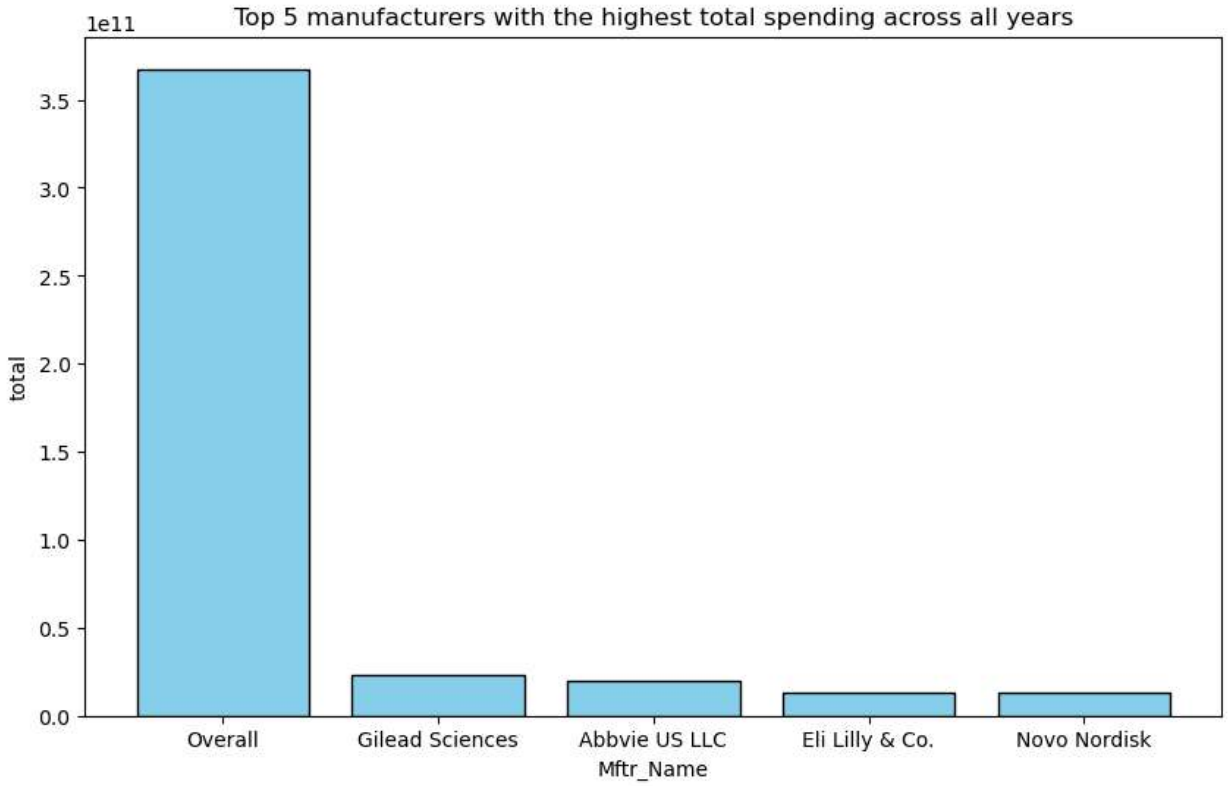
Below is the table showing top brand names with the highest average spending per clai
m 2019

Out[137]:

	Brand Names	Highest Average spending Per Claim 2019
0	Zolgensma	3.435723e+06
1	Revcovi	3.743077e+05
2	Gamifant	3.584478e+05
3	Cablivi	2.803050e+05
4	Spinraza	2.387944e+05
5	Novoseven RT	1.780818e+05
6	Mepsevii	1.618121e+05
7	Myalept	1.589701e+05
8	Ruconest	1.459186e+05
9	Feiba NF	1.354486e+05

In [446…

```
# Question 9: Generate a table showing the top 5 mnufacturers with the highest total s

df_new = df_mdata[['Mftr_Name', 'Tot_Spndng_2017', 'Tot_Spndng_2018','Tot_Spndng_2019'

df_new = df_new.assign(total = ((df_new['Tot_Spndng_2017'] + df_new['Tot_Spndng_2018']

top_five_manufacturers_highest_tot_spending = df_new.groupby(['Mftr_Name'])['total'].s

#plotting chart
plt.figure(figsize=(10,6))
plt.bar(top_five_manufacturers_highest_tot_spending.index, top_five_manufacturers_high
plt.xlabel('Mftr_Name')
plt.ylabel('total')
plt.title('Top 5 manufacturers with the highest total spending across all years')
plt.show()

# creating a dataframe to generate a table
table = pd.DataFrame({'Manufacturers':top_five_manufacturers_highest_tot_spending.inde
print("Below is the table showing top 5 manufacturers with highest total spending acro
table
```



Below is the table showing top 5 manufacturers with highest total spending across all years

Out[446]:

	Manufacturers	Highest Total Spending
0	Overall	3.675242e+11
1	Gilead Sciences	2.296129e+10
2	Abbvie US LLC	1.947680e+10
3	Eli Lilly & Co.	1.323722e+10
4	Novo Nordisk	1.287389e+10

In [163…

```
# Question 10: (Including missing values that I have filled with zero) Identify the ge

# 1st table is for value including 0
df_new = df_mdata[['Gnrc_Name', 'Tot_Spndng_2017','Tot_Spndng_2019']]

df_new = df_new.assign(percentage_increase = ((df_new['Tot_Spndng_2019'] - df_new['Tot

df_new.nlargest(1, 'percentage_increase')
```

Out[163]:

	Gnrc_Name	Tot_Spndng_2017	Tot_Spndng_2019	percentage_increase
0	Acetaminophen	0.0	733.42	inf

In [39]:

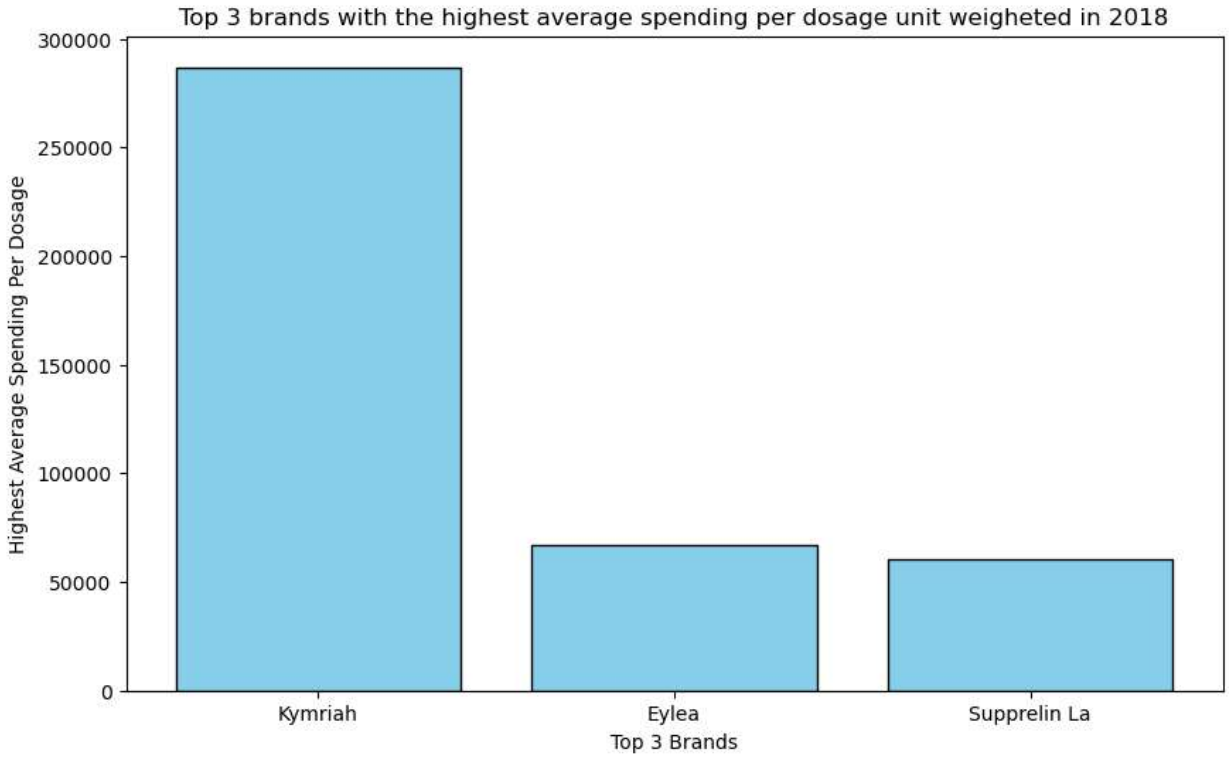
```
# Question 10: (Not Including missing values that I have filled with zero)
df_new = df_mdata[['Gnrc_Name', 'Tot_Spndng_2017','Tot_Spndng_2019']]

df_new = df_new.loc[(df_new != 0).all(axis=1), :].assign(percentage_increase = ((df_ne

df_new.nlargest(1, 'percentage_increase')
```

Out[39]:

	Gnrc_Name	Tot_Spndng_2017	Tot_Spndng_2019	percentage_increase
13781	Succinylcholine Chloride	3.2	867264.61	2.710192e+07

In [465…

```
#Question 11: Create a table displaying the top 3 brands with the highest average spen

brand_name = df_mdata.groupby(['Brnd_Name'])
avg_spend_per_dosage_unit_2018 = brand_name['Avg_Spnd_Per_Dsg_Unt_Wghtd_2018'].sum()
sorting_of_brand_names = avg_spend_per_dosage_unit_2018.sort_values(ascending=False)
top_three_brands_highest_avg_spending_per_dosage_2018 = sorting_of_brand_names.head(3)



plt.figure(figsize=(10,6))

plt.bar(top_three_brands_highest_avg_spending_per_dosage_2018.index, top_three_brands_
plt.xlabel('Top 3 Brands')
plt.ylabel('Highest Average Spending Per Dosage')
plt.title('Top 3 brands with the highest average spending per dosage unit weigheted in
plt.show()

# creating a dataframe to generate a table
table = pd.DataFrame({'Brand Names':top_three_brands_highest_avg_spending_per_dosage_2
table
```



Out[465]:

	Brand Names	Highest Average Spending Per Dosage Unit 2018
0	Kymriah	286905.171580
1	Eylea	66715.142344
2	Supprelin La	60573.203590

In [42]:
```
# Question 12: Generate a table showing the percentage change in total claims from 201

df_percent_change_claim = df_mdata[['Mftr_Name', 'Tot_Clms_2017','Tot_Clms_2019']]

df_percent_change_claim = df_percent_change_claim.loc[(df_percent_change_claim != 0).a

df_percent_change_claim
```

Out[42]:

	Mftr_Name	Tot_Clms_2017	Tot_Clms_2019	percentage_change
4	Overall	47.0	17.0	-63.829787
5	Bayer Healthcar	47.0	17.0	-63.829787
8	Overall	67.0	34.0	-49.253731
9	Hospira/Pfizer	67.0	34.0	-49.253731
10	Overall	453.0	187.0	-58.719647
...	...	...	...	...
16141	Pharmaci/Pfizer	314.0	54.0	-82.802548
16142	Overall	1308.0	690.0	-47.247706
16143	Pharmaci/Pfizer	987.0	362.0	-63.323202
16144	Phar-Prep/Pfize	169.0	190.0	12.426036
16145	Phar-Nov/Pfizer	152.0	138.0	-9.210526

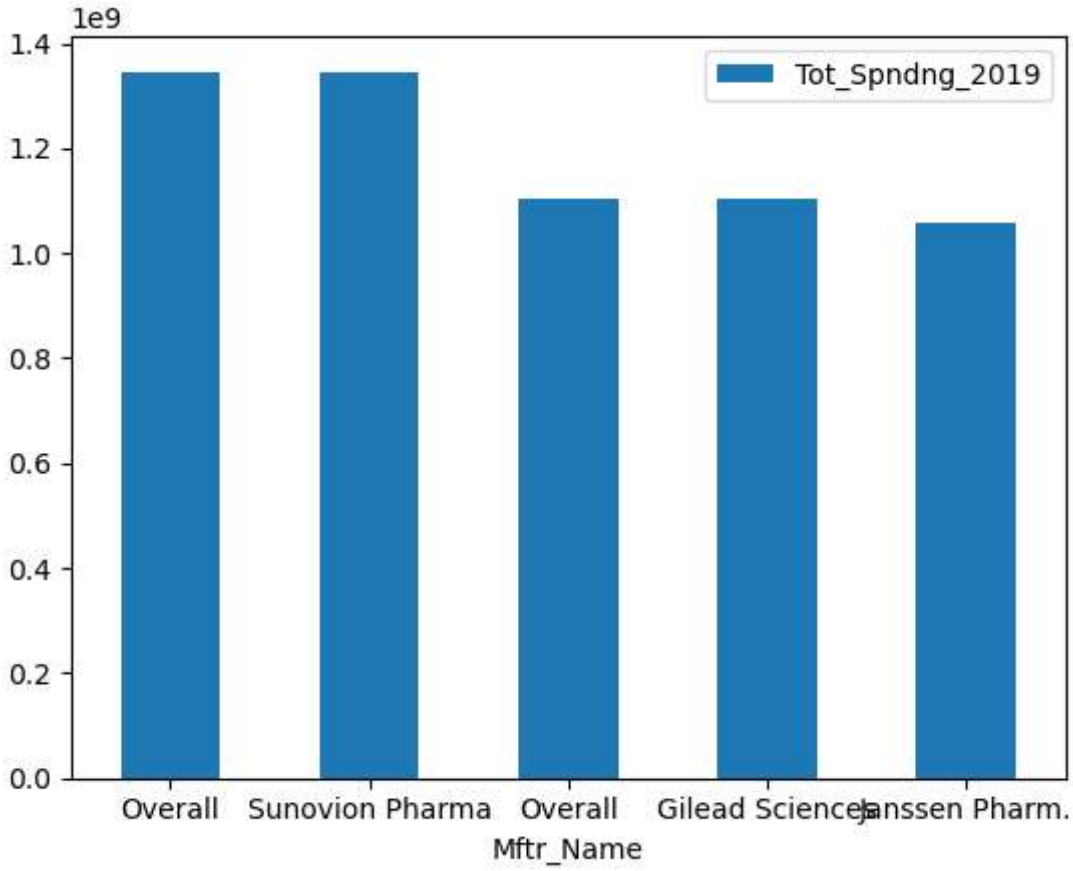
11029 rows × 4 columns

In [173…
```
# Question 13: List the top 5 rows with the highest total spending in 2019. Plot a his


df_top_five_rows_highest_tot_spending_2019 = df_mdata.sort_values('Tot_Spndng_2019',as

df_new = df_top_five_rows_highest_tot_spending_2019[['Mftr_Name', 'Tot_Spndng_2019']]

df_new.plot.bar(x='Mftr_Name', y='Tot_Spndng_2019', rot=0)

df_top_five_rows_highest_tot_spending_2019
```

Out[173]:

	Brnd_Name	Gnrc_Name	Tot_Mftr	Mftr_Name	Tot_Spndng_2017	Tot_Dsg_Unts_20
8172	Latuda	Lurasidone HCl	1	Overall	1.105327e+09	2.961813e+
8173	Latuda	Lurasidone HCl	1	Sunovion Pharma	1.105327e+09	2.961813e+
1647	Biktarvy	Bictegrav/Emtricit/Tenofov Ala	1	Overall	0.000000e+00	0.000000e+
1648	Biktarvy	Bictegrav/Emtricit/Tenofov Ala	1	Gilead Sciences	0.000000e+00	0.000000e+
7543	Invega Sustenna	Paliperidone Palmitate	1	Janssen Pharm.	7.545156e+08	4.926159e+

5 rows × 36 columns



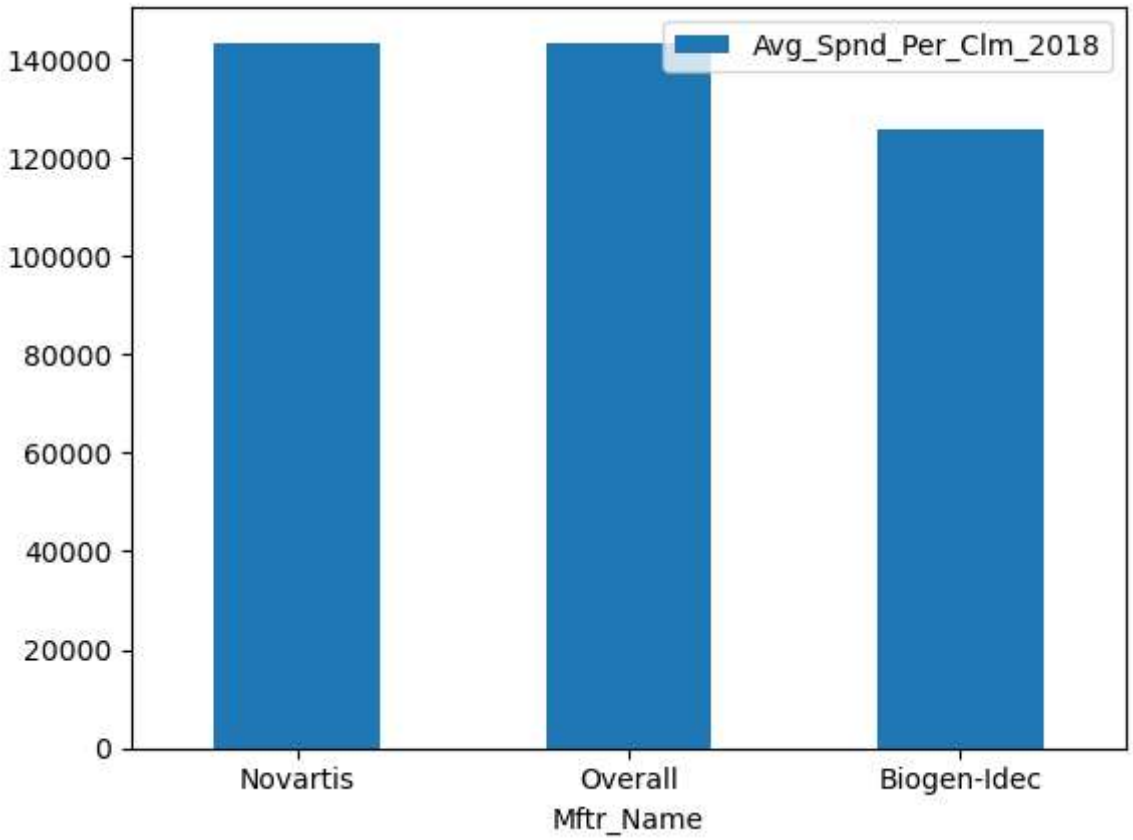
In [175…

```
# Question 14: Build a table listing the top 3 rows with the highest average spending

df_top_three_rows_highest_avg_spending_per_claim_2018 = df_mdata.sort_values('Avg_Spnd

df_top_three_rows_highest_avg_spending_per_claim_2018.plot.bar(x='Mftr_Name', y='Avg_S

df_top_three_rows_highest_avg_spending_per_claim_2018
```

Out[175]:

	Brnd_Name	Gnrc_Name	Tot_Mftr	Mftr_Name	Tot_Spndng_2017	Tot_Dsg_Unts_2017	Tot_
7974	Kymriah	Tisagenlecleucel	1	Novartis	0.000000e+00	0.000	
7973	Kymriah	Tisagenlecleucel	1	Overall	0.000000e+00	0.000	
13675	Spinraza	Nusinersen Sodium/PF	1	Biogen-Idec	1.318874e+08	5718.499	

3 rows × 36 columns



In [398…

```
# Question 15: Identify the manufacturer with the highest total spending in 2017, cons

df_new = df_mdata[['Mftr_Name', 'Brnd_Name', 'Tot_Clms_2017', 'Tot_Clms_2018']]

df_new1 = df_new.groupby(['Brnd_Name'])[['Tot_Clms_2017', 'Tot_Clms_2018']].sum()

df_new1 = df_new1.loc[df_new1['Tot_Clms_2018'] > df_new1['Tot_Clms_2017']]

df_top_spending = pd.merge(df_new1, df_mdata, on='Brnd_Name', how='inner')

df_top_spending.groupby(['Mftr_Name'])['Tot_Spndng_2017'].sum().nlargest(1)
```

Out[398]:

```
Mftr_Name
Overall    2.925735e+10
Name: Tot_Spndng_2017, dtype: float64
```

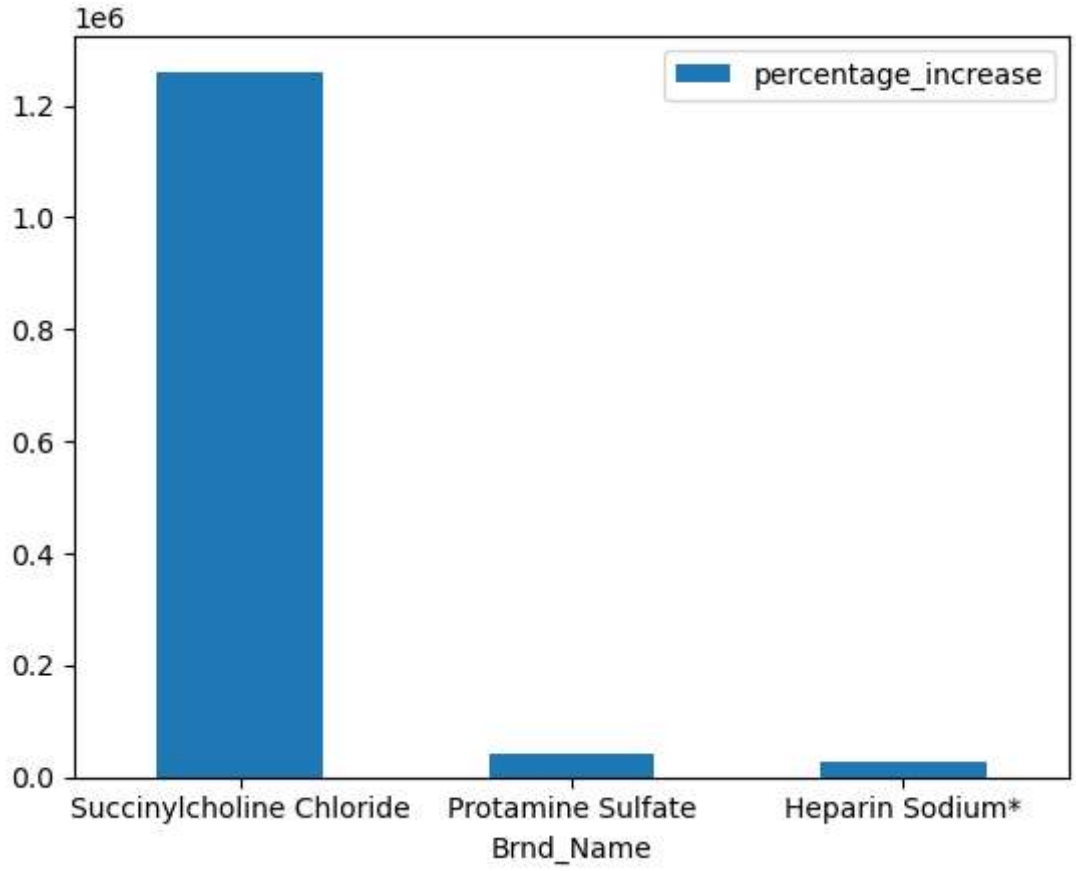
In [233…

```
# Question 16: Create a table displaying the top 3 brands with the highest percentage

#Filtering out column values where average spending is zero(missing data)
df_percent_increase_avg_spend_claim = df_mdata[['Brnd_Name', 'Avg_Spnd_Per_Clm_2017','
```

```
df_percent_increase_avg_spend_claim = df_percent_increase_avg_spend_claim.loc[(df_perc
df_percent_increase_avg_spend_claim = df_percent_increase_avg_spend_claim.nlargest(3,
df_percent_increase_avg_spend_claim


df_percent_increase_avg_spend_claim.plot.bar(x='Brnd_Name', y='percentage_increase', r
df_percent_increase_avg_spend_claim
```

Out[233]:

	Brnd_Name	Avg_Spnd_Per_Clm_2017	Avg_Spnd_Per_Clm_2019	percentage_increase
13781	Succinylcholine Chloride	0.037209	469.299031	1.261141e+06
12529	Protamine Sulfate	1.722110	704.011952	4.078078e+04
6897	Heparin Sodium*	8.354444	2322.270000	2.769682e+04



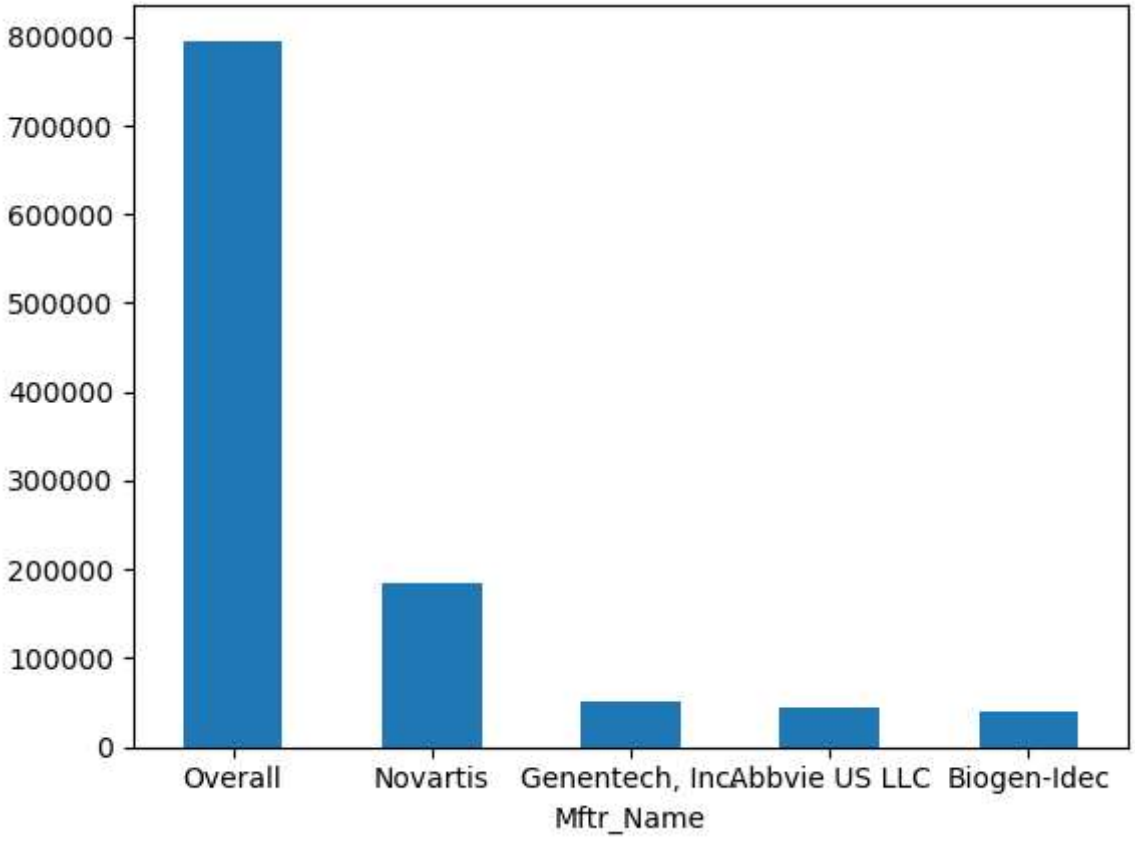
In [422…

```
# Question 17: Identify the brand with the highest average spending per claim in 2019,

df_new = df_mdata[['Brnd_Name', 'Avg_Spnd_Per_Clm_2018', 'Avg_Spnd_Per_Clm_2019']]

df_new1 = df_new.groupby(['Brnd_Name'])[['Avg_Spnd_Per_Clm_2018', 'Avg_Spnd_Per_Clm_20

df_new1 = df_new1.loc[df_new1['Avg_Spnd_Per_Clm_2018'] > df_new1['Avg_Spnd_Per_Clm_201

# Joined the full data with filtered brands
df_top_spending = pd.merge(df_new1, df_mdata[['Brnd_Name', 'Avg_Spnd_Per_Clm_2019']],

df_top_spending

df_top_spending.groupby(['Brnd_Name'])['Avg_Spnd_Per_Clm_2019_y'].sum().nlargest(1)
```

Out[422]:
```
Brnd_Name
Spinraza    238794.38158
Name: Avg_Spnd_Per_Clm_2019_y, dtype: float64
```

In [245…
```
# Question 18: Create a table displaying the top 5 manufacturers with the highest aver


mftr_name_filter = df_mdata[['Mftr_Name', 'Outlier_Flag_2018', 'Avg_Spnd_Per_Dsg_Unt_W

avg_spend_per_dosage_unit_2018 = mftr_name_filter.groupby(['Mftr_Name'])['Avg_Spnd_Per


avg_spend_per_dosage_unit_2018.plot.bar(x='Mftr_Name', y='Avg_Spnd_Per_Dsg_Unt_Wghtd_2

avg_spend_per_dosage_unit_2018
```

Out[245]:
```
Mftr_Name
Overall            796020.342851
Novartis           183744.870830
Genentech, Inc.     51340.074490
Abbvie US LLC       43610.421109
Biogen-Idec         40341.431553
Name: Avg_Spnd_Per_Dsg_Unt_Wghtd_2018, dtype: float64
```



In [440…
```
#Question 19: Find manufacturers that have names with an odd numbers of characters.
# Calculate the total spending for brands associated with these manufacturers in 2019

mftr_characters = df_mdata[['Mftr_Name']]
odd_number_char_manufacturers = [x for i,x in enumerate(mftr_characters['Mftr_Name'])

odd_number_char_manufacturers
```

```
df_new = df_mdata[['Mftr_Name','Tot_Spndng_2019']]

df_new = df_new[df_new['Mftr_Name'].isin(odd_number_char_manufacturers)]

df_new.groupby(['Mftr_Name'])['Tot_Spndng_2019'].sum().nlargest(3)
```

Out[440]:
```
Mftr_Name
Overall            7.207820e+10
Gilead Sciences    4.401205e+09
Abbvie US LLC      3.932866e+09
Name: Tot_Spndng_2019, dtype: float64
```

In [434…
```
# Question 20: Identify the manufacturer with the highest total spending across all br

df_mftr_tot_spend_2018 = df_mdata[['Mftr_Name','Tot_Spndng_2018', 'Brnd_Name']]

tot_spend_2018 = df_mftr_tot_spend_2018.groupby(['Mftr_Name'])['Tot_Spndng_2018'].sum(

# Find out the manufacturer with highest total spending.
sorting_of_manufacturer = tot_spend_2018.sort_values(ascending=False).nlargest(1)

# Filter to only manufacturer with highest total spending.
top_mfg = df_mdata[['Mftr_Name','Tot_Spndng_2018', 'Brnd_Name']].loc[df_mdata['Mftr_Na

top_3_brands_associated_with_highest_spending_mftr = top_mfg.groupby(['Mftr_Name','Brr

top_3_brands_associated_with_highest_spending_mftr
```

Out[434]:
```
Mftr_Name  Brnd_Name
Overall    Humira Pen    1.394367e+09
           Latuda        1.250847e+09
           Mavyret       1.025846e+09
Name: Tot_Spndng_2018, dtype: float64
```

In [438…
```
# Question: 21 For each of the top 3 brands identified in Question 20, calculate the p

df_new = df_mdata[['Brnd_Name', 'Mftr_Name', 'Avg_Spnd_Per_Clm_2018','Avg_Spnd_Per_Clm

df_new = df_new.loc[((df_new['Brnd_Name'] == top_3_brands_associated_with_highest_sper

top_3_brands_perct_increase = df_new.loc[(df_new != 0).all(axis=1), :].assign(percenta

top_3_brands_perct_increase
```

Out[438]:

	Brnd_Name	Mftr_Name	Avg_Spnd_Per_Clm_2018	Avg_Spnd_Per_Clm_2019	percentage_increase
6965	Humira Pen	Overall	5211.964489	5883.829444	12.890820
6966	Humira Pen	Abbvie US LLC	5211.964489	5883.829444	12.890820
8172	Latuda	Overall	1241.935093	1255.782467	1.114984
8173	Latuda	Sunovion Pharma	1241.935093	1255.782467	1.114984
8957	Mavyret	Overall	11323.304565	12424.829259	9.727944
8958	Mavyret	Abbvie US LLC	11323.304565	12424.829259	9.727944

In [437…

```
# Question 22: Among the brands from Question 21 that have a positive percenatge incre

df_new = df_mdata[['Brnd_Name', 'Tot_Spndng_2019']]

top_3_brands_positive = top_3_brands_perct_increase.loc[top_3_brands_perct_increase['p

merged_df = pd.merge(df_new, top_3_brands_positive, on='Brnd_Name', how='inner')

merged_df.groupby(['Brnd_Name'])['Tot_Spndng_2019'].sum().nlargest(1)
```

Out[437]:

```
Brnd_Name
Latuda    5.384755e+09
Name: Tot_Spndng_2019, dtype: float64
```

In [432…

```
# Question 23: Top 3 manufacturers with names starting with 'A' or 'a', based on total

df_mftr_tot_spend_2018 = df_mdata[['Mftr_Name','Tot_Spndng_2018']]

df_mftr_tot_spend_2018 = df_mftr_tot_spend_2018.loc[df_mftr_tot_spend_2018['Mftr_Name'

tot_spend_2018 = df_mftr_tot_spend_2018.groupby(['Mftr_Name'])['Tot_Spndng_2018'].sum(

sorting_of_manufacturer = tot_spend_2018.sort_values(ascending=False).nlargest(3)

sorting_of_manufacturer

#creating a dataframe to generate a table
table = pd.DataFrame({'Manufacturers': sorting_of_manufacturer.index,'Total Spending f
print("Below is the table showing manufacturer names starting with 'A' or 'a', based c
table
```

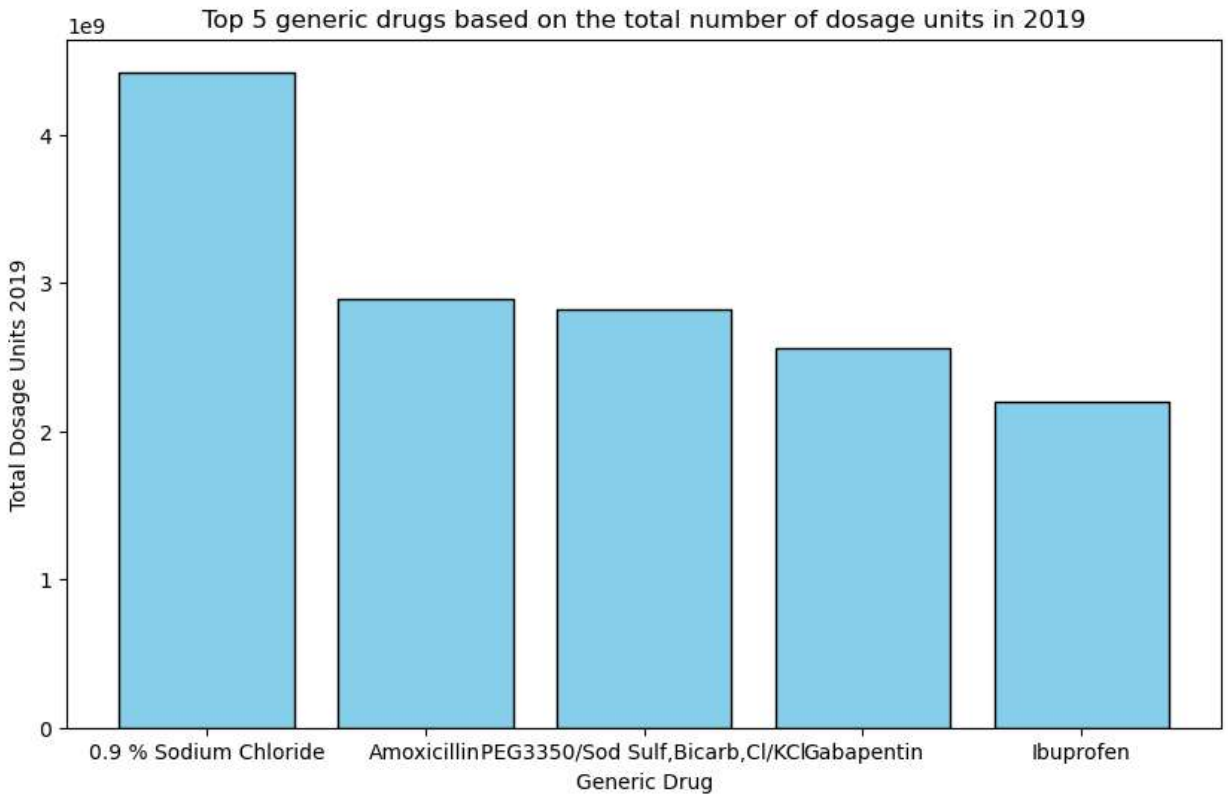
```
Below is the table showing manufacturer names starting with 'A' or 'a', based on tota
l spending for Brands in 2018
```

Out[432]:

	Manufacturers	Total Spending for Brands 2018
0	Abbvie US LLC	3.441056e+09
1	Amgen	1.846425e+09
2	Astrazeneca	1.364785e+09

In [448…

```
# Question 24: Identify the top 5 generic drugs based on the total number of dosage un

generic_drug = df_mdata.groupby(['Gnrc_Name'])
tot_dosage_unit_2019 = generic_drug['Tot_Dsg_Unts_2019'].sum()
sorting_of_generic_drugs = tot_dosage_unit_2019.sort_values(ascending=False)
top_five_generic_drugs_tot_number_dosage_units_2019 = sorting_of_generic_drugs.head(5)



plt.figure(figsize=(10,6))

plt.bar(top_five_generic_drugs_tot_number_dosage_units_2019.index, top_five_generic_dr
plt.xlabel('Generic Drug')
plt.ylabel('Total Dosage Units 2019')
plt.title('Top 5 generic drugs based on the total number of dosage units in 2019')

plt.show()

# creating a dataframe to generate a table
table = pd.DataFrame({'Generic Drugs':top_five_generic_drugs_tot_number_dosage_units_2
print("Below is the table showing brand names with highest average spending per claim
table
```




```
Below is the table showing brand names with highest average spending per claim in the
Year 2019
```

Out[448]:

	Generic Drugs	Top Total dosage Units 2019
0	0.9 % Sodium Chloride	4.420203e+09
1	Amoxicillin	2.889964e+09
2	PEG3350/Sod Sulf,Bicarb,Cl/KCl	2.815912e+09
3	Gabapentin	2.562289e+09
4	Ibuprofen	2.195557e+09

In [ ]: