Health Insights Dashboard

June 16, 2024

1 Author : Harshal Devidas Baviskar

2 Project Name: Health Insights Dashboard

[1]:
```python
import pandas as pd
```

[2]:
```python
# Load the dataset
file_path = r"C:\Users\Lenovo\Downloads\archive (1).zip"
df = pd.read_csv(file_path)
```

[3]:
```python
# General processes
# View the first few rows of the DataFrame
print(df.head())
```

```
            Name  Age  Gender Blood Type Medical Condition Date of Admission  \
0  Bobby JacksOn   30    Male         B-            Cancer        2024-01-31
1   LesLie TErRy   62    Male         A+           Obesity        2019-08-20
2    DaNnY sMitH   76  Female         A-           Obesity        2022-09-22
3   andrEw waTtS   28  Female         O+          Diabetes        2020-11-18
4  adrIENNE bEll   43  Female        AB+            Cancer        2022-09-19

            Doctor                    Hospital Insurance Provider  \
0    Matthew Smith             Sons and Miller         Blue Cross
1  Samantha Davies                     Kim Inc           Medicare
2  Tiffany Mitchell                   Cook PLC              Aetna
3      Kevin Wells  Hernandez Rogers and Vang,           Medicare
4   Kathleen Hanna                 White-White              Aetna

   Billing Amount  Room Number Admission Type Discharge Date   Medication  \
0    18856.281306          328         Urgent     2024-02-02  Paracetamol
1    33643.327287          265      Emergency     2019-08-26    Ibuprofen
2    27955.096079          205      Emergency     2022-10-07      Aspirin
3    37909.782410          450       Elective     2020-12-18    Ibuprofen
4    14238.317814          458         Urgent     2022-10-09   Penicillin

   Test Results
0        Normal
1  Inconclusive
```

```
2        Normal
3      Abnormal
4      Abnormal
```

[4]:
```
# Summary statistics
print(df.describe())
```

```
                Age  Billing Amount   Room Number
count  55500.000000    55500.000000  55500.000000
mean      51.539459    25539.316097    301.134829
std       19.602454    14211.454431    115.243069
min       13.000000    -2008.492140    101.000000
25%       35.000000    13241.224652    202.000000
50%       52.000000    25538.069376    302.000000
75%       68.000000    37820.508436    401.000000
max       89.000000    52764.276736    500.000000
```

[5]:
```
# Check for missing values
print(df.isnull().sum())
```

```
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
```

[6]:
```
# Column information
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 55500 entries, 0 to 55499
Data columns (total 15 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Name                55500 non-null  object
 1   Age                 55500 non-null  int64
 2   Gender              55500 non-null  object
 3   Blood Type          55500 non-null  object
```

```
 4   Medical Condition   55500 non-null  object
 5   Date of Admission   55500 non-null  object
 6   Doctor              55500 non-null  object
 7   Hospital            55500 non-null  object
 8   Insurance Provider  55500 non-null  object
 9   Billing Amount      55500 non-null  float64
 10  Room Number         55500 non-null  int64
 11  Admission Type      55500 non-null  object
 12  Discharge Date      55500 non-null  object
 13  Medication          55500 non-null  object
 14  Test Results        55500 non-null  object
dtypes: float64(1), int64(2), object(12)
memory usage: 6.4+ MB
None
```

[7]:

```
# Unique values in 'Medical Condition' column
print(df['Medical Condition'].unique())
```

```
['Cancer' 'Obesity' 'Diabetes' 'Asthma' 'Hypertension' 'Arthritis']
```

[8]:

```
# Value counts for 'Medical Condition'
print(df['Medical Condition'].value_counts())
```

```
Medical Condition
Arthritis       9308
Diabetes        9304
Hypertension    9245
Obesity         9231
Cancer          9227
Asthma          9185
Name: count, dtype: int64
```

[9]:

```
# Filter data for patients with 'Diabetes'
diabetes_patients = df[df['Medical Condition'] == 'Diabetes']
print(diabetes_patients.head())
```

```
                 Name  Age  Gender Blood Type Medical Condition  \
3         andrEw waTtS   28  Female         O+          Diabetes
6        edwArD EDWaRDs   21  Female        AB-          Diabetes
12        connOR HANsEn   75  Female         A+          Diabetes
27   mr. KenNEth MoORE   34  Female         A+          Diabetes
34     NicOlE RodriGUEz   30  Female        AB+          Diabetes

   Date of Admission           Doctor                    Hospital  \
3         2020-11-18      Kevin Wells  Hernandez Rogers and Vang,
6         2020-11-03     Kelly Olson             Group Middleton
12        2019-12-12  Kenneth Fletcher   Powers Miller, and Flores
27        2022-06-21      James Ellis               Serrano-Dixon
34        2020-01-17       Lynn Young                   Poole Inc
```

```
   Insurance Provider  Billing Amount  Room Number Admission Type  \
3            Medicare    37909.782410          450       Elective
6            Medicare    19580.872345          389      Emergency
12              Cigna    43282.283358          134      Emergency
27   UnitedHealthcare    18834.801341          157      Emergency
34         Blue Cross     8408.949354          285      Emergency

   Discharge Date   Medication  Test Results
3      2020-12-18    Ibuprofen      Abnormal
6      2020-11-15  Paracetamol  Inconclusive
12     2019-12-28   Penicillin      Abnormal
27     2022-06-30      Lipitor      Abnormal
34     2020-02-10      Lipitor        Normal
```

[10]:

```python
# Sort the DataFrame by 'Age'
df_sorted = df.sort_values(by='Age')
print(df_sorted.head())
```

```
                   Name  Age  Gender Blood Type Medical Condition  \
50908      rOnalD daVis   13    Male         A+           Obesity
51900    jAcob wILLiAMs   13  Female         A+           Obesity
51873  doRothY hoffMAn   13  Female         O-            Cancer
50823    DEaNnA pALMeR   13    Male        AB-           Obesity
51833  JoNaTHAn JacksoN   13  Female         B-           Obesity

      Date of Admission           Doctor                     Hospital  \
50908        2023-12-06  Shannon Butler    Kelly, and Gomez Williams
51900        2020-03-03  Samantha Scott        and Huerta, Cox Price
51873        2020-07-18   Suzanne Jones       Davis Davis, and Davis
50823        2020-09-20  Barbara Butler  and Sanchez Phillips, Brown
51833        2020-02-07  Kevin Friedman              Spencer-Shields

      Insurance Provider  Billing Amount  Room Number Admission Type  \
50908              Cigna     3014.565852          241      Emergency
51900           Medicare    42349.109219          373         Urgent
51873              Cigna    22316.169323          379       Elective
50823           Medicare    23941.759486          163      Emergency
51833           Medicare    30075.230981          472         Urgent

      Discharge Date   Medication  Test Results
50908     2024-01-04  Paracetamol        Normal
51900     2020-03-20    Ibuprofen        Normal
51873     2020-08-10    Ibuprofen      Abnormal
50823     2020-09-23   Penicillin  Inconclusive
51833     2020-02-24      Aspirin  Inconclusive
```

[11]:
```
#Group by 'Medical Condition' and get mean 'Billing Amount'
df_grouped = df.groupby('Medical Condition')['Billing Amount'].mean()
print(df_grouped)
```

```
Medical Condition
Arthritis       25497.327056
Asthma          25635.249359
Cancer          25161.792707
Diabetes        25638.405577
Hypertension    25497.095761
Obesity         25805.971259
Name: Billing Amount, dtype: float64
```

[12]:
```
# Intermediate level processes
# Handle missing values
df['Billing Amount'].fillna(df['Billing Amount'].mean(), inplace=True)
df.dropna(subset=['Doctor'], inplace=True)
```

[13]:
```
# Convert data types
df['Date of Admission'] = pd.to_datetime(df['Date of Admission'])
df['Discharge Date'] = pd.to_datetime(df['Discharge Date'])
```

[14]:
```
# Create new columns
df['Length of Stay'] = (df['Discharge Date'] - df['Date of Admission']).dt.days
```

[15]:
```
# Apply functions to columns
df['Age Group'] = df['Age'].apply(lambda x: 'Senior' if x > 65 else 'Adult')
```

[16]:
```
# Merge DataFrames (example)
df1 = pd.DataFrame({'ID': [1, 2], 'Value': [10, 20]})
df2 = pd.DataFrame({'ID': [1, 2], 'Label': ['A', 'B']})
merged_df = pd.merge(df1, df2, on='ID')
print(merged_df)
```

```
   ID  Value Label
0   1     10     A
1   2     20     B
```

[17]:
```
# Create pivot table
pivot_df = df.pivot_table(values='Billing Amount', index='Hospital',␣
  ↪columns='Medical Condition', aggfunc='mean')
print(pivot_df)
```

```
Medical Condition                  Arthritis  Asthma  Cancer      Diabetes  \
Hospital
Abbott Inc                      38052.041917     NaN     NaN           NaN
Abbott Ltd                      29877.586483     NaN     NaN           NaN
Abbott Moore and Williams,               NaN     NaN     NaN           NaN
Abbott and Thompson, Sullivan            NaN     NaN     NaN           NaN
```

```
Abbott, Peters and Hoffman                 NaN      NaN      NaN  18842.396863
…                                        …     …      …           …
and Zimmerman Sons                         NaN      NaN      NaN           NaN
and Zuniga Davis Carlson,                  NaN      NaN      NaN  42867.041298
and Zuniga Francis Peterson,               NaN      NaN      NaN  33689.630726
and Zuniga Sons                            NaN      NaN      NaN           NaN
and Zuniga Thompson, Blake        22067.428763      NaN      NaN           NaN

Medical Condition               Hypertension       Obesity
Hospital
Abbott Inc                               NaN           NaN
Abbott Ltd                               NaN           NaN
Abbott Moore and Williams,               NaN  24799.596339
Abbott and Thompson, Sullivan   16738.569765           NaN
Abbott, Peters and Hoffman               NaN           NaN
…                                        …           …
and Zimmerman Sons                       NaN  32706.652625
and Zuniga Davis Carlson,                NaN           NaN
and Zuniga Francis Peterson,             NaN           NaN
and Zuniga Sons                 33950.170483           NaN
and Zuniga Thompson, Blake               NaN           NaN

[39876 rows x 6 columns]
```

[18]:
```
# Reshape data using melt
melted_df = df.melt(id_vars=['Name'], value_vars=['Age', 'Billing Amount'])
print(melted_df.head())
```

```
            Name variable  value
0   Bobby JacksOn      Age   30.0
1    LesLie TErRy      Age   62.0
2     DaNnY sMitH      Age   76.0
3    andrEw waTtS      Age   28.0
4   adrIENNE bEll      Age   43.0
```

[19]:
```
# Remove duplicate rows
df.drop_duplicates(subset=['Name', 'Date of Admission'], inplace=True)
```

[20]:
```
# Create conditional columns
df['High Billing'] = df['Billing Amount'].apply(lambda x: 'Yes' if x > 20000␣
 ↪else 'No')
```

[21]:
```
# Perform string operations
df['Name'] = df['Name'].str.upper()
df['Doctor'] = df['Doctor'].str.strip()
```

[22]:
```
# Print the cleaned and transformed DataFrame
print(df.head())
```

```
            Name  Age  Gender Blood Type Medical Condition Date of Admission  \
0  BOBBY JACKSON   30    Male         B-            Cancer        2024-01-31
1   LESLIE TERRY   62    Male         A+           Obesity        2019-08-20
2    DANNY SMITH   76  Female         A-           Obesity        2022-09-22
3   ANDREW WATTS   28  Female         O+          Diabetes        2020-11-18
4  ADRIENNE BELL   43  Female        AB+            Cancer        2022-09-19

            Doctor                    Hospital Insurance Provider  \
0    Matthew Smith             Sons and Miller         Blue Cross
1   Samantha Davies                    Kim Inc           Medicare
2  Tiffany Mitchell                   Cook PLC              Aetna
3       Kevin Wells  Hernandez Rogers and Vang,          Medicare
4    Kathleen Hanna                White-White              Aetna

   Billing Amount  Room Number Admission Type Discharge Date   Medication  \
0    18856.281306          328         Urgent     2024-02-02  Paracetamol
1    33643.327287          265      Emergency     2019-08-26    Ibuprofen
2    27955.096079          205      Emergency     2022-10-07      Aspirin
3    37909.782410          450       Elective     2020-12-18    Ibuprofen
4    14238.317814          458         Urgent     2022-10-09   Penicillin

   Test Results  Length of Stay Age Group High Billing
0        Normal               2     Adult           No
1  Inconclusive               6     Adult          Yes
2        Normal              15    Senior          Yes
3      Abnormal              30     Adult          Yes
4      Abnormal              20     Adult           No
```

[23]:
```
# Save the cleaned DataFrame for use in visualizations
df.to_csv('cleaned_healthcare_dataset_2.csv', index=False)
```

3 Visualizations

[24]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

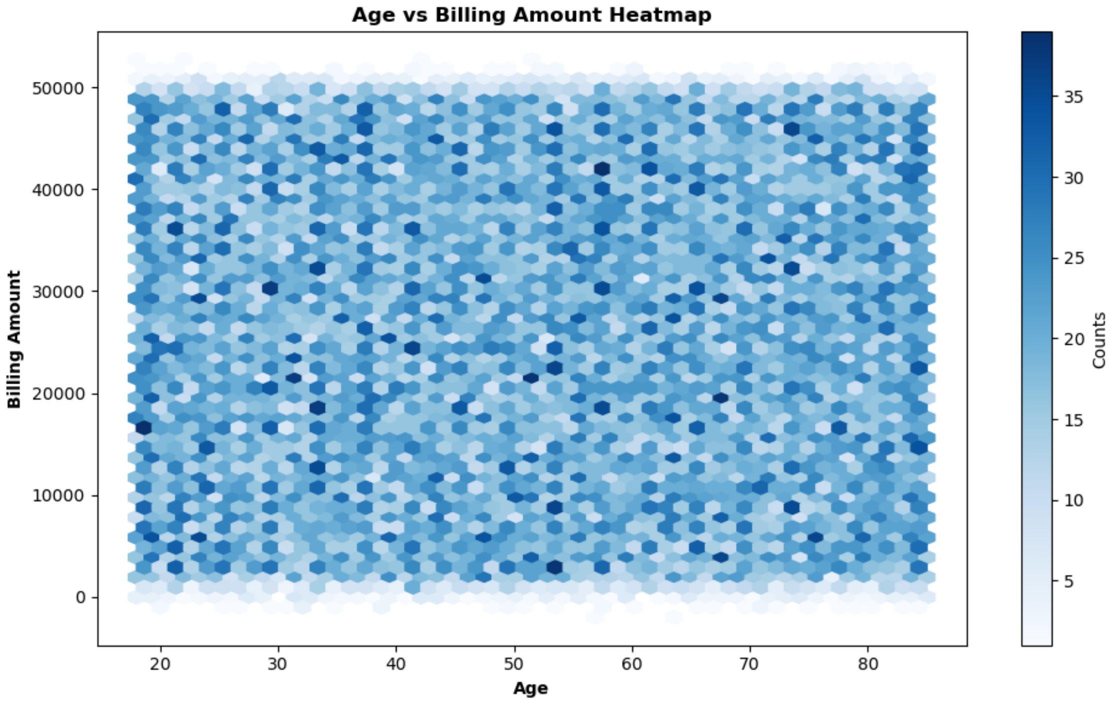
[25]:
```
# Load the cleaned dataset
file_path = 'cleaned_healthcare_dataset_2.csv'
df = pd.read_csv(file_path)
```

[26]:
```
# Convert date columns to datetime if they are not already
df['Date of Admission'] = pd.to_datetime(df['Date of Admission'])
df['Discharge Date'] = pd.to_datetime(df['Discharge Date'])
```

[27]:
```
# Calculate length of stay
df['Length of Stay'] = (df['Discharge Date'] - df['Date of Admission']).dt.days
```

4 Relationship Visualizations

[28]:
```
# 1. How does age correlate with billing amount?
# Create a heatmap to show the density of Age vs Billing Amount
plt.figure(figsize=(10, 6))
heatmap_data = df[['Age', 'Billing Amount']]

# Create a 2D histogram (heatmap)
plt.hexbin(heatmap_data['Age'], heatmap_data['Billing Amount'], gridsize=50,␣
↪cmap='Blues', mincnt=1)
plt.colorbar(label='Counts')
plt.title('Age vs Billing Amount Heatmap',fontweight='bold')
plt.xlabel('Age',fontweight='bold')
plt.ylabel('Billing Amount',fontweight='bold')

# Insights
insights = """
Insights for Hospitals:
1. Younger patients tend to have lower billing amounts, possibly due to fewer␣
↪chronic conditions.
2. Billing amounts increase with age, highlighting the need for targeted cost␣
↪management for older patients.

Insights for Patients:
1. Younger patients might incur lower healthcare costs, indicating potentially␣
↪fewer health issues.
2. Older patients should be prepared for higher medical expenses and plan their␣
↪finances accordingly.
"""

plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center',␣
↪fontsize=10)
plt.tight_layout()
plt.show()
```



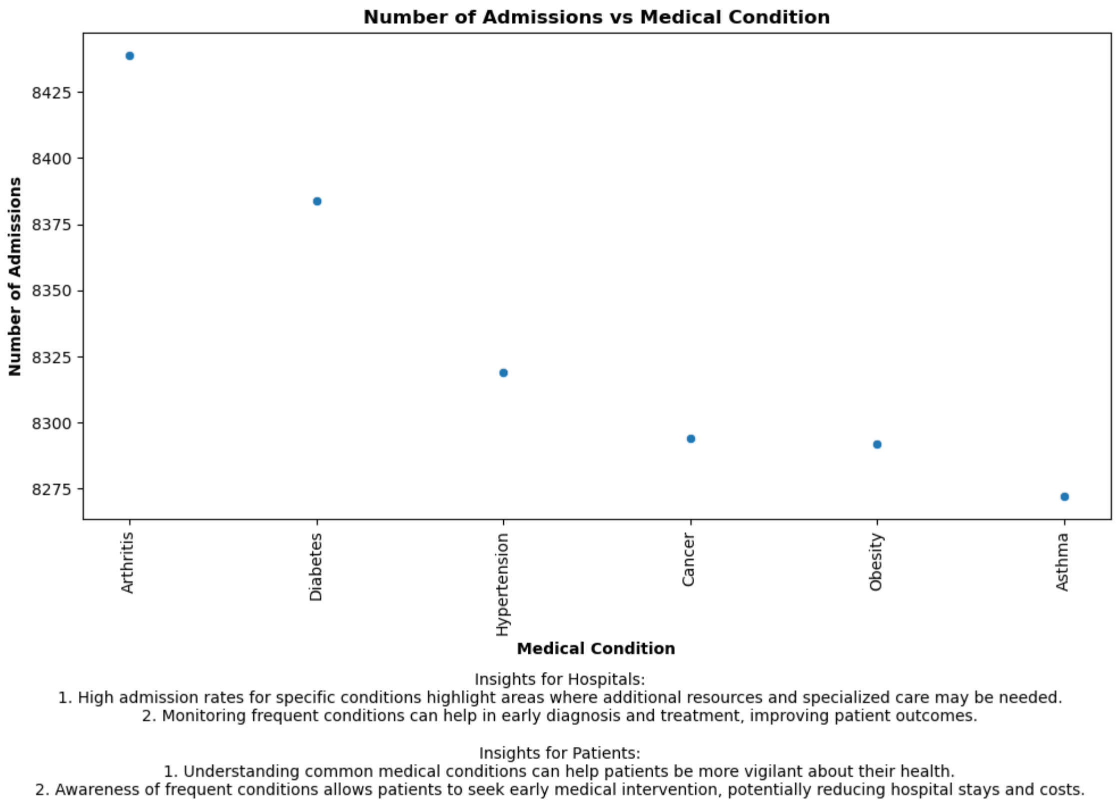
Insights for Hospitals:
1. Younger patients tend to have lower billing amounts, possibly due to fewer chronic conditions.
2. Billing amounts increase with age, highlighting the need for targeted cost management for older patients.

Insights for Patients:
1. Younger patients might incur lower healthcare costs, indicating potentially fewer health issues.
2. Older patients should be prepared for higher medical expenses and plan their finances accordingly.

[29]:

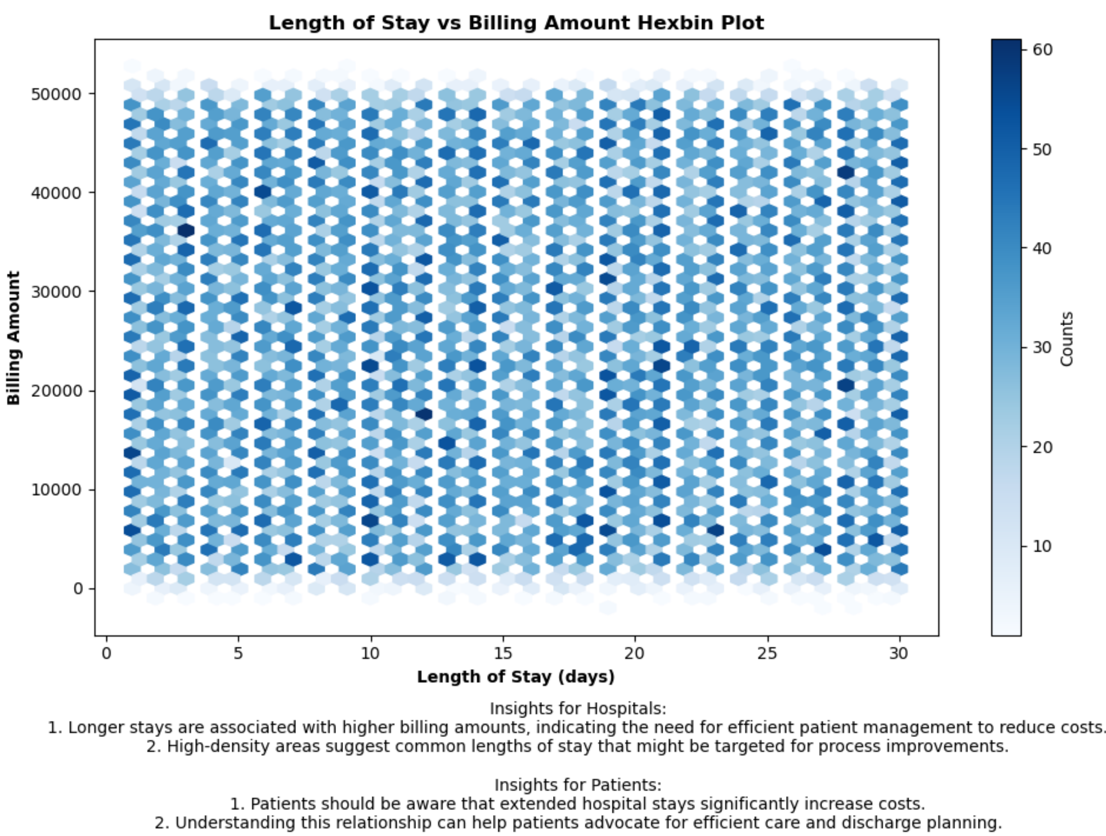
```
# 2. What is the relationship between the number of admissions and the medical␣
↪condition?
admissions_per_condition = df['Medical Condition'].value_counts()
plt.figure(figsize=(10, 6))
sns.scatterplot(x=admissions_per_condition.index, y=admissions_per_condition.
↪values)
plt.title('Number of Admissions vs Medical Condition',fontweight='bold')
plt.xlabel('Medical Condition',fontweight='bold')
plt.ylabel('Number of Admissions',fontweight='bold')
plt.xticks(rotation=90)

# Insights
insights = """
Insights for Hospitals:
1. High admission rates for specific conditions highlight areas where␣
↪additional resources and specialized care may be needed.
2. Monitoring frequent conditions can help in early diagnosis and treatment,␣
↪improving patient outcomes.
```

```
Insights for Patients:
1. Understanding common medical conditions can help patients be more vigilant␣
  ↪about their health.
2. Awareness of frequent conditions allows patients to seek early medical␣
  ↪intervention, potentially reducing hospital stays and costs.
"""

plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center',␣
  ↪fontsize=10)
plt.tight_layout()
plt.show()
```

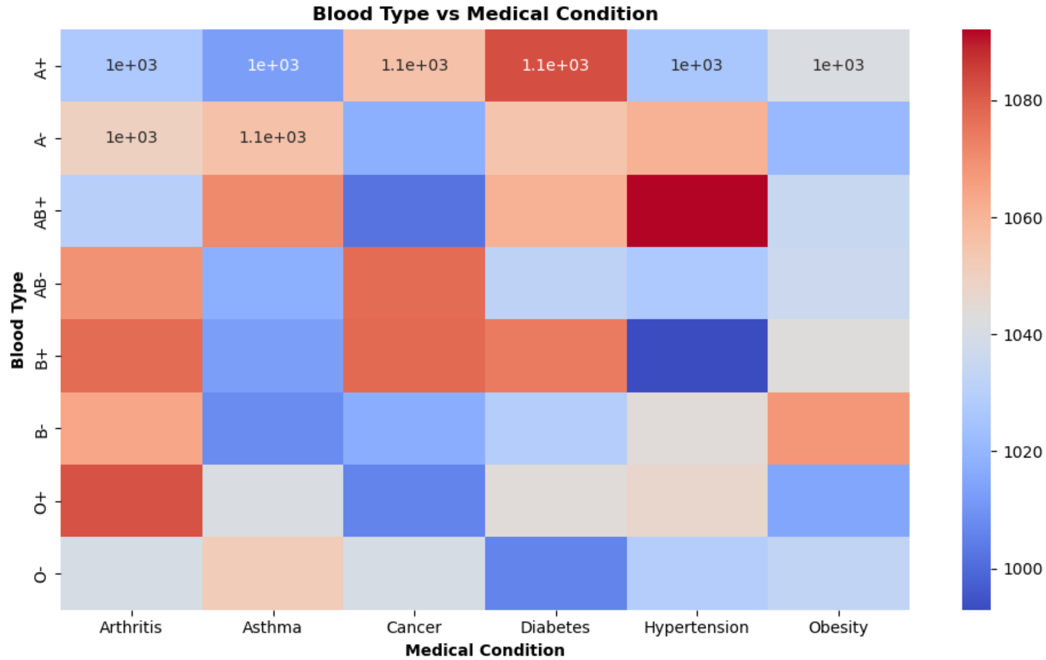


[30]:
```
# 3. How does the length of stay correlate with billing amount?
plt.figure(figsize=(10, 6))
plt.hexbin(df['Length of Stay'], df['Billing Amount'], gridsize=50,␣
  ↪cmap='Blues', mincnt=1)
plt.colorbar(label='Counts')
plt.title('Length of Stay vs Billing Amount Hexbin Plot',fontweight='bold')
plt.xlabel('Length of Stay (days)',fontweight='bold')
plt.ylabel('Billing Amount',fontweight='bold')
```

```python
# Insights
insights = """
Insights for Hospitals:
1. Longer stays are associated with higher billing amounts, indicating the need for efficient patient management to reduce costs.
2. High-density areas suggest common lengths of stay that might be targeted for process improvements.

Insights for Patients:
1. Patients should be aware that extended hospital stays significantly increase costs.
2. Understanding this relationship can help patients advocate for efficient care and discharge planning.
"""

plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center', fontsize=10)
plt.tight_layout()
plt.show()
```



[31]:
```
# 4. Is there a relationship between blood type and medical conditions?
blood_type_condition = pd.crosstab(df['Blood Type'], df['Medical Condition'])
plt.figure(figsize=(10, 6))
sns.heatmap(blood_type_condition, annot=True, cmap='coolwarm')
plt.title('Blood Type vs Medical Condition',fontweight='bold')
plt.xlabel('Medical Condition',fontweight='bold')
plt.ylabel('Blood Type',fontweight='bold')

# Insights
insights = """
Insights for Hospitals:
1. Specific medical conditions may be more prevalent among certain blood types,␣
↪aiding in targeted screening and prevention programs.
2. Understanding these patterns can help in resource allocation and␣
↪personalized patient care.

Insights for Patients:
1. Patients with certain blood types can be more vigilant about conditions␣
↪prevalent in their group.
2. Awareness of these trends can encourage regular check-ups and proactive␣
↪health management.
"""

plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center',␣
↪fontsize=10)
plt.tight_layout()
plt.show()
```

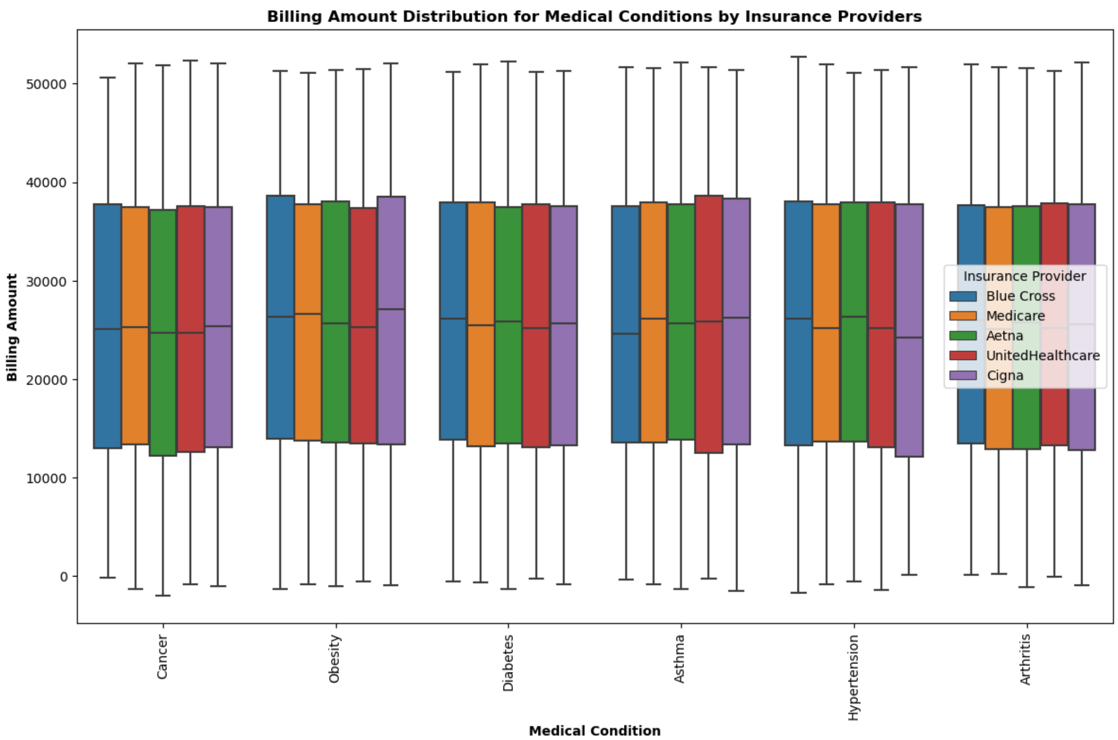


Insights for Hospitals:
1. Specific medical conditions may be more prevalent among certain blood types, aiding in targeted screening and prevention programs.
2. Understanding these patterns can help in resource allocation and personalized patient care.

Insights for Patients:
1. Patients with certain blood types can be more vigilant about conditions prevalent in their group.
2. Awareness of these trends can encourage regular check-ups and proactive health management.

[32]:
```
# 5. How do different insurance providers affect the billing amounts for␣
↪different medical conditions?
plt.figure(figsize=(12, 8))
sns.boxplot(x='Medical Condition', y='Billing Amount', hue='Insurance␣
↪Provider', data=df)
plt.title('Billing Amount Distribution for Medical Conditions by Insurance␣
↪Providers',fontweight='bold')
plt.xlabel('Medical Condition',fontweight='bold')
plt.ylabel('Billing Amount',fontweight='bold')
plt.xticks(rotation=90)
plt.legend(title='Insurance Provider')

# Insights
insights = """
Insights for Hospitals:
1. Variation in billing amounts across insurance providers for the same␣
↪condition highlights potential discrepancies in coverage and negotiated␣
↪rates.
2. Hospitals can use this data to negotiate better rates with insurance␣
↪providers and ensure fair pricing for all patients.
```

```
Insights for Patients:
1. Patients should be aware that their insurance provider can significantly affect their billing amount for the same medical condition.
2. Choosing an insurance provider with better coverage and negotiated rates can lead to lower out-of-pocket expenses.
"""

plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center', fontsize=10)
plt.tight_layout()
plt.show()
```

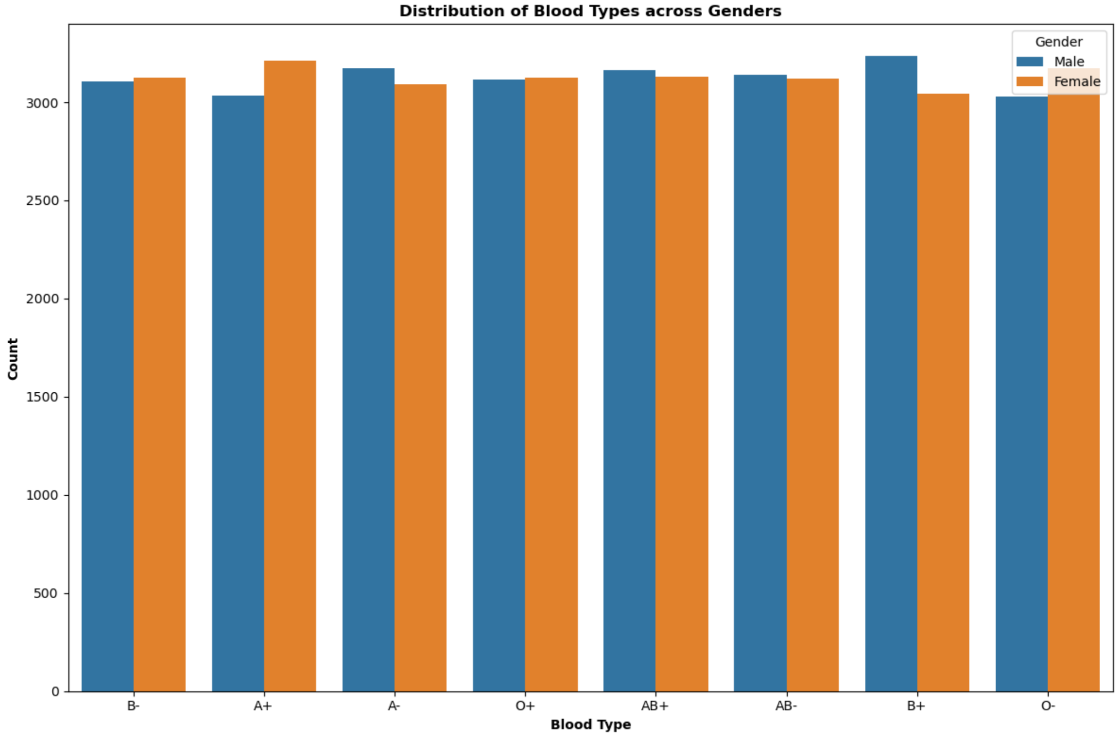


Insights for Hospitals:
1. Variation in billing amounts across insurance providers for the same condition highlights potential discrepancies in coverage and negotiated rates.
2. Hospitals can use this data to negotiate better rates with insurance providers and ensure fair pricing for all patients.

Insights for Patients:
1. Patients should be aware that their insurance provider can significantly affect their billing amount for the same medical condition.
2. Choosing an insurance provider with better coverage and negotiated rates can lead to lower out-of-pocket expenses.

5 Comparison Visualizations

[33]:
```
# 1. What is the distribution of different blood types across genders?
plt.figure(figsize=(12, 8))
sns.countplot(x='Blood Type', hue='Gender', data=df)
plt.title('Distribution of Blood Types across Genders',fontweight='bold')
plt.xlabel('Blood Type',fontweight='bold')
plt.ylabel('Count',fontweight='bold')

# Insights
insights = """
Insights for Hospitals:
1. Understanding the distribution of blood types across genders helps in␣
↪effective blood bank management and preparedness for transfusions.
2. Hospitals can ensure a balanced supply of different blood types to meet the␣
↪demand for both genders.

Insights for Patients:
1. Awareness of blood type distribution can help patients understand the␣
↪availability of their blood type for emergencies.
2. Patients can consider donating blood to maintain an adequate supply of rare␣
↪blood types.
"""

plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center',␣
↪fontsize=10)
plt.tight_layout()
plt.show()
```

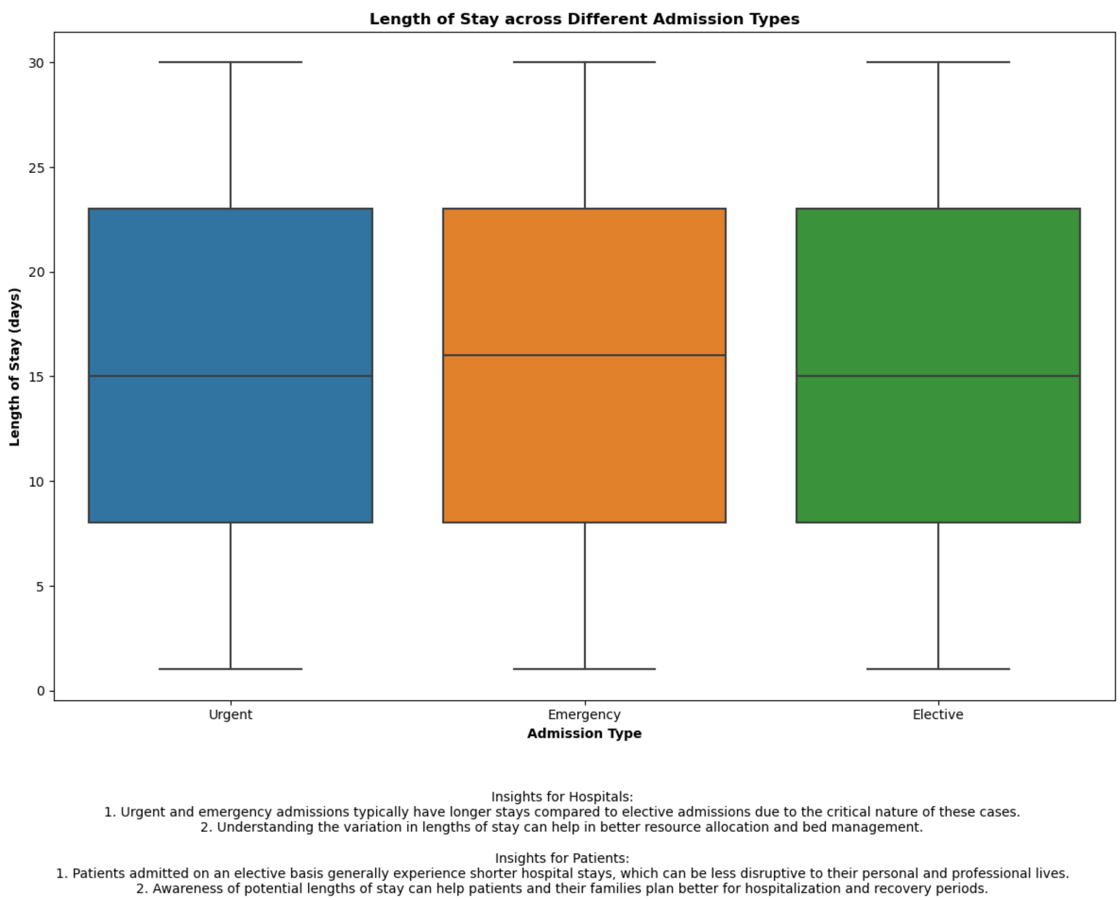


Insights for Hospitals:
1. Understanding the distribution of blood types across genders helps in effective blood bank management and preparedness for transfusions.
2. Hospitals can ensure a balanced supply of different blood types to meet the demand for both genders.

Insights for Patients:
1. Awareness of blood type distribution can help patients understand the availability of their blood type for emergencies.
2. Patients can consider donating blood to maintain an adequate supply of rare blood types.

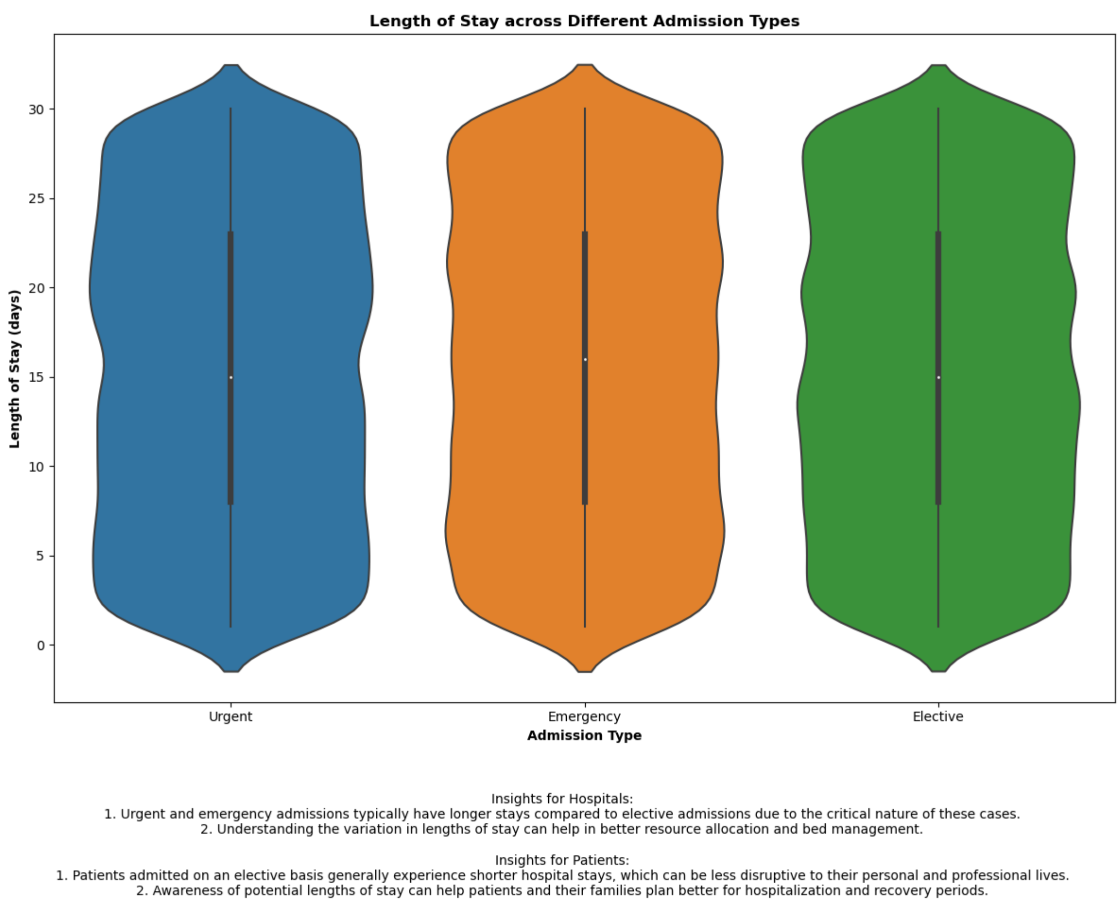
[34]:
```
# 2. How do the lengths of stay differ across different admission types (e.g.,␣
↪urgent, emergency, elective)?
plt.figure(figsize=(12, 8))
sns.boxplot(x='Admission Type', y='Length of Stay', data=df)
plt.title('Length of Stay across Different Admission Types',fontweight='bold')
plt.xlabel('Admission Type',fontweight='bold')
plt.ylabel('Length of Stay (days)',fontweight='bold')


# Insights
insights = """
Insights for Hospitals:
1. Urgent and emergency admissions typically have longer stays compared to␣
↪elective admissions due to the critical nature of these cases.
2. Understanding the variation in lengths of stay can help in better resource␣
↪allocation and bed management.

Insights for Patients:
```

```
1. Patients admitted on an elective basis generally experience shorter hospital stays, which can be less disruptive to their personal and professional lives.
2. Awareness of potential lengths of stay can help patients and their families plan better for hospitalization and recovery periods.
"""


plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center', fontsize=10)
plt.tight_layout()
plt.show()
```



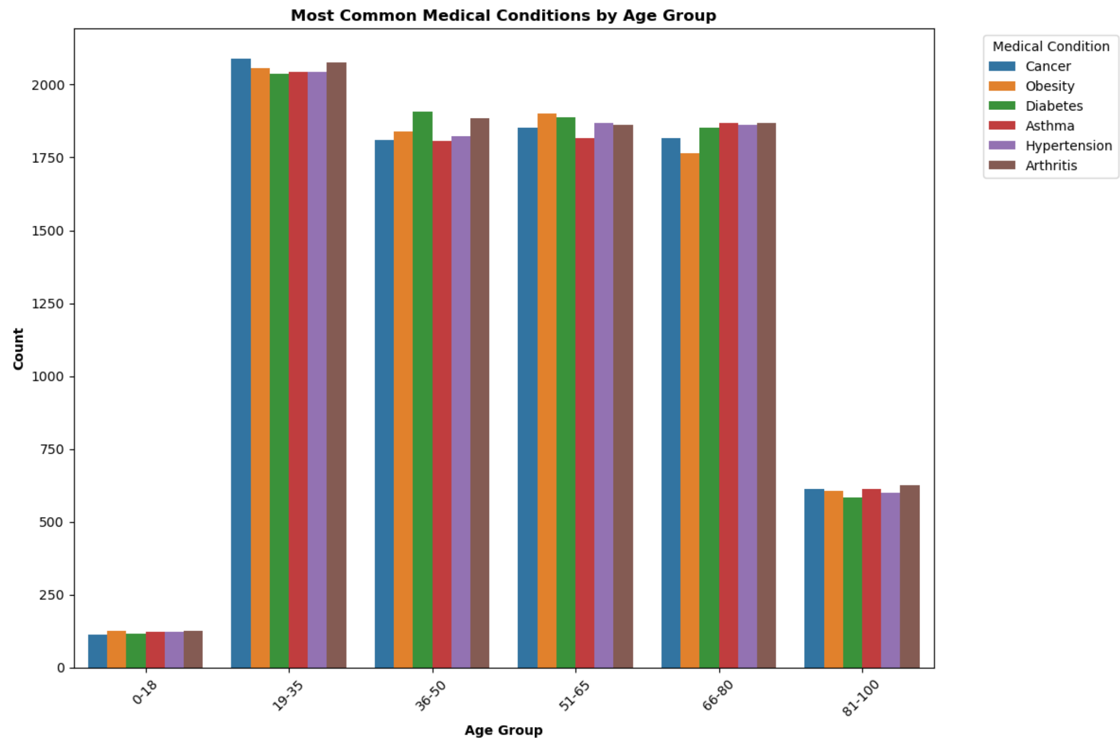
[35]:
```
# Alternative visualization using violin plot
plt.figure(figsize=(12, 8))
sns.violinplot(x='Admission Type', y='Length of Stay', data=df)
plt.title('Length of Stay across Different Admission Types',fontweight='bold')
plt.xlabel('Admission Type',fontweight='bold')
plt.ylabel('Length of Stay (days)',fontweight='bold')
```

```python
# Insights
insights = """
Insights for Hospitals:
1. Urgent and emergency admissions typically have longer stays compared to elective admissions due to the critical nature of these cases.
2. Understanding the variation in lengths of stay can help in better resource allocation and bed management.

Insights for Patients:
1. Patients admitted on an elective basis generally experience shorter hospital stays, which can be less disruptive to their personal and professional lives.
2. Awareness of potential lengths of stay can help patients and their families plan better for hospitalization and recovery periods.
"""


plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center', fontsize=10)
plt.tight_layout()
plt.show()
```



[36]:
```python
# 3. What are the most common medical conditions by age group?
age_bins = [0, 18, 35, 50, 65, 80, 100]
age_labels = ['0-18', '19-35', '36-50', '51-65', '66-80', '81-100']
df['Age Group'] = pd.cut(df['Age'], bins=age_bins, labels=age_labels)

plt.figure(figsize=(12, 8))
sns.countplot(x='Age Group', hue='Medical Condition', data=df)
plt.title('Most Common Medical Conditions by Age Group',fontweight='bold')
plt.xlabel('Age Group',fontweight='bold')
plt.ylabel('Count',fontweight='bold')
plt.xticks(rotation=45)
plt.legend(title='Medical Condition', bbox_to_anchor=(1.05, 1), loc='upper␣
↪left')

# Insights
insights = """
Insights for Hospitals:
1. Understanding the prevalence of medical conditions across age groups helps␣
↪in planning specialized care and resource allocation.
2. Hospitals can design targeted prevention and treatment programs based on the␣
↪most common conditions in each age group.

Insights for Patients:
1. Awareness of common medical conditions in their age group can help␣
↪individuals take preventive measures.
2. Patients can engage in age-appropriate health screenings and lifestyle␣
↪adjustments to mitigate risks associated with prevalent conditions.
"""

plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center',␣
↪fontsize=10)
plt.tight_layout()
plt.show()
```

```
D:\anaconda\Lib\site-packages\seaborn\categorical.py:641: FutureWarning: The
default of observed=False is deprecated and will be changed to True in a future
version of pandas. Pass observed=False to retain current behavior or
observed=True to adopt the future default and silence this warning.
  grouped_vals = vals.groupby(grouper)
D:\anaconda\Lib\site-packages\seaborn\categorical.py:641: FutureWarning: The
default of observed=False is deprecated and will be changed to True in a future
version of pandas. Pass observed=False to retain current behavior or
observed=True to adopt the future default and silence this warning.
  grouped_vals = vals.groupby(grouper)
```



Insights for Hospitals:
1. Understanding the prevalence of medical conditions across age groups helps in planning specialized care and resource allocation.
2. Hospitals can design targeted prevention and treatment programs based on the most common conditions in each age group.

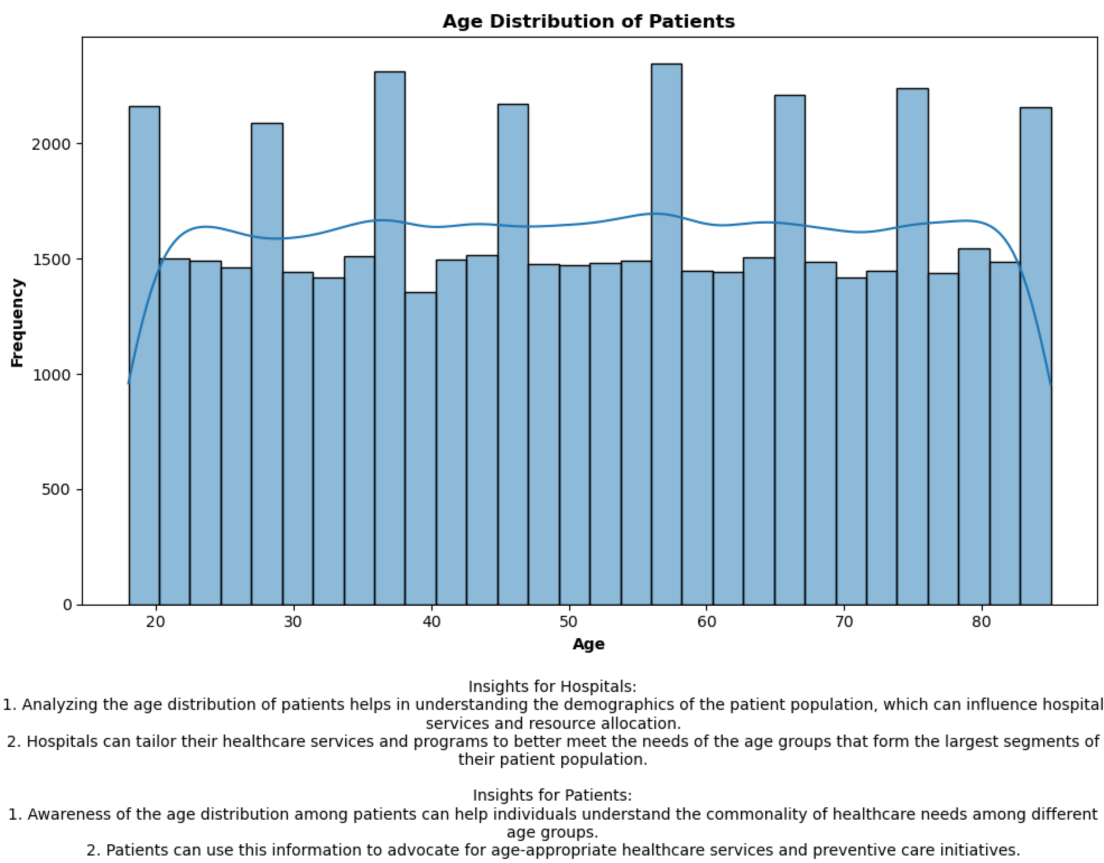
Insights for Patients:
1. Awareness of common medical conditions in their age group can help individuals take preventive measures.
2. Patients can engage in age-appropriate health screenings and lifestyle adjustments to mitigate risks associated with prevalent conditions.

6 Distribution Visualizations

[37]:
```
# 1. What is the age distribution of patients in the dataset?
plt.figure(figsize=(10, 6))
sns.histplot(df['Age'], bins=30, kde=True)
plt.title('Age Distribution of Patients',fontweight='bold')
plt.xlabel('Age',fontweight='bold')
plt.ylabel('Frequency',fontweight='bold')

# Insights specific to Age Distribution of Patients
insights = """
Insights for Hospitals:
1. Analyzing the age distribution of patients helps in understanding the demographics of the patient population, which can influence hospital services and resource allocation.
2. Hospitals can tailor their healthcare services and programs to better meet the needs of the age groups that form the largest segments of their patient population.
```

```
Insights for Patients:
1. Awareness of the age distribution among patients can help individuals␣
 ↪understand the commonality of healthcare needs among different age groups.
2. Patients can use this information to advocate for age-appropriate healthcare␣
 ↪services and preventive care initiatives.
"""

plt.figtext(0.5, -0.3, insights, wrap=True, horizontalalignment='center',␣
 ↪fontsize=10)
plt.tight_layout()
plt.show()
```

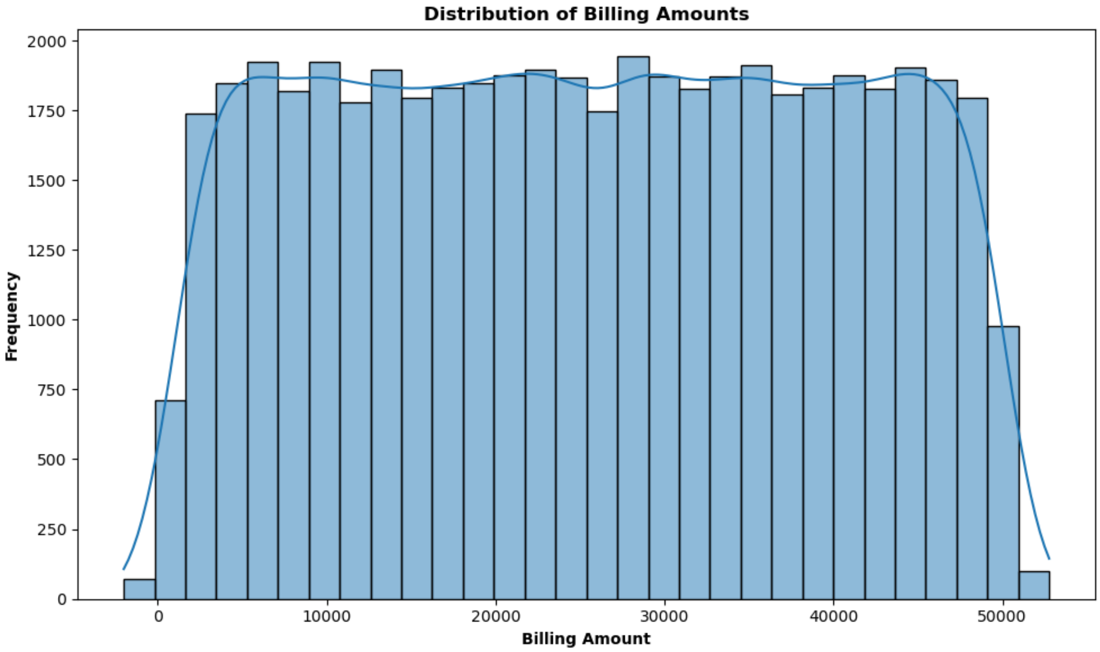
```
D:\anaconda\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning:
use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



[38]:

```python
# 2. What is the distribution of billing amounts?
plt.figure(figsize=(10, 6))
sns.histplot(df['Billing Amount'], bins=30, kde=True)
plt.title('Distribution of Billing Amounts',fontweight='bold')
plt.xlabel('Billing Amount',fontweight='bold')
plt.ylabel('Frequency',fontweight='bold')


# Insights
insights = """
Insights for Hospitals:
1. Understanding the distribution of billing amounts can help in identifying␣
↪common cost ranges for different treatments and services.
2. Hospitals can use this data to evaluate their pricing strategies and ensure␣
↪they are competitive while covering costs effectively.
3. Analyzing billing amounts can also help in detecting any anomalies or␣
↪outliers that may indicate billing errors or fraudulent activities.

Insights for Patients:
1. Awareness of the common billing amounts can help patients anticipate and␣
↪plan for potential healthcare expenses.
2. Patients can use this information to compare costs across different␣
↪healthcare providers and make informed decisions about their care.
3. Understanding the distribution of billing amounts can also empower patients␣
↪to ask relevant questions about the costs of their treatments and services.
"""

plt.figtext(0.5, -0.3, insights, wrap=True, horizontalalignment='center',␣
↪fontsize=10)
plt.tight_layout()
plt.show()
```

```
D:\anaconda\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning:
use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

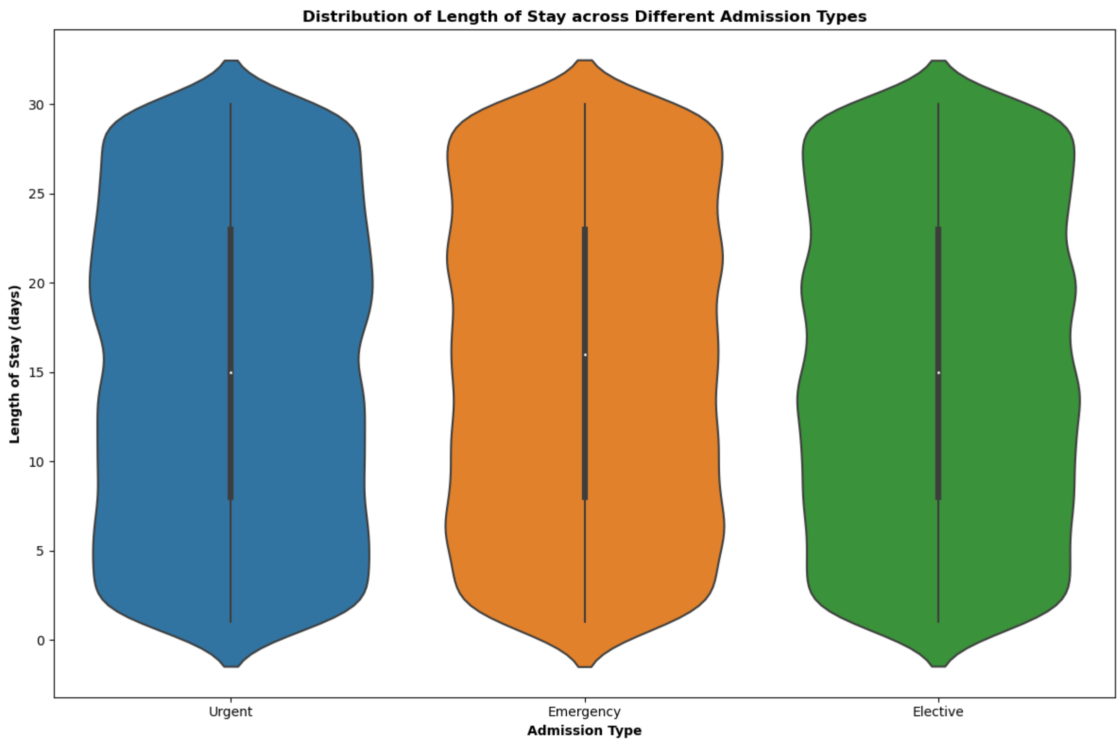


Insights for Hospitals:
1. Understanding the distribution of billing amounts can help in identifying common cost ranges for different treatments and services.
2. Hospitals can use this data to evaluate their pricing strategies and ensure they are competitive while covering costs effectively.
3. Analyzing billing amounts can also help in detecting any anomalies or outliers that may indicate billing errors or fraudulent activities.

Insights for Patients:
1. Awareness of the common billing amounts can help patients anticipate and plan for potential healthcare expenses.
2. Patients can use this information to compare costs across different healthcare providers and make informed decisions about their care.
3. Understanding the distribution of billing amounts can also empower patients to ask relevant questions about the costs of their treatments and services.

[39]:
```
# 3. How is the distribution of length of stay across different admission types?
plt.figure(figsize=(12, 8))
sns.violinplot(x='Admission Type', y='Length of Stay', data=df)
plt.title('Distribution of Length of Stay across Different Admission␣
↪Types',fontweight='bold')
plt.xlabel('Admission Type',fontweight='bold')
plt.ylabel('Length of Stay (days)',fontweight='bold')

insights = """
Insights for Hospitals:
1. The length of stay varies significantly across different admission types,␣
↪with emergency and urgent admissions typically resulting in longer hospital␣
↪stays due to the severity of conditions.
2. Elective admissions tend to have shorter stays as these are planned␣
↪procedures and patients are often in better health beforehand.
3. Understanding these patterns helps in optimizing bed management and resource␣
↪allocation to ensure efficient patient flow and hospital operations.

Insights for Patients:
```

```
1. Patients admitted for elective procedures can expect shorter hospital stays,␣
↪which can help in better planning for post-discharge care and recovery.
2. Awareness of the potential length of stay based on admission type can help␣
↪patients and their families prepare for the duration of
   hospitalization and associated costs."""

plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center',␣
↪fontsize=10)
plt.tight_layout()
plt.show()
```

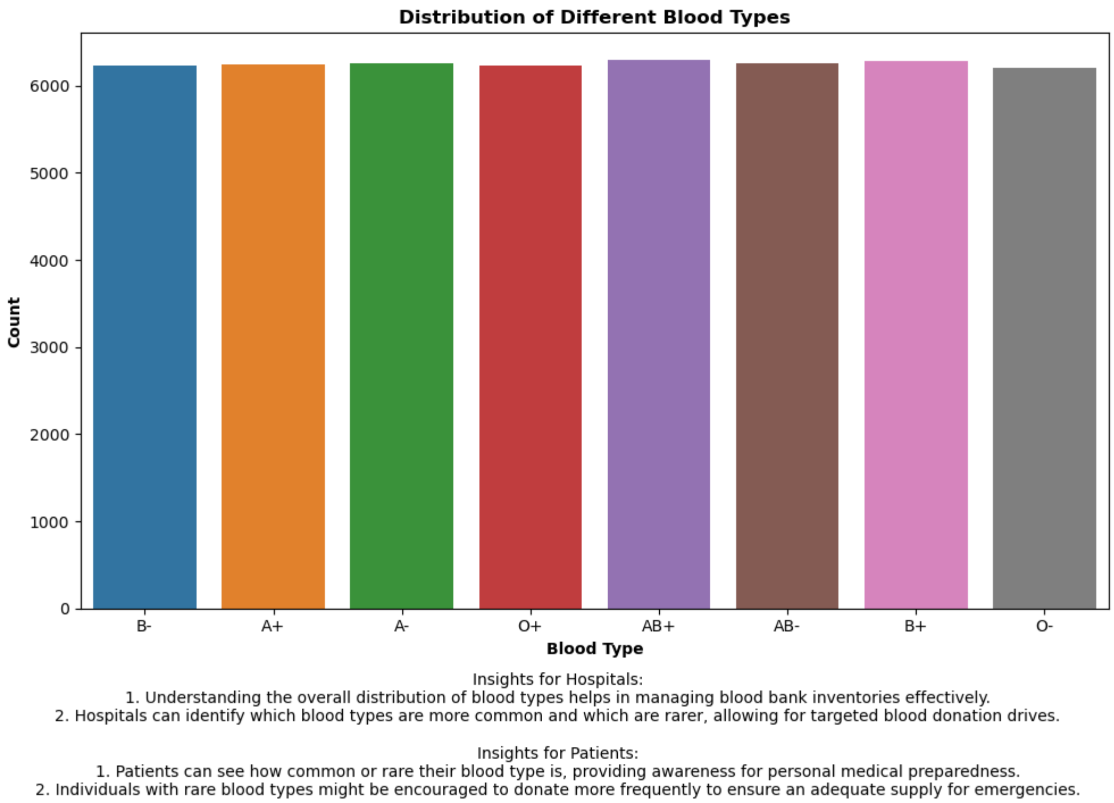


Insights for Hospitals:
1. The length of stay varies significantly across different admission types, with emergency and urgent admissions typically resulting in longer hospital stays due to the severity of conditions.
2. Elective admissions tend to have shorter stays as these are planned procedures and patients are often in better health beforehand.
3. Understanding these patterns helps in optimizing bed management and resource allocation to ensure efficient patient flow and hospital operations.

Insights for Patients:
1. Patients admitted for elective procedures can expect shorter hospital stays, which can help in better planning for post-discharge care and recovery.
2. Awareness of the potential length of stay based on admission type can help patients and their families prepare for the duration of hospitalization and associated costs.

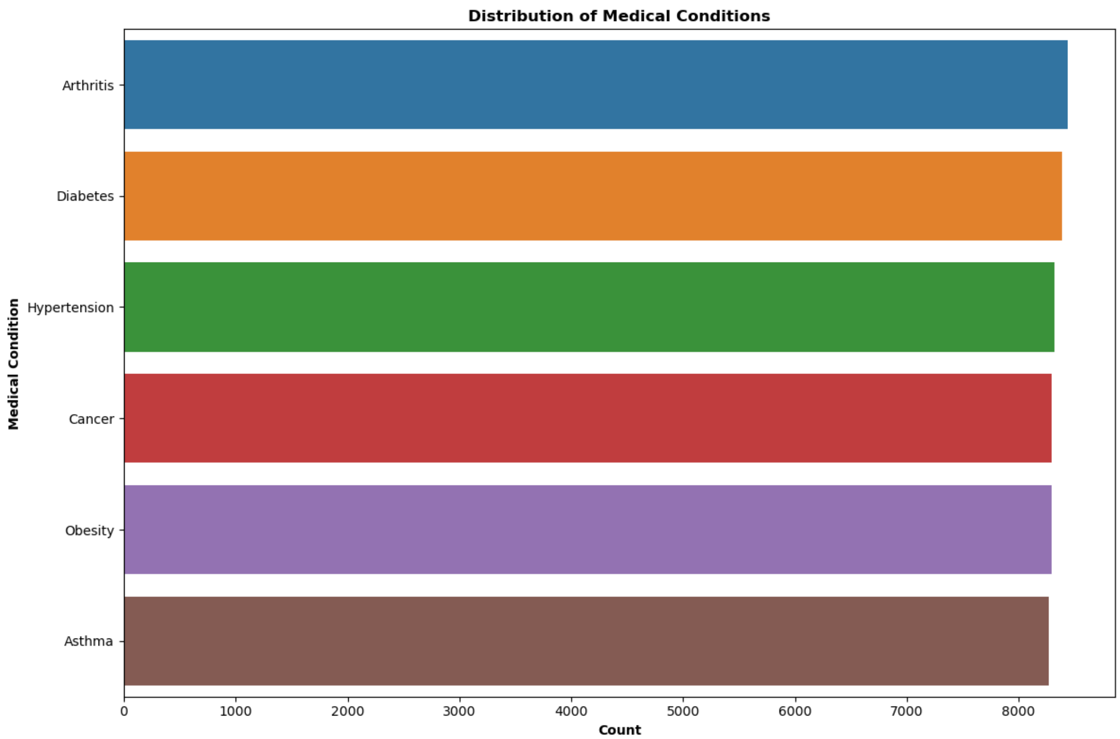
[40]:
```
# 4. What is the distribution of different blood types?
plt.figure(figsize=(10, 6))
sns.countplot(x='Blood Type', data=df)
plt.title('Distribution of Different Blood Types',fontweight='bold')
plt.xlabel('Blood Type',fontweight='bold')
plt.ylabel('Count',fontweight='bold')
```

```
# Insights
insights = """
Insights for Hospitals:
1. Understanding the overall distribution of blood types helps in managing␣
↪blood bank inventories effectively.
2. Hospitals can identify which blood types are more common and which are␣
↪rarer, allowing for targeted blood donation drives.

Insights for Patients:
1. Patients can see how common or rare their blood type is, providing awareness␣
↪for personal medical preparedness.
2. Individuals with rare blood types might be encouraged to donate more␣
↪frequently to ensure an adequate supply for emergencies.
"""

plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center',␣
↪fontsize=10)
plt.tight_layout()
plt.show()
```



[41]:

```
# 5. What is the distribution of medical conditions across the dataset?
plt.figure(figsize=(12, 8))
sns.countplot(y='Medical Condition', data=df, order=df['Medical Condition'].
 ↪value_counts().index)
plt.title('Distribution of Medical Conditions',fontweight='bold')
plt.xlabel('Count',fontweight='bold')
plt.ylabel('Medical Condition',fontweight='bold')


# Insights
insights = """
Insights for Hospitals:
1. Identifying the most common medical conditions can help in resource␣
 ↪allocation and specialized staff training.
2. Hospitals can develop targeted treatment programs and preventive measures␣
 ↪for prevalent conditions.

Insights for Patients:
1. Understanding the prevalence of certain medical conditions can help patients␣
 ↪be more proactive about their health.
2. Patients can seek information and resources about common conditions to␣
 ↪better manage their health and prevent complications.
"""

plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center',␣
 ↪fontsize=10)
plt.tight_layout()
plt.show()
```



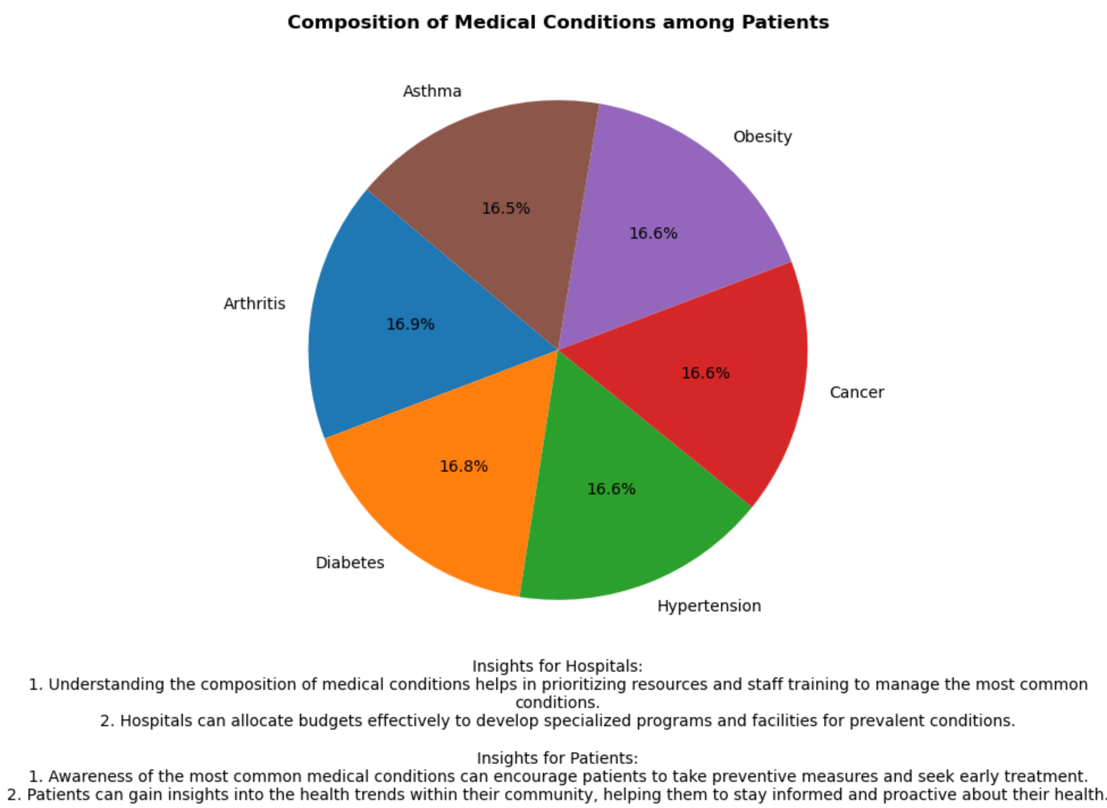
Insights for Hospitals:
1. Identifying the most common medical conditions can help in resource allocation and specialized staff training.
2. Hospitals can develop targeted treatment programs and preventive measures for prevalent conditions.

Insights for Patients:
1. Understanding the prevalence of certain medical conditions can help patients be more proactive about their health.
2. Patients can seek information and resources about common conditions to better manage their health and prevent complications.

7 Composition Visualizations

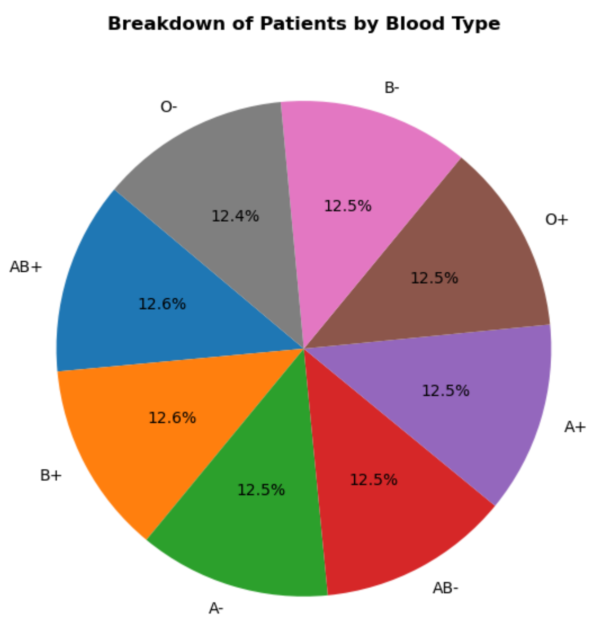
[42]:
```
# 1. What is the composition of medical conditions among patients?
plt.figure(figsize=(10, 6))
df['Medical Condition'].value_counts().plot.pie(autopct='%1.1f%%',␣
↪startangle=140)
plt.title('Composition of Medical Conditions among Patients',fontweight='bold')
plt.ylabel('')

# Insights
insights = """
Insights for Hospitals:
1. Understanding the composition of medical conditions helps in prioritizing␣
↪resources and staff training to manage the most common conditions.
2. Hospitals can allocate budgets effectively to develop specialized programs␣
↪and facilities for prevalent conditions.
```

```
Insights for Patients:
1. Awareness of the most common medical conditions can encourage patients to take preventive measures and seek early treatment.
2. Patients can gain insights into the health trends within their community, helping them to stay informed and proactive about their health.
"""

plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center', fontsize=10)
plt.tight_layout()
plt.show()
```



[43]:
```
# 2. What is the breakdown of patients by blood type?
plt.figure(figsize=(10, 6))
df['Blood Type'].value_counts().plot.pie(autopct='%1.1f%%', startangle=140)
plt.title('Breakdown of Patients by Blood Type',fontweight='bold')
plt.ylabel('')

# Insights
insights = """
```

```
Insights for Hospitals:
1. Understanding the distribution of blood types among patients helps in␣
↪managing blood inventory and ensuring the availability of all blood types.
2. Hospitals can plan blood donation drives more effectively, targeting blood␣
↪types that are in lower supply.

Insights for Patients:
1. Patients can understand the prevalence of their blood type within the␣
↪community, which can help in emergency situations.
2. Individuals with rare blood types might be encouraged to donate blood more␣
↪frequently to ensure an adequate supply for others in need.
"""

plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center',␣
↪fontsize=10)
plt.tight_layout()
plt.show()
```

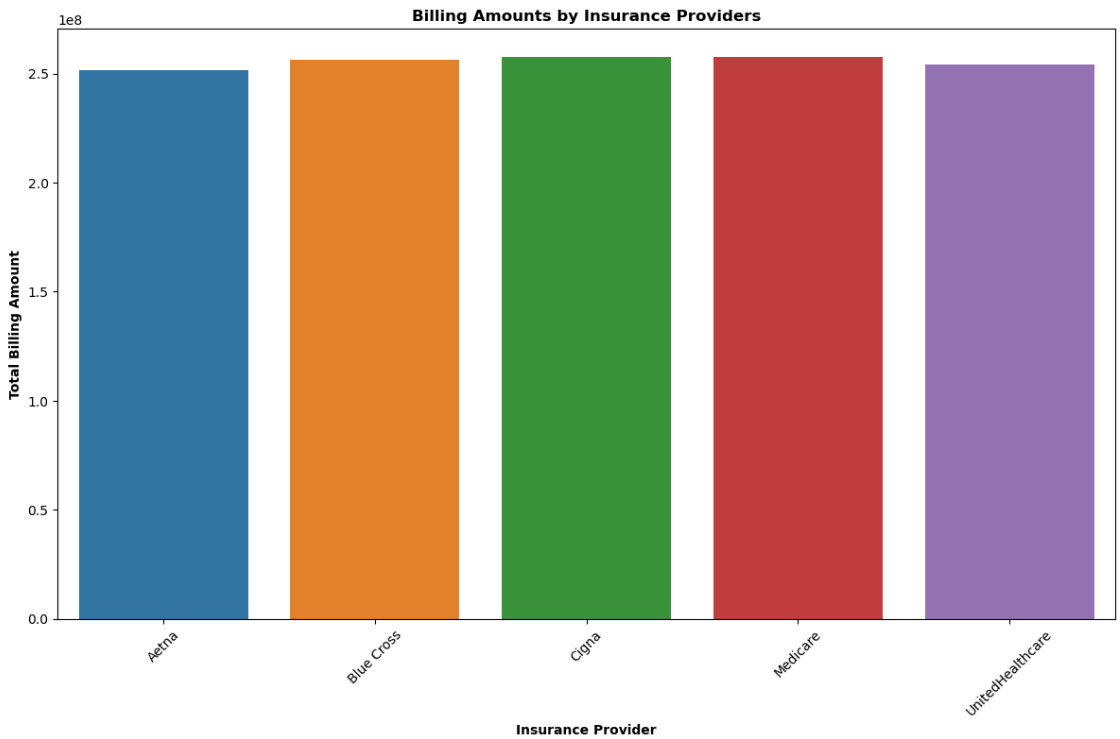


Insights for Hospitals:
1. Understanding the distribution of blood types among patients helps in managing blood inventory and ensuring the availability of all blood types.
2. Hospitals can plan blood donation drives more effectively, targeting blood types that are in lower supply.

Insights for Patients:
1. Patients can understand the prevalence of their blood type within the community, which can help in emergency situations.
2. Individuals with rare blood types might be encouraged to donate blood more frequently to ensure an adequate supply for others in need.

[44]:
```
# 3. How are billing amounts distributed across different insurance providers?
```

```
insurance_billing = df.groupby('Insurance Provider')['Billing Amount'].sum().
  ↪reset_index()
plt.figure(figsize=(12, 8))
sns.barplot(x='Insurance Provider', y='Billing Amount', data=insurance_billing)
plt.title('Billing Amounts by Insurance Providers',fontweight='bold')
plt.xlabel('Insurance Provider',fontweight='bold')
plt.ylabel('Total Billing Amount',fontweight='bold')
plt.xticks(rotation=45)


# Insights
insights = """
Insights for Hospitals:
1. Identifying the insurance providers that contribute the most to billing␣
  ↪amounts can help hospitals negotiate better terms and understand patient␣
  ↪demographics.
2. Hospitals can focus on improving relationships with top insurance providers␣
  ↪to streamline billing processes and improve cash flow.

Insights for Patients:
1. Patients can see which insurance providers are most commonly used, helping␣
  ↪them make informed decisions when choosing or switching providers.
2. Understanding billing distributions can encourage patients to review their␣
  ↪insurance coverage to ensure it meets their healthcare needs and financial␣
  ↪situation.
"""

plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center',␣
  ↪fontsize=10)
plt.tight_layout()
plt.show()
```

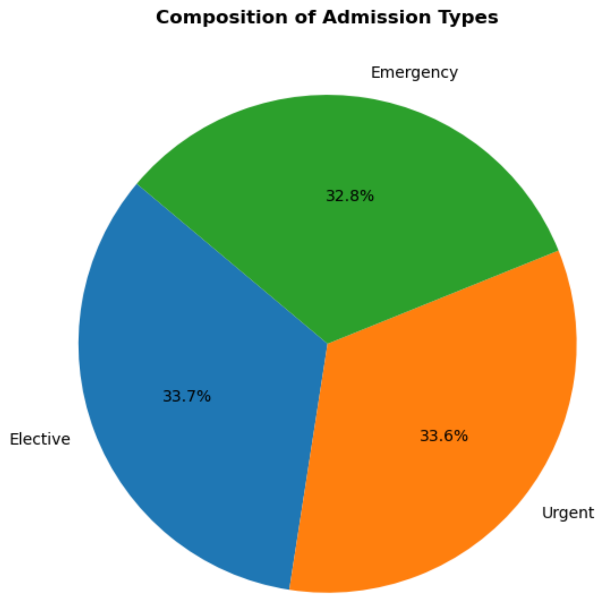


Insights for Hospitals:
1. Identifying the insurance providers that contribute the most to billing amounts can help hospitals negotiate better terms and understand patient demographics.
2. Hospitals can focus on improving relationships with top insurance providers to streamline billing processes and improve cash flow.

Insights for Patients:
1. Patients can see which insurance providers are most commonly used, helping them make informed decisions when choosing or switching providers.
2. Understanding billing distributions can encourage patients to review their insurance coverage to ensure it meets their healthcare needs and financial situation.

[45]:
```
# 4. What is the composition of admission types (urgent, emergency, elective)?
plt.figure(figsize=(10, 6))
df['Admission Type'].value_counts().plot.pie(autopct='%1.1f%%', startangle=140)
plt.title('Composition of Admission Types',fontweight='bold')
plt.ylabel('')


# Insights
insights = """
Insights for Hospitals:
1. Understanding the distribution of admission types helps in resource␣
↪allocation, ensuring that staff and facilities are prepared for different␣
↪types of admissions.
2. Hospitals can use this information to improve emergency response protocols␣
↪and manage elective surgeries more efficiently.

Insights for Patients:
```

```
1. Awareness of the distribution of admission types can help patients␣
 ↪understand the hospital's operational focus and preparedness for different␣
 ↪medical situations.
2. Patients can make informed decisions about their healthcare, knowing the␣
 ↪hospital's capabilities and response times for various types of admissions.
"""

plt.figtext(0.5, -0.2, insights, wrap=True, horizontalalignment='center',␣
 ↪fontsize=10)
plt.tight_layout()
plt.show()
```

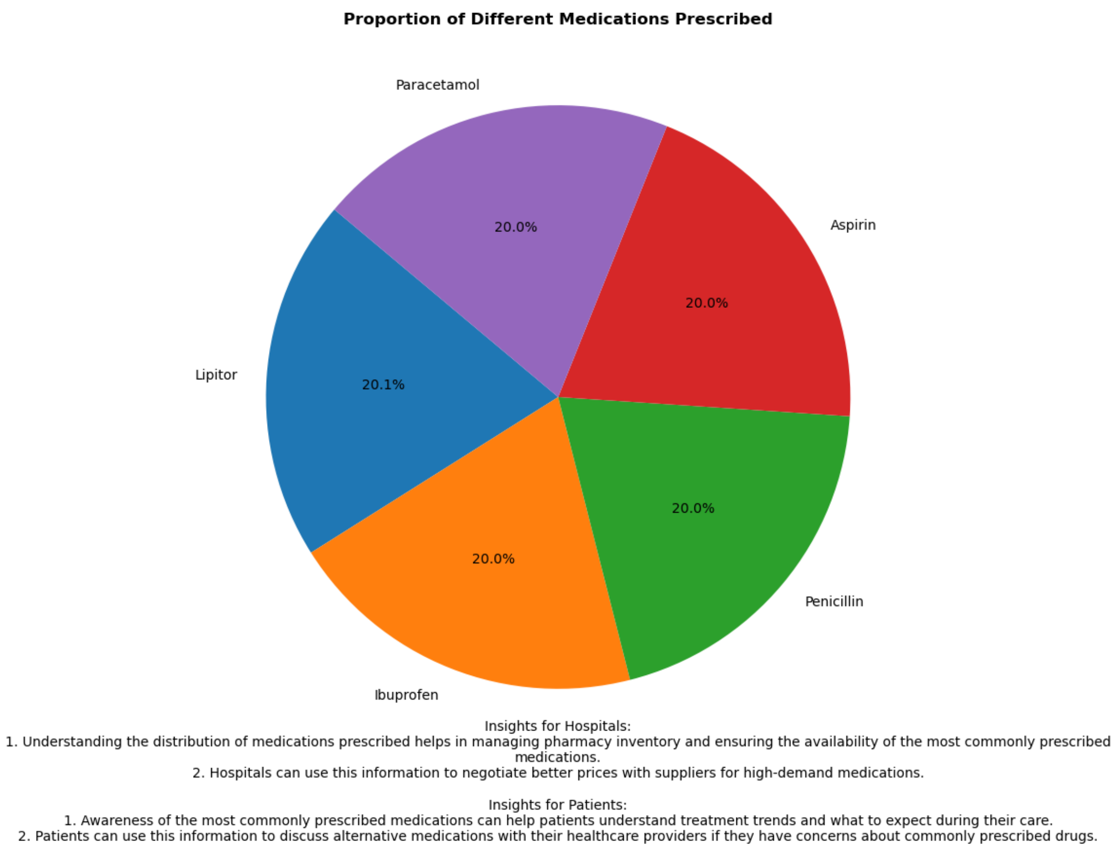


Insights for Hospitals:
1. Understanding the distribution of admission types helps in resource allocation, ensuring that staff and facilities are prepared for different types of admissions.
2. Hospitals can use this information to improve emergency response protocols and manage elective surgeries more efficiently.

Insights for Patients:
1. Awareness of the distribution of admission types can help patients understand the hospital's operational focus and preparedness for different medical situations.
2. Patients can make informed decisions about their healthcare, knowing the hospital's capabilities and response times for various types of admissions.

[46]:
```
# 5. What is the proportion of different medications prescribed?
plt.figure(figsize=(12, 8))
df['Medication'].value_counts().plot.pie(autopct='%1.1f%%', startangle=140)
plt.title('Proportion of Different Medications Prescribed',fontweight='bold')
plt.ylabel('')

# Insights
insights = """
Insights for Hospitals:
```

```
1. Understanding the distribution of medications prescribed helps in managing␣
  ↪pharmacy inventory and ensuring the availability of the most commonly␣
  ↪prescribed medications.
2. Hospitals can use this information to negotiate better prices with suppliers␣
  ↪for high-demand medications.

Insights for Patients:
1. Awareness of the most commonly prescribed medications can help patients␣
  ↪understand treatment trends and what to expect during their care.
2. Patients can use this information to discuss alternative medications with␣
  ↪their healthcare providers if they have concerns about commonly prescribed␣
  ↪drugs.
"""

plt.figtext(0.5, -0.1, insights, wrap=True, horizontalalignment='center',␣
  ↪fontsize=10)
plt.tight_layout()
plt.show()
```



```
[ ]:
```