Hospital Patient Analytics Dashboard

OBJECTIVE - This project analyzes hospital patient data using SQL for storing patient and treatment records, Python for data cleaning and analysis, and Power BI for creating interactive dashboards. It helps hospitals track patient admissions, department workload, and recovery trends.

TOOL USED - MySQL for data storage and querying, Python for data cleaning, preprocessing and analysis and Power BI for visualization and dashboard creation.

DATASET INFO -Found the dataset from a GitHub repo:- https://github.com/mattdelaune/PowerBI_Healthcare_Dashboard.git

Every other dataset has a greater number of rows this has 1000 rows exact that will be helpful for importing the data without facing any error in SQL.

There are 2 files Found out that it is in .xlsx format also contain null value so first step I have decided to clean and preprocess the data in python and run EDA on that data. Lately import the data in SQL and run some queries based on the data followed by the final dashboarding in Power BI.

First opened the file in excel adjusted the width of the column add a table format to the columns and then converted the file type to csv and made it a duplicated data so that all cleaning and imputation can be run on this duplicated csv file without altering anything on the original data. Next Step is Data Cleaning and preprocessing in python.

HOSPITAL DATA CLEANING, PREPROCESSING AND EDA

IMPORTING LIBRARIES

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
data1 = pd.read_csv(r"C:\Users\Lenovo\Downloads\HealthCare_Dataset.csv")
print(data1)
```

```
     PatientID      PatientName  Age  Gender BloodType     Diagnosis  \
0            1    David Johnson    3   Other        A+           Flu
1            2              NaN   82   Other        A-      Covid-19
2            3   William Taylor   56   Other        B+  Hypertension
3            4    William Davis   36   Other       AB+      Covid-19
```

```
4            5     Robert Davis   78   Male        B+           Flu
..         ...              ...  ...    ...       ...           ...
995        996      Linda Lopez    3   Male        B-      Covid-19
996        997     David Martin    3  Other        A-  Hypertension
997        998              NaN   70  Other        O+      Covid-19
998        999  Joseph Martinez   37  Other        A-      Diabetes
999       1000       John Moore   10   Male        O-        Asthma

      Treatment        AdmissionDate        DischargeDate     TotalBill  \
0    Medication  2021-01-01 00:00:00  2021-01-02 00:00:00  14383.782350
1    Medication  2021-01-02 00:00:00  2021-01-03 00:00:00  15512.302210
2       Therapy  2021-01-03 00:00:00  2021-01-04 00:00:00   4039.296436
3       Therapy  2021-01-04 00:00:00  2021-01-05 00:00:00   4226.498069
4       Surgery  2021-01-05 00:00:00  2021-01-06 00:00:00   2562.768983
..          ...                  ...                  ...           ...
995     Therapy  2023-09-23 00:00:00  2023-09-24 00:00:00  18796.590760
996     Surgery  2023-09-24 00:00:00  2023-09-25 00:00:00   7505.551827
997     Surgery  2023-09-25 00:00:00  2023-09-26 00:00:00  13635.508700
998  Medication  2023-09-26 00:00:00  2023-09-27 00:00:00   6075.690059
999  Medication  2023-09-27 00:00:00  2023-09-28 00:00:00  19775.994120

                             Full Prescription Details
0    Furosemide 40mg, three times a day for 5 days;...
1    Losartan 50mg, twice a day for 7 days; Amoxici...
2    Amlodipine 5mg, twice a day for 5 days; Gabape...
3    Azithromycin 250mg, three times a day as neede...
4    Duloxetine 60mg, three times a day for 5 days;...
..                                                 ...
995  Gabapentin 300mg, once a day for 10 days; Omep...
996  Insulin Glargine 100 units/mL, once a day as n...
997  Furosemide 40mg, once a day for 5 days; Ibupro...
998  Atorvastatin 10mg, three times a day for 5 day...
999  Hydrochlorothiazide 25mg, once a day for 7 day...

[1000 rows x 11 columns]
```

```python
data2 = pd.read_csv(r"C:\Users\Lenovo\Downloads\HealthCare_Dataset2.csv")
print(data2)
```

```
     PatientID                     Hospital       DoctorName  RoomNumber  \
0            1           Riverside Hospital     Joseph Lopez         178
1            2  Green Valley Medical Center      James Moore         368
2            3           Riverside Hospital    Michael Lopez         260
3            4           Cedar Sinai Clinic  Linda Rodriguez         228
4            5           Riverside Hospital   Mary Hernandez         167
..         ...                          ...              ...         ...
995        996      Silver Oak Medical Plaza   Charles Martin         438
996        997  Green Valley Medical Center     Linda Martin         255
997        998           Cedar Sinai Clinic      Mary Martin         351
998        999      Silver Oak Medical Plaza  James Martinez         142
```

```
999       1000     Silver Oak Medical Plaza    Barbara Martin         260

       DailyCost      TreatmentType  RecoveryRating
0     359.006021            Surgery            10.0
1     933.915694            Surgery             4.0
2    1272.088112         Counseling             NaN
3     402.609932         Counseling             3.0
4     483.129350   Physical Therapy             NaN
..           ...                ...             ...
995  1625.316045            Surgery             7.0
996   348.339523   Physical Therapy             8.0
997  1485.272908   Physical Therapy             5.0
998  1630.479191            Surgery             3.0
999  1759.963492   Physical Therapy             1.0

[1000 rows x 7 columns]
```

DATA PROFILING

```
data1.columns
```

```
Index(['PatientID', 'PatientName', 'Age', 'Gender', 'BloodType', 'Diagnosis',
       'Treatment', 'AdmissionDate', 'DischargeDate', 'TotalBill',
       'Full Prescription Details'],
      dtype='object')
```

```
data2.columns
```

```
Index(['PatientID', 'Hospital', 'DoctorName', 'RoomNumber', 'DailyCost',
       'TreatmentType', 'RecoveryRating'],
      dtype='object')
```

```
total_columns = data1.shape[1] , data2.shape[1]
total_columns
```

```
(11, 7)
```

```
total_rows = data1.shape[0] , data2.shape[0]
total_rows
```

```
(1000, 1000)
```

AS YOU CAN SEE DATA1 HAS 11 COLUMNS AND DATA2 HAS 7 BUT BOTH OF THEM CONSIST OF 1000 ROWS

```
data1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 11 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
```

```
 0   PatientID                  1000 non-null   int64
 1   PatientName                940 non-null    object
 2   Age                        1000 non-null   int64
 3   Gender                     1000 non-null   object
 4   BloodType                  1000 non-null   object
 5   Diagnosis                  1000 non-null   object
 6   Treatment                  1000 non-null   object
 7   AdmissionDate              1000 non-null   object
 8   DischargeDate              1000 non-null   object
 9   TotalBill                  940 non-null    float64
 10  Full Prescription Details  1000 non-null   object
dtypes: float64(1), int64(2), object(8)
memory usage: 86.1+ KB
```

```
data1.isnull().sum()
```

```
PatientID                     0
PatientName                  60
Age                           0
Gender                        0
BloodType                     0
Diagnosis                     0
Treatment                     0
AdmissionDate                 0
DischargeDate                 0
TotalBill                    60
Full Prescription Details     0
dtype: int64
```

```
miss= data1.isnull().sum()
miss
```

```
PatientID                     0
PatientName                  60
Age                           0
Gender                        0
BloodType                     0
Diagnosis                     0
Treatment                     0
AdmissionDate                 0
DischargeDate                 0
TotalBill                    60
Full Prescription Details     0
dtype: int64
```

```
miss_percentage = (data1.isnull().sum()/len(data1))* 100
miss_percentage
```

```
PatientID                    0.0
PatientName                  6.0
Age                          0.0
```

```
Gender                         0.0
BloodType                      0.0
Diagnosis                      0.0
Treatment                      0.0
AdmissionDate                  0.0
DischargeDate                  0.0
TotalBill                      6.0
Full Prescription Details      0.0
dtype: float64
```

PatientName has 6% missing value which can be treated as unknown as name is a critical column we just can't directly remove the blanks as rest oof the data are present also TotalBill got 6% null value which will be imputated in the later stage

```
data2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   PatientID       1000 non-null   int64
 1   Hospital        922 non-null    object
 2   DoctorName      1000 non-null   object
 3   RoomNumber      1000 non-null   int64
 4   DailyCost       1000 non-null   float64
 5   TreatmentType   1000 non-null   object
 6   RecoveryRating  922 non-null    float64
dtypes: float64(2), int64(2), object(3)
memory usage: 54.8+ KB
```

```
data2.isnull().sum()
```

```
PatientID          0
Hospital          78
DoctorName         0
RoomNumber         0
DailyCost          0
TreatmentType      0
RecoveryRating    78
dtype: int64
```

```
miss1= data2.isnull().sum()
miss1
```

```
PatientID          0
Hospital          78
DoctorName         0
RoomNumber         0
DailyCost          0
TreatmentType      0
```

```
RecoveryRating    78
dtype: int64
```

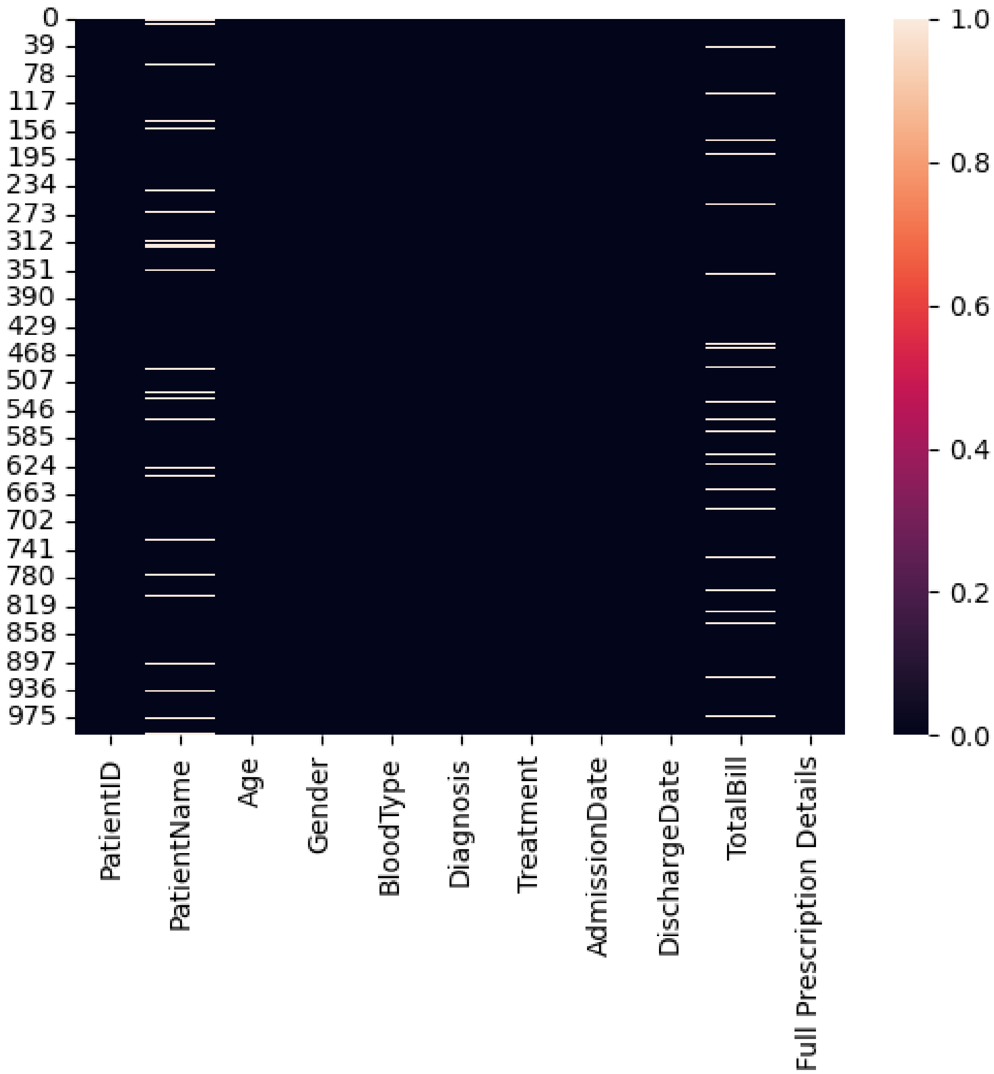
```
miss_percentage1 = (data2.isnull().sum()/len(data2))* 100
miss_percentage1
```

```
PatientID         0.0
Hospital          7.8
DoctorName        0.0
RoomNumber        0.0
DailyCost         0.0
TreatmentType     0.0
RecoveryRating    7.8
dtype: float64
```

Just Like in data1 data2 also got few null value about 7.8% in columns Hospital and RecoveryRating Hospital column blanks cannot be omitted as it is one of the primary column out there. So unknown will be filled in place of blanks. For RecoveryRating we are going to find the average of it and fill accordingly

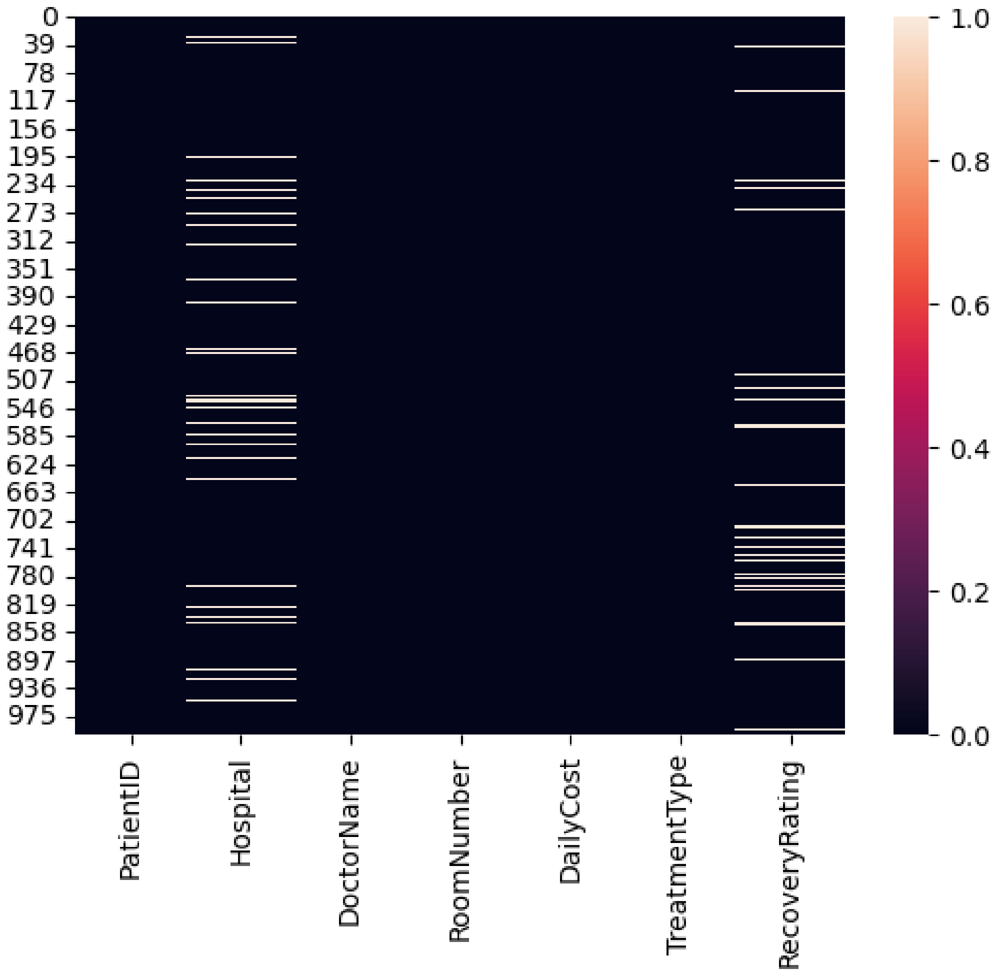
```
sns.heatmap(data1.isnull())
```

```
<Axes: >
```




```
sns.heatmap(data2.isnull())
```

```
<Axes: >
```



By this it is clearly been seen that the rows that is blank in hospital is not the same blanked rows in recoverrating likewise for patient name and total bill. So we need to deal with both individually and assess different technique for imputations

DATA CLEANING AND PRE PROCESSING

```
data1.drop('Full Prescription Details', axis=1,inplace=True)
```

```
data1.columns
```

```
Index(['PatientID', 'PatientName', 'Age', 'Gender', 'BloodType', 'Diagnosis',
       'Treatment', 'AdmissionDate', 'DischargeDate', 'TotalBill'],
      dtype='object')
```

DROPPED the Description column to reduce the data redundancy as it was mainly a string column not so useful for analysis

```
data1.shape
```

```
(1000, 10)
```

```
data1['PatientName'] = data1['PatientName'].fillna('unknown')
```

```
data1['PatientName'].value_counts()
```

```
PatientName
unknown                60
Patricia Hernandez      7
Robert Rodriguez        7
Elizabeth Lopez         7
Jessica Rodriguez       7
                       ..
Sarah Jackson           1
James Thomas            1
Mary Gonzalez           1
Michael Moore           1
Joseph Martinez         1
Name: count, Length: 369, dtype: int64
```

MISSING Patient Name has been treated as per possible. It may not be the best way but still better than having a blank in the selection

```
data2.columns
```

```
Index(['PatientID', 'Hospital', 'DoctorName', 'RoomNumber', 'DailyCost',
       'TreatmentType', 'RecoveryRating'],
      dtype='object')
```

```
data2['Hospital'] = data2['Hospital'].fillna('n/a')
```

```
data2['Hospital'].value_counts()
```

```
Hospital
Green Valley Medical Center    207
Silver Oak Medical Plaza       185
Cedar Sinai Clinic             184
Maple Grove Health Facility    178
Riverside Hospital             168
n/a                             78
Name: count, dtype: int64
```

Dealing with duplicate rows Finding number of duplicate rows in the datasets then Droping the duplicate entries from the dataset.

```
data1[data1.duplicated()]
```

```
Empty DataFrame
Columns: [PatientID, PatientName, Age, Gender, BloodType, Diagnosis,
Treatment, AdmissionDate, DischargeDate, TotalBill]
Index: []
```

```
data1.duplicated().sum()
```

```
0
```

```
data2[data2.duplicated()]
```

```
Empty DataFrame
Columns: [PatientID, Hospital, DoctorName, RoomNumber, DailyCost,
TreatmentType, RecoveryRating]
Index: []
```

```
data2.duplicated().sum()
```

```
0
```

NO DUPLICATES FOUND

```
data1.describe()
```

```
         PatientID          Age     TotalBill
count  1000.000000  1000.000000    940.000000
mean    500.500000    50.500000  10038.866970
std     288.819436    28.599859   5801.795268
min       1.000000     0.000000    200.928022
25%     250.750000    26.000000   4883.315196
50%     500.500000    51.500000  10152.880440
75%     750.250000    75.000000  14872.452167
max    1000.000000    99.000000  19979.201530
```

```
data1.groupby('Diagnosis')['TotalBill'].mean()
```

```
Diagnosis
Asthma           9779.536904
Covid-19        10581.567618
Diabetes         9469.119830
Flu              9771.535403
Hypertension    10615.723800
Name: TotalBill, dtype: float64
```

```
data1.groupby('Treatment')['TotalBill'].mean()
```

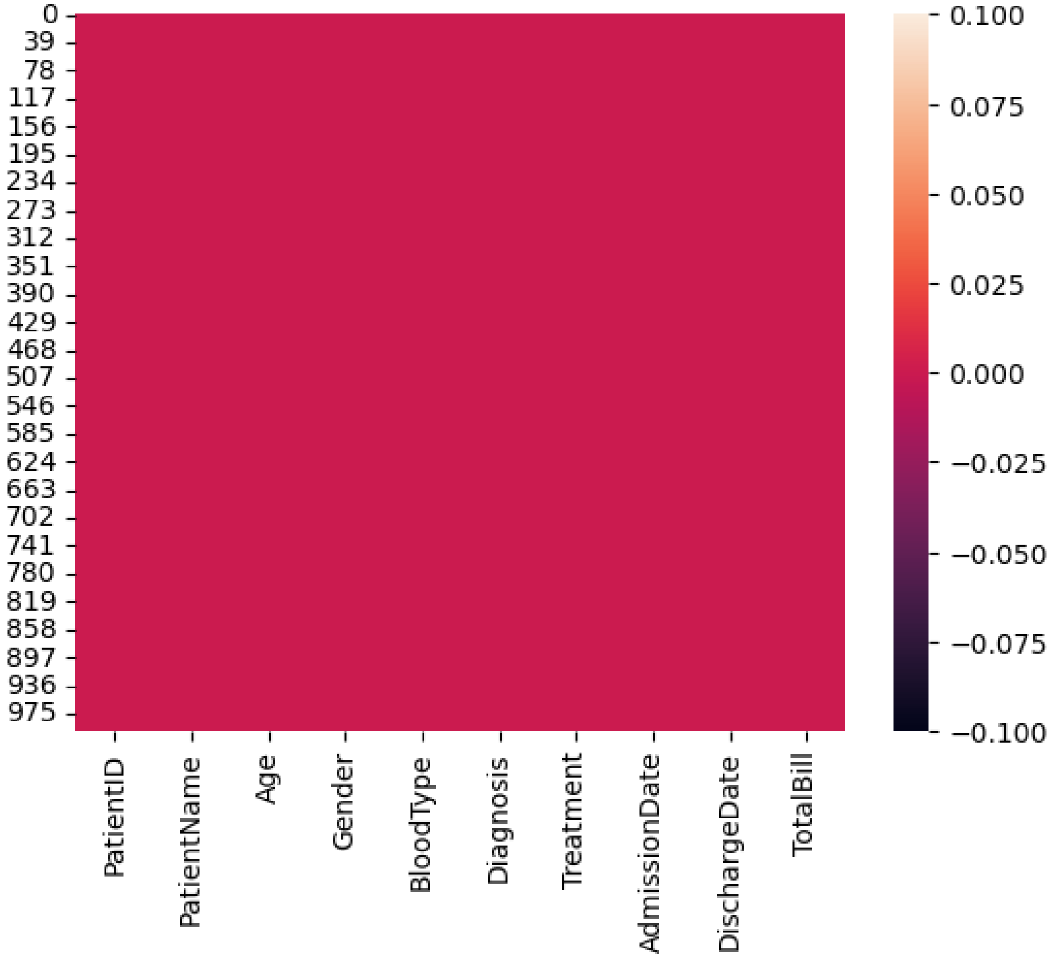
```
Treatment
Medication    10204.701944
Surgery        9865.776236
Therapy       10026.439343
Name: TotalBill, dtype: float64
```

```
data1['TotalBill'] = data1.groupby('Treatment')['TotalBill'].transform(lambda
x: x.fillna(x.mean()))
```

Filled the blank value in total bill with their respective treatment mean.It distributed the mean evenly based on their treatment for which they have visited the hospital

```
sns.heatmap(data1.isnull())
```

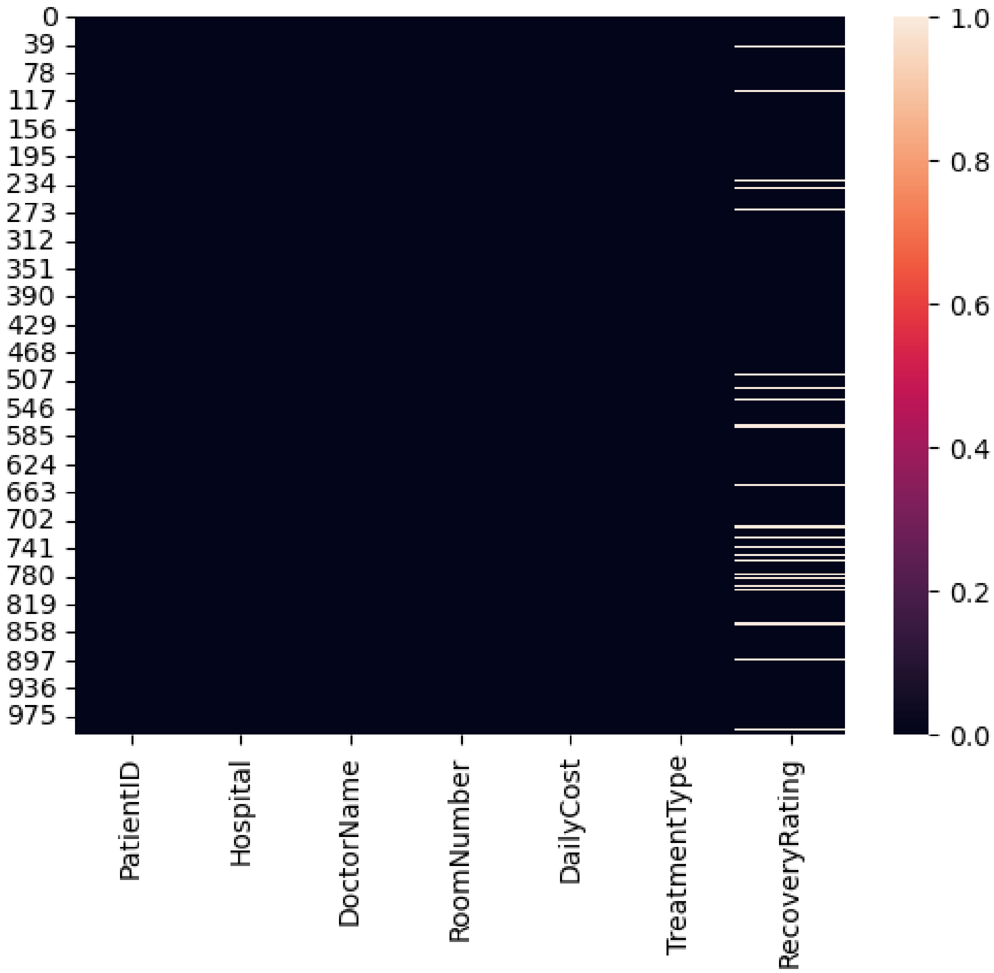
```
<Axes: >
```



Data1 is cleaned and standardized

```
sns.heatmap(data2.isnull())
```

```
<Axes: >
```



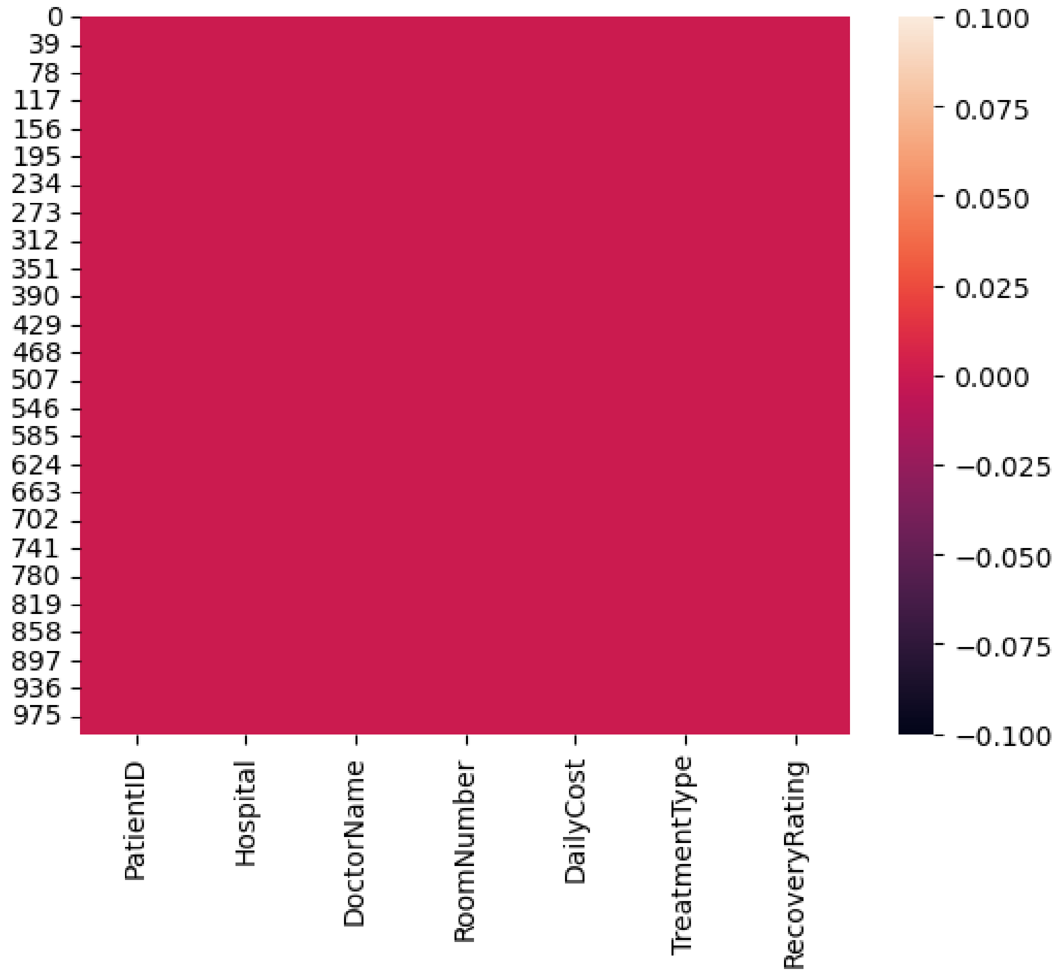

```
data2['RecoveryRating'].mean()
```

5.436008676789588

```
mean_recovery_rating = round(data2['RecoveryRating'].mean())
data2['RecoveryRating'] =
data2['RecoveryRating'].fillna(mean_recovery_rating)
```

```
sns.heatmap(data2.isnull())
```

<Axes: >



Data2 is cleared and standardized

```
data1.to_csv('hospital1_clean.csv')
```

```
data2.to_csv('hospital2_clean.csv')
```

EDA

```
data1.dtypes
```

```
PatientID           int64
PatientName        object
Age                 int64
Gender             object
BloodType          object
Diagnosis          object
Treatment          object
AdmissionDate      object
DischargeDate      object
TotalBill         float64
dtype: object
```

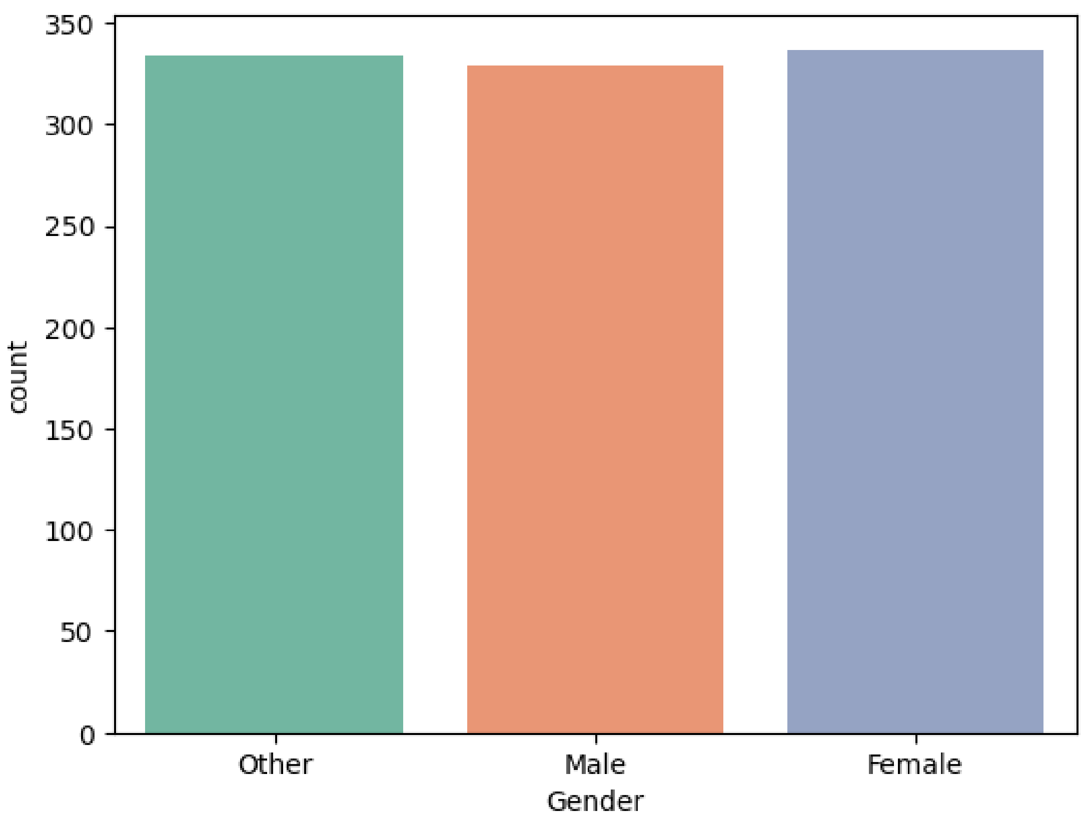
```
data1['AdmissionDate'] = pd.to_datetime(data1['AdmissionDate'])
data1['DischargeDate'] = pd.to_datetime(data1['DischargeDate'])
```

```
data2.dtypes
```

```
PatientID            int64
Hospital            object
DoctorName          object
RoomNumber           int64
DailyCost          float64
TreatmentType       object
RecoveryRating     float64
dtype: object
```

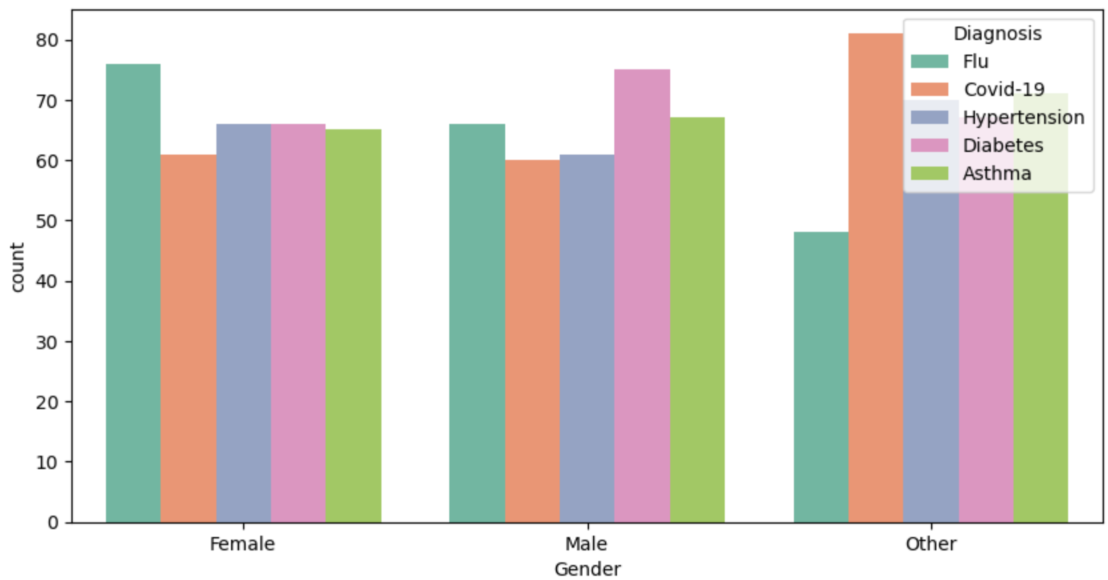
```
sns.countplot(x='Gender',data=data1,palette='Set2')
plt.show()
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_21484\1666141477.py:1:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

  sns.countplot(x='Gender',data=data1,palette='Set2')
```



GENDER IS ALMOST SAME FOR ALL TYPE

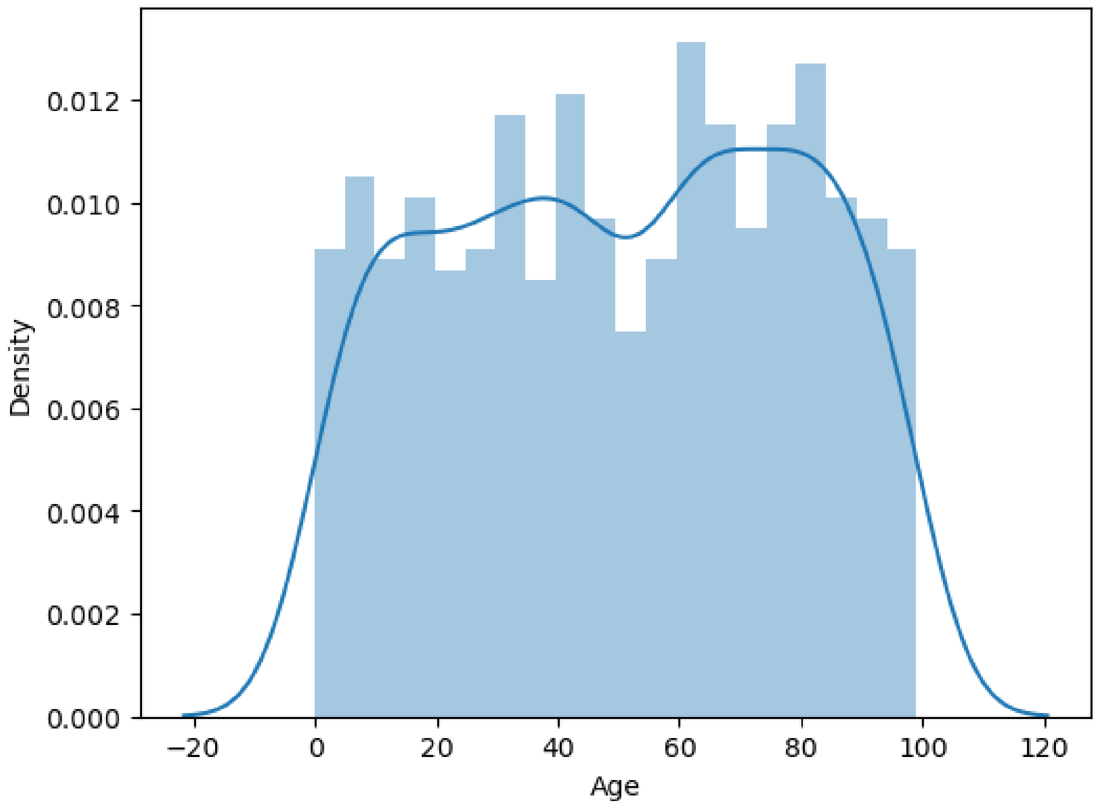
```
plt.figure(figsize=(10, 5))
sns.countplot(x='Gender',hue='Diagnosis',data=data1,palette='Set2')
plt.xticks([0,1,2],['Female','Male','Other'])
plt.show()
```



INSIGHTS:-FOUND SOME INTERESTING INSIGHTS WHICH WILL BE FURTHER DISCUSSED IN THE LATER PART OF THE PROJECT

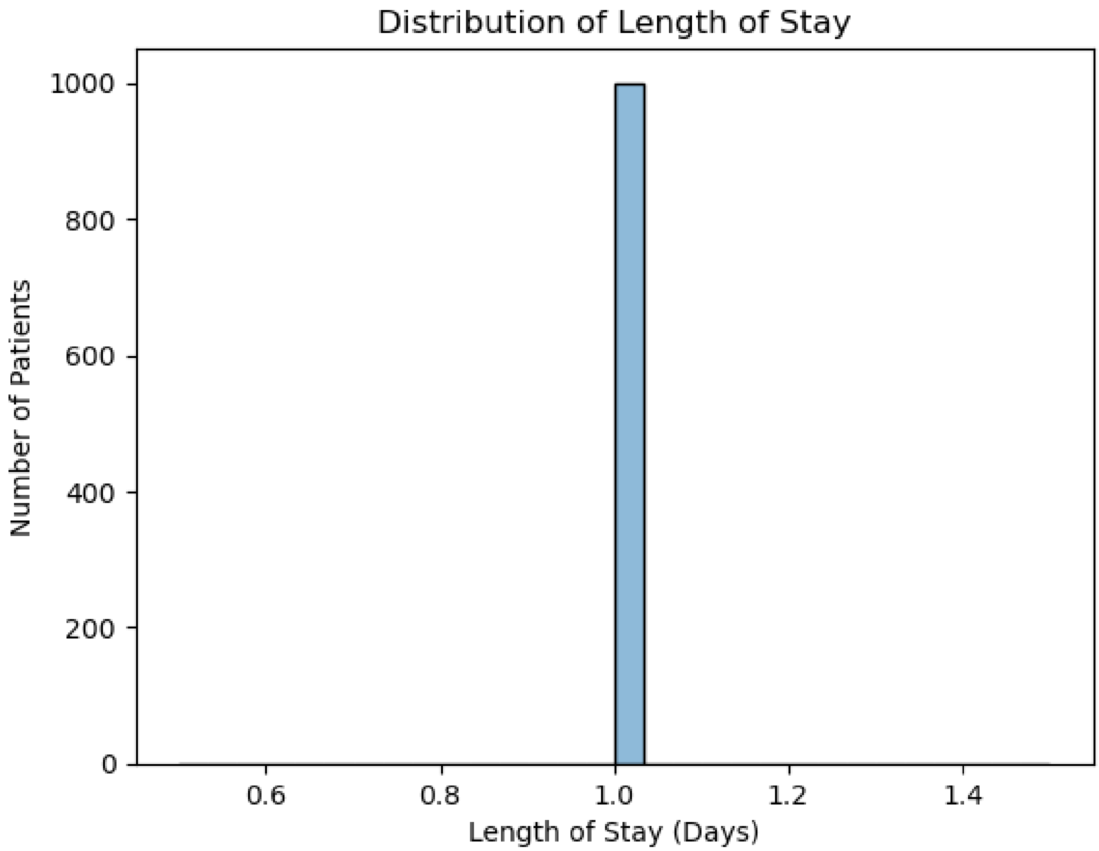
```
sns.distplot(data1['Age'],bins=20)
plt.show()
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_21484\1951830225.py:1: 
UserWarning: 

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with 
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data1['Age'],bins=20)
```



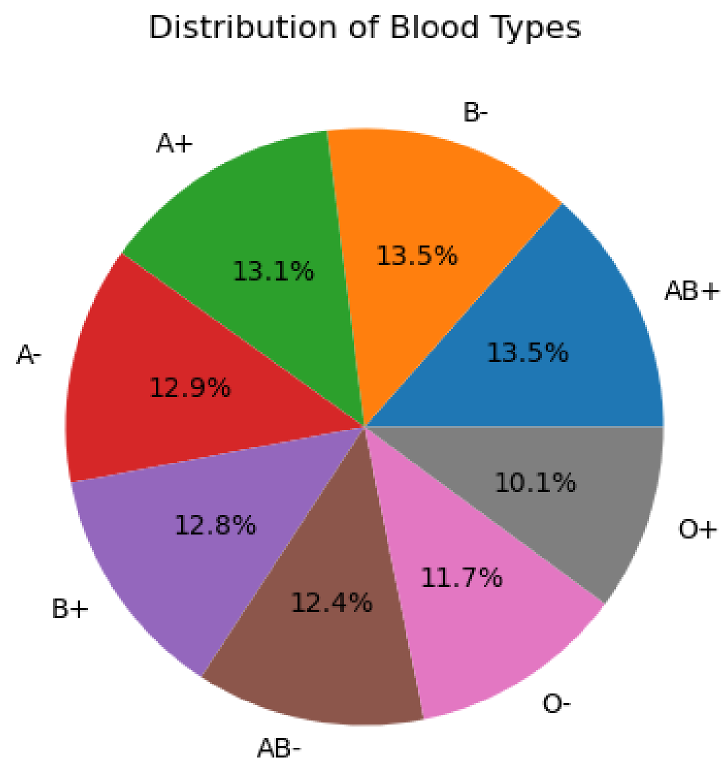
DATA HAS NO OUTLIER IN THE AGE

```
data1['LengthOfStay'] = (data1['DischargeDate'] -
data1['AdmissionDate']).dt.days
sns.histplot(data1['LengthOfStay'], bins=30, kde=True)
plt.title('Distribution of Length of Stay')
plt.xlabel('Length of Stay (Days)')
plt.ylabel('Number of Patients')
plt.show()
```



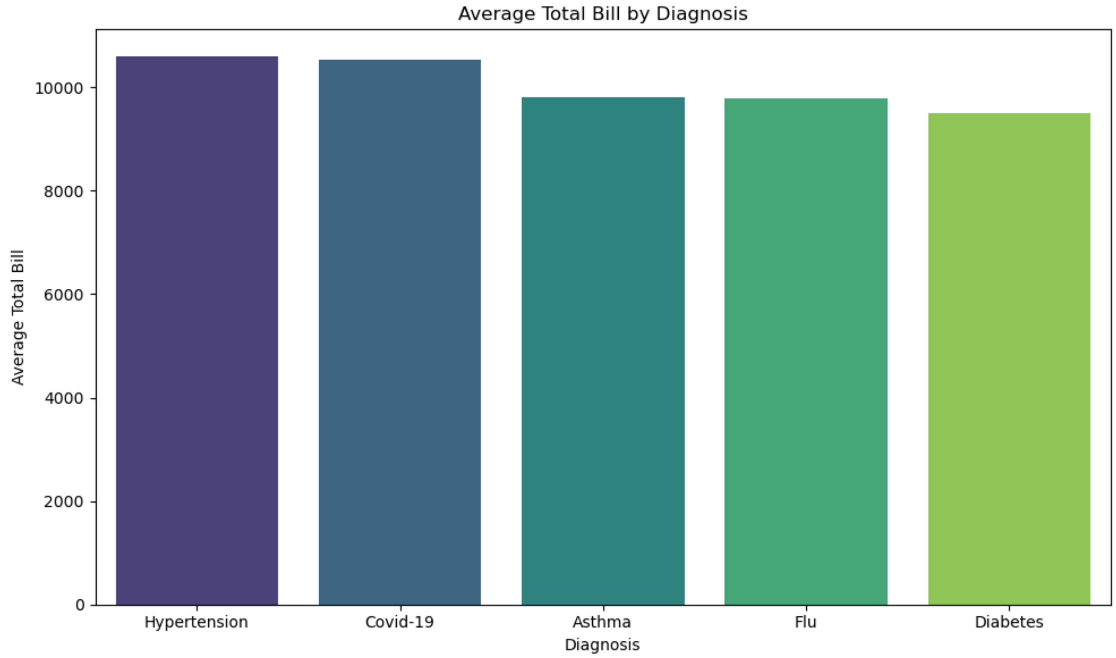
SEEMS THAT THIS DATASET IS A MADEUP DATASET IT IS SHOWING EVERY 1000 PATIENT STAYED ONLY FOR 1 DAY THIS CLARIFIES IT IS FICTIONAL

```
blood_type_counts = data1['BloodType'].value_counts()
plt.figure(figsize=(5, 5))
plt.pie(blood_type_counts, labels=blood_type_counts.index, autopct='%1.1f%%')
plt.title('Distribution of Blood Types')
plt.show()
```



NO SUCH SIGNIFICANT INSIGHTS FOUND

```
avg_bill_by_diagnosis =
data1.groupby('Diagnosis')['TotalBill'].mean().sort_values(ascending=False)
plt.figure(figsize=(10, 6))
sns.barplot(x=avg_bill_by_diagnosis.index, y=avg_bill_by_diagnosis.values,
hue=avg_bill_by_diagnosis.index, palette='viridis', legend=False)
plt.title('Average Total Bill by Diagnosis')
plt.xlabel('Diagnosis')
plt.ylabel('Average Total Bill')
plt.tight_layout()
plt.show()
```



INSIGHTS:- HYPERTENSION AND COVID-19 GENERATE A LOT OF REVENUE

```
data2.dtypes
```

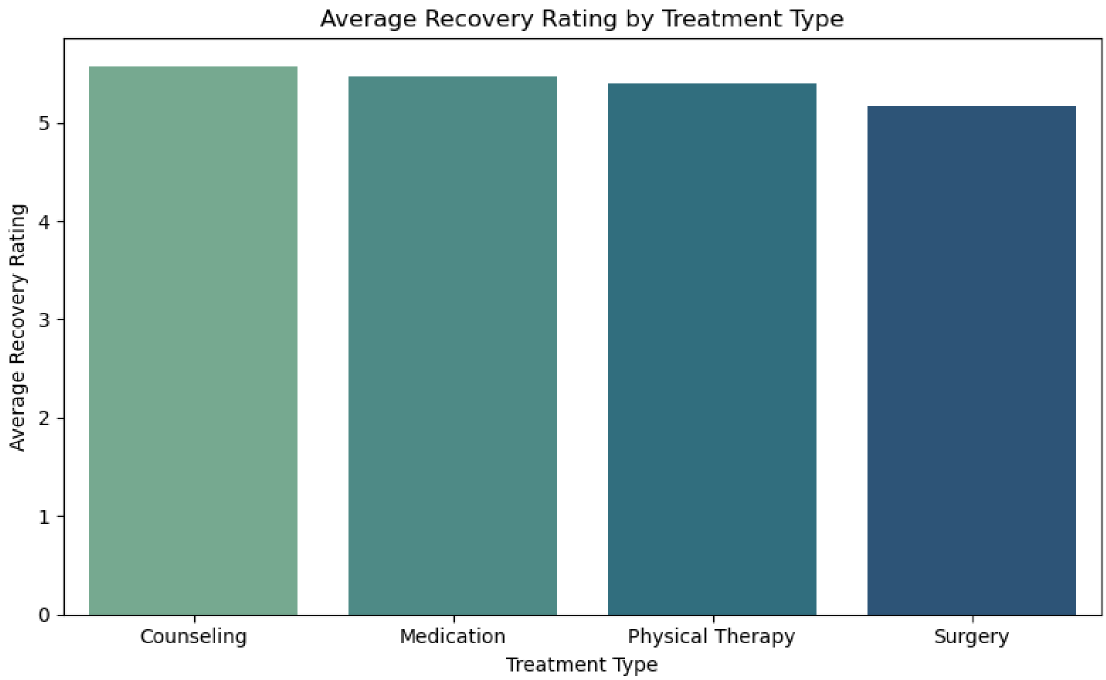
```
PatientID            int64
Hospital            object
DoctorName          object
RoomNumber           int64
DailyCost          float64
TreatmentType       object
RecoveryRating     float64
dtype: object
```

```
data2.head(3)
```

```
   PatientID                    Hospital     DoctorName  RoomNumber  \
0          1          Riverside Hospital   Joseph Lopez         178
1          2  Green Valley Medical Center    James Moore         368
2          3          Riverside Hospital  Michael Lopez         260

     DailyCost TreatmentType  RecoveryRating
0   359.006021       Surgery            10.0
1   933.915694       Surgery             4.0
2  1272.088112    Counseling             5.0
```

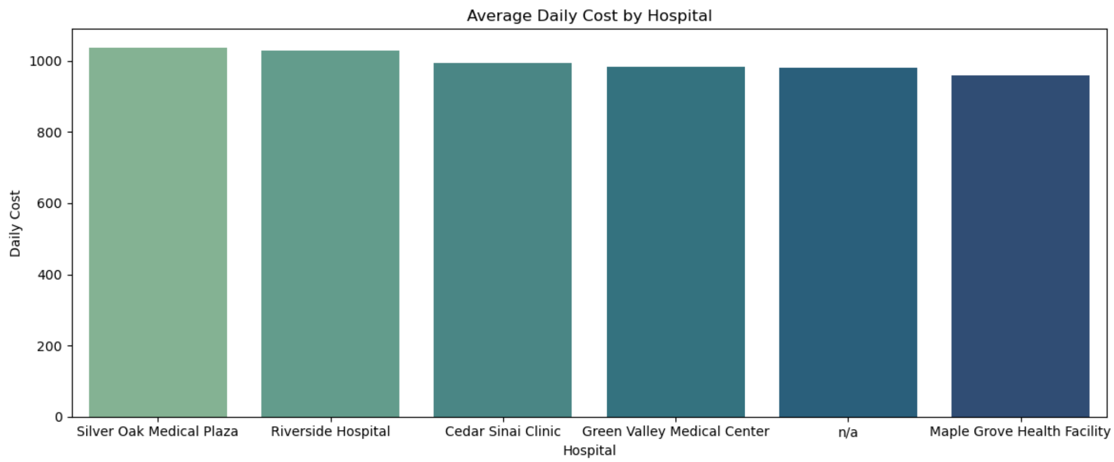
```
avg_recovery_by_treatment =
data2.groupby('TreatmentType')['RecoveryRating'].mean().sort_values(ascending
=False)
plt.figure(figsize=(8, 5))
sns.barplot(x=avg_recovery_by_treatment.index,
y=avg_recovery_by_treatment.values, hue=avg_recovery_by_treatment.index,
```

```
palette='crest', legend=False)
plt.title('Average Recovery Rating by Treatment Type')
plt.xlabel('Treatment Type')
plt.ylabel('Average Recovery Rating')
plt.tight_layout()
plt.show()
```



INSIGHTS:- COUNSELING HAS THE BEST AVERAGE RATING COMPARED TO THE REST

```
avg_dailycost_by_hospital =
data2.groupby('Hospital')['DailyCost'].mean().sort_values(ascending=False)
plt.figure(figsize=(12, 5))
sns.barplot(x=avg_dailycost_by_hospital.index,
y=avg_dailycost_by_hospital.values, hue=avg_dailycost_by_hospital.index,
palette='crest', legend=False)
plt.title('Average Daily Cost by Hospital')
plt.xlabel('Hospital')
plt.ylabel('Daily Cost')
plt.tight_layout()
plt.show()
```



INSIGHTS:- SILVER OAK MEDICAL PLAZA IS THE MOST EXPENSIVE WHEN IT COMES TO DAILY BILL

After this next step is in SQL (querying the data)

CREATE DATABASE hospital;

USE hospital;

SELECT * FROM hospital1;

ALTER TABLE hospital1

DROP COLUMN MyUnknownColumn;

SELECT * FROM hospital2;

ALTER TABLE hospital2

DROP COLUMN MyUnknownColumn;

ALTER TABLE hospital1 ADD PRIMARY KEY (PatientId);

ALTER TABLE hospital2

ADD CONSTRAINT fk_hospital2_patientid

FOREIGN KEY (PatientId) REFERENCES hospital1 (PatientId);

-- QUERIES

-- 1. List all patient names, age, gender, diagnosis, and total bill from hospital1

SELECT PatientName, Age, Gender, Diagnosis, TotalBill

FROM hospital1;

-- 2. Display patient name, blood type, treatment, and admission date for patients whose total bill > 5000

SELECT PatientName, BloodType, Treatment, AdmissionDate

```sql
FROM hospital1
WHERE TotalBill > 5000;
-- INSIGHTS:- 761 PATIENT TOTAL BILL WENT MORE THAN 5000
-- 3. Show patient name, age, and gender for patients diagnosed with 'Covid-19'
SELECT PatientName, Age, Gender
FROM hospital1
WHERE Diagnosis = 'Covid-19';
-- INSIGHTS :- 202 PATIENT HAS BEEN ADMITTED DUE TO BEING AFFECTED BY COVID-19
-- 4. Display patient name, hospital name, doctor name, and room number for every patient
SELECT h1.PatientName, h2.Hospital, h2.DoctorName, h2.RoomNumber
FROM hospital1 h1
INNER JOIN hospital2 h2 ON h1.PatientID = h2.PatientID;
-- 5. List patient name, diagnosis, treatment type, and recovery rating
SELECT h1.PatientName, h1.Diagnosis, h2.TreatmentType, h2.RecoveryRating
FROM hospital1 h1
INNER JOIN hospital2 h2 ON h1.PatientID = h2.PatientID;
-- 6. Show patient name, doctor name, treatment, and daily cost
SELECT h1.PatientName, h2.DoctorName, h1.Treatment, h2.DailyCost
FROM hospital1 h1
INNER JOIN hospital2 h2 ON h1.PatientID = h2.PatientID;
-- 7. Display patient name, room number, admission date, discharge date, and total bill
SELECT h1.PatientName, h2.RoomNumber, h1.AdmissionDate, h1.DischargeDate, h1.TotalBill
FROM hospital1 h1
INNER JOIN hospital2 h2 ON h1.PatientID = h2.PatientID;
-- 8. Patients treated in 'Riverside Hospital' hospital with doctor name and diagnosis
SELECT h1.PatientName, h2.DoctorName, h1.Diagnosis
FROM hospital1 h1
INNER JOIN hospital2 h2 ON h1.PatientID = h2.PatientID
WHERE h2.Hospital = 'Riverside Hospital';
-- INSIGHTS:- 168 PATIENTS ARE ADMITTED IN RIVERSIDE HOSPITAL
```

```
-- 9. Patient name, doctor name, recovery rating > 8

SELECT h1.PatientName, h2.DoctorName, h2.RecoveryRating

FROM hospital1 h1

INNER JOIN hospital2 h2 ON h1.PatientID = h2.PatientID

WHERE h2.RecoveryRating > 8;

-- INSIGHTS:- 190 PATIENT GOT RECOVERY RATING MORE THAN 8

-- 10. Patient name, hospital, total bill where DailyCost > 1500

SELECT h1.PatientName, h2.Hospital, h2.DailyCost

FROM hospital1 h1

INNER JOIN hospital2 h2 ON h1.PatientID = h2.PatientID

WHERE h2.DailyCost > 1500;

-- INSIGHTS:- 227 PATIENT HAS MORE THAN 1500 DAILYCOST

-- 11. Female patients: name, gender, diagnosis, doctor name

SELECT h1.PatientName, h1.Gender, h1.Diagnosis, h2.DoctorName

FROM hospital1 h1

INNER JOIN hospital2 h2 ON h1.PatientID = h2.PatientID

WHERE h1.Gender = 'Female';

-- 12. Total bill per hospital (in descending order)

SELECT h2.Hospital, ROUND(SUM(h1.TotalBill),2) AS TotalBillSum

FROM hospital1 h1

INNER JOIN hospital2 h2 ON h1.PatientID = h2.PatientID

GROUP BY h2.Hospital

ORDER BY TotalBillSum DESC;

-- INSIGHTS:- GREEN VALLEY MEDICAL CENTER GENERATES THE MOST MONEY FOLLOWED BY

-- CEDAR SINAI CLINIC AND MAPLE GROVE HEALTH FACILITY

-- 13. Rank doctors by number of patients treated (most patients = rank 1)

SELECT h2.DoctorName,

        COUNT(*) AS PatientCount,

        DENSE_RANK() OVER (ORDER BY COUNT(*) DESC) AS DoctorRank

FROM hospital1 h1
```

```sql
INNER JOIN hospital2 h2 ON h1.PatientID = h2.PatientID

GROUP BY h2.DoctorName;

-- INSIGHTS:- DR.DAVID MOORE GOT THE MOST PATIENT COUNT FOLLOWED BY MICHAEL THOMAS

-- JENNIFER JOHNSON AND PATRICIA WILSON

-- 14. Average daily cost per hospital (in descending order)

SELECT h2.Hospital, ROUND(AVG(h2.DailyCost),2) AS AvgDailyCost

FROM hospital1 h1

INNER JOIN hospital2 h2 ON h1.PatientID = h2.PatientID

GROUP BY h2.Hospital

ORDER BY AvgDailyCost DESC;

-- INSIGHTS:- SILVER OAK MEDICAL PLAZA GENERATED THE HIGHEST AVERAGE DAILYCOST

-- 15. Max, min, avg recovery rating per treatment type

SELECT h2.TreatmentType, MAX(h2.RecoveryRating) AS MaxRating,MIN(h2.RecoveryRating) AS MinRating,

ROUND(AVG(h2.RecoveryRating),1) AS AvgRating

FROM hospital1 h1

INNER JOIN hospital2 h2 ON h1.PatientID = h2.PatientID

GROUP BY h2.TreatmentType;

-- 16. Top 5 highest total bills with hospital

SELECT h1.PatientName, h2.Hospital, h1.TotalBill

FROM hospital1 h1

INNER JOIN hospital2 h2 ON h1.PatientID = h2.PatientID

ORDER BY h1.TotalBill DESC

LIMIT 5;

-- INSIGHTS:- MAPLE GROVE HEALTH FACILITY TOPS THE LIST

-- 17. Top 3 patients by total bill within each hospital

SELECT PatientName, Hospital, TotalBill, BillRank

FROM (

    SELECT h1.PatientName, h2.Hospital, h1.TotalBill,

            RANK() OVER (PARTITION BY h2.Hospital ORDER BY h1.TotalBill DESC) AS BillRank
```

```sql
    FROM hospital1 h1
    INNER JOIN hospital2 h2 ON h1.PatientID = h2.PatientID
) ranked_patients
WHERE BillRank <= 3
ORDER BY Hospital, BillRank;
-- 18. Categorize patients by age group
SELECT PatientName, Age, Gender,
       CASE
           WHEN Age < 18 THEN 'Minor'
           WHEN Age BETWEEN 18 AND 60 THEN 'Adult'
           ELSE 'Senior'
       END AS AgeGroup
FROM hospital1;
-- 19. Categorize total bill amounts
SELECT PatientName, TotalBill,
       CASE
           WHEN TotalBill < 5000 THEN 'Low Cost'
           WHEN TotalBill BETWEEN 5000 AND 15000 THEN 'Medium Cost'
           ELSE 'High Cost'
       END AS BillCategory
FROM hospital1;
-- 20. Recovery rating performance labels per doctor
SELECT h1.PatientName, h2.DoctorName, h2.RecoveryRating,
       CASE
           WHEN h2.RecoveryRating >= 9 THEN 'Excellent'
           WHEN h2.RecoveryRating >= 7 THEN 'Good'
           WHEN h2.RecoveryRating >= 5 THEN 'Average'
           ELSE 'Poor'
       END AS RecoveryLabel
FROM hospital1 h1
```

INNER JOIN hospital2 h2 ON h1.PatientID = h2.PatientID;

Last step importing the dataset in Power BI and creating a dashboard to retrieve some insights follow-up by recommendation.

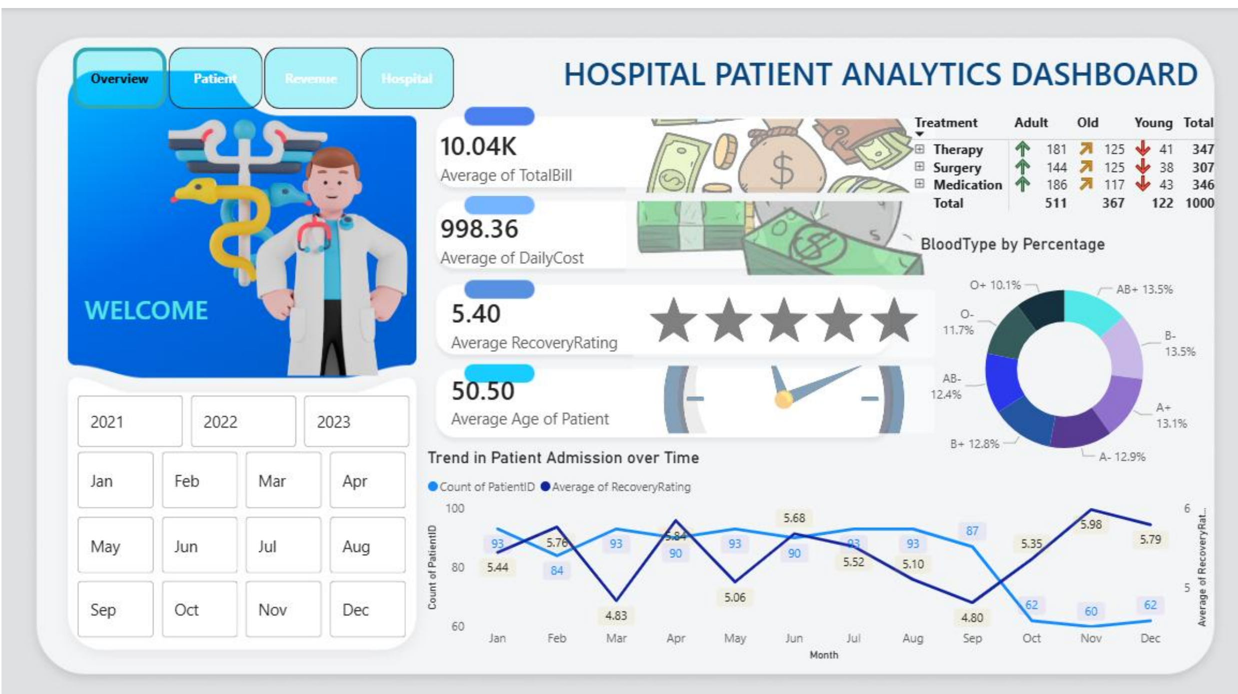
POWER BI DASHBOARD SUMMARY

INTRODUCTION

The Hospital Patient Analytics Dashboard is a multi-page Power BI solution designed to monitor clinical outcomes, patient demographics, and financial performance across hospitals. It provides a unified view of how patients are treated, how resources are utilized, and how these decisions impact revenue and recovery.

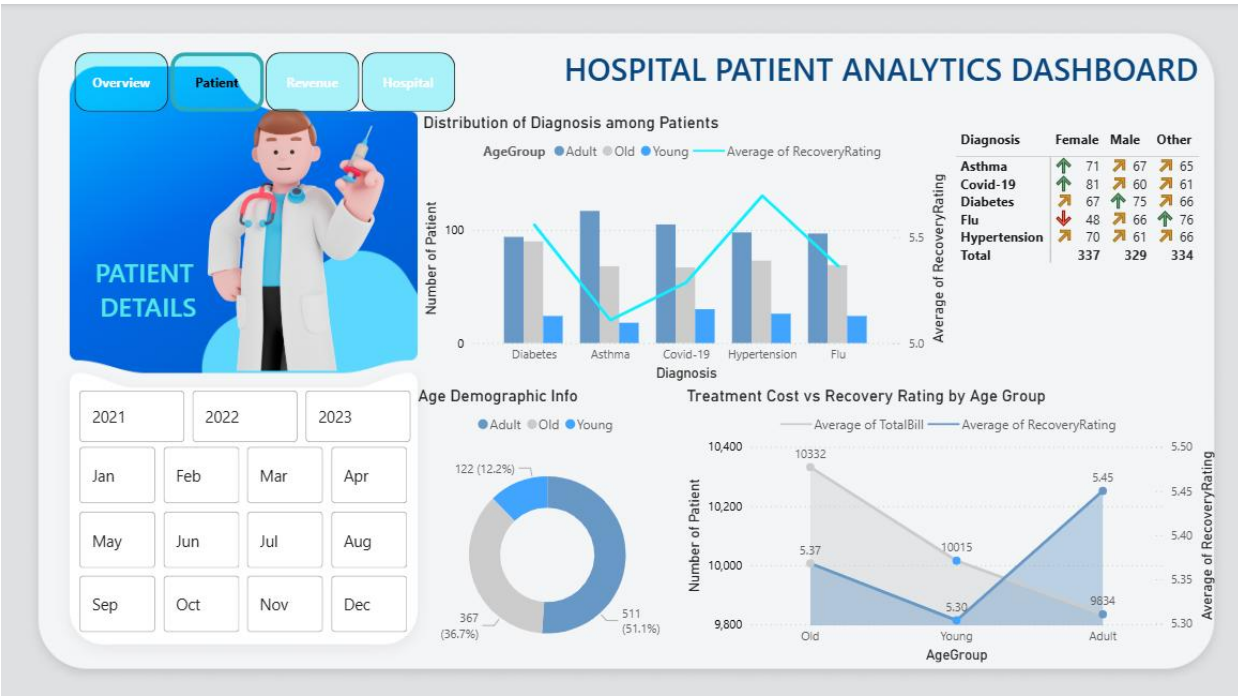
1. **Overview Dashboard**

The Overview page highlights key KPIs such as average total bill, average daily cost, average recovery rating, and average patient age. It also shows the trend of patient admissions and recovery over time, helping users quickly assess overall performance and seasonality.



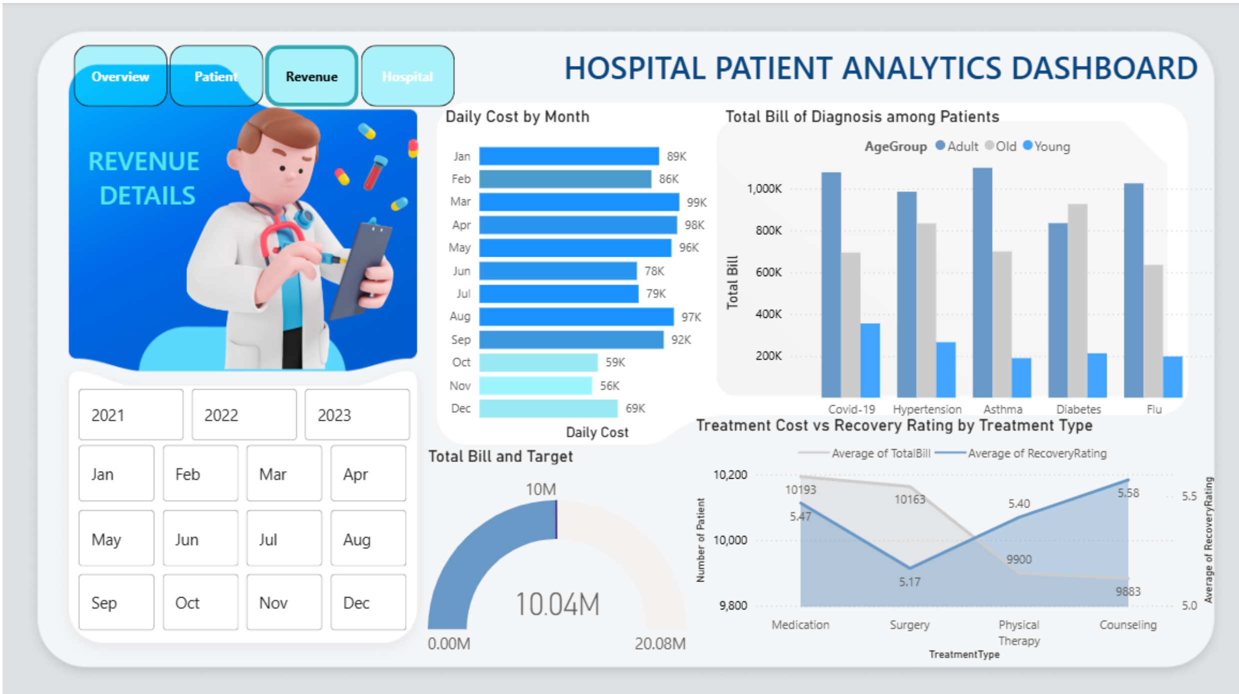
2. Patient Dashboard

The Patient page focuses on clinical patterns and demographics. It analyzes the distribution of diagnoses by age group and gender, age-group proportions, and the relationship between treatment cost and recovery rating by age group. This view supports patient-centric decisions and identification of high-risk segments.



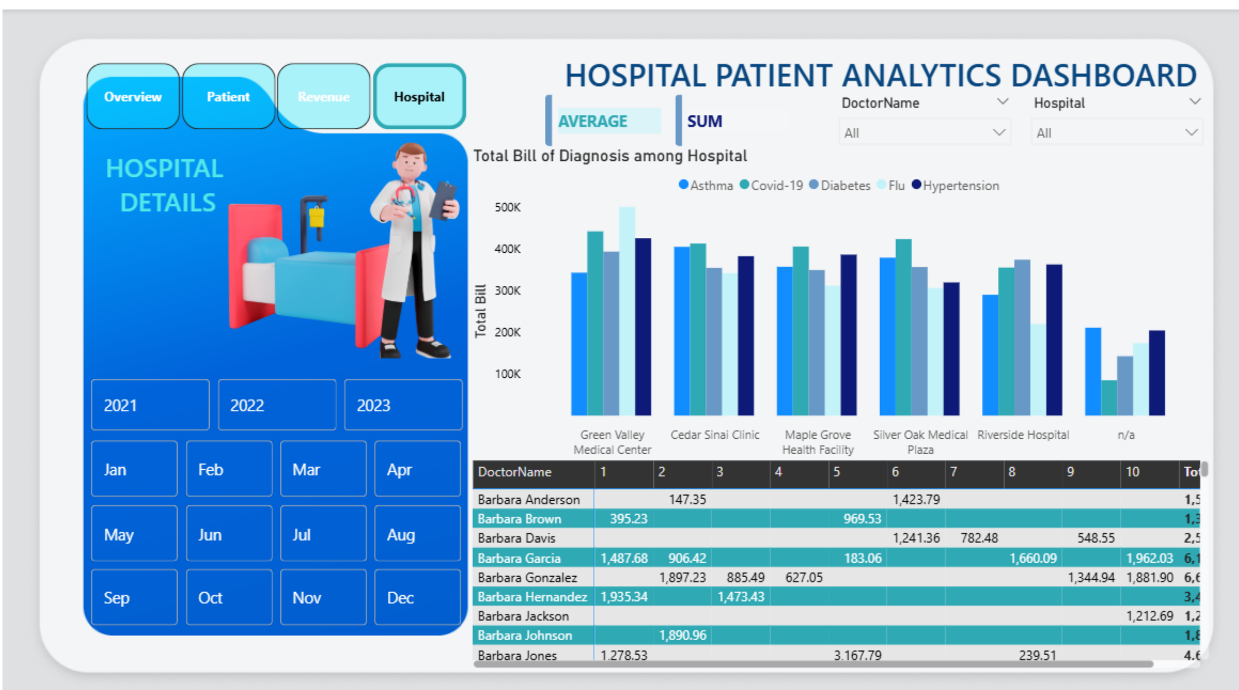
3. Revenue Dashboard

The Revenue page examines financial efficiency, including daily cost by month, total bill by diagnosis, comparison of total bill versus target and total cost vs recovery rating by treatment type. It also shows how different treatment types perform in terms of cost and recovery, enabling management to control expenses while maintaining quality.



4. Hospital Dashboard

The Hospital page compares hospitals and doctors on total bill by diagnosis and summarizes performance at provider level. It helps identify variation in billing and outcomes across facilities and clinicians, guiding standardization efforts and best-practice sharing.



All the dashboards have been supported by slicer: - Month Slicer and Year Slicer which helps in further filtrations in the dashboard.

INSIGHTS

- Revenue and bills

Total bill is about 10.04M against a 10M target, indicating slight over-achievement but also possible overspending or cost creep.

Average daily cost is about 998, so even small reductions per patient day can materially improve margins at current volume levels.

- Recovery and patient profile

Average recovery rating is roughly 5.4, which is good but leaves room to push toward best-in-class by focusing on diagnoses and treatments with lower scores.

Average age is about 50, and age demographics show adults forming roughly half of patients, with smaller but material old and young segments that may need tailored pathways.

- Time trends

Patient admissions are fairly stable but with visible monthly spikes and dips; some months have higher recovery ratings and others show drops, suggesting process or staffing variability.

Daily cost by month peaks around Mar–May and again Aug–Sep, while some later months show lower spend, indicating inconsistent cost control through the year.

- Diagnosis and treatment patterns

Covid-19 and Asthma/Hypertension appear among the highest total-bill diagnoses, while Flu and Diabetes generate moderate bills; some diagnoses show higher bills but not proportionally higher recovery ratings.

Treatment-type view shows differences in cost vs recovery: for example, Surgery is cost-intensive with only slightly better recovery than cheaper options, while Counseling/Physical Therapy show decent recovery at lower or mid-level cost.

- Hospital and doctor variation

Hospitals differ meaningfully in total bill by diagnosis; some centers (e.g., those with the highest bars for Asthma or Covid-19) bill significantly more than others for similar case types.

Doctor-level table shows spread in total bill per doctor, which likely reflects a mix of case complexity and practice style but still indicates opportunities for standardizing protocols.

- Demographics and blood type

Age-group breakdown (Adult, Old, Young) shows that adults dominate volume, but old patients incur higher cost and slightly lower recovery ratings on average.

Blood-type distribution is relatively balanced; it mainly matters for inventory and transfusion planning rather than revenue, but can support better blood-bank stocking.

RECOMMENDATION

Improve cost efficiency without hurting outcomes

Implement clinical pathways for high-bill diagnoses (Covid-19, Asthma, Hypertension) to standardize tests, imaging, and length of stay so that cost per case converges toward the best-performing hospital or doctor.

Introduce monthly cost dashboards at department level that show daily cost by month vs benchmark, with alerts when a department's cost exceeds target bands for more than 2 consecutive months.

Raise recovery ratings where they lag

For treatments where cost is high but recovery rating is only average (e.g., Surgery), conduct case-mix adjusted benchmarking across hospitals and surgeons, and revise pre- and post-operative protocols aiming to lift recovery rating by at least 0.2–0.3 points.

Expand lower-cost treatments that show good recovery (e.g., Counseling, Physical Therapy) through care bundles and early referrals, reducing reliance on medication-only approaches.

Optimize operations by month and capacity

Use the monthly admission and cost trend to reallocate staffing and beds: increase capacity and fast-track teams in the months with peak admissions and cost spikes, and schedule elective surgeries or maintenance in low-volume months.

Set monthly cost and recovery targets per department; review variance in a short operations meeting every month and assign specific actions (e.g., reduce average stay by 0.2 days for selected diagnoses).

Standardize best practices across hospitals and doctors

Identify hospitals with the best combination of lower total bill and higher recovery rating for each major diagnosis, and codify their protocols into standard order sets in the EMR.

For doctors with significantly higher total bills, set up peer review and feedback sessions, combining cost data with quality metrics (readmission, complications) to align practices.

Targeted programs by age group and diagnosis

For older patients with higher cost and slightly lower recovery, design geriatric-focused care pathways (fall-risk checks, medication review, early physiotherapy) to reduce complications and length of stay.

For chronic diagnoses (Diabetes, Hypertension), expand outpatient follow-up and education to reduce readmissions and high inpatient bills, tracking whether total bill per chronic patient falls over time.

--THANK YOU--

PRESENTED BY

- ARIJEET MUKHERJEE