1 Project: HealthCare Data Analysis and Prediction

2 Domain: Healthcare

3 Organization: Vigor Council

4 Interns Name: Kirti, Nancy, Vishal

[1]:
```python
# Import libraries
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
```

[2]:
```python
# import data from excel
df=pd.read_excel("healthcare_dataset.xlsx")
df
```

[2]:
```
                  Name Age Gender Blood Type Medical Condition \
0       Tiffany Ramirez  81 Female         O-          Diabetes

1           Ruben Burns35     Male         O+            Asthma

2             Chad Byrd61     Male         B-           Obesity

3     Antonio Frederick49     Male         B-            Asthma

4   Mrs. Brandy Flowers51     Male         O-         Arthritis

…                     … …    …         …              …

9995         James Hood83     Male         A+           Obesity

9996    Stephanie Evans  47 Female        AB+         Arthritis

9997       Christopher54     Male         B-         Arthritis
Martinez
9998        Amanda Duke84     Male         A+         Arthritis

9999          Eric King20     Male         B-         Arthritis

                                                    Hospital
   Date of Admission          Doctor                \
0         2022-11-17    Patrick Parker         Wallace-Hamilton
```

```
1         2023-06-01     Diane Jackson Burke, Griffin and
                         Cooper
2         2019-01-09       Paul Baker              Walton LLC
3         2020-05-02    Brian Chandler             Garcia Ltd
4         2021-07-09    Dustin GriffinJones, Brown and Murray
…              …              …                    …
9995       2022-07-29  Samuel Moody     Wood, Martin and Simmons
9996       2022-01-06 Christopher Yates      Nash-Krueger
9997       2022-07-01 Robert Nicholson Larson and Sons 9998 2020-02-06
           Jamie Lewis Wilson-Lyons
9999      2023-03-22     Tasha Avila Torres, Young and Stewart

Insurance Provider Billing Amount Room Number Admission Type \
0             Medicare   37490.983364        146     Elective
1     UnitedHealthcare   47304.064845        404    Emergency
2             Medicare   36874.896997        292    Emergency
3             Medicare   23303.322092        480       Urgent
4     UnitedHealthcare   18086.344184        477       Urgent
…                …            …         …           …

9995 UnitedHealthcare  39606.840083        110     Elective
9996        Blue Cross   5995.717488        244    Emergency
9997        Blue Cross  49559.202905        312     Elective
9998 UnitedHealthcare  25236.344761        420       Urgent
9999             Aetna  37223.965865        290    Emergency

     Discharge   Date   Medication   Test
Results 0 2022-12-01 Aspirin Inconclusive
1     2023-06-15 Lipitor    Normal
2     2019-02-08 Lipitor    Normal
3     2020-05-03 Penicillin Abnormal 4
2021-08-02 Paracetamol Normal
…          …        …          …
9995  2022-08-02 Ibuprofen  Abnormal 9996
      2022-01-29 Ibuprofen  Normal
9997    2022-07-15    Ibuprofen  Normal
9998    2020-02-26    Penicillin Normal
9999    2023-04-15    Penicillin Abnormal
[10000 rows x 15 columns]
```

[3]:
```
# for number of rows and columns
df.shape
```

[3]: (10000, 15)

[4]:
```
# checking the top 5 records of data
df.head()
```

[4]:
```
                Name Age Gender Blood Type Medical Condition \
0     Tiffany Ramirez 81  Female        O-         Diabetes
1        Ruben Burns  35    Male        O+           Asthma
2           Chad Byrd  61    Male        B-          Obesity
3    Antonio Frederick 49    Male  B-    Asthma
4    Mrs. Brandy Flowers     51    Male  O-    Arthritis

 Date of Admission        Doctor                 Hospital \
0     2022-11-17 Patrick Parker  Wallace-Hamilton 1
2023-06-01 Diane Jackson Burke, Griffin and Cooper
2     2019-01-09 Paul Baker Walton LLC 3     2020-05-02
Brian Chandler   Garcia Ltd
4      2021-07-09 Dustin Griffin Jones, Brown and Murray

Insurance Provider Billing Amount Room Number Admission Type
                              \
0         Medicare  37490.983364       146      Elective
1  UnitedHealthcare  47304.064845       404     Emergency
2         Medicare  36874.896997       292     Emergency
3         Medicare  23303.322092       480        Urgent
4 UnitedHealthcare  18086.344184       477        Urgent

  Discharge   Date   Medication   Test
Results     0     2022-12-01     Aspirin
Inconclusive
1     2023-06-15 Lipitor    Normal
2     2019-02-08 Lipitor    Normal
3     2020-05-03 Penicillin Abnormal 4
      2021-08-02 Paracetamol      Normal
```

[5]:
```python
# checking the name of columns
df.columns
```

[5]:
```
Index(['Name', 'Age', 'Gender', 'Blood Type', 'Medical Condition',
       'Date of Admission', 'Doctor', 'Hospital', 'Insurance Provider',
 'Billing Amount', 'Room Number', 'Admission Type', 'Discharge Date',
       'Medication', 'Test Results'],
      dtype='object')
```

[6]:
```python
# checking the data type of each column
df.info()
```

```
<class
'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to
9999 Data columns (total 15
columns):
```

```
 #   Column              Non-Null Count Dtype
---  ------              -------------- -----
 0   Name                10000 non-null object
 1   Age                 10000 non-null int64
 2   Gender              10000 non-null object
 3   Blood Type          10000 non-null object
 4   Medical Condition   10000 non-null object
 5   Date of Admission   10000          non-null
datetime64[ns]
 6   Doctor  10000 non-null object
 7   Hospital     10000 non-null object
 8   Insurance Provider 10000 non-null object
 9   Billing Amount      10000 non-null float64
 10  Room Number         10000 non-null int64
 11  Admission Type      10000 non-null object
 12  Discharge Date      10000          non-null
datetime64[ns]
 13  Medication          10000 non-null object
 14  Test Results        10000 non-null object
dtypes: datetime64[ns](2), float64(1), int64(2),
object(10) memory usage: 1.1+ MB
```

[7]:
```
# checking there is any null value or not
df.isnull().sum()
```

[7]:
```
Name                  0
Age                   0
Gender                0
Blood Type            0
Medical Condition     0
Date of Admission     0
Doctor                0
Hospital              0
Insurance Provider    0
Billing Amount        0
Room Number           0
Admission Type        0
Discharge Date        0
Medication            0
Test Results          0
dtype: int64
```

5 Exploratory Data Analysis

6 1. Age Distribution:

Analyze the age distribution to understand the demographics of patients admitted.

[8]:
```
pd.crosstab(index=df["Age"],columns=df['Age'])
```

[8]:
```
Age   18   19   20   21      22 23 24 25 26 27 … 76 77 78 79 80 \
Age                                            …
18     164  0    0    0    0    0    0    0    0    0 …  0    0
            0    0    0
19     0 1320    0    0    0    0    0    0    0 …  0    0    0
            0    0
20     0     0 169 0    0    0    0    0    0    0 …  0    0    0
            0    0
21     0     0    0 153 0    0    0    0    0    0 …  0    0    0
            0    0
22     0     0    0    0 123 0    0    0    0    0 …  0    0    0
            0    0
..       … … … … … .. .. .. .. .. … .. .. .. .. ..
81     0     0    0    0    0    0    0    0    0    0 …  0    0
            0    0    0
82     0     0    0    0    0    0    0    0    0    0 …  0    0
            0    0    0
83     0     0    0    0    0    0    0    0    0    0 …  0    0
            0    0    0
84     0     0    0    0    0    0    0    0    0    0 …  0    0
            0    0    0
85     0     0    0    0    0    0    0    0    0    0 …  0    0
            0    0    0

Age   81   82   83   84   85
Age
18     0     0    0    0    0
19     0     0    0    0    0
20     0     0    0    0    0
21     0     0    0    0    0
22     0     0    0    0    0
..     … … … … …
81     159  0    0    0    0
82     0 1470    0    0
83     0     0 131 0    0
84     0     0    0 133 0
85     0     0    0    0 123

[68 rows x 68 columns]
```

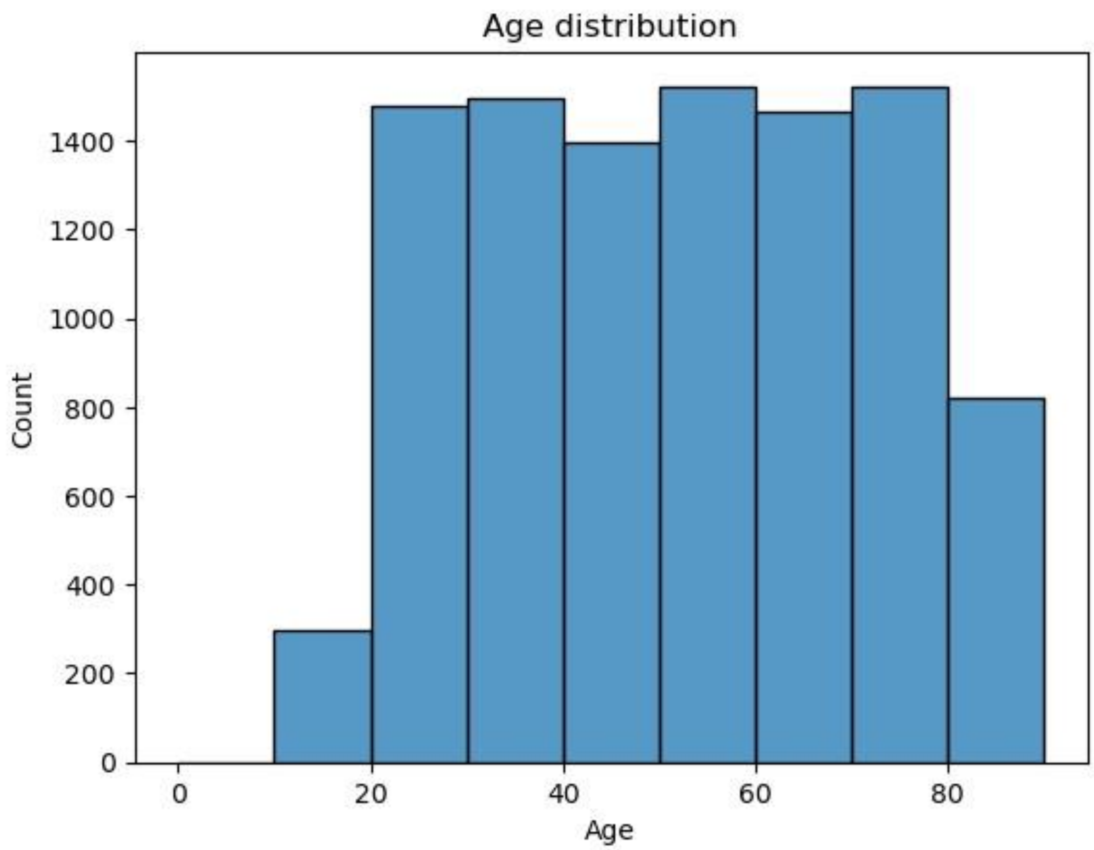
[9]:
```
df.groupby(pd.cut(df['Age'],bins=[0,10,20,30,40,50,60,70,80,90]))["Age"].count()
```

[9]: Age

```
(0, 10]        0
(10, 20]     465
(20, 30]    1438
(30, 40]    1504
(40, 50]    1389
(50, 60]    1543
(60, 70]    1448
(70, 80]    1520
(80, 90]     693
Name: Age, dtype: int64
```

```
[10]:
g=sns.histplot(data=df,x=df['Age'],bins=[0,10,20,30,40,50,60,70,80,90])
g.set_title("Age distribution")
```

```
[10]: Text(0.5, 1.0, 'Age distribution')
```



Research Analysis: From above analysis we conclude that age group of 50-60 people admitted most in the hospital

7 2. Gender Ratio:

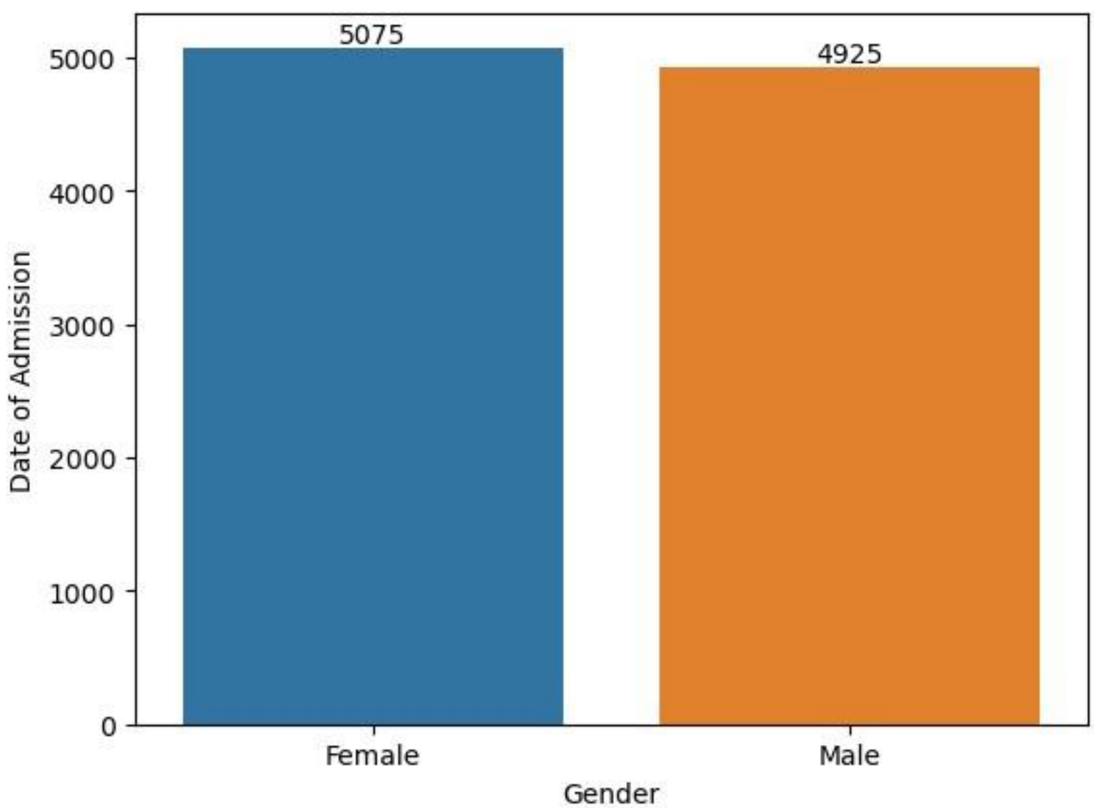
Determine the gender ratio of admitted patients to identify any gender-specific healthcare trends.

[11]:
```
pd.crosstab(index=df["Medical Condition"],columns=df['Gender'])
```

[11]:
```
Gender             Female  Male
Medical Condition
Arthritis             815   835
Asthma                874   834
Cancer                887   816
Diabetes              825   798
Hypertension          836   852
Obesity               838   790
```

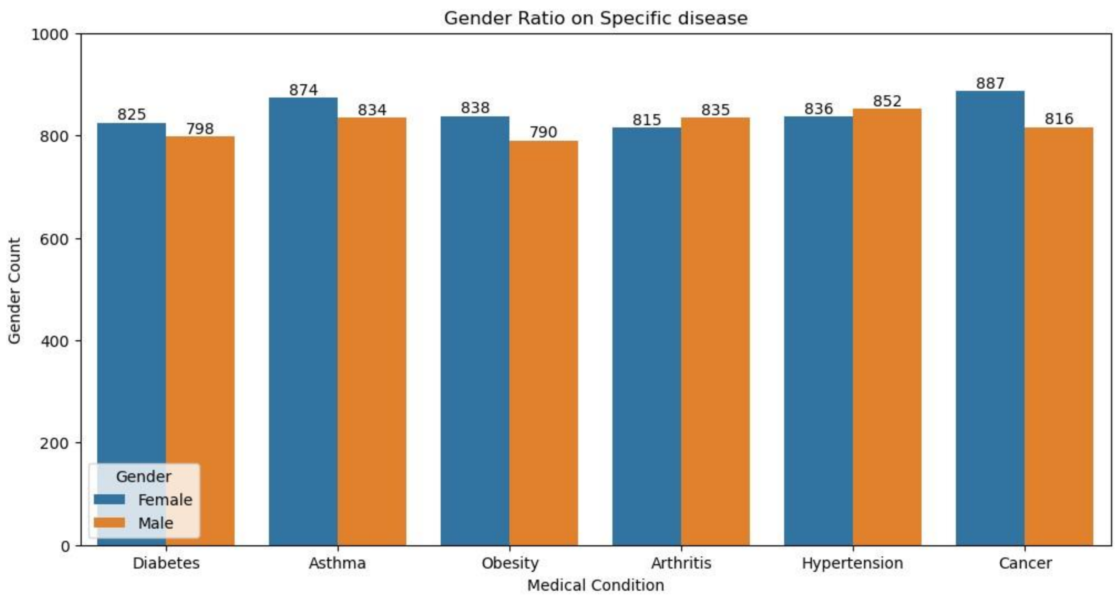
[12]:
```
gen=df.groupby(["Gender"],as_index=False)["Date of Admission"].count().
 ↪sort_values(by='Date of Admission', ascending
=False) print(gen) ax=sns.barplot(data=gen,
x="Gender", y="Date of Admission") for bar in
ax.containers: ax.bar_label(bar)
```

```
   Gender Date of Admission
0    Female  5075
1    Male    4925
```



[13]:
```
plt.figure(figsize=(12,6))
gr=sns.countplot(data=df,x="Medical
Condition",hue='Gender')
gr.set_title("Gender Ratio on Specific
disease") gr.set_ylabel("Gender Count")
for bars in gr.containers:
gr.bar_label(bars) gr.set_ylim(ymax=1000)
    gr.figure.get_axes()[0].legend(title="Gender",
                    loc="lower left")
```

[13]:
```
<matplotlib.legend.Legend at 0x17320a8f2d0>
```



Reseach Analysis: from the above insights we conclude that there is gender specific disease in admitted patients i.e Cancer in Females and Hypertension in Males . Highest number of admission is taken by Females.

8 3. Blood Type Frequency:

Examine the frequency of different blood types among patients for potential correlation with medical conditions or treatments.

[14]:
```
BT=df.groupby(["Blood Type"],as_index=False)["Date of
Admission"].count(). ↪sort_values(by="Date of Admission",
ascending=False) BT
```

[14]:

	Blood Type	Date of Admission
3	AB-	1275

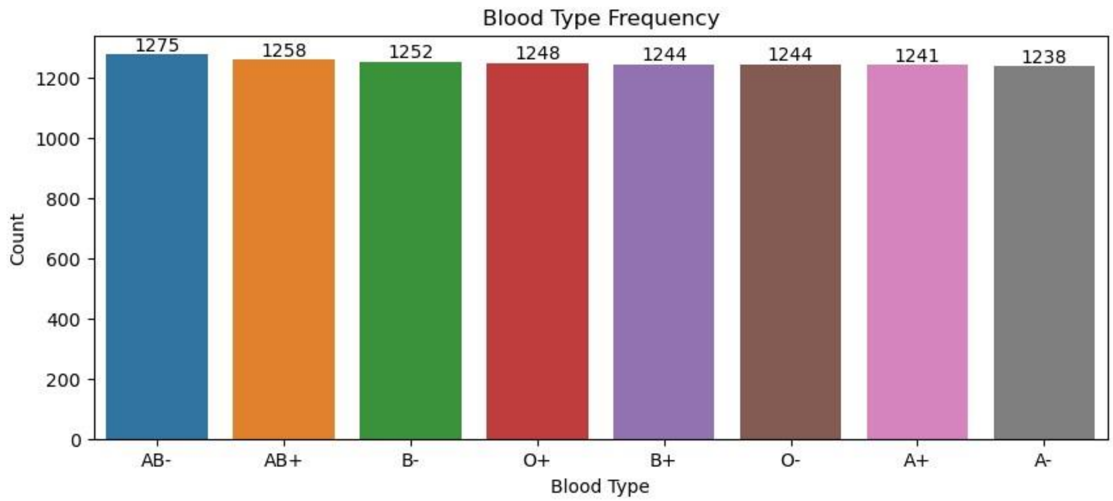
```
2      AB+            1258
5       B-            1252
6       O+            1248
4       B+            1244
7       O-            1244
0       A+            1241
1       A-            1238
```

[15]:
```
df.groupby("Blood Type")["Date of Admission"].count()
```

[15]:
```
Blood Type
A+     1241
A-     1238
AB+    1258
AB-    1275
B+     1244
B-     1252
O+     1248
O-     1244
Name: Date of Admission, dtype: int64
```

[16]:
```
plt.figure(figsize=(10,4))
bf=sns.barplot(data=BT, y="Date of Admission",x="Blood Type")
for bar in bf.containers:
    bf.bar_label(bar)
bf.set_title("Blood Type Frequency")
bf.set_ylabel("Count")
```

[16]:
```
Text(0, 0.5, 'Count')
```

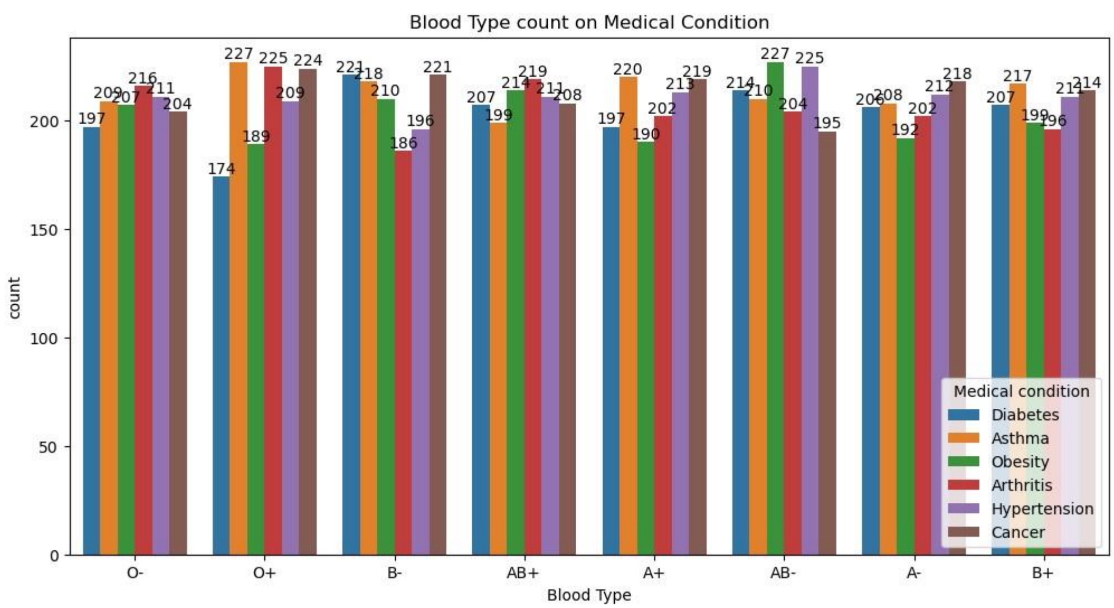


[17]:
```
pd.crosstab(index=df["Blood Type"],columns=df['Medical Condition'])
```

```
[17]: Medical Condition Arthritis Asthma Cancer Diabetes Hypertension
     Obesity Blood Type
     A+                        202    220    219     197          213     190
     A-                        202    208    218     206          212     192
     AB+                       219    199    208     207          211     214
     AB-                       204    210    195     214          225     227
     B+                        196    217    214     207          211     199
     B-                        186    218    221     221          196     210
     O+                        225    227    224     174          209     189
     O-                        216    209    204     197          211     207
```

```
[18]: plt.figure(figsize=(12,6)) bf=sns.countplot(data = df, hue =
     'Medical Condition' , x = "Blood Type")
     bf.set_title("Blood Type count on Medical
     Condition") for bars in bf.containers:
         bf.bar_label(bars)
          bf.figure.get_axes()[0].legend(title="Medical condition",
                                    loc="lower right")
```

```
[18]: <matplotlib.legend.Legend at 0x17320b2bc10>
```



Research Analysis: From the above graph we can see that AB- have the highest rate of patients and A- have the lowest rate.

9 4. Common Medical Conditions:

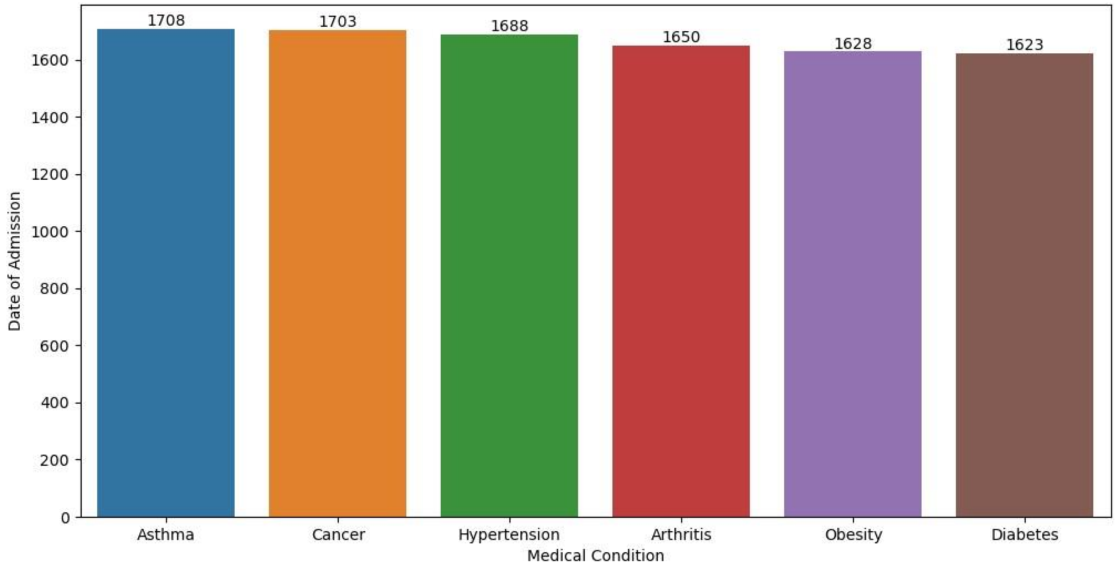
Identify the most prevalent medical conditions among admitted patients to prioritize resources and healthcare services.

[19]:
```
med_condition=df["Medical Condition"].value_counts()
med_condition
```

[19]:
```
Medical Condition
Asthma          1708
Cancer          1703
Hypertension    1688
Arthritis       1650
Obesity         1628
Diabetes        1623
Name: count, dtype: int64
```

[20]:
```
plt.figure(figsize=(12,6))
Med = df.groupby(['Medical Condition'], as_index=False)['Date of Admission'].
 ↪count().sort_values(by='Date of Admission',
ascending=False).head(10) print(Med) ax=sns.barplot(data = Med,
x = 'Medical Condition',y= 'Date of Admission') for bars in
ax.containers:
    ax.bar_label(bars)
```

```
  Medical Condition  Date of Admission
1            Asthma               1708
2            Cancer               1703
4      Hypertension               1688
0         Arthritis               1650
5           Obesity               1628
3          Diabetes               1623
```



Research Analysis: from the insights we conclude that the most common medical condition is Asthma

10 5. Admission Trends Over Time:

Analyze the dates of admission to identify any seasonal or temporal patterns in hospital admissions.

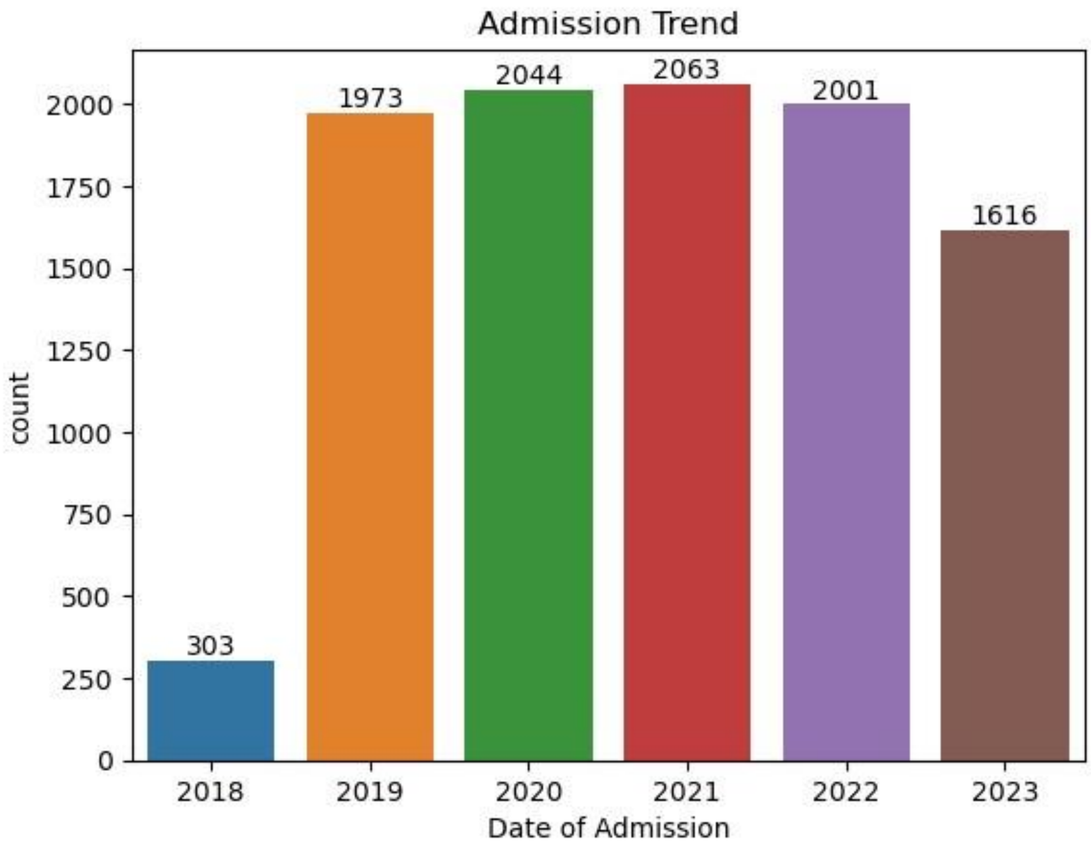
[21]:
```
df["Date of Admission"]=pd.to_datetime(df['Date of
Admission']) df["Date of Admission"]=df["Date of
Admission"].dt.year
```

[22]:
```
admission_per_year=df["Date of Admission"].value_counts()
admission_per_year
```

[22]:
```
Date of Admission
2021  2063
2020   2044
2022   2001
2019   1973
2023   1616
2018    303
```

```
Name: count, dtype: int64
```

```
[23]: DOA=sns.countplot(data= df,  x = df["Date of Admission"])
      DOA.set_title('Admission Trend')
      plt.figure(figsize=(12,8))
      for bar in DOA.containers:
          DOA.bar_label(bar)
```




```
<Figure size 1200x800 with 0 Axes>
```

Research Analysis: from the above graph we can see that in 2021 there is highest number of admission.

11 6. Attending Doctors:

Assess the performance and workload of different doctors based on the number of admissions they handle.

```
[24]: doc=df.groupby(["Doctor"],as_index=False)['Date of
Admission'].count().
```

```
↪sort_values(by='Date of Admission',
ascending=False).head(20) doc
```

[24]:
```
                  Doctor  Date of Admission
6460  Michael Johnson                  7
6216    Matthew Smith                  5
6522    Michael Smith                  5
6572 Michelle Anderson                 5
7593     Robert Brown                  5
4087   Jennifer Smith                  5
3724      James Perez                  5
3753   James Williams                  5
809    Ashley Jackson                  4
1753 Christopher Davis                 4
1789 Christopher Jones                 4
9340 William Rodriguez                 4
6400    Michael Brown                  4
2302    David Johnson                  4
7649    Robert Miller                  4
4222   Jessica Wilson                  3
2179     Daniel Smith                  3
3852       Jason Hall                  3
8761     Thomas Brown                  3
7847    Ryan Thompson                  3
```

[25]:
```
df["Doctor"].duplicated().any()
```

[25]:
```
True
```

[26]:
```
df["Doctor"].value_counts()
```

[26]:
```
Doctor

Michael Johnson    7
 Robert Brown      5
 Michelle Anderson 5
Matthew Smith      5
Jennifer Smith     5
                  ..
 Sandra Howard     1
 Steven Fuller     1
 Benjamin Lawson   1
 Allison Woods     1
 Tasha Avila       1
 Name: count, Length: 9416, dtype: int64
```

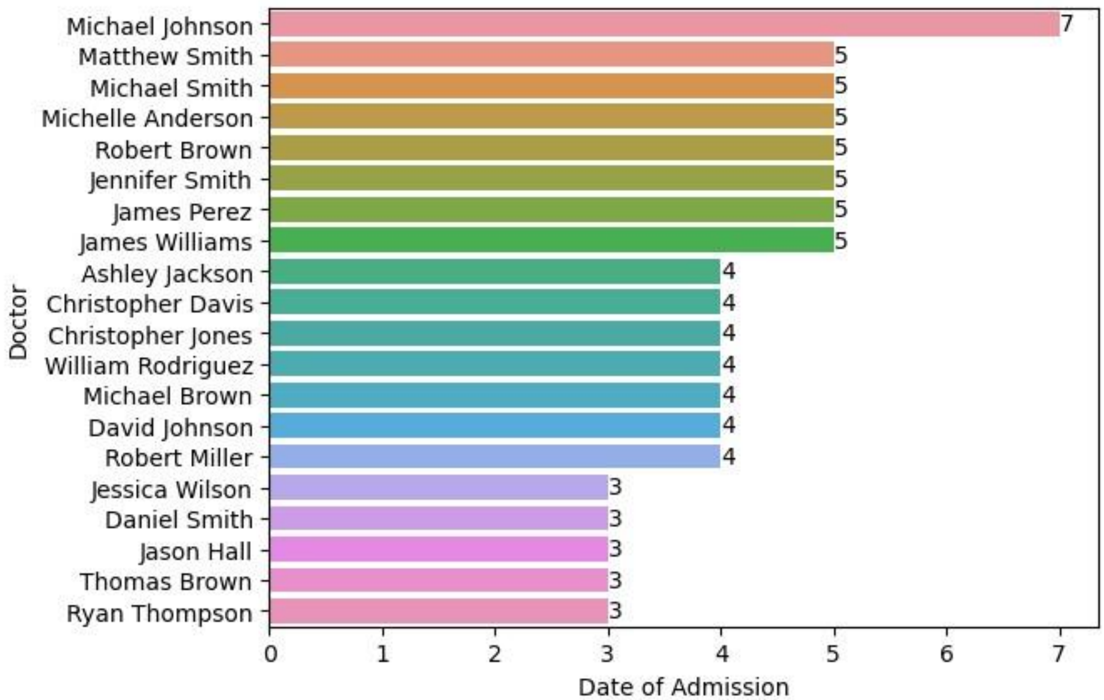
[27]:
```
df[df['Doctor']=='Michael Johnson']
```

[27]:
```
             Name Age Gender Blood Type Medical Condition \
```

```
1862 Sherri Mckinney 67   Male       O+         Asthma
5908 Brittany Glover 57   Male       A+         Asthma
6397   Maria Carter  59 Female      AB-       Diabetes
6411  Joshua Bailey  78 Female       A+        Obesity
6875   Rebecca King  45 Female       O+         Cancer
9085  Peter Matthews  30   Male       B-         Asthma
9909  Jonathan Perry  24   Male       A-      Arthritis

     Date of Admission       Doctor                  Hospital \
1862            2022 Michael Johnson             Harris-Cowan
5908            2021 Michael Johnson             Harrison LLC
6397             2022 Michael Johnson Jackson, Thompson and
                 Thomas
6411            2019 Michael Johnson          Thomas-Franklin
6875            2021 Michael Johnson              Farrell Inc
9085            2019 Michael Johnson           Fletcher Group
9909            2022 Michael Johnson         Watkins and Sons
    Insurance Provider Billing Amount Room Number Admission Type \
1862             Aetna   49559.841901       288        Urgent
5908          Medicare   33099.519497       192      Elective
6397        Blue Cross    1428.619493       461        Urgent
6411             Aetna   38310.284764       386     Emergency
6875 UnitedHealthcare    3678.787754       457      Elective
9085 UnitedHealthcare   27108.266411       241      Elective
9909          Medicare   28391.155073       198     Emergency

    Discharge Date Medication Test Results
1862 2022-03-05 Lipitor Inconclusive
5908   2021-12-20   Ibuprofen    Abnormal
6397   2022-02-14     Aspirin    Abnormal
6411  2019-12-04  Ibuprofen  Abnormal  6875
2021-10-10 Paracetamol Inconclusive
9085   2019-06-21     Aspirin    Abnormal
9909   2022-01-20     Lipitor    Abnormal
```

[28]:
```
d=sns.barplot(data=doc, y="Doctor", x="Date of Admission")
plt.figure(figsize=(15,8))
```

```
for bar in d.containers:
    d.bar_label(bar)
```




```
<Figure size 1500x800 with 0 Axes>
```

Research Analysis from the above graph we can see that the doctor Michael Johnson have the more workload .

12 7. Hospital Utilization:

Determine which hospitals have the highest admission rates and assess their capacity to handle patient influx.

[29]:
```
hospital=df["Hospital"].value_counts().head(120)
hospital
```

[29]:
```
Hospital
Smith PLC         19
Smith and Sons    17
Smith Ltd         14
Smith Inc         14
```

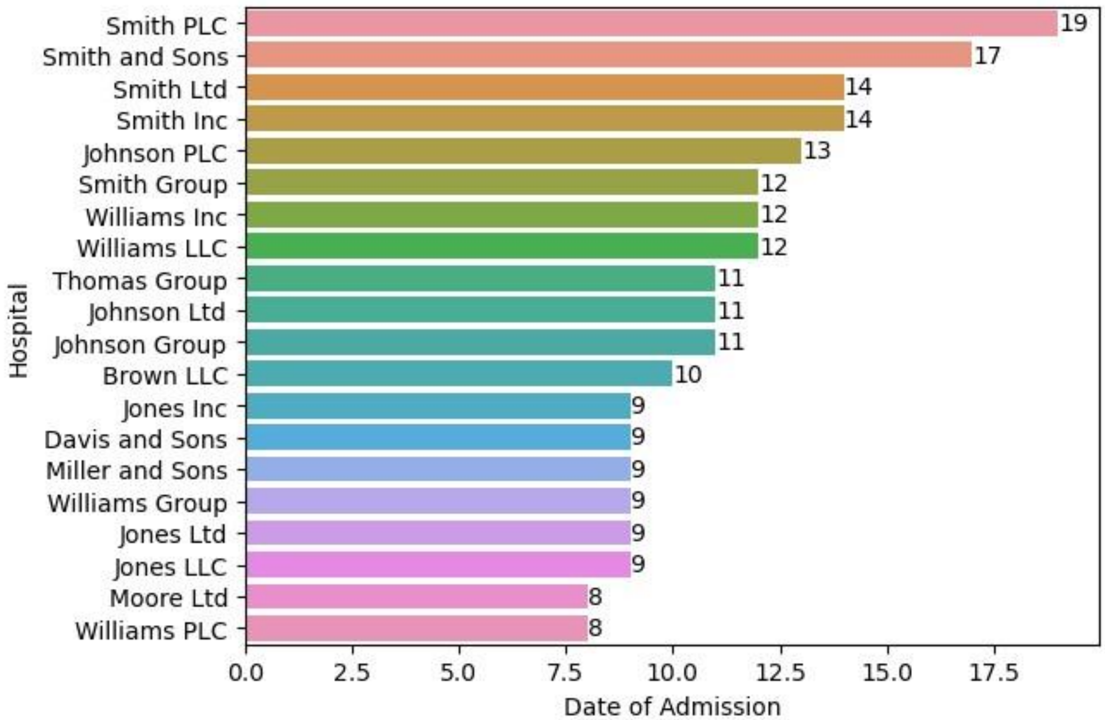
```
    Johnson PLC     13
..
    Alvarez Inc      4
    Bell LLC         4
    Morgan Ltd       4
    Allen Group      4
    West PLC         4
    Name: count, Length: 120, dtype: int64
```

```
[30]: hos=df.groupby(["Hospital"],as_index=False)["Date of Admission"].count().
      ↪sort_values(by='Date of Admission', ascending=False).head(20)
      hos
```

```
[30]:             Hospital  Date of Admission
7114         Smith PLC                 19
7115    Smith and Sons                 17
7113         Smith Ltd                 14
7111         Smith Inc                 14
3769       Johnson PLC                 13
7110       Smith Group                 12
8282      Williams Inc                 12
8283      Williams LLC                 12
7561      Thomas Group                 11
3768       Johnson Ltd                 11
3765     Johnson Group                 11
835          Brown LLC                 10
3899         Jones Inc                  9
1797    Davis and Sons                  9
5002   Miller and Sons                  9
8281    Williams Group                  9
3901         Jones Ltd                  9
3900         Jones LLC                  9
5151         Moore Ltd                  8
```

```
8285    Williams PLC                  8
```

[31]:
```
h=sns.barplot(data=hos, y='Hospital',  ='Date of Admission')
for bar in h.containers:
    h.bar_label(bar)
```



Research Analysis: from the above graph we can see that the SMITH PLC hospital have the highest rate of admissions.

13 8.Insurance Coverage:

Analyze the distribution of insurance providers among admitted patients to understand coverage gaps or preferences.

[32]:
```
pd.crosstab(index=df["Insurance Provider"],columns=df["Date of Admission"])
```

[32]:

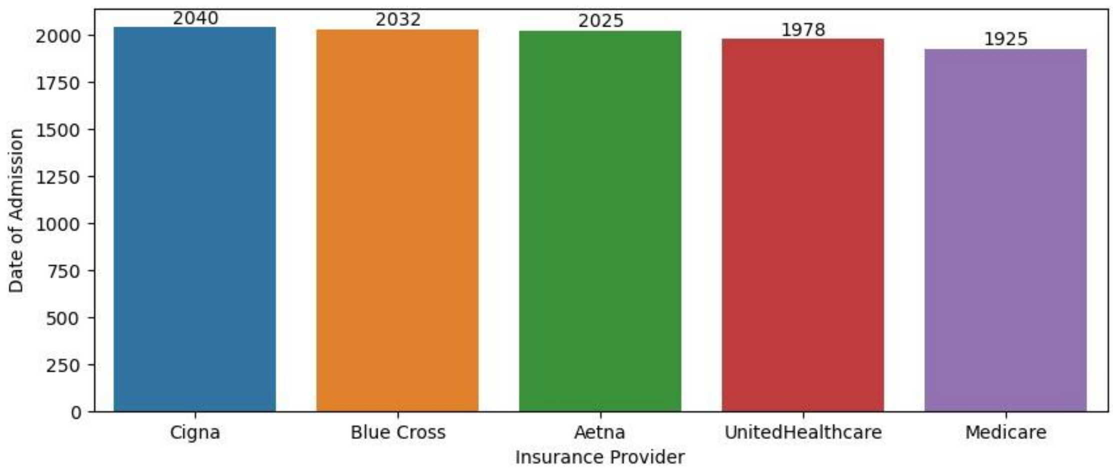
Date of Admission	2018	2019	2020	2021	2022	2023
Insurance Provider						
Aetna	63	401	421	406	401	333
Blue Cross	56	403	429	420	407	317
Cigna	65	385	427	438	401	324
Medicare	59	374	390	385	396	321

```
UnitedHealthcare    60   410   377   414   396   321
```

[33]:
```
plt.figure(figsize=(10,4))
IP=df.groupby(["Insurance Provider"], as_index=False)["Date of Admission"].
 ↪count().sort_values(by='Date of Admission', ascending=False)
print(IP)
ins=sns.barplot(data=IP, x="Insurance Provider", y="Date of
Admission")
```

```
for bar in ins.containers:
    ins.bar_label(bar)
```

```
  Insurance Provider Date of Admission
2             Cigna              2040
1        Blue Cross              2032
0             Aetna              2025
4  UnitedHealthcare              1978
3          Medicare              1925
```



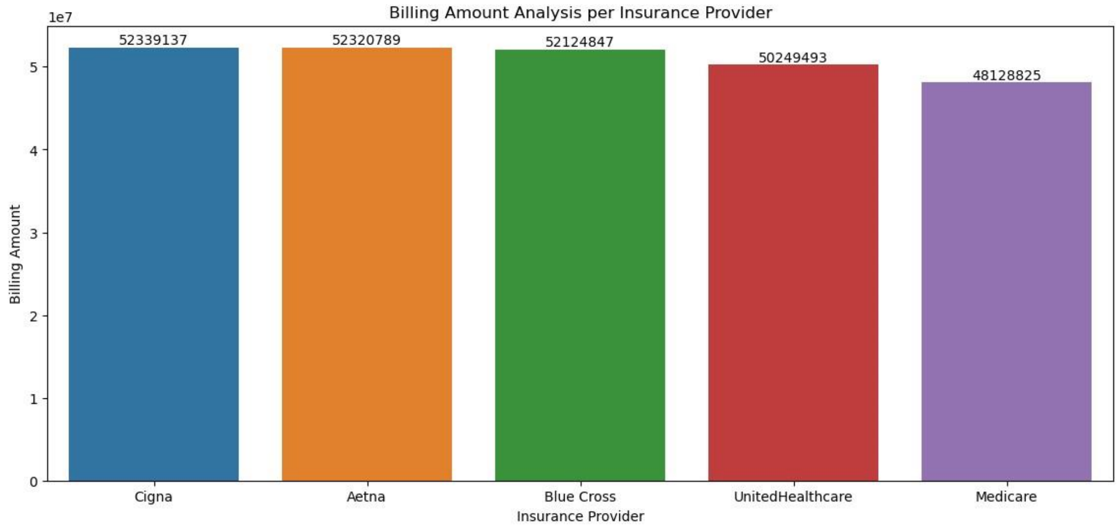
[34]:
```
df['Billing Amount']=df['Billing Amount'].astype('int64')
```

[35]:
```
plt.figure(figsize=(14,6))
     BA = df.groupby(['Insurance Provider'], as_index=False)['Billing
                                                   Amount'].sum().
 ↪sort_values(by='Billing Amount', ascending=False)
print(BA) ax=sns.barplot(x = 'Insurance Provider',y=
'Billing Amount',data = BA)
ax.bar_label(container=ax.containers[0], labels=BA['Billing
Amount']) ax.set_title("Billing Amount Analysis per
Insurance Provider")
```

```
  Insurance Provider Billing Amount
```

```
2          Cigna      52339137
0          Aetna      52320789
1     Blue Cross      52124847
4  UnitedHealthcare   50249493
3       Medicare      48128825
```

[35]: Text(0.5, 1.0, 'Billing Amount Analysis per Insurance Provider')



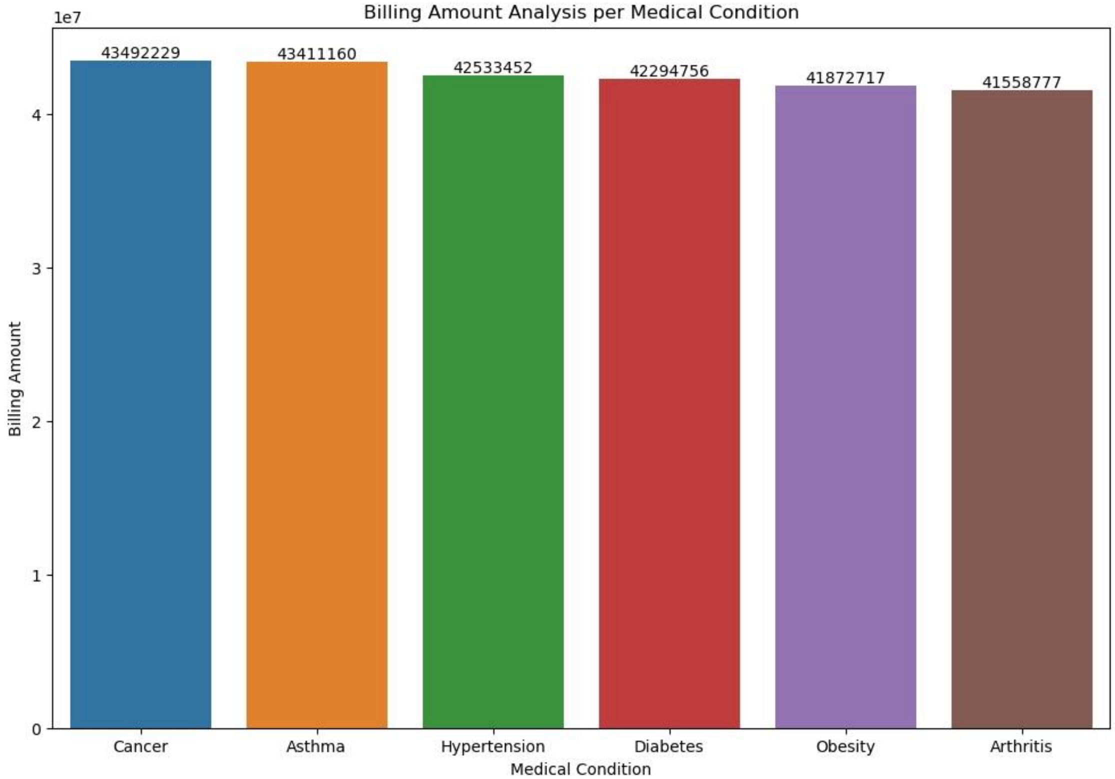
Research analysis: from the above graph we can see that the Cigna is the most prefered insurance provider chosen by the patients and generating highest amount of billing.

9. Billing Amount Analysis: Investigate the billing amounts to identify any outliers or trends in healthcare costs.

[36]:
```
plt.figure(figsize=(12,8))
BA = df.groupby(['Medical Condition'], as_index=False)['Billing Amount'].sum().
 ↪sort_values(by="Billing Amount", ascending=False)
print(BA) ax=sns.barplot(x = 'Medical Condition',y=
'Billing Amount',data = BA)
ax.bar_label(container=ax.containers[0], labels=BA['Billing
Amount']) ax.set_title("Billing Amount Analysis per Medical
Condition")
```

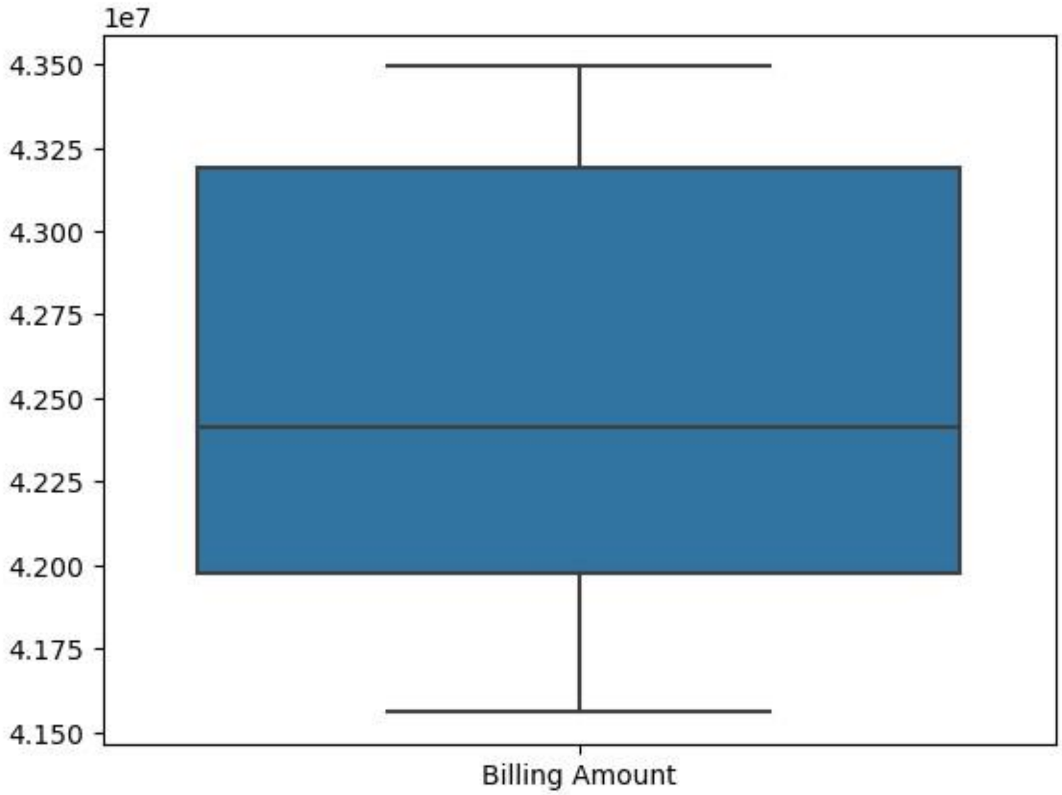
```
  Medical Condition Billing Amount
2            Cancer       43492229
1            Asthma       43411160
4      Hypertension       42533452
3          Diabetes       42294756
5           Obesity       41872717
0         Arthritis       41558777
```

[36]: Text(0.5, 1.0, 'Billing Amount Analysis per Medical Condition')



[37]:
```
sns.boxplot(data=BA)
```

[37]: <Axes: >



[38]:
```
df['Billing Amount'].describe()
```

[38]:
```
count    10000.00000
mean     25516.30910
std      14067.29156
min       1000.00000
25%      13506.25000
50%      25257.50000
75%      37733.00000
max      49995.00000
Name: Billing Amount, dtype: float64
```

Research analysis: from the above graph we see that Cancer has the highest Billing Amount and no outlier is the billing amount.

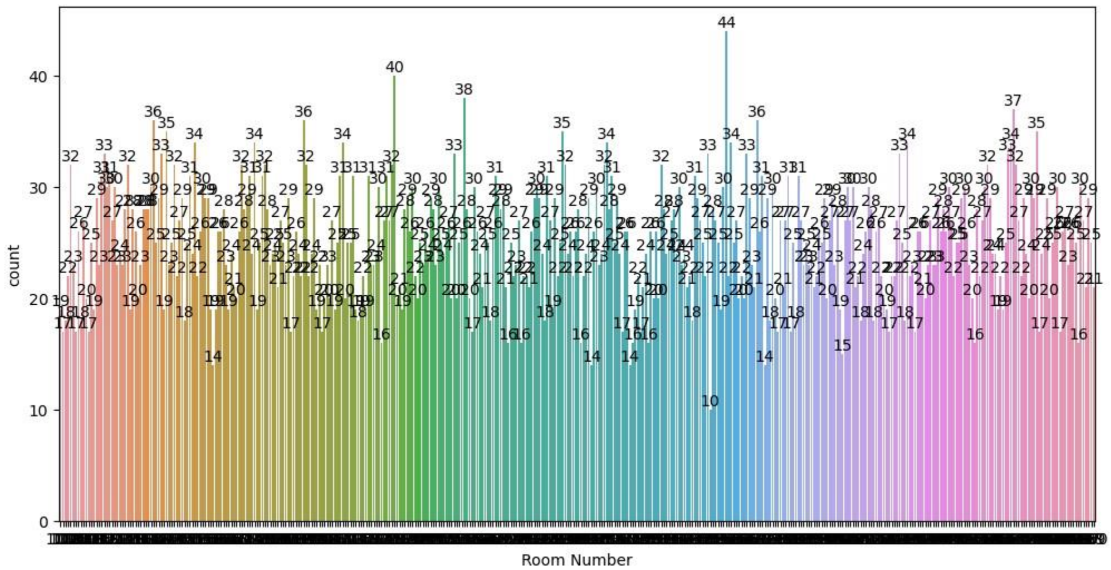
14 10.Room Occupancy:

Examine the distribution of room numbers to optimize room allocation and utilization

[39]:
```
df['Room Number'].value_counts()
```

```
[39]: Room Number
358    44
230    40
257    38
469    37
195    36
       ..
160    14
306    14
321    14
373    14
352    10
Name: count, Length: 400, dtype: int64
```

```
[40]: plt.figure(figsize=(12,6))
ax = sns.countplot(data=df, x=df['Room Number'])
for data in ax.containers:
    ax.bar_label(data)
```



Research Analysis: From the above graph we see that the 358 room number is more utilize according to other rooms.

15 11. Admission Type:

Differentiate between planned admissions (e.g., elective surgeries) and emergency admissions to understand healthcare demands

[41]:
```
pd.crosstab(index=df["Admission Type"], columns=df["Medical Condition"])
```

[41]:

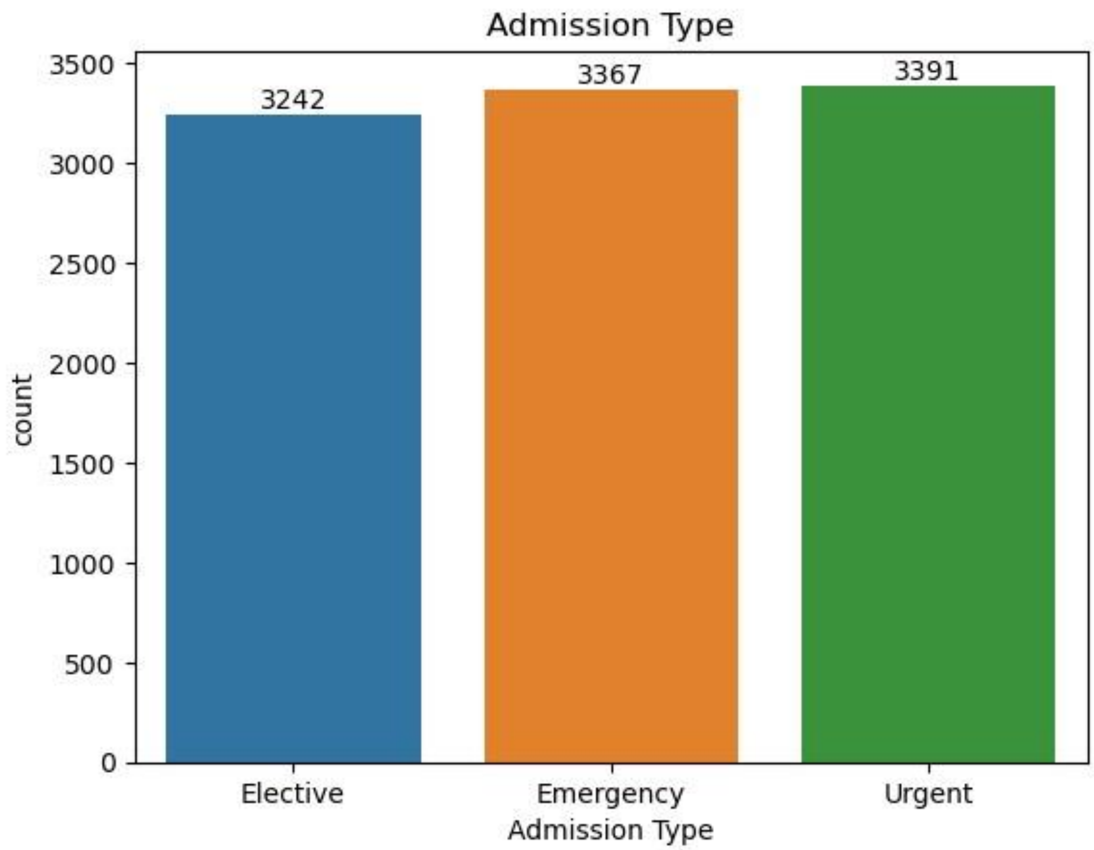
Medical Condition Admission Type	Arthritis	Asthma	Cancer	Diabetes	Hypertension	Obesity
Elective	569	570	555	528	515	505
Emergency	529	556	578	557	578	569
Urgent	552	582	570	538	595	554

[42]:
```
df["Admission Type"].value_counts()
```

[42]:
```
Admission Type
Urgent       3391
Emergency    3367
Elective     3242
Name: count, dtype: int64
```

[43]:
```
AT= sns.countplot(data=df, x="Admission Type")
for bar in AT.containers:
    AT.bar_label(bar)
plt.figure(figsize=(12,4))
AT.set_title("Admission Type")
```

[43]:
```
Text(0.5, 1.0, 'Admission Type')
```



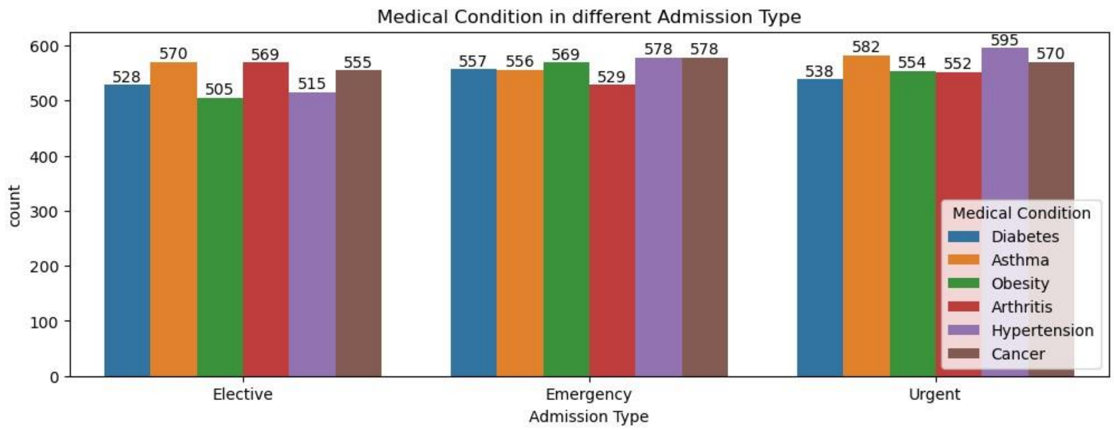

```
<Figure size 1200x400 with 0 Axes>
```

[44]:

```
plt.figure(figsize=(12,4)) ax = sns.countplot(data = df, x =
'Admission Type', hue = 'Medical Condition')

for bars in ax.containers:
    ax.bar_label(bars)
ax.set_title('Medical Condition in different Admission Type')
ax.figure.get_axes()[0].legend(title="Medical Condition", loc="lower
right")
```

[44]:

```
<matplotlib.legend.Legend at 0x17324933850>
```



Research Analysis: from the above graph we can see that in EMERGENCY , cancer and hypertension has the highest rate of admission

16 12.Length of Stay:

Calculate the duration of hospital stays to identify any prolonged admissions or trends in discharge times.

[45]:
```
from datetime import date
```

[46]:
```
df["Date of Admission"]=pd.to_datetime(df['Date of Admission'])
df["Date of Admission"]=df["Date of Admission"].dt.day
```

[47]:
```
df["Discharge Date"]=pd.to_datetime(df['Discharge Date'])
df["Discharge Date"]=df["Discharge Date"].dt.day
```

[48]:
```
df["stay_length"]=(df["Discharge Date"]-df["Date of Admission"])
```

[49]:
```
df['stay_length'].describe()
```

[49]:
```
count    10000.000000
mean        14.591200
std          8.809827
min          0.000000
25%          7.000000
50%         14.000000
75%         22.000000
max         30.000000
Name: stay_length, dtype: float64
```

Research Analysis From the insight we can see that length of stays vary greatly, from 0 to 30 days.

17 13.Medication Usage:

Analyze the types and frequencies of medications prescribed to patients to monitor treatment patterns and effectiveness.

[50]:
```
pd.crosstab(index=df["Medication"], columns=df["Medical Condition"])
```

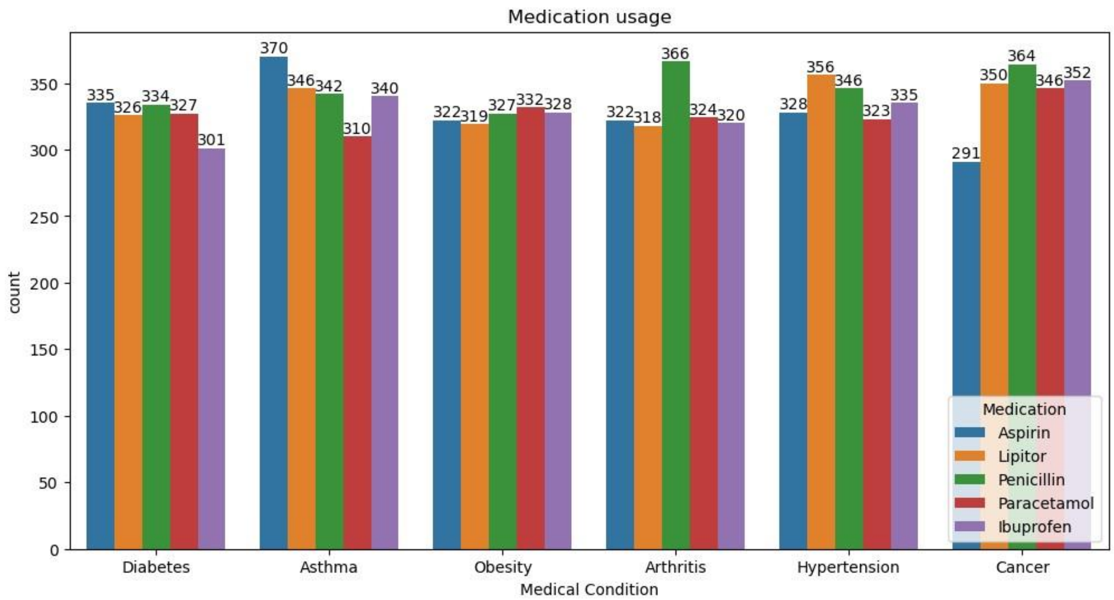
[50]:
```
Medical Condition Arthritis Asthma Cancer Diabetes Hypertension
Obesity
Medication
Aspirin              322    370    291     335        328     322
Ibuprofen            320    340    352     301        335     328
Lipitor              318    346    350     326        356     319
Paracetamol          324    310    346     327        323     332
Penicillin           366    342    364     334        346     327
```

[51]:
```
df["Medication"].value_counts()
```

[51]:
```
Medication

Penicillin    2079
Lipitor       2015
Ibuprofen     1976
Aspirin       1968
Paracetamol   1962
Name: count, dtype: int64
```

[52]:
```
plt.figure(figsize=(12,6))    ax=sns.countplot(data=df,
x="Medical  Condition",   hue="Medication")   for   bar   in
ax.containers:
    ax.bar_label(bar)
ax.figure.get_axes()[0].legend(title="Medication", loc="lower right")
ax.set_title("Medication usage")
```

[52]:
```
Text(0.5, 1.0, 'Medication usage')
```



Research Analysis: from the above graph we conclude that Penicillin medication is frequently prescribed.

18 14.Test Results Trends:

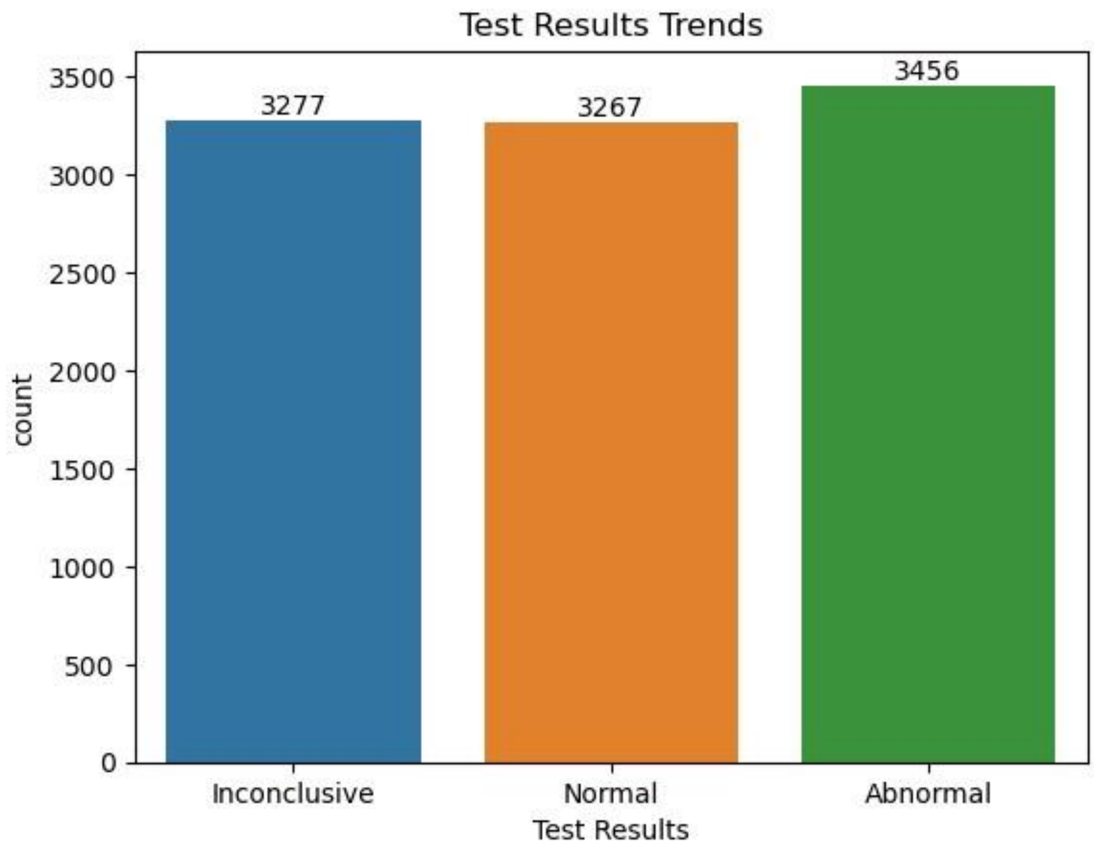
Identify any patterns or abnormalities in test results to improve diagnostic and treatment protocols.

[53]:
```
test=df["Test Results"].value_counts()
test
```

[53]:
```
Test Results
Abnormal        3456
Inconclusive    3277
Normal          3267
Name: count, dtype: int64
```

[54]:
```
ax=sns.countplot( x = df["Test Results"])
for bar in ax.containers:
    ax.bar_label(bar)
ax.set_title("Test Results Trends")
```

[54]:
```
Text(0.5, 1.0, 'Test Results Trends')
```



Research Analysis: From the above we can see that there is highest number of Abnormal test results.

19 15. Readmission Rates:

Track instances of readmission to assess the effectiveness of initial treatments and follow-up care

[55]:
```
df[["Name","Age","Gender","Blood Type"]].value_counts()
```

[55]:
```
Name              Age Gender Blood Type

Aaron Burnett     54  Female A-          1
Melanie Clark     46  Male   AB+         1
Meghan Jordan     52  Male   A-          1
Meghan Lee        56  Male   O+          1
Meghan Robinson   71  Male   B+          1
                                        ..
Gabriella Ware    61  Female O+          1
Gabrielle Francis 81 Male    A-          1
Gabrielle Mcclain 68 Male    A-          1
```

```
Gabrielle Russell 68  Male   AB+          1
Zoe Moore          74  Male   O-           1
Name: count, Length: 10000, dtype: int64
```

```
[56]: df.groupby(["Name"], as_index=False)["Date of Admission"].value_counts().
    ↪sort_values(by="Date of Admission", ascending=False)
```

```
[56]:                Name  Date of Admission  count
0         Aaron Burnett                  1      1
6263    Meghan Robinson                  1      1
6247     Megan Phillips                  1      2
6248       Megan Rogers                  1      1
6249        Megan Short                  1      1
…                   …                  …      …
3127  Frederick Sherman                  1      1
3128  Frederick Williams                 1      1
3129     Gabriel Flores                  1      1
3130  Gabriel Henderson                  1      1
9377          Zoe Moore                  1      1

[9378 rows x 3 columns]
```

```
[57]: df[['Name','Age',"Gender"]].duplicated().any()
```

```
[57]: True
```

```
[58]: df[df["Name"]== 'Megan Phillips']
```

```
[58]:             Name  Age  Gender Blood Type Medical Condition  \
6836  Megan Phillips   29    Male        AB-          Diabetes
9689  Megan Phillips   74  Female         A+      Hypertension

      Date of Admission          Doctor         Hospital Insurance Provider  \
6836                  1  Robert Gonzalez       Martin LLC UnitedHealthcare
9689                  1   Andrew Cannon Sanchez and Sons         Medicare

      Billing Amount  Room Number Admission Type  Discharge DateMedication  \
6836           11639          457         Urgent              25 Penicillin
9689           30105          192       Elective              26 Paracetamol

    Test Results  stay_length
```

```
6836       Normal         24
9689  Inconclusive        25
```

Research Analysis: From the above data we can see that there is duplicate data but there Age and Blood Type is different.
So we conclude that there no redamission.

20 16. Age and Medical Condition Correlation:

Investigate if certain medical conditions are more prevalent in specific age groups.

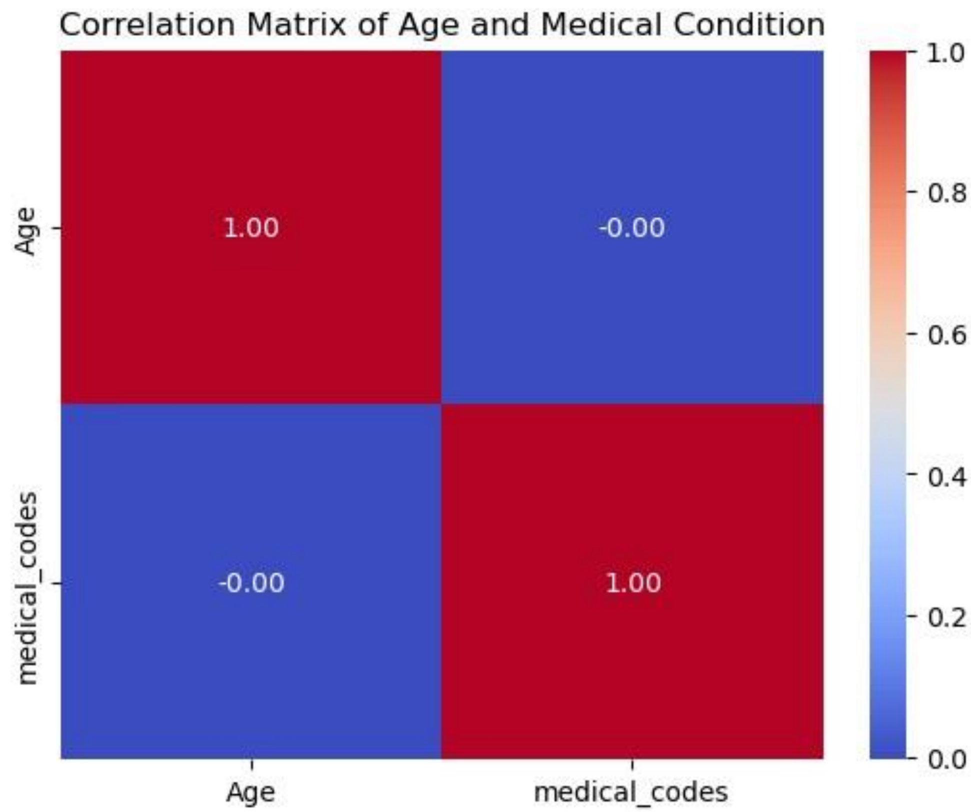
[59]:
```
# Convert to categorical and get codes
df['medical_codes'] = pd.Categorical(df['Medical
Condition']).codes df.head()
```

[59]:
```
                 Name Age Gender Blood Type Medical Condition \
0          Tiffany Ramirez 81 Female  O-    Diabetes
1          Ruben Burns      35    Male  O+    Asthma
2          Chad Byrd  61    Male  B-    Obesity
3          Antonio Frederick     49    Male  B-    Asthma
4          Mrs. Brandy Flowers   51    Male  O-    Arthritis

  Date of Admission        Doctor                Hospital \
0                 1 Patrick Parker Wallace-Hamilton
1                 1     Diane Jackson Burke, Griffin and Cooper
2                 1     Paul Baker Walton LLC
3                 1 Brian Chandler Garcia Ltd
4                 1 Dustin Griffin Jones, Brown and Murray

  Insurance Provider Billing Amount Room Number Admission Type \
0         Medicare          37490         146      Elective
1 UnitedHealthcare          47304         404     Emergency
2         Medicare          36874         292     Emergency
3         Medicare          23303         480        Urgent
4 UnitedHealthcare          18086         477        Urgent
  Discharge DateMedication Test Results stay_length medical_codes
0              1     Aspirin Inconclusive          0             3
1             15      Lipitor       Normal         14             1
2              8      Lipitor       Normal          7             5
3              3  Penicillin     Abnormal          2             1
4              2  Paracetamol       Normal          1             0
```

[65]:
```
                         data=df[["Age","medical_codes"]]
corr_matrix=data.corr()          sns.heatmap(corr_matrix,
annot=True,          cmap='coolwarm',          fmt='.2f')
```

```
plt.title("Correlation Matrix of Age and Medical
Condition")
```

[65]: Text(0.5, 1.0, 'Correlation Matrix of Age and Medical Condition')



Research Analysis The above correlation matrix show that there is no significant correlation between Age and Medical condition.

21 17. Gender and Medical Condition Correlation:

Determine if there are gender-based disparities in the prevalence or treatment outcomes of certain medical conditions.

[61]:
```
df['gender_codes'] = pd.Categorical(df['Gender']).codes
df.head()
```

[61]:
```
                Name  Age  Gender Blood Type Medical Condition  \
0    Tiffany Ramirez   81  Female         O-          Diabetes
1        Ruben Burns   35    Male         O+            Asthma
2          Chad Byrd   61    Male         B-           Obesity
3  Antonio Frederick   49    Male         B-            Asthma
```

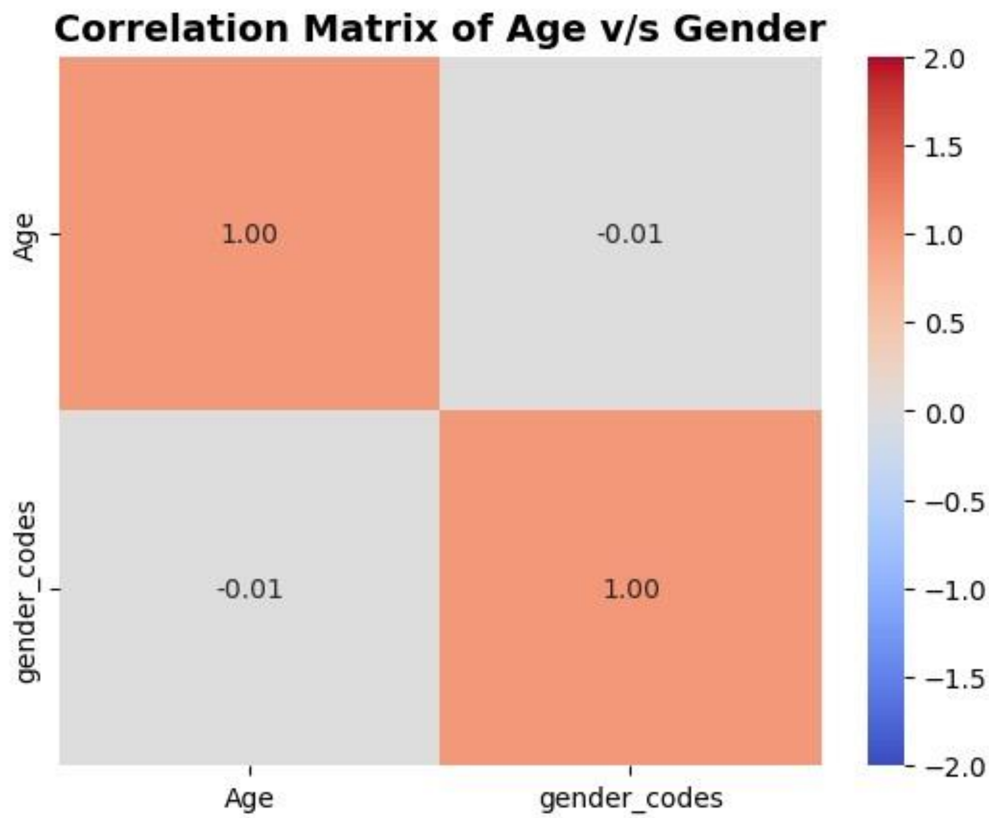
```
4 Mrs. Brandy Flowers 51    Male        O-        Arthritis

                                                  Hospital
   Date of Admission       Doctor                 \
0                 1 Patrick Parker        Wallace-Hamilton
1                 1     Diane Jackson Burke, Griffin and Cooper
2                 1     Paul Baker Walton LLC
3                 1 Brian Chandler Garcia Ltd
4                 1 Dustin Griffin Jones, Brown and Murray

Insurance Provider Billing Amount Room Number Admission Type
                              \
0          Medicare          37490        146      Elective
1 UnitedHealthcare          47304        404     Emergency
2          Medicare          36874        292     Emergency
3          Medicare          23303        480        Urgent
4 UnitedHealthcare          18086        477        Urgent
  Discharge DateMedication Test Results stay_length medical_codes \
0              1  Aspirin Inconclusive  0     3
1              15 Lipitor     Normal     14    1
2              8  Lipitor     Normal     7     5
3              3  Penicillin Abnormal    2     1

4              2  Paracetamol Normal     1     0

   gender_codes
0             0
1             1
2             1
3             1
4             1
```

[68]:

```python
data=df[["Age","gender_codes"]]
corr_matrix=data.corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt='.2f', vmin=-2,
 vmax=2)

plt.title("Correlation Matrix of Age v/s Gender ", fontsize=14,
 fontweight='bold')
```

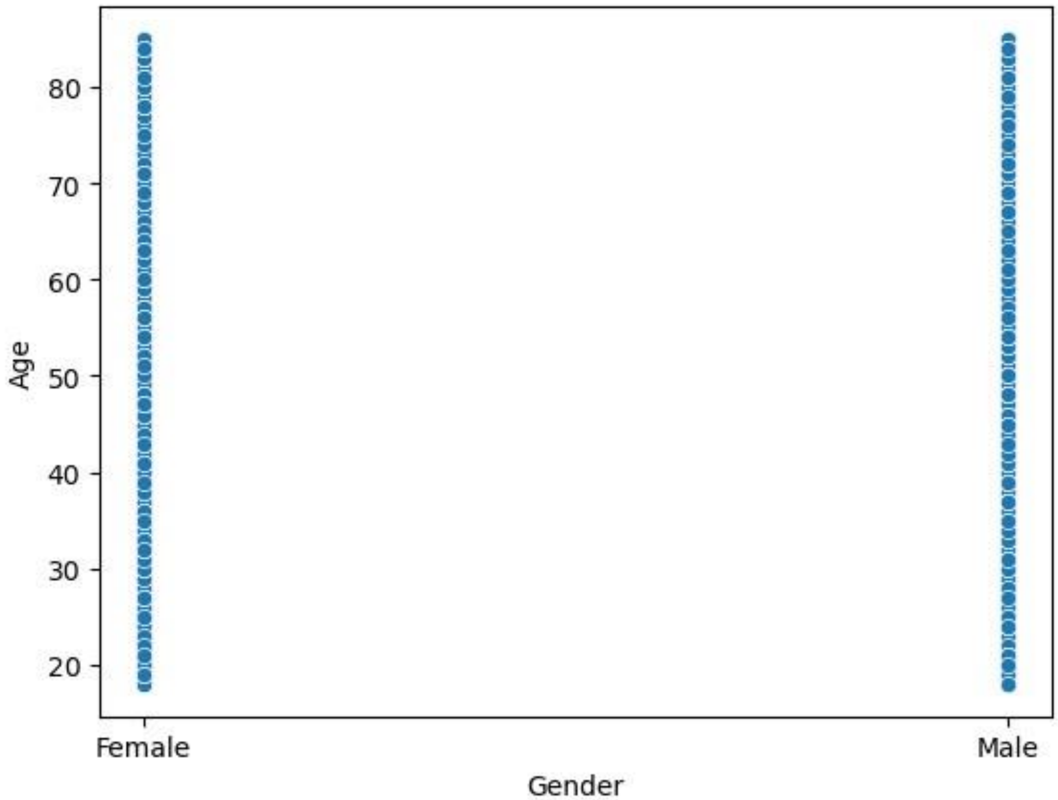
[68]:

```
Text(0.5, 1.0, 'Correlation Matrix of Age v/s Gender')
```



[79]:
```
sns.scatterplot(y='Age', x="Gender", data=df)
```

[79]: <Axes: xlabel='Gender', ylabel='Age'>



Research Analysis: In correlation matrix -0.01 indicates there is no correlation between Age and Gender.

22 18. Insurance Coverage and Billing Amount:

Analyze if there's any correlation between the patient's insurance provider and the billed amount for services.

[69]:
```
df['Insurance_codes'] = pd.Categorical(df['Insurance
Provider']).codes df.head()
```

[69]:
```
                  Name Age Gender Blood Type Medical Condition \
0     Tiffany Ramirez  81 Female         O-          Diabetes
1         Ruben Burns  35   Male         O+            Asthma
2           Chad Byrd  61   Male         B-           Obesity
3   Antonio Frederick  49   Male         B-            Asthma
4 Mrs. Brandy Flowers 51   Male         O-         Arthritis

                                                     Hospital
   Date of Admission       Doctor                   \
0                  1 Patrick Parker         Wallace-Hamilton
1                   1     Diane Jackson Burke, Griffin and Cooper
```

```
2                1     Paul Baker Walton LLC
3                1  Brian Chandler Garcia Ltd
4                1  Dustin Griffin Jones, Brown and Murray

Insurance Provider Billing Amount Room Number Admission Type
                                       \
0         Medicare        37490       146      Elective
1 UnitedHealthcare        47304       404     Emergency
2         Medicare        36874       292     Emergency
3         Medicare        23303       480        Urgent
4 UnitedHealthcare        18086       477        Urgent
  Discharge DateMedication Test Results stay_length medical_codes \
```

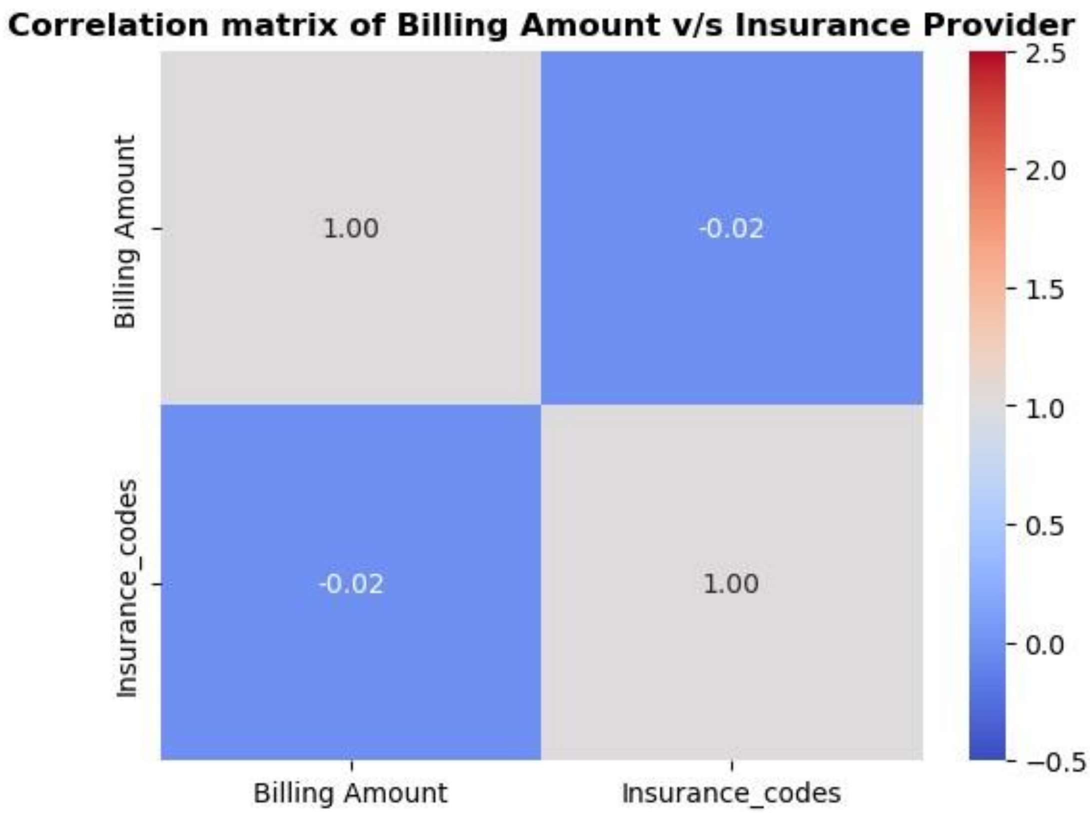
0	1	Aspirin	Inconclusive	0	3
1	15	Lipitor	Normal	14	1
2	8	Lipitor	Normal	7	5
3	3	Penicillin	Abnormal	2	1
4	2	Paracetamol	Normal	1	0

gender_codes Insurance_codes

0	0	3
1	1	4
2	1	3
3	1	3
4	1	4

[75]:
```
data=df[["Billing Amount", "Insurance_codes"]]
corr_matrix=data.corr() sns.heatmap(corr_matrix,annot=True,
fmt='.2f', cmap="coolwarm", vmin=-0.5,␣
↪vmax=2.5) plt.title("Correlation matrix of Billing Amount v/s
Insurance Provider",␣↪fontweight="bold")
```

[75]:
```
Text(0.5, 1.0, 'Correlation matrix of Billing Amount v/s Insurance
Provider')
```



Research Analysis: There is weak or almost no linear relationship between Billing amount and Insurance provider

23 Conclusion

A comprehensive analysis of various aspects of healthcare data, including demographics, medical conditions, billing, and insurance. The data suggests that females in the 50-60 age groups are the most admitted patients, with cancer being the predominant diagnosis, while males are often diagnosed with hypertension. Hence, promoting healthcare awareness among this age group, particularly for Cancer and Hypertension, is crucial.

Additionally, individuals with AB negative blood group should be educated about these diseases as they represent a significant portion of admitted patients.

Moreover, raising awareness about Asthma is essential since it's the most prevalent medical condition among all patients.

[ ]:

```

```