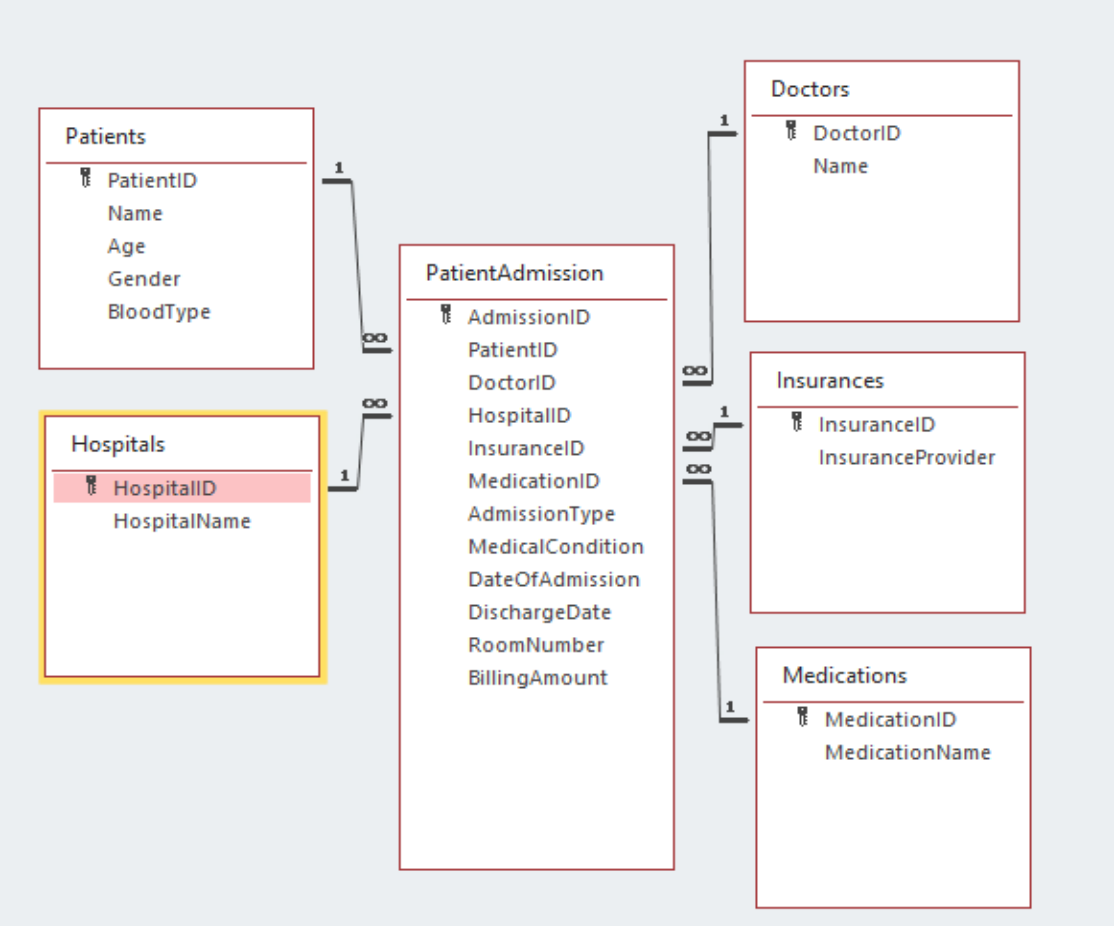
Healthcare Data Analysis

Steps Performed

Step 1: Read the text using Python
Step 2: Used Python for Data Cleaning / Data Transformation
Step 3: Got cleaned CSV file an output of data cleaning task performed in Step 2
Step 4: Create tables in SQL
Step 5: Performed SQL Scripting
Step 6: Performed Python Scripting (Data Visualization, Data Cleaning, Coding task)

Data Provided (Relational Schema):

- A SQL database with the following tables: Patients, Doctors, Hospitals, Insurance, Medications, PatientAdmission



SQL & BI tasks:

1. Exploration Data Analysis to understand the distribution of data.

a. Number of distinct patients

```
SELECT COUNT(DISTINCT PatientID) AS NumberOfDistinctPatients
FROM Patients;
```

NumberOfDistinctPatients
10000

b. Group patients by gender

```
SELECT Gender, COUNT(*) AS NumberOfPatients
FROM Patients
GROUP BY Gender; SELECT Gender, COUNT(*) AS NumberOfPatients
FROM Patients
GROUP BY Gender;
```

Gender	NumberOfPatients
Female	5075
Male	4925

c. Group patients by BloodType

```
SELECT BloodType, COUNT(*) AS NumberOfPatients
FROM Patients
GROUP BY BloodType
ORDER BY NumberOfPatients DESC;
```

BloodType	NumberOfPatients
AB-	1275
AB+	1258
B-	1252
O+	1248
B+	1244
O-	1244
A+	1241
A-	1238

d. Group patients by Medical Condition

```
SELECT MedicalCondition, COUNT(*) AS NumberOfPatients
FROM PatientAdmission
GROUP BY MedicalCondition
ORDER BY 2 DESC;
```

MedicalCondition	NumberOfPatients
Asthma	1708
Cancer	1703
Hypertension	1688
Arthritis	1650
Obesity	1628
Diabetes	1623

e. Group patients by Insurance Provider

```
SELECT InsuranceProvider, COUNT(*) AS NumberOfPatients
FROM Insurances i
JOIN PatientAdmission pa ON i.InsuranceID = pa.InsuranceID
GROUP BY InsuranceProvider
ORDER BY 2 DESC;
```

InsuranceProvider	NumberOfPatients
Cigna	2040
Blue Cross	2032
Aetna	2025
UnitedHealthcare	1978
Medicare	1925

f. Group patients by Medication

```
SELECT MedicationName, COUNT(*) AS NumberOfPatients FROM Medications m
JOIN PatientAdmission pa ON m.MedicationID = pa.MedicationID
GROUP BY MedicationName
ORDER BY 2 DESC;
```

MedicationName	NumberOfPatients
Penicillin	2079
Lipitor	2015
Ibuprofen	1976
Aspirin	1968
Paracetamol	1962

2. Query to showcase the distribution of medical conditions across different genders

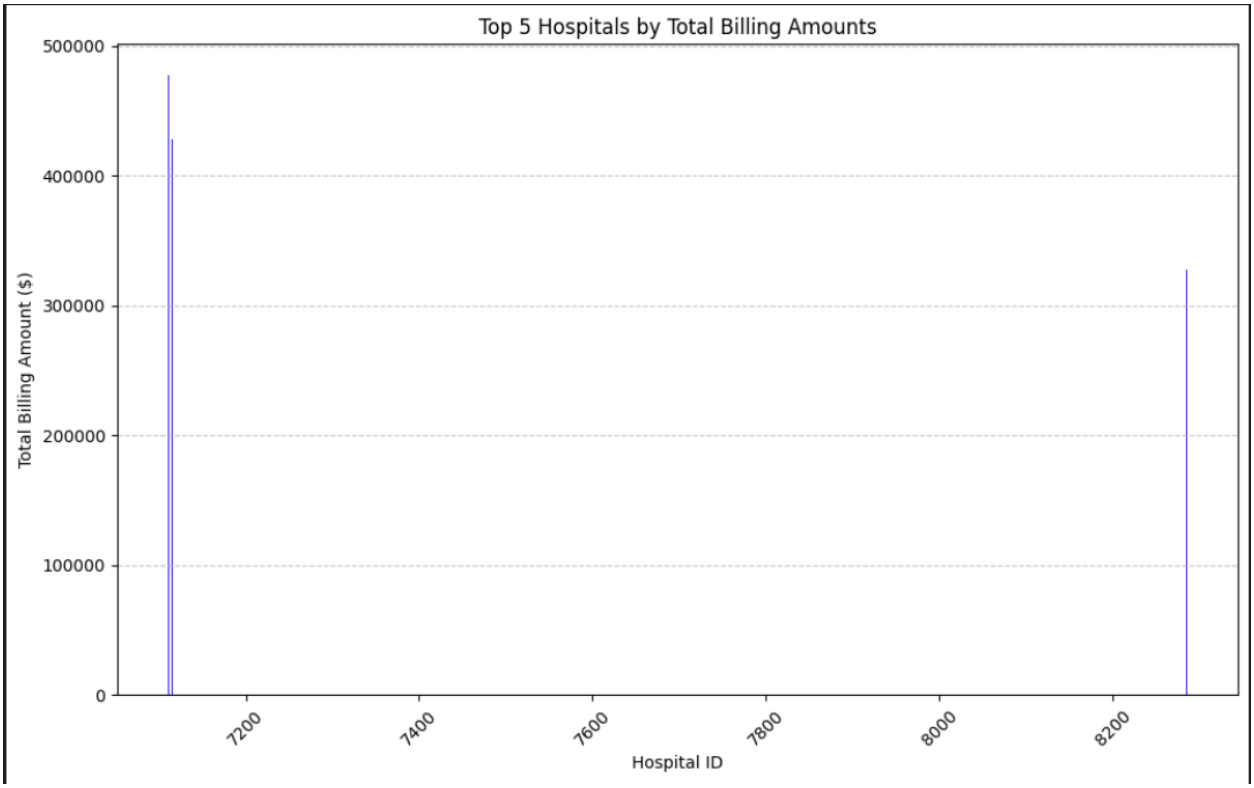
```
SELECT MedicalCondition, Gender, COUNT(*) AS NumberOfPatients
FROM Patients
JOIN PatientAdmission
ON Patients.PatientID = PatientAdmission.PatientID
GROUP BY medicalCondition, Gender
ORDER BY 3 DESC;
```

MedicalCondition	Gender	NumberOfPatients
Cancer	Female	887
Asthma	Female	874
Hypertension	Male	852
Obesity	Female	838
Hypertension	Female	836
Arthritis	Male	835
Asthma	Male	834
Diabetes	Female	825
Cancer	Male	816
Arthritis	Female	815
Diabetes	Male	798
Obesity	Male	790

3. Top 5 hospitals with the highest total billing amounts and a bar chart to display these amounts.

```
SELECT HospitalID, SUM(BillingAmount) AS TotalBilling
FROM PatientAdmission
GROUP BY HospitalID
ORDER BY TotalBilling DESC
lIMIT 5;
```

HospitalID	TotalBilling
7111	477638.88
7116	432283.57
7115	428163.07
7113	351463.89
8285	327522.47



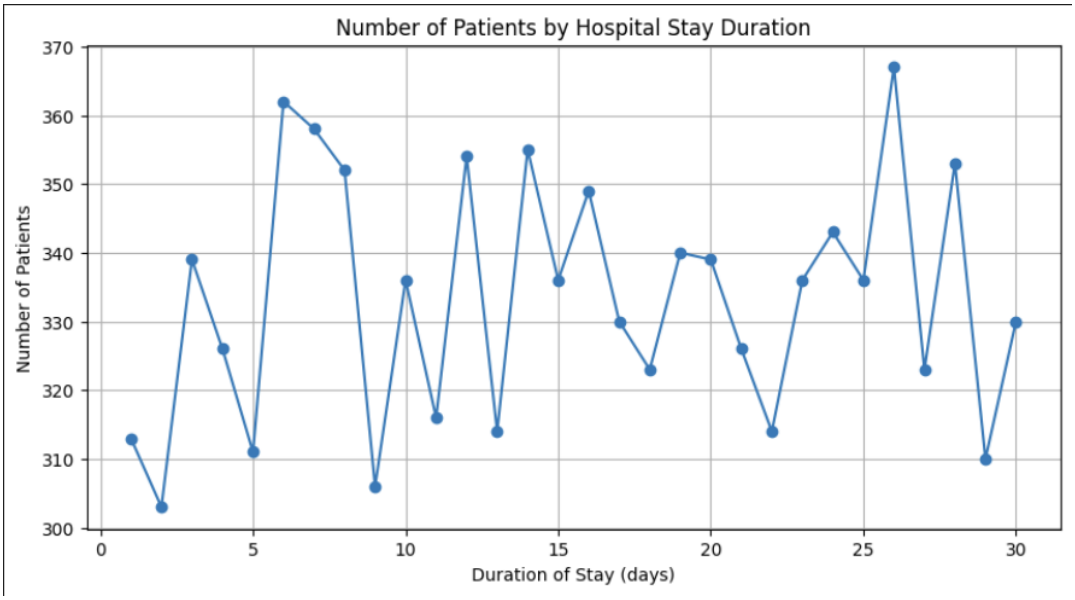
4. SQL query to count the number of doctors whose last names start with the letter 'W'

```
SELECT MedicalCondition, Gender, COUNT(*) AS NumberOfPatients
FROM Patients
JOIN PatientAdmission
ON Patients.PatientID = PatientAdmission.PatientID
GROUP BY medicalCondition, Gender
ORDER BY 3 DESC;
```

NumberOfDoctors
761

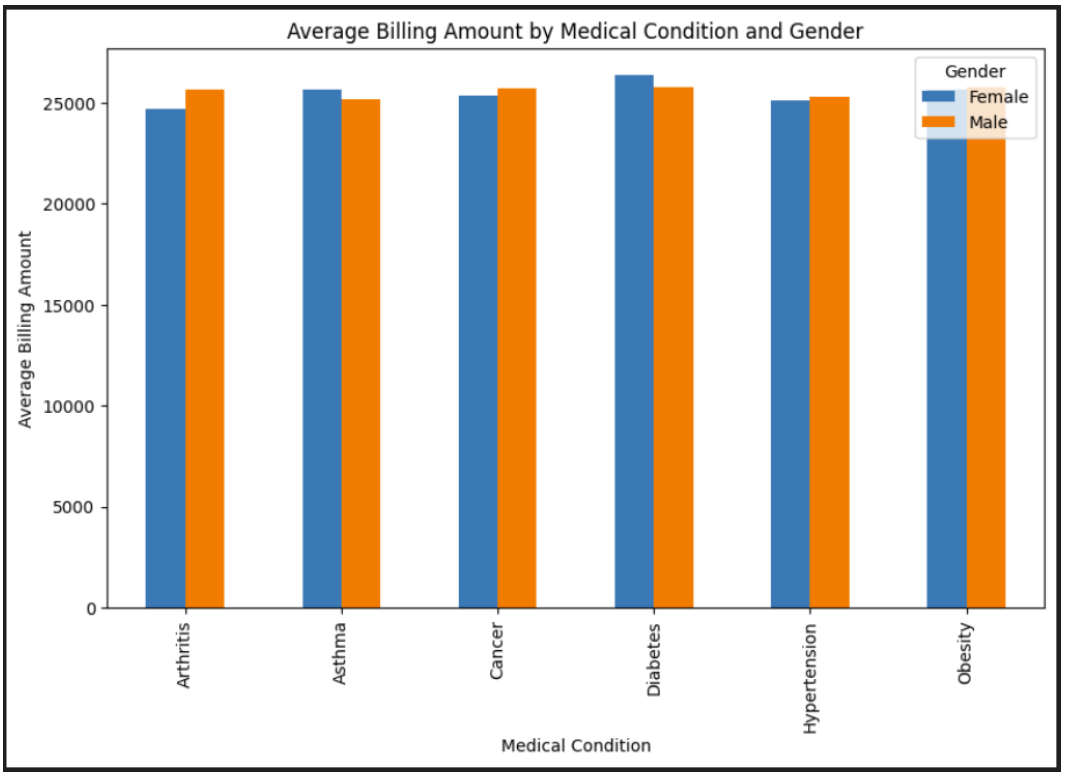
5. Plot that displays the number of patients grouped by the duration of their hospital stay

```
SELECT HospitalID, SUM(BillingAmount) AS TotalBilling
FROM PatientAdmission
GROUP BY HospitalID
ORDER BY TotalBilling DESC
lIMIT 5;
```



6. Bar graph that shows the average billing amount for Medical Conditions among Males & Females

```
SELECT MedicalCondition, Gender, round(AVG(BillingAmount),2) AS AverageBilling
FROM PatientAdmission
JOIN Patients ON PatientAdmission.PatientID = Patients.PatientID
GROUP BY MedicalCondition, Gender
ORDER BY AverageBilling DESC;
```



Reflecting on the dataset:

1. **What are your thoughts on this dataset? It is designed to mimic a real-world healthcare scenario using an RDBMS with simulated data. Do you believe this database closely mimics a real-world setting?**

   This dataset provides a simple but functional representation of a healthcare system's relational database that captures the most important entities, such as patients, doctors, and hospitals.

   It captures essential data points for admissions, treatments, and healthcare providers to offer a simple framework useful in educational purposes or building a system from scratch. Due to the limitations

   in insurance and medication options, it does not include the complexity of a real-world setting. A real database would have much more extensive records and complex, interwoven data relationships, bound by strict regulations.

   This simulated dataset is a good educational tool but needs to be expanded to really reflect the full scope and scale of real healthcare data

2. **Please list any data cleaning or data transformation steps you performed when loading the relational tables into the database or when writing the queries**

Sol :

1) **Convert Text to CSV**: Load the raw data from a text file into a structured format using pandas, ensuring it's in a table-like format (CSV).

2) **Validate Data**: Implement specific functions to validate each column based on predefined rules (e.g., digit check for IDs, alphabetic check for names, age range check, and correct date format).

3) **Filter Valid Entries**: Apply these validation checks to each row, marking each field as valid or invalid, and then filter out rows that contain any invalid fields.

4) **Save Clean Data**: Export the cleaned and validated data back into a CSV file, ensuring that only data that meets all validation criteria is saved

Coding Task:

The code for the **Coding Task** and Automation Code for basic data Cleaning/validation is in the Python file attached