healthcare-data-analysis (1)

December 1, 2025

[3]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime
import warnings
warnings.filterwarnings('ignore')
```

[4]:
```
# Set up visualization style
plt.style.use('seaborn-v0_8')
sns.set_palette("husl")

# Load the dataset
df = pd.read_csv('/kaggle/input/healthcare-data/healthcare_dataset.csv')

# Display basic information about the dataset
print("Dataset Shape:", df.shape)
print("\nFirst few rows:")
print(df.head())
```

```
Dataset Shape: (10000, 15)

First few rows:
                  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
```

1 Data exploration

[5]:
```
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  object
 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  object
 13  Medication          10000 non-null  object
 14  Test Results        10000 non-null  object
dtypes: float64(1), int64(2), object(12)
memory usage: 1.1+ MB
```

[6]:
```
print("\nDataset Info:")
print("\nMissing Values:")
print(df.isnull().sum())
```

```
Dataset Info:

Missing Values:
```

```
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
```

[7]:
```
df.head()
```

[7]:

	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
0	Tiffany Ramirez	81	Female	O-	Diabetes	2022-11-17	Patrick Parker	Wallace-Hamilton	Medicare	37490.983364	146	Elective	2022-12-01	Aspirin	Inconclusive
1	Ruben Burns	35	Male	O+	Asthma	2023-06-01	Diane Jackson	Burke, Griffin and Cooper	UnitedHealthcare	47304.064845	404	Emergency	2023-06-15	Lipitor	Normal
2	Chad Byrd	61	Male	B-	Obesity	2019-01-09	Paul Baker	Walton LLC	Medicare	36874.896997	292	Emergency	2019-02-08	Lipitor	Normal
3	Antonio Frederick	49	Male	B-	Asthma	2020-05-02	Brian Chandler	Garcia Ltd	Medicare	23303.322092	480	Urgent	2020-05-03	Penicillin	Abnormal
4	Mrs. Brandy Flowers	51	Male	O-	Arthritis	2021-07-09	Dustin Griffin	Jones, Brown and Murray	UnitedHealthcare	18086.344184	477	Urgent	2021-08-02	Paracetamol	Normal

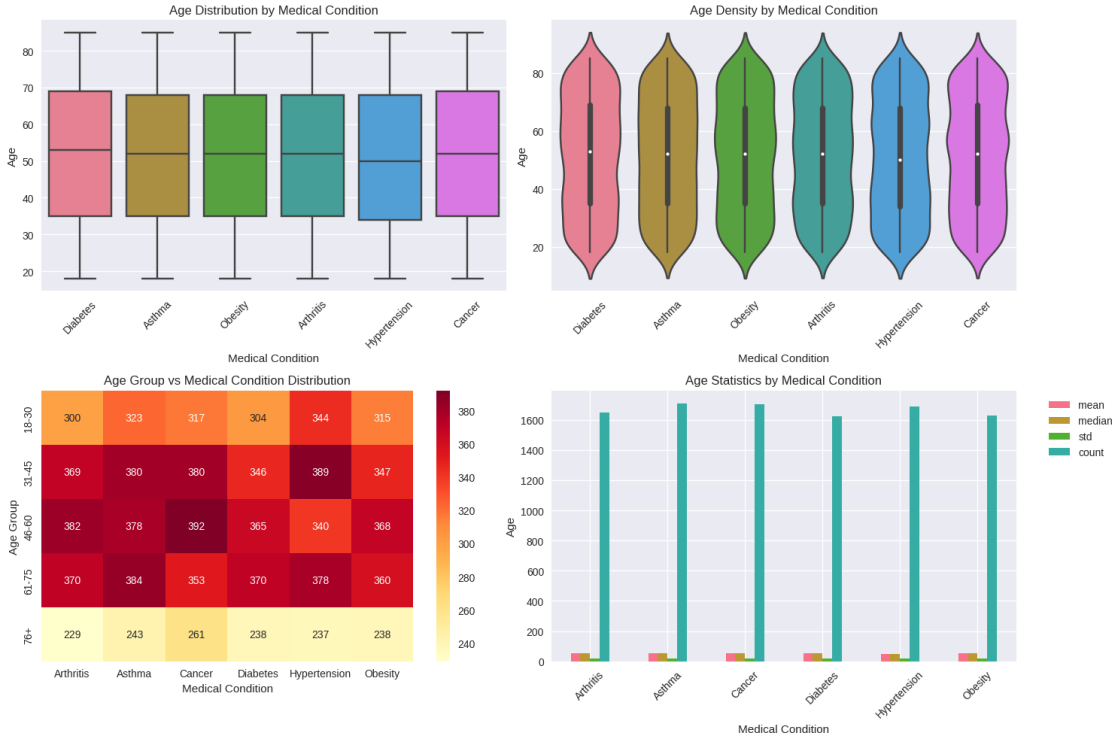
[8]:
```
# Data preprocessing
df['Date of Admission'] = pd.to_datetime(df['Date of Admission'])
df['Discharge Date'] = pd.to_datetime(df['Discharge Date'])
df['Days Hospitalized'] = (df['Discharge Date'] - df['Date of Admission']).dt.
 ↪days

# Create age groups
def create_age_group(age):
    if age <= 30:
        return '18-30'
    elif age <= 45:
        return '31-45'
    elif age <= 60:
        return '46-60'
    elif age <= 75:
        return '61-75'
    else:
        return '76+'

df['Age Group'] = df['Age'].apply(create_age_group)

print("Dataset loaded and preprocessed successfully!")
print(f"Dataset shape: {df.shape}")
```

```
Dataset loaded and preprocessed successfully!
Dataset shape: (10000, 17)
```

2 1. Age Distribution Across Medical Conditions

[9]:
```
# Question 1: What is the age distribution of patients across different medical␣
 ↪conditions?
plt.figure(figsize=(15, 10))

# Box plot
plt.subplot(2, 2, 1)
sns.boxplot(data=df, x='Medical Condition', y='Age')
plt.title('Age Distribution by Medical Condition')
plt.xticks(rotation=45)
plt.ylabel('Age')

# Violin plot
plt.subplot(2, 2, 2)
sns.violinplot(data=df, x='Medical Condition', y='Age')
plt.title('Age Density by Medical Condition')
plt.xticks(rotation=45)
plt.ylabel('Age')
```

```python
# Heatmap of age groups vs conditions
plt.subplot(2, 2, 3)
age_condition = pd.crosstab(df['Age Group'], df['Medical Condition'])
sns.heatmap(age_condition, annot=True, fmt='d', cmap='YlOrRd')
plt.title('Age Group vs Medical Condition Distribution')
plt.ylabel('Age Group')

# Statistical summary
plt.subplot(2, 2, 4)
condition_age_stats = df.groupby('Medical Condition')['Age'].agg(['mean',␣
↪'median', 'std', 'count']).round(1)
condition_age_stats.plot(kind='bar', ax=plt.gca())
plt.title('Age Statistics by Medical Condition')
plt.xticks(rotation=45)
plt.ylabel('Age')
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

plt.tight_layout()
plt.show()

print("Average Age by Medical Condition:")
print(condition_age_stats)
```

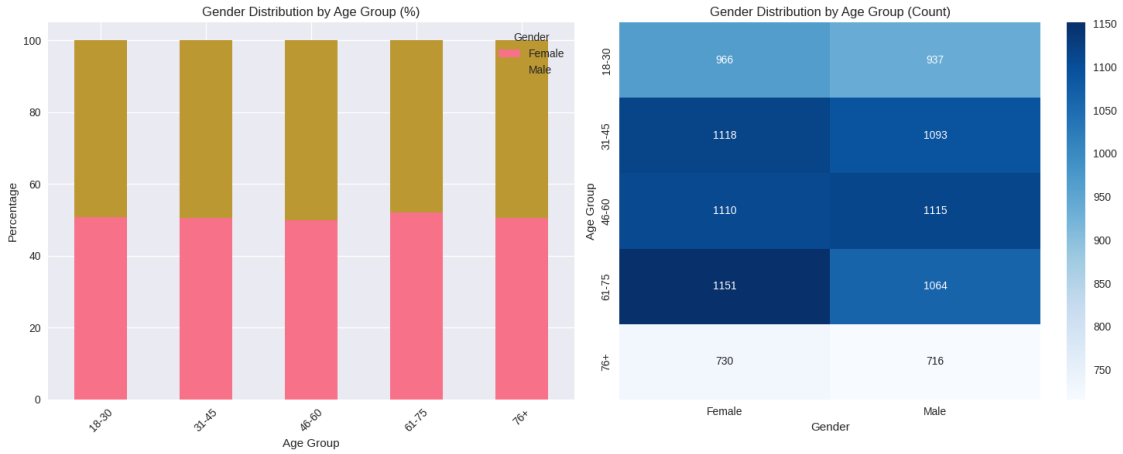


Average Age by Medical Condition:

```
                    mean  median   std  count
Medical Condition
Arthritis           51.5    52.0  19.4   1650
Asthma              51.4    52.0  19.5   1708
Cancer              51.6    52.0  19.5   1703
Diabetes            51.8    53.0  19.8   1623
Hypertension        50.7    50.0  19.8   1688
Obesity             51.6    52.0  19.5   1628
```

3 2. Gender Distribution Across Age Groups

[10]:
```python
# Question 2: How does gender distribution vary across different age groups?
plt.figure(figsize=(15, 6))

# Stacked bar chart
plt.subplot(1, 2, 1)
gender_age = pd.crosstab(df['Age Group'], df['Gender'])
gender_age_pct = gender_age.div(gender_age.sum(axis=1), axis=0) * 100
gender_age_pct.plot(kind='bar', stacked=True, ax=plt.gca())
plt.title('Gender Distribution by Age Group (%)')
plt.xlabel('Age Group')
plt.ylabel('Percentage')
plt.legend(title='Gender')
plt.xticks(rotation=45)

# Heatmap
plt.subplot(1, 2, 2)
sns.heatmap(gender_age, annot=True, fmt='d', cmap='Blues')
plt.title('Gender Distribution by Age Group (Count)')
plt.xlabel('Gender')
plt.ylabel('Age Group')

plt.tight_layout()
plt.show()

print("Gender Distribution by Age Group:")
print(gender_age)
print("\nPercentage Distribution:")
print(gender_age_pct.round(1))
```

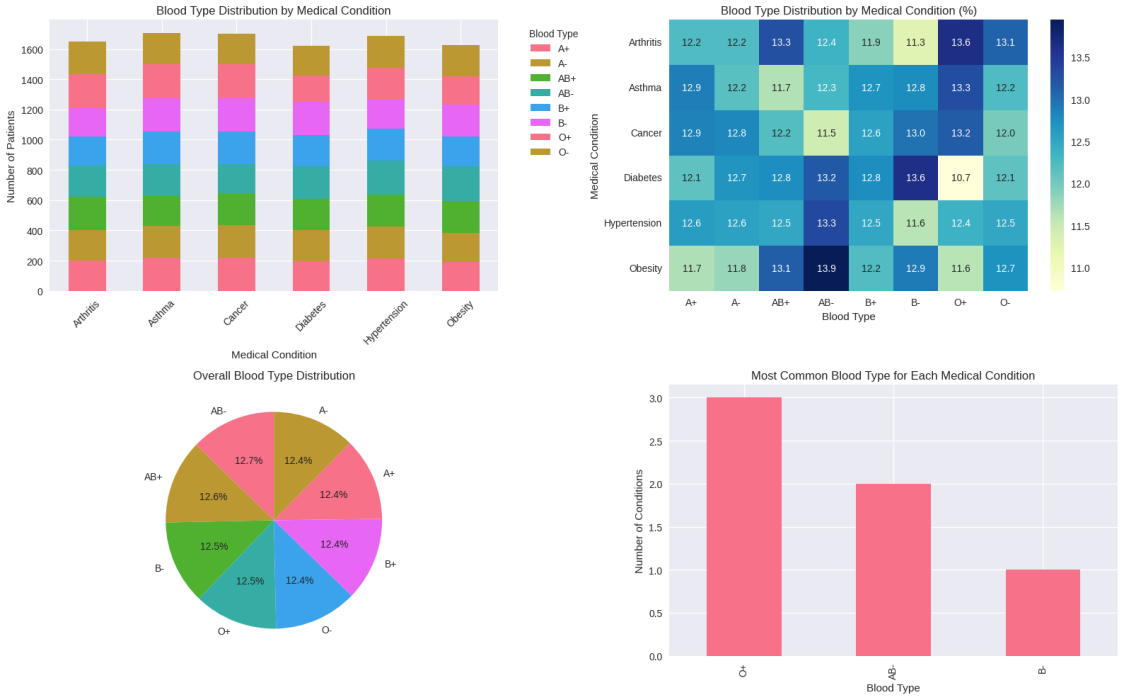



```
Gender Distribution by Age Group:
Gender     Female  Male
Age Group
18-30         966   937
31-45        1118  1093
46-60        1110  1115
61-75        1151  1064
76+           730   716

Percentage Distribution:
Gender     Female  Male
Age Group
18-30        50.8  49.2
31-45        50.6  49.4
46-60        49.9  50.1
61-75        52.0  48.0
76+          50.5  49.5
```

4 3. Blood Type Distribution by Medical Conditions

[11]:
```
# Question 3: What is the blood type distribution among patients with different␣
↪medical conditions?
plt.figure(figsize=(16, 10))

# Stacked bar chart
plt.subplot(2, 2, 1)
blood_condition = pd.crosstab(df['Medical Condition'], df['Blood Type'])
blood_condition.plot(kind='bar', stacked=True, ax=plt.gca())
plt.title('Blood Type Distribution by Medical Condition')
plt.xlabel('Medical Condition')
plt.ylabel('Number of Patients')
```

```
plt.legend(title='Blood Type', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.xticks(rotation=45)

# Heatmap
plt.subplot(2, 2, 2)
blood_condition_pct = blood_condition.div(blood_condition.sum(axis=1), axis=0)␣
 ↪* 100
sns.heatmap(blood_condition_pct, annot=True, fmt='.1f', cmap='YlGnBu')
plt.title('Blood Type Distribution by Medical Condition (%)')
plt.xlabel('Blood Type')
plt.ylabel('Medical Condition')

# Blood type overall distribution
plt.subplot(2, 2, 3)
blood_dist = df['Blood Type'].value_counts()
plt.pie(blood_dist.values, labels=blood_dist.index, autopct='%1.1f%%',␣
 ↪startangle=90)
plt.title('Overall Blood Type Distribution')

# Most common blood type by condition
plt.subplot(2, 2, 4)
most_common_blood = df.groupby('Medical Condition')['Blood Type'].agg(lambda x:␣
 ↪x.mode().iloc[0] if not x.mode().empty else 'Unknown')
most_common_blood.value_counts().plot(kind='bar')
plt.title('Most Common Blood Type for Each Medical Condition')
plt.xlabel('Blood Type')
plt.ylabel('Number of Conditions')

plt.tight_layout()
plt.show()

print("Blood Type Distribution by Medical Condition:")
print(blood_condition)
```

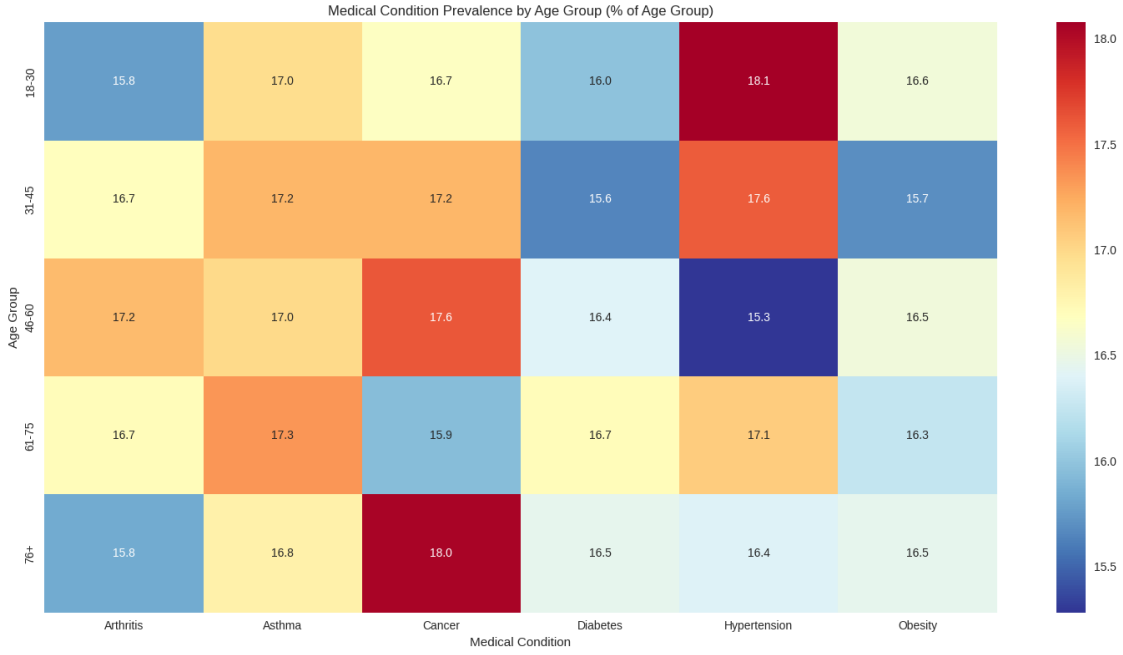



```
Blood Type Distribution by Medical Condition:
Blood Type           A+   A-   AB+  AB-   B+   B-   O+   O-
Medical Condition
Arthritis           202  202  219  204  196  186  225  216
Asthma              220  208  199  210  217  218  227  209
Cancer              219  218  208  195  214  221  224 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

5 4. Age Group Susceptibility to Medical Conditions

[12]:
```
# Question 4: Which age groups are most susceptible to specific medical␣
 ↪conditions?
plt.figure(figsize=(15, 8))

# Calculate susceptibility (percentage of age group with condition)
age_condition_pct = pd.crosstab(df['Age Group'], df['Medical Condition'],␣
 ↪normalize='index') * 100

# Heatmap
sns.heatmap(age_condition_pct, annot=True, fmt='.1f', cmap='RdYlBu_r')
plt.title('Medical Condition Prevalence by Age Group (% of Age Group)')
plt.xlabel('Medical Condition')
```

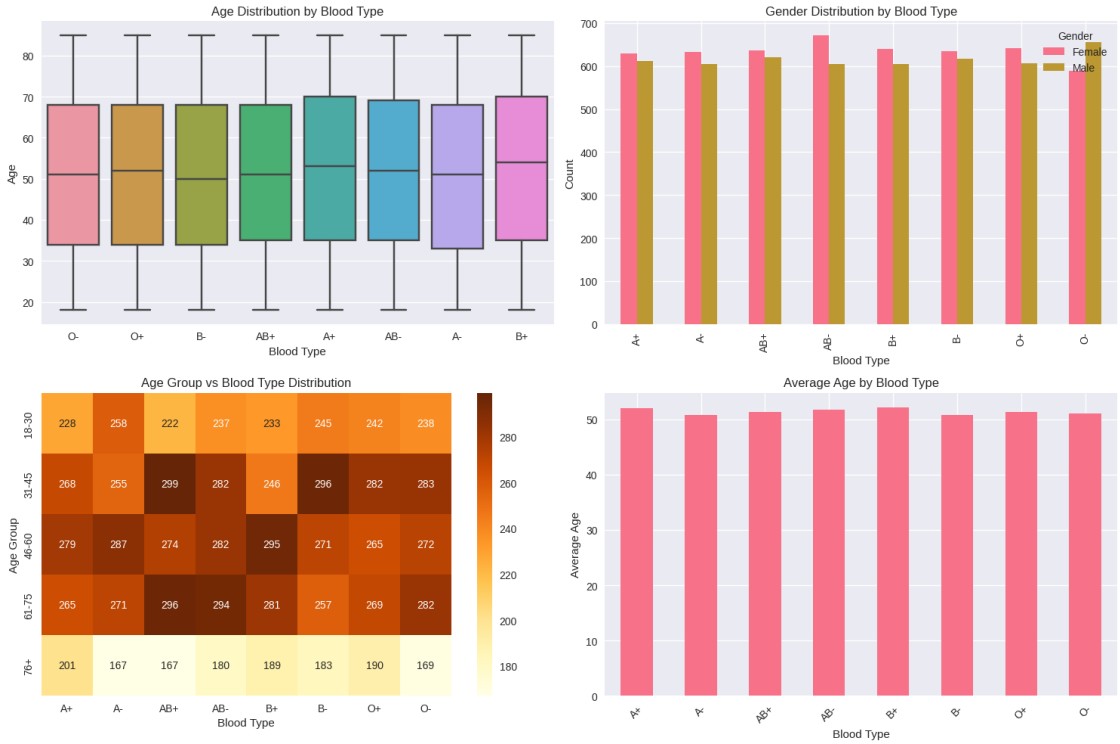
```
plt.ylabel('Age Group')
plt.tight_layout()
plt.show()

# Find most susceptible age group for each condition
print("Most Susceptible Age Group for Each Medical Condition:")
for condition in df['Medical Condition'].unique():
    susceptible_age = age_condition_pct[condition].idxmax()
    percentage = age_condition_pct[condition].max()
    print(f"  {condition}: {susceptible_age} ({percentage:.1f}% of this age␣
 ↪group)")
```




```
Most Susceptible Age Group for Each Medical Condition:
  Diabetes: 61-75 (16.7% of this age group)
  Asthma: 61-75 (17.3% of this age group)
  Obesity: 18-30 (16.6% of this age group)
  Arthritis: 46-60 (17.2% of this age group)
  Hypertension: 18-30 (18.1% of this age group)
  Cancer: 76+ (18.0% of this age group)
```

6 5. Demographics Correlation with Blood Types

[13]:
```
# Question 5: How does patient demographics (age, gender) correlate with blood types?
plt.figure(figsize=(15, 10))

# Age vs Blood Type
plt.subplot(2, 2, 1)
sns.boxplot(data=df, x='Blood Type', y='Age')
plt.title('Age Distribution by Blood Type')
plt.ylabel('Age')

# Gender vs Blood Type
plt.subplot(2, 2, 2)
gender_blood = pd.crosstab(df['Blood Type'], df['Gender'])
gender_blood.plot(kind='bar', ax=plt.gca())
plt.title('Gender Distribution by Blood Type')
plt.xlabel('Blood Type')
plt.ylabel('Count')
plt.legend(title='Gender')

# Age Group vs Blood Type
plt.subplot(2, 2, 3)
agegroup_blood = pd.crosstab(df['Age Group'], df['Blood Type'])
sns.heatmap(agegroup_blood, annot=True, fmt='d', cmap='YlOrBr')
plt.title('Age Group vs Blood Type Distribution')
plt.xlabel('Blood Type')
plt.ylabel('Age Group')

# Statistical analysis
plt.subplot(2, 2, 4)
blood_age_stats = df.groupby('Blood Type')['Age'].agg(['mean', 'std', 'count']).round(1)
blood_age_stats['mean'].plot(kind='bar', ax=plt.gca())
plt.title('Average Age by Blood Type')
plt.xlabel('Blood Type')
plt.ylabel('Average Age')
plt.xticks(rotation=45)

plt.tight_layout()
plt.show()

print("Age Statistics by Blood Type:")
print(blood_age_stats)
print("\nGender Distribution by Blood Type:")
print(gender_blood)
```

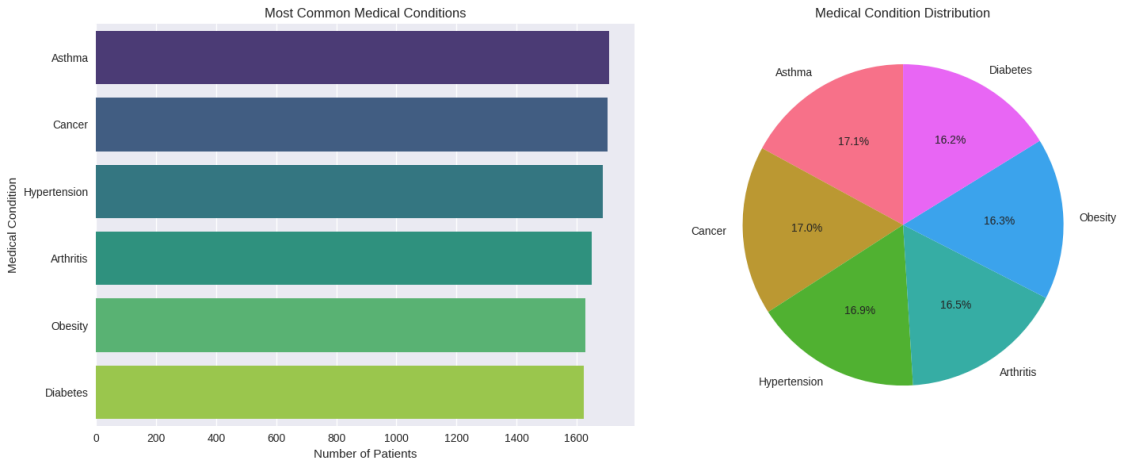



```
Age Statistics by Blood Type:
              mean   std  count
Blood Type
A+            52.1  19.9   1241
A-            50.8  19.6   1238
AB+           51.4  19.3   1258
AB-           51.8  19.5   1275
B+            52.2  19.6   1244
B-            50.8  19.6   1252
O+            51.4  19.6   1248
O-            51.1  19.5   1244

Gender Distribution by Blood Type:
Gender       Female  Male
Blood Type
A+              629   612
A-              633   605
AB+             637   621
AB-             671   604
B+              640   604
B-              635   617
O+              641   607
O-              589   655
```

7 Medical Conditions & Health Analysis

6. Most Common Medical Conditions

[14]:
```
# Question 6: What are the most common medical conditions in the dataset?
plt.figure(figsize=(15, 6))

# Count plot
plt.subplot(1, 2, 1)
condition_counts = df['Medical Condition'].value_counts()
sns.barplot(x=condition_counts.values, y=condition_counts.index,␣
 ↪palette='viridis')
plt.title('Most Common Medical Conditions')
plt.xlabel('Number of Patients')
plt.ylabel('Medical Condition')

# Pie chart
plt.subplot(1, 2, 2)
plt.pie(condition_counts.values, labels=condition_counts.index, autopct='%1.
 ↪1f%%', startangle=90)
plt.title('Medical Condition Distribution')

plt.tight_layout()
plt.show()

print("Medical Condition Frequency:")
for condition, count in condition_counts.items():
    percentage = (count / len(df)) * 100
    print(f"  {condition}: {count} patients ({percentage:.1f}%)")
```



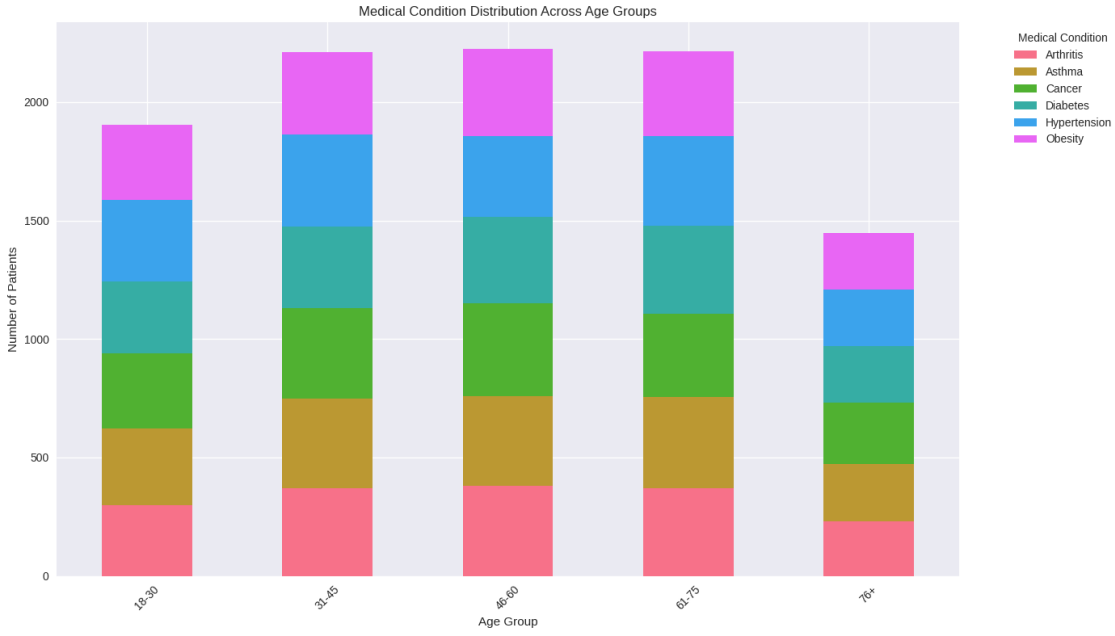

```
Medical Condition Frequency:
  Asthma: 1708 patients (17.1%)
```

```
Cancer: 1703 patients (17.0%)
Hypertension: 1688 patients (16.9%)
Arthritis: 1650 patients (16.5%)
Obesity: 1628 patients (16.3%)
Diabetes: 1623 patients (16.2%)
```

8 7. Medical Conditions Distribution Across Age Groups

[15]:
```
# Question 7: How are medical conditions distributed across different age groups?
plt.figure(figsize=(16, 10))

# Stacked bar chart
condition_age_dist = pd.crosstab(df['Age Group'], df['Medical Condition'])
condition_age_dist.plot(kind='bar', stacked=True, figsize=(14, 8))
plt.title('Medical Condition Distribution Across Age Groups')
plt.xlabel('Age Group')
plt.ylabel('Number of Patients')
plt.legend(title='Medical Condition', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

print("Medical Condition Distribution by Age Group:")
print(condition_age_dist)
```

```
<Figure size 1600x1000 with 0 Axes>
```



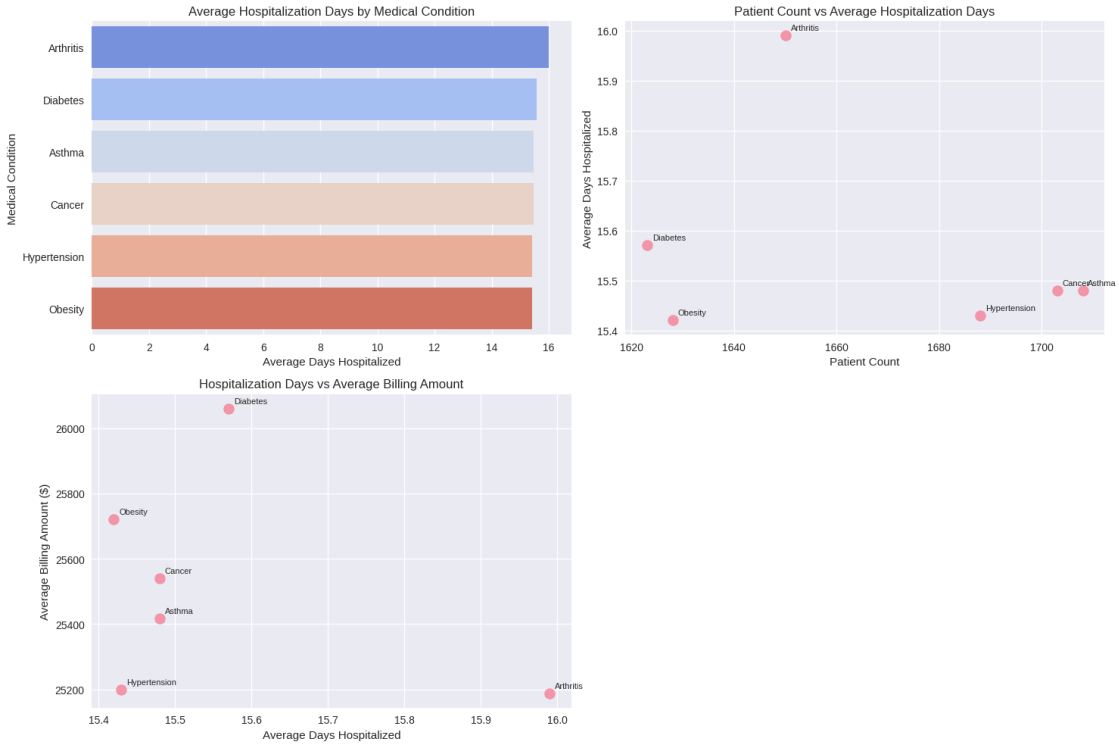
```
Medical Condition Distribution by Age Group:
```

Medical Condition	Arthritis	Asthma	Cancer	Diabetes	Hypertension	Obesity
Age Group						
18-30	300	323	317	304	344	315
31-45	369	380	380	346	389	347
46-60	382	378	392	365	340	368
61-75	370	384	353	370	378	360
76+	229	243	261	238	237	238

9 8. Medical Conditions with Highest Hospitalization Rates

[16]:
```
# Question 8: Which medical conditions have the highest hospitalization rates?
condition_hospitalization = df.groupby('Medical Condition').agg({
    'Days Hospitalized': ['mean', 'median', 'count'],
    'Billing Amount': 'mean'
}).round(2)

condition_hospitalization.columns = ['Avg Days Hospitalized', 'Median Days␣
 ↪Hospitalized', 'Patient Count', 'Avg Billing Amount']
condition_hospitalization = condition_hospitalization.sort_values('Avg Days␣
 ↪Hospitalized', ascending=False)

plt.figure(figsize=(15, 10))

# Average hospitalization days
plt.subplot(2, 2, 1)
sns.barplot(x=condition_hospitalization['Avg Days Hospitalized'],␣
 ↪y=condition_hospitalization.index, palette='coolwarm')
plt.title('Average Hospitalization Days by Medical Condition')
plt.xlabel('Average Days Hospitalized')

# Patient count vs hospitalization days
plt.subplot(2, 2, 2)
plt.scatter(condition_hospitalization['Patient Count'],␣
 ↪condition_hospitalization['Avg Days Hospitalized'], s=100, alpha=0.7)
for i, condition in enumerate(condition_hospitalization.index):
    plt.annotate(condition, (condition_hospitalization['Patient Count'].
 ↪iloc[i], condition_hospitalization['Avg Days Hospitalized'].iloc[i]),
                 xytext=(5, 5), textcoords='offset points', fontsize=8)
plt.title('Patient Count vs Average Hospitalization Days')
plt.xlabel('Patient Count')
plt.ylabel('Average Days Hospitalized')

# Billing vs hospitalization
```

```
plt.subplot(2, 2, 3)
plt.scatter(condition_hospitalization['Avg Days Hospitalized'],␣
 ↪condition_hospitalization['Avg Billing Amount'], s=100, alpha=0.7)
for i, condition in enumerate(condition_hospitalization.index):
    plt.annotate(condition, (condition_hospitalization['Avg Days Hospitalized'].
 ↪iloc[i], condition_hospitalization['Avg Billing Amount'].iloc[i]),
                 xytext=(5, 5), textcoords='offset points', fontsize=8)
plt.title('Hospitalization Days vs Average Billing Amount')
plt.xlabel('Average Days Hospitalized')
plt.ylabel('Average Billing Amount ($)')

plt.tight_layout()
plt.show()

print("Hospitalization Statistics by Medical Condition:")
print(condition_hospitalization)
```




```
Hospitalization Statistics by Medical Condition:
                   Avg Days Hospitalized  Median Days Hospitalized  \
Medical Condition
Arthritis                          15.99                      16.0
Diabetes                           15.57                      16.0
Asthma                             15.48                      16.0
```

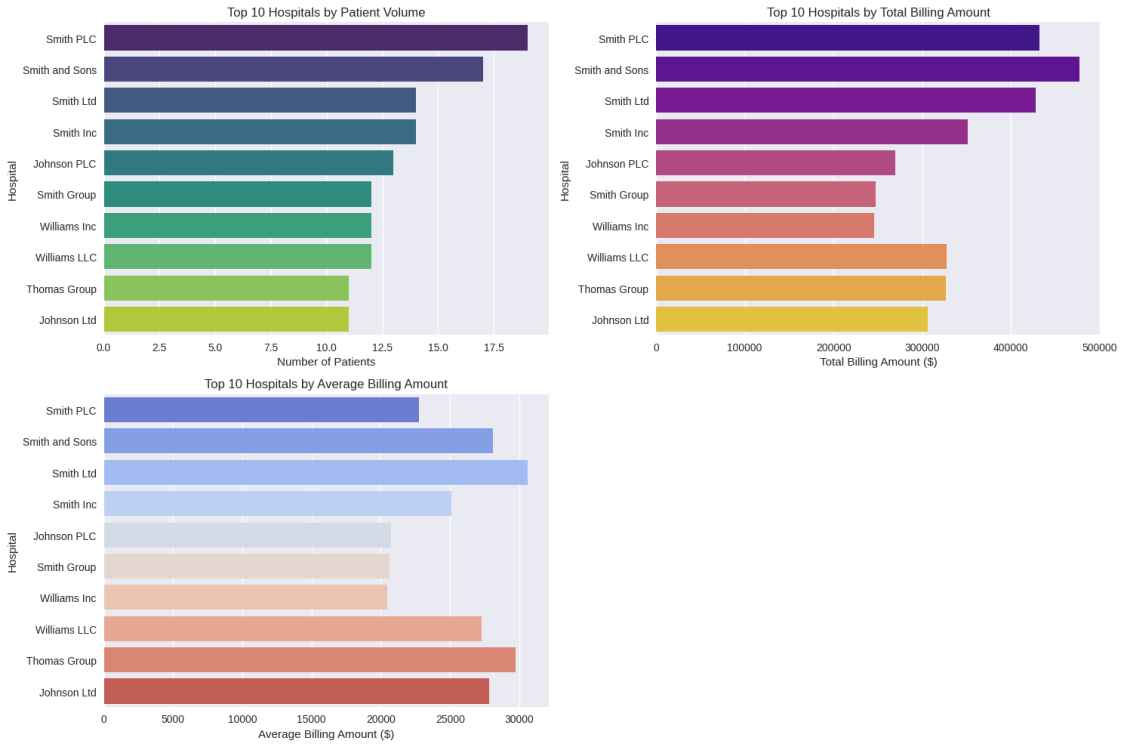
```
Cancer                          15.48                    15.0
Hypertension                    15.43                    15.0
Obesity                         15.42                    15.0

                   Patient Count  Avg Billing Amount
Medical Condition
Arthritis                   1650            25187.63
Diabetes                    1623            26060.12
Asthma                      1708            25416.87
Cancer                      1703            25539.10
Hypertension                1688            25198.03
Obesity                     1628            25720.84
```

10 Hospital & Healthcare Provider Analysis

11 11. Hospitals with Highest Patient Volume

[17]:
```python
# Question 11: Which hospitals have the highest patient volume?
hospital_stats = df.groupby('Hospital').agg({
    'Name': 'count',
    'Billing Amount': ['sum', 'mean'],
    'Days Hospitalized': 'mean'
}).round(2)

hospital_stats.columns = ['Patient Count', 'Total Billing', 'Avg Billing', 'Avg␣
 ↪Hospitalization Days']
hospital_stats = hospital_stats.sort_values('Patient Count', ascending=False)

plt.figure(figsize=(15, 10))

# Patient count
plt.subplot(2, 2, 1)
sns.barplot(x=hospital_stats['Patient Count'].head(10), y=hospital_stats.
 ↪head(10).index, palette='viridis')
plt.title('Top 10 Hospitals by Patient Volume')
plt.xlabel('Number of Patients')

# Total billing
plt.subplot(2, 2, 2)
sns.barplot(x=hospital_stats['Total Billing'].head(10), y=hospital_stats.
 ↪head(10).index, palette='plasma')
plt.title('Top 10 Hospitals by Total Billing Amount')
plt.xlabel('Total Billing Amount ($)')

# Average billing
plt.subplot(2, 2, 3)
```

```
sns.barplot(x=hospital_stats['Avg Billing'].head(10), y=hospital_stats.head(10).
 ↪index, palette='coolwarm')
plt.title('Top 10 Hospitals by Average Billing Amount')
plt.xlabel('Average Billing Amount ($)')

plt.tight_layout()
plt.show()

print("Top 10 Hospitals by Patient Volume:")
print(hospital_stats[['Patient Count', 'Total Billing', 'Avg Billing']].
 ↪head(10))
```

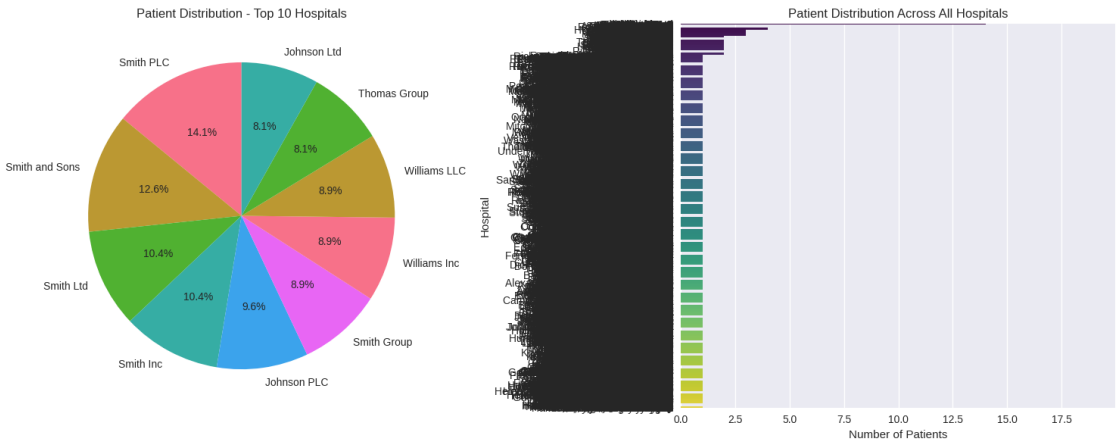



```
Top 10 Hospitals by Patient Volume:
               Patient Count  Total Billing  Avg Billing
Hospital
Smith PLC                 19      432283.55     22751.77
Smith and Sons            17      477638.88     28096.40
Smith Ltd                 14      428163.07     30583.08
Smith Inc                 14      351463.89     25104.56
Johnson PLC               13      269777.54     20752.12
Smith Group               12      247635.08     20636.26
Williams Inc              12      245434.30     20452.86
Williams LLC              12      327522.47     27293.54
Thomas Group              11      327045.37     29731.40
```

```
Johnson Ltd                 11       305971.19      27815.56
```

12 12. Patient Distribution Across Hospitals

[18]:
```
# Question 12: What is the patient distribution across different hospitals?
plt.figure(figsize=(15, 6))

# Pie chart for top 10 hospitals
plt.subplot(1, 2, 1)
top_hospitals = hospital_stats.head(10)
plt.pie(top_hospitals['Patient Count'], labels=top_hospitals.index, autopct='%1.
  ↪1f%%', startangle=90)
plt.title('Patient Distribution - Top 10 Hospitals')

# All hospitals patient count
plt.subplot(1, 2, 2)
sns.barplot(x=hospital_stats['Patient Count'], y=hospital_stats.index,␣
  ↪palette='viridis')
plt.title('Patient Distribution Across All Hospitals')
plt.xlabel('Number of Patients')

plt.tight_layout()
plt.show()

print("Patient Distribution Summary:")
print(f"Total hospitals: {len(hospital_stats)}")
print(f"Average patients per hospital: {hospital_stats['Patient Count'].mean():.
  ↪1f}")
print(f"Hospital with most patients: {hospital_stats['Patient Count'].idxmax()}␣
  ↪({hospital_stats['Patient Count'].max()} patients)")
print(f"Hospital with fewest patients: {hospital_stats['Patient Count'].
  ↪idxmin()} ({hospital_stats['Patient Count'].min()} patients)")
```

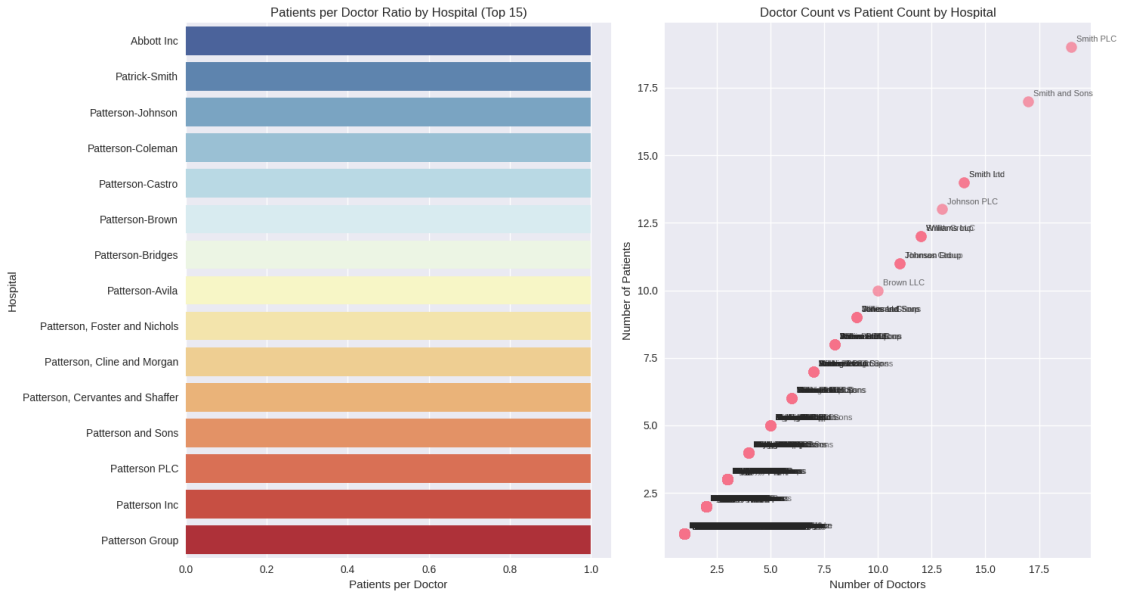


```
Patient Distribution Summary:
Total hospitals: 8639
Average patients per hospital: 1.2
Hospital with most patients: Smith PLC (19 patients)
Hospital with fewest patients: Payne-Bailey (1 patients)
```

13 13. Doctor-to-Patient Ratio

[19]:
```python
# Question 13: What is the doctor-to-patient ratio across different hospitals?
doctor_stats = df.groupby(['Hospital', 'Doctor']).size().
 ↪reset_index(name='Patient Count')
hospital_doctor_stats = doctor_stats.groupby('Hospital').agg({
    'Doctor': 'count',
    'Patient Count': 'sum'
}).rename(columns={'Doctor': 'Doctor Count'})

hospital_doctor_stats['Patients per Doctor'] = (hospital_doctor_stats['Patient␣
 ↪Count'] /
                                                hospital_doctor_stats['Doctor␣
 ↪Count']).round(1)
hospital_doctor_stats = hospital_doctor_stats.sort_values('Patients per␣
 ↪Doctor', ascending=False)

plt.figure(figsize=(15, 8))

# Patients per doctor
plt.subplot(1, 2, 1)
sns.barplot(x=hospital_doctor_stats['Patients per Doctor'].head(15),
            y=hospital_doctor_stats.head(15).index, palette='RdYlBu_r')
plt.title('Patients per Doctor Ratio by Hospital (Top 15)')
plt.xlabel('Patients per Doctor')

# Doctor count vs patient count
plt.subplot(1, 2, 2)
plt.scatter(hospital_doctor_stats['Doctor Count'],␣
 ↪hospital_doctor_stats['Patient Count'], alpha=0.7, s=100)
for i, hospital in enumerate(hospital_doctor_stats.index):
    plt.annotate(hospital, (hospital_doctor_stats['Doctor Count'].iloc[i],␣
 ↪hospital_doctor_stats['Patient Count'].iloc[i]),
                 xytext=(5, 5), textcoords='offset points', fontsize=8, alpha=0.
 ↪7)
plt.title('Doctor Count vs Patient Count by Hospital')
plt.xlabel('Number of Doctors')
plt.ylabel('Number of Patients')
```

```
plt.tight_layout()
plt.show()

print("Doctor-to-Patient Ratio by Hospital:")
print(hospital_doctor_stats[['Doctor Count', 'Patient Count', 'Patients per␣
↪Doctor']].sort_values('Patients per Doctor', ascending=False).head(10))
```

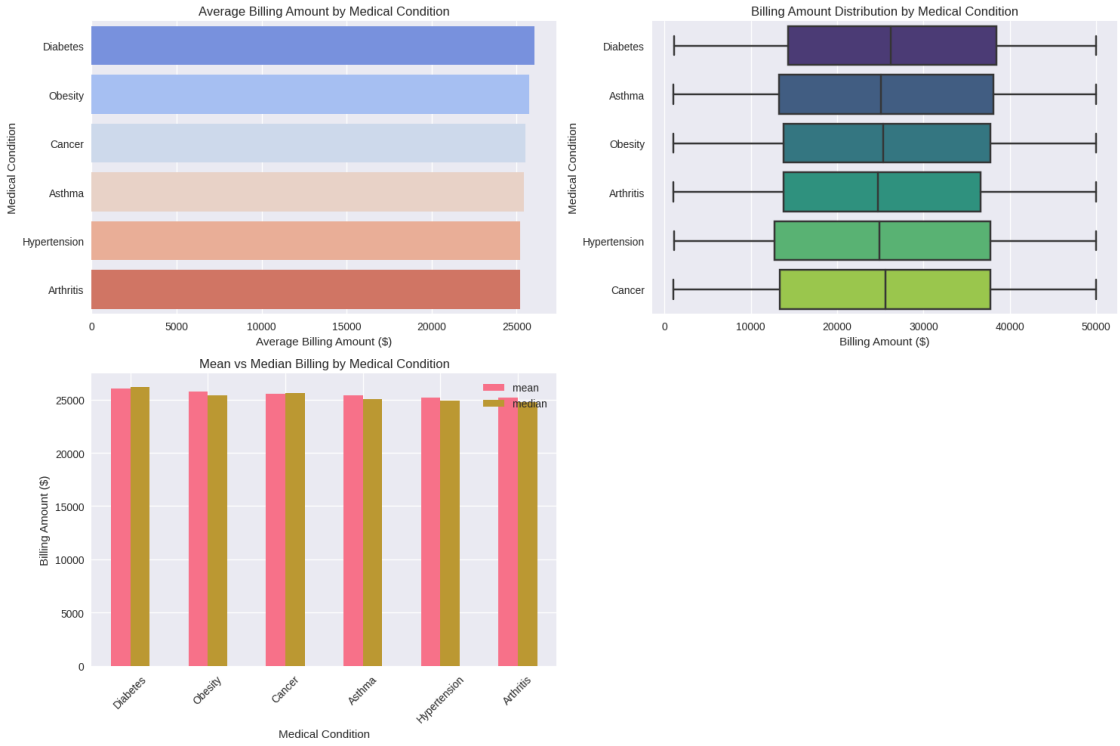



```
Doctor-to-Patient Ratio by Hospital:
                          Doctor Count  Patient Count  Patients per Doctor
Hospital
Abbott Inc                           1              1                  1.0
Acosta-Holmes                        1              1                  1.0
Abbott, Curry and Moore              1              1                  1.0
Abbott-Jordan                        1              1                  1.0
Abbott-Phillips                      1              1                  1.0
Abbott-Shea                          1              1                  1.0
Acevedo LLC                          1              1                  1.0
Acevedo and Sons                     1              1                  1.0
Acevedo, Rojas and Smith             1              1                  1.0
Acosta PLC                           1              1                  1.0
```

14 14. Average Billing by Medical Condition

[20]:
```
# Question 14: What is the average billing amount for different medical␣
↪conditions?
condition_billing = df.groupby('Medical Condition')['Billing Amount'].
↪agg(['mean', 'median', 'std', 'count']).round(2)
```

```
condition_billing = condition_billing.sort_values('mean', ascending=False)

plt.figure(figsize=(15, 10))

# Average billing by condition
plt.subplot(2, 2, 1)
sns.barplot(x=condition_billing['mean'], y=condition_billing.index,␣
 ↪palette='coolwarm')
plt.title('Average Billing Amount by Medical Condition')
plt.xlabel('Average Billing Amount ($)')

# Box plot of billing by condition
plt.subplot(2, 2, 2)
sns.boxplot(data=df, x='Billing Amount', y='Medical Condition',␣
 ↪palette='viridis')
plt.title('Billing Amount Distribution by Medical Condition')
plt.xlabel('Billing Amount ($)')

# Billing statistics
plt.subplot(2, 2, 3)
condition_billing[['mean', 'median']].plot(kind='bar', ax=plt.gca())
plt.title('Mean vs Median Billing by Medical Condition')
plt.xlabel('Medical Condition')
plt.ylabel('Billing Amount ($)')
plt.xticks(rotation=45)
plt.legend()

plt.tight_layout()
plt.show()

print("Billing Statistics by Medical Condition:")
print(condition_billing)
```

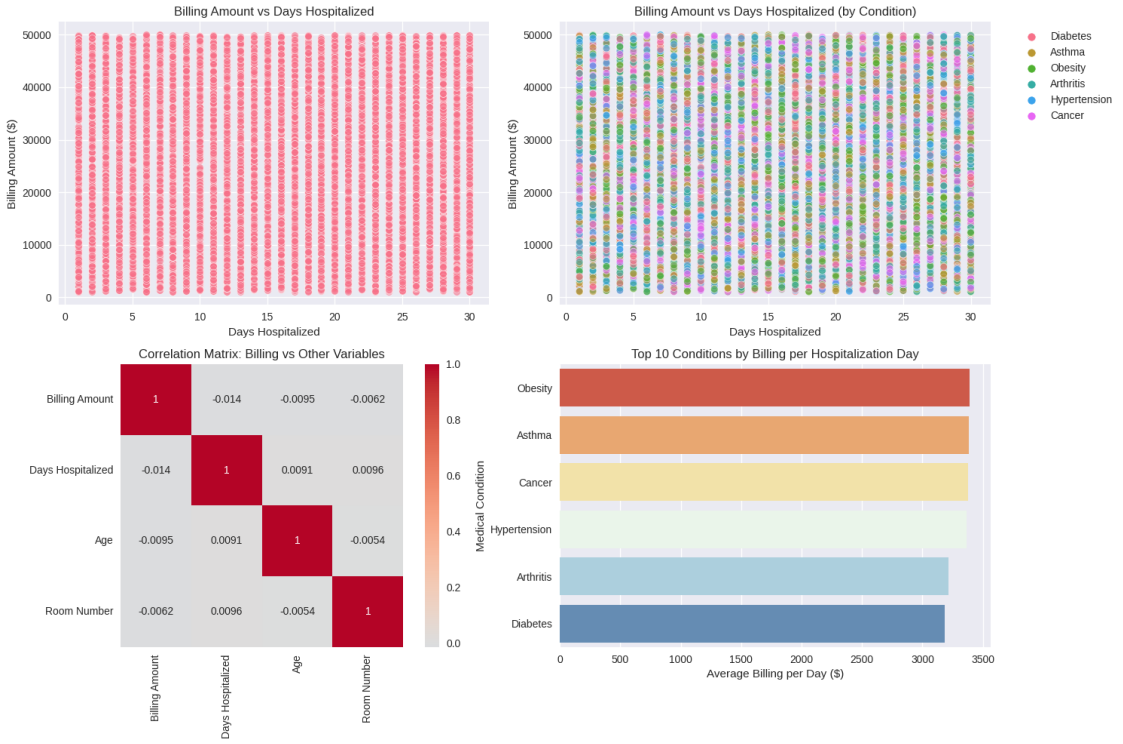



```
Billing Statistics by Medical Condition:
                        mean    median       std  count
Medical Condition
Diabetes            26060.12  26162.20  14013.92   1623
Obesity             25720.84  25365.02  14040.79   1628
Cancer              25539.10  25610.64  14081.99   1703
Asthma              25416.87  25073.45  14346.78   1708
Hypertension        25198.03  24920.46  14137.14   1688
Arthritis           25187.63  24739.17  13765.17   1650
```

15 15. Billing Amount vs Length of Stay

[21]:
```
# Question 16: How does billing amount correlate with length of stay?
plt.figure(figsize=(15, 10))

# Scatter plot
plt.subplot(2, 2, 1)
sns.scatterplot(data=df, x='Days Hospitalized', y='Billing Amount', alpha=0.6)
plt.title('Billing Amount vs Days Hospitalized')
plt.xlabel('Days Hospitalized')
plt.ylabel('Billing Amount ($)')

# By medical condition
```

```
plt.subplot(2, 2, 2)
sns.scatterplot(data=df, x='Days Hospitalized', y='Billing Amount',␣
 ↪hue='Medical Condition', alpha=0.6)
plt.title('Billing Amount vs Days Hospitalized (by Condition)')
plt.xlabel('Days Hospitalized')
plt.ylabel('Billing Amount ($)')
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

# Correlation heatmap
plt.subplot(2, 2, 3)
correlation_matrix = df[['Billing Amount', 'Days Hospitalized', 'Age', 'Room␣
 ↪Number']].corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', center=0,␣
 ↪square=True)
plt.title('Correlation Matrix: Billing vs Other Variables')

# Billing per day
df['Billing per Day'] = df['Billing Amount'] / df['Days Hospitalized']
billing_per_day_by_condition = df.groupby('Medical Condition')['Billing per␣
 ↪Day'].mean().sort_values(ascending=False)

plt.subplot(2, 2, 4)
sns.barplot(x=billing_per_day_by_condition.head(10).values,␣
 ↪y=billing_per_day_by_condition.head(10).index, palette='RdYlBu')
plt.title('Top 10 Conditions by Billing per Hospitalization Day')
plt.xlabel('Average Billing per Day ($)')

plt.tight_layout()
plt.show()

# Correlation statistics
correlation = df['Billing Amount'].corr(df['Days Hospitalized'])
print(f"Correlation between Billing Amount and Days Hospitalized: {correlation:.
 ↪3f}")
print(f"Average billing per day: ${df['Billing per Day'].mean():.2f}")
```

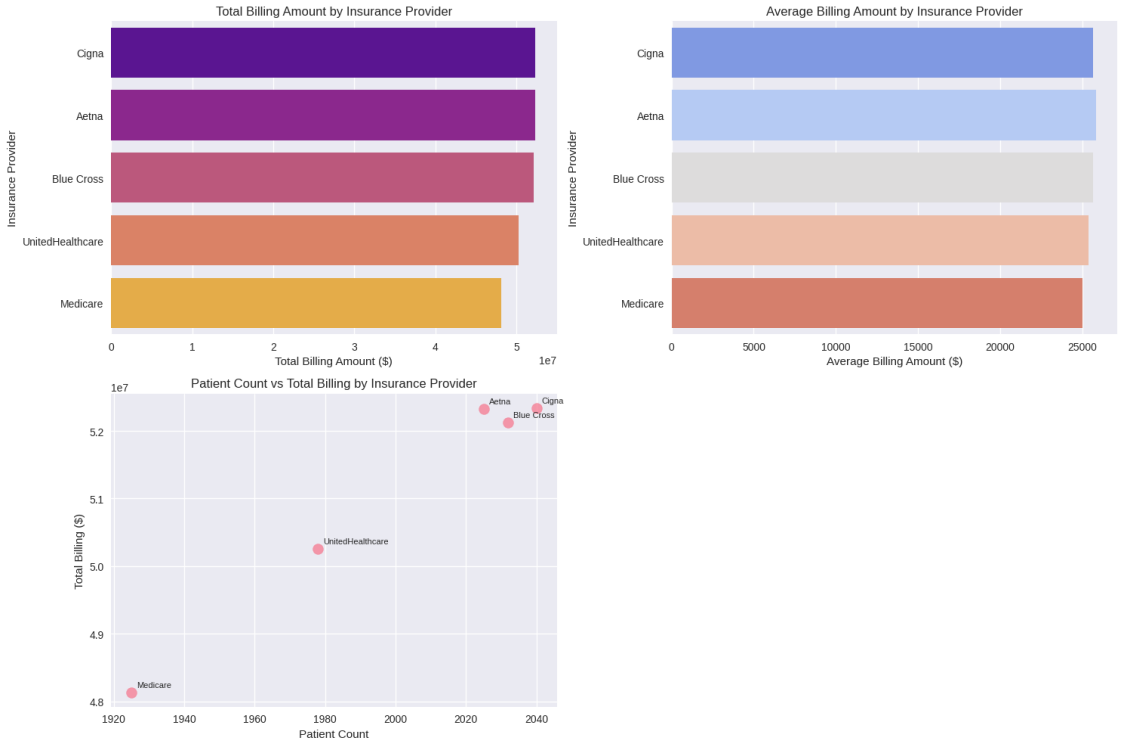



```
Correlation between Billing Amount and Days Hospitalized: -0.014
Average billing per day: $3321.20
```

16 16. Insurance Providers Covering Highest Billing Amounts

[22]:
```
# Question 17: Which insurance providers cover the highest billing amounts?
insurance_billing = df.groupby('Insurance Provider').agg({
    'Billing Amount': ['sum', 'mean', 'count'],
    'Days Hospitalized': 'mean'
}).round(2)

insurance_billing.columns = ['Total Billing', 'Average Billing', 'Patient␣
 ↪Count', 'Avg Hospitalization Days']
insurance_billing = insurance_billing.sort_values('Total Billing',␣
 ↪ascending=False)

plt.figure(figsize=(15, 10))

# Total billing by insurance
plt.subplot(2, 2, 1)
sns.barplot(x=insurance_billing['Total Billing'], y=insurance_billing.index,␣
 ↪palette='plasma')
plt.title('Total Billing Amount by Insurance Provider')
```

```
plt.xlabel('Total Billing Amount ($)')

# Average billing by insurance
plt.subplot(2, 2, 2)
sns.barplot(x=insurance_billing['Average Billing'], y=insurance_billing.index,␣
 ↪palette='coolwarm')
plt.title('Average Billing Amount by Insurance Provider')
plt.xlabel('Average Billing Amount ($)')

# Patient count vs total billing
plt.subplot(2, 2, 3)
plt.scatter(insurance_billing['Patient Count'], insurance_billing['Total␣
 ↪Billing'], s=100, alpha=0.7)
for i, provider in enumerate(insurance_billing.index):
    plt.annotate(provider, (insurance_billing['Patient Count'].iloc[i],␣
 ↪insurance_billing['Total Billing'].iloc[i]),
                 xytext=(5, 5), textcoords='offset points', fontsize=8)
plt.title('Patient Count vs Total Billing by Insurance Provider')
plt.xlabel('Patient Count')
plt.ylabel('Total Billing ($)')

plt.tight_layout()
plt.show()

print("Insurance Providers by Billing Amount:")
print(insurance_billing)
```

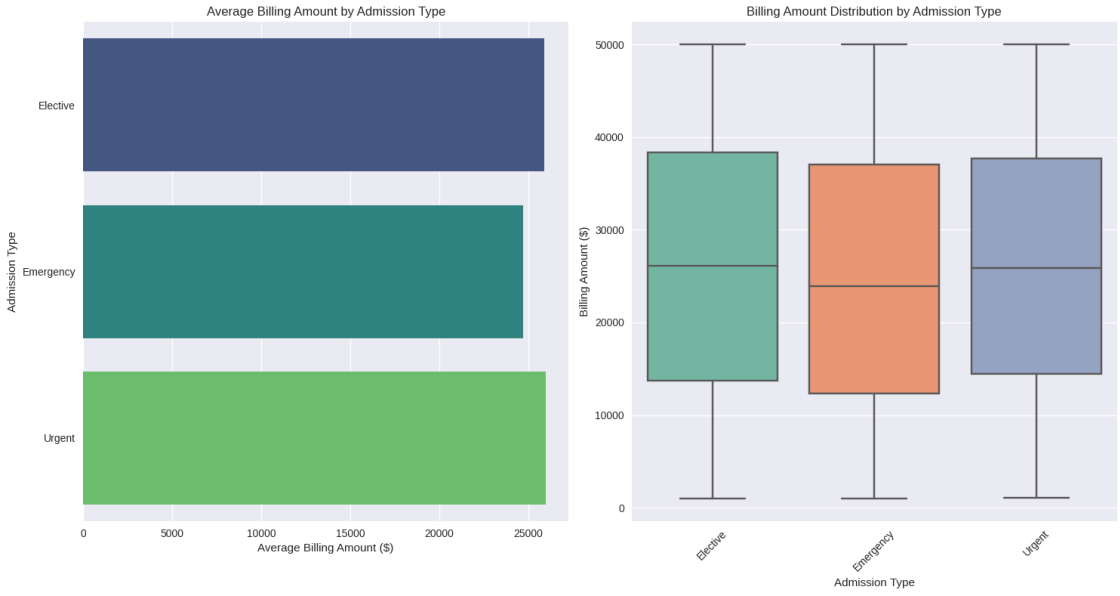



```
Insurance Providers by Billing Amount:
                    Total Billing  Average Billing  Patient Count  \
Insurance Provider
Cigna                 52340171.59         25656.95           2040
Aetna                 52321794.76         25837.92           2025
Blue Cross            52125858.90         25652.49           2032
UnitedHealthcare      50250467.70         25404.69           1978
Medicare              48129774.83         25002.48           1925

                    Avg Hospitalization Days
Insurance Provider
Cigna                                  15.48
Aetna                                  15.62
Blue Cross                             15.57
UnitedHealthcare                       15.34
Medicare                               15.80
```

17 17. Average Billing by Admission Type

[23]:
```
# Question 18: What is the average billing amount by admission type?
admission_billing = df.groupby('Admission Type').agg({
    'Billing Amount': ['mean', 'median', 'sum', 'count'],
    'Days Hospitalized': 'mean'
```

```
}).round(2)

admission_billing.columns = ['Avg Billing', 'Median Billing', 'Total Billing',␣
 ↪'Patient Count', 'Avg Hospitalization Days']

plt.figure(figsize=(15, 8))

# Average billing by admission type
plt.subplot(1, 2, 1)
sns.barplot(x=admission_billing['Avg Billing'], y=admission_billing.index,␣
 ↪palette='viridis')
plt.title('Average Billing Amount by Admission Type')
plt.xlabel('Average Billing Amount ($)')

# Box plot
plt.subplot(1, 2, 2)
sns.boxplot(data=df, x='Admission Type', y='Billing Amount', palette='Set2')
plt.title('Billing Amount Distribution by Admission Type')
plt.xlabel('Admission Type')
plt.ylabel('Billing Amount ($)')
plt.xticks(rotation=45)

plt.tight_layout()
plt.show()

print("Billing Statistics by Admission Type:")
print(admission_billing)
```

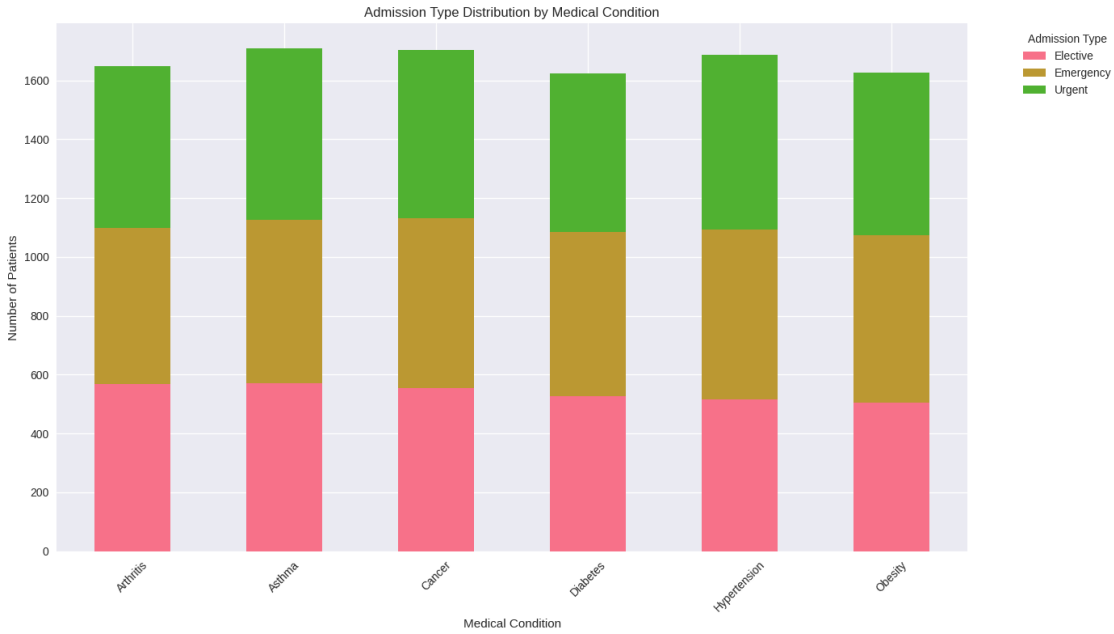


```
Billing Statistics by Admission Type:
                Avg Billing  Median Billing  Total Billing  Patient Count  \
Admission Type
Elective           25891.83        26120.77    83941321.51           3242
Emergency          24708.51        23900.17    83193559.68           3367
Urgent             25960.83        25850.43    88033186.59           3391

                Avg Hospitalization Days
Admission Type
Elective                           15.60
Emergency                          15.61
Urgent                             15.48
```

18 Admission & Stay Analysis

[24]:
```
# Question 20: What are the most common admission types for different medical conditions?
plt.figure(figsize=(16, 10))

# Admission type by condition
admission_condition = pd.crosstab(df['Medical Condition'], df['Admission Type'])

# Stacked bar chart
admission_condition.plot(kind='bar', stacked=True, figsize=(14, 8))
plt.title('Admission Type Distribution by Medical Condition')
plt.xlabel('Medical Condition')
plt.ylabel('Number of Patients')
plt.legend(title='Admission Type', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

print("Most Common Admission Type for Each Medical Condition:")
for condition in df['Medical Condition'].unique():
    common_admission = df[df['Medical Condition'] == condition]['Admission Type'].mode().iloc[0]
    count = (df[df['Medical Condition'] == condition]['Admission Type'] == common_admission).sum()
    total = len(df[df['Medical Condition'] == condition])
    percentage = (count / total) * 100
    print(f"  {condition}: {common_admission} ({percentage:.1f}%)")
```

```
<Figure size 1600x1000 with 0 Axes>
```

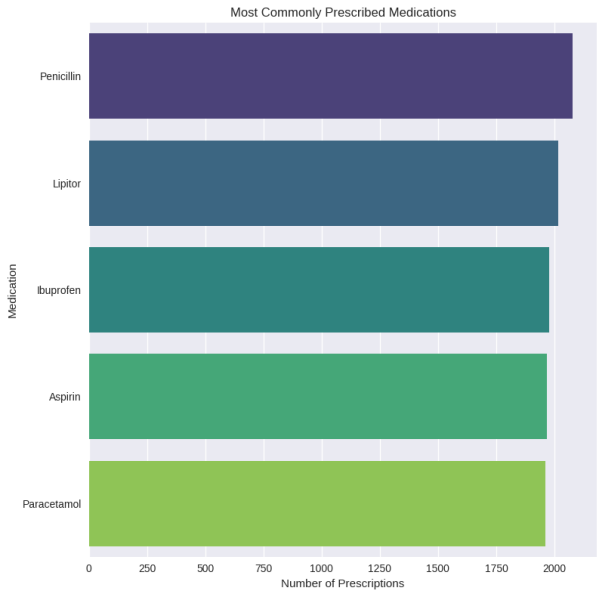


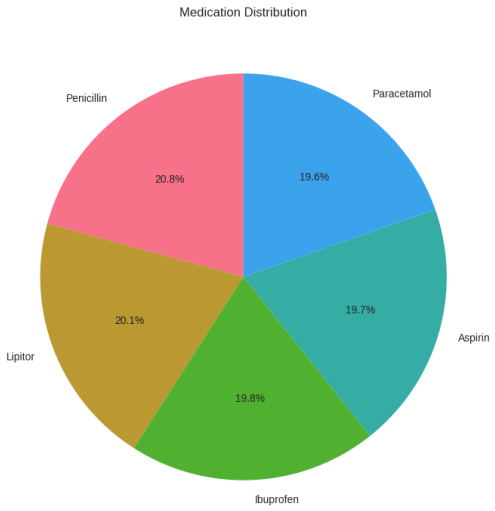

```
Most Common Admission Type for Each Medical Condition:
  Diabetes: Emergency (34.3%)
  Asthma: Urgent (34.1%)
  Obesity: Emergency (35.0%)
  Arthritis: Elective (34.5%)
  Hypertension: Urgent (35.2%)
  Cancer: Emergency (33.9%)
```

19 Medication & Treatment Analysis

[25]:
```
# Question 27: What are the most commonly prescribed medications?
plt.figure(figsize=(15, 8))

# Medication distribution
medication_counts = df['Medication'].value_counts()

# Bar chart
plt.subplot(1, 2, 1)
sns.barplot(x=medication_counts.values, y=medication_counts.index,␣
 ↪palette='viridis')
plt.title('Most Commonly Prescribed Medications')
plt.xlabel('Number of Prescriptions')

# Pie chart
plt.subplot(1, 2, 2)
```

```
plt.pie(medication_counts.values, labels=medication_counts.index, autopct='%1.
  ↪1f%%', startangle=90)
plt.title('Medication Distribution')

plt.tight_layout()
plt.show()

print("Medication Prescription Frequency:")
for medication, count in medication_counts.items():
    percentage = (count / len(df)) * 100
    print(f"  {medication}: {count} prescriptions ({percentage:.1f}%)")
```






```
Medication Prescription Frequency:
  Penicillin: 2079 prescriptions (20.8%)
  Lipitor: 2015 prescriptions (20.2%)
  Ibuprofen: 1976 prescriptions (19.8%)
  Aspirin: 1968 prescriptions (19.7%)
  Paracetamol: 1962 prescriptions (19.6%)
```

20 29. Relationship Between Medications and Test Results

[ ]:
```
# Question 29: Is there a relationship between medications and test results?
plt.figure(figsize=(15, 10))

# Test results by medication
medication_test = pd.crosstab(df['Medication'], df['Test Results'],␣
  ↪normalize='index') * 100
```

```
# Stacked bar chart
medication_test.plot(kind='bar', stacked=True, figsize=(14, 8))
plt.title('Test Results Distribution by Medication (%)')
plt.xlabel('Medication')
plt.ylabel('Percentage')
plt.legend(title='Test Results', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# Abnormal test rate by medication
abnormal_by_medication = (df[df['Test Results'] == 'Abnormal'].
 ↪groupby('Medication').size() /
                          df.groupby('Medication').size() * 100).
 ↪sort_values(ascending=False)

print("Medications with Highest Abnormal Test Results Rate:")
for medication, rate in abnormal_by_medication.head().items():
    total_patients = len(df[df['Medication'] == medication])
    print(f"  {medication}: {rate:.1f}% abnormal results ({total_patients}␣
 ↪patients)")
```

21 COMPREHENSIVE SUMMARY

[27]:
```
print("\n" + "=" * 80)
print("COMPREHENSIVE ANALYSIS SUMMARY")
print("=" * 80)

# Key findings summary
print("\nKEY FINDINGS SUMMARY:")
print(f"1. Dataset Overview: {len(df):,} patient records from {df['Date of␣
 ↪Admission'].min().strftime('%Y-%m-%d')} to {df['Date of Admission'].max().
 ↪strftime('%Y-%m-%d')}")
print(f"2. Total Billing: ${df['Billing Amount'].sum():,.2f}")
print(f"3. Most Common Condition: {df['Medical Condition'].value_counts().
 ↪index[0]} ({df['Medical Condition'].value_counts().iloc[0]} patients)")
print(f"4. Average Hospital Stay: {df['Days Hospitalized'].mean():.1f} days")
print(f"5. Gender Distribution: {df['Gender'].value_counts().to_dict()}")
print(f"6. Top Insurance Provider: {df['Insurance Provider'].value_counts().
 ↪index[0]}")
print(f"7. Most Prescribed Medication: {df['Medication'].value_counts().
 ↪index[0]}")
print(f"8. Test Results: {df['Test Results'].value_counts().to_dict()}")

# Financial insights
```

```
print("\nFINANCIAL INSIGHTS:")
print(f"• Highest average billing condition: {df.groupby('Medical␣
 ↪Condition')['Billing Amount'].mean().idxmax()} (${df.groupby('Medical␣
 ↪Condition')['Billing Amount'].mean().max():.2f})")
print(f"• Most profitable condition: {df.groupby('Medical Condition')['Billing␣
 ↪Amount'].sum().idxmax()} (${df.groupby('Medical Condition')['Billing␣
 ↪Amount'].sum().max():,.2f})")
print(f"• Hospital with highest revenue: {df.groupby('Hospital')['Billing␣
 ↪Amount'].sum().idxmax()} (${df.groupby('Hospital')['Billing Amount'].sum().
 ↪max():,.2f})")

# Operational insights
print("\nOPERATIONAL INSIGHTS:")
print(f"• Longest average stay condition: {df.groupby('Medical␣
 ↪Condition')['Days Hospitalized'].mean().idxmax()} ({df.groupby('Medical␣
 ↪Condition')['Days Hospitalized'].mean().max():.1f} days)")
print(f"• Most common admission type: {df['Admission Type'].value_counts().
 ↪index[0]} ({df['Admission Type'].value_counts().iloc[0]} cases)")
print(f"• Busiest hospital: {df['Hospital'].value_counts().idxmax()}␣
 ↪({df['Hospital'].value_counts().max()} patients)")

# Clinical insights
print("\nCLINICAL INSIGHTS:")
print(f"• Most common blood type: {df['Blood Type'].value_counts().index[0]}␣
 ↪({df['Blood Type'].value_counts().iloc[0]} patients)")
print(f"• Average patient age: {df['Age'].mean():.1f} years")

print("\n" + "=" * 80)
print("ANALYSIS COMPLETED SUCCESSFULLY!")
print("=" * 80)
```

```
================================================================================
COMPREHENSIVE ANALYSIS SUMMARY
================================================================================

KEY FINDINGS SUMMARY:
1. Dataset Overview: 10,000 patient records from 2018-10-30 to 2023-10-30
2. Total Billing: $255,168,067.78
3. Most Common Condition: Asthma (1708 patients)
4. Average Hospital Stay: 15.6 days
5. Gender Distribution: {'Female': 5075, 'Male': 4925}
6. Top Insurance Provider: Cigna
7. Most Prescribed Medication: Penicillin
8. Test Results: {'Abnormal': 3456, 'Inconclusive': 3277, 'Normal': 3267}

FINANCIAL INSIGHTS:
```

- Highest average billing condition: Diabetes ($26060.12)
- Most profitable condition: Cancer ($43,493,080.71)
- Hospital with highest revenue: Smith and Sons ($477,638.88)

OPERATIONAL INSIGHTS:
- Longest average stay condition: Arthritis (16.0 days)
- Most common admission type: Urgent (3391 cases)
- Busiest hospital: Smith PLC (19 patients)

CLINICAL INSIGHTS:
- Most common blood type: AB- (1275 patients)
- Average patient age: 51.5 years

==
ANALYSIS COMPLETED SUCCESSFULLY!
==