Technical Report for Project:

Hospital Data Management and Analysis

ALGONQUIN COLLEGE

CST2102-Database Analytics

Instructor - Yasser Jafer

By:

Prajna Ganji, Shiva Shiva, Neetika Upadhyay and Maruta Zalane

April 11, 2025

Abstract

This project presents a fully developed relational database system for a hospital, created to manage and analyze key healthcare operations such as patient records, doctor information, appointments, prescriptions, billing, room assignments, and staff data. The system incorporates real-world medical scenarios with accurate synthetic data generation and enforces constraints such as primary and foreign keys, not-null requirements, and unique identifiers. Comprehensive SQL queries and views were crafted to answer business questions, and a Power BI dashboard was developed to provide visual insights into hospital performance, enabling effective operational and strategic decisions.

These dashboards are designed to support a range of hospital stakeholders:

- **Executives and Administrators** for strategic planning and performance monitoring
- **Department Heads and Doctors** for appointment management and workload tracking
- **Finance and Billing Teams** for revenue analysis and outstanding payments
- **Pharmacy and Clinical Quality Teams** for prescription reviews and treatment oversight
- **Operations and Ward Managers** for optimizing room utilization and patient flow

This project demonstrates how integrated database systems combined with interactive analytics can enhance hospital efficiency, transparency, and decision-making across multiple departments.

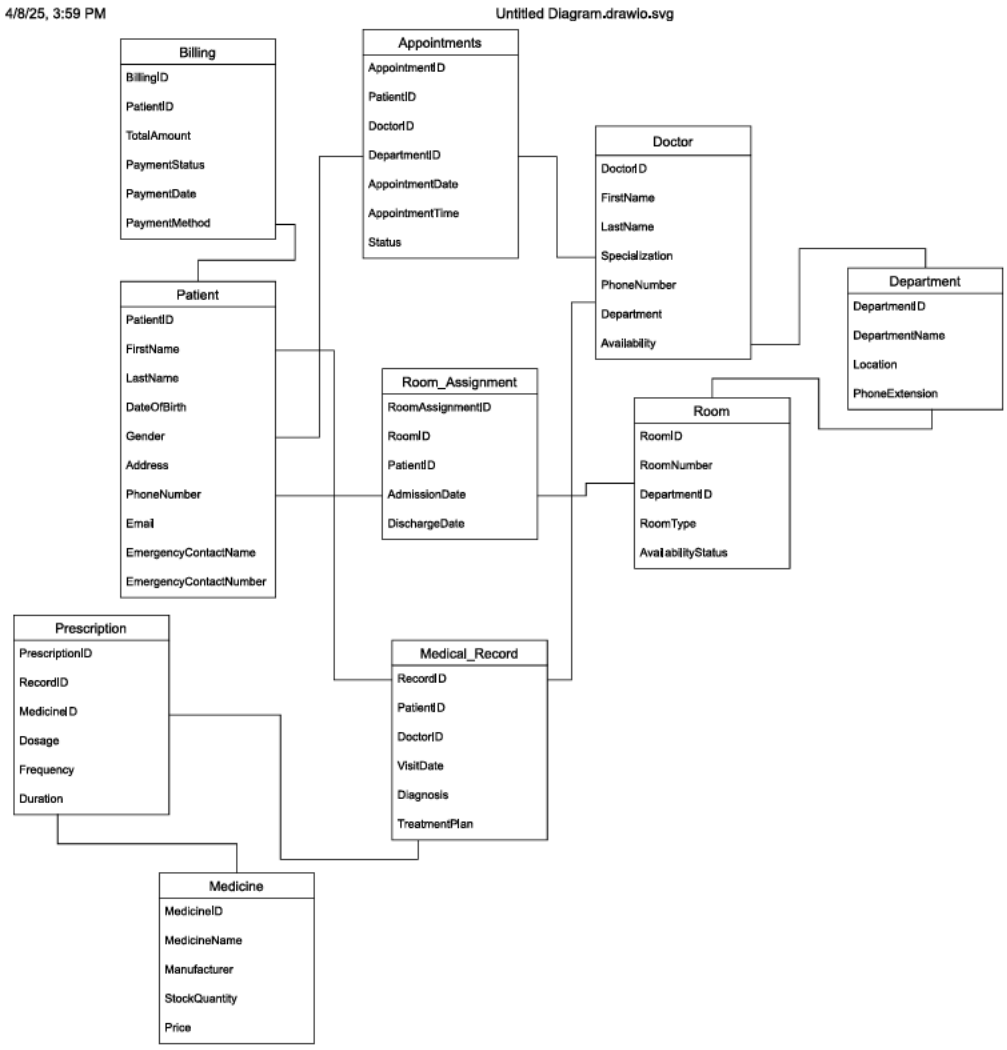
Introduction

Hospitals generate and manage vast amounts of data ranging from patient records to appointment scheduling and financial operations. To handle this effectively, we implemented a relational database design that facilitates efficient storage, retrieval, and analysis of hospital-related data. This report outlines the objectives, including data normalization, data population from staging sources, and developing SQL queries for operational insights. We also used Power BI to visualize the data across five key categories.

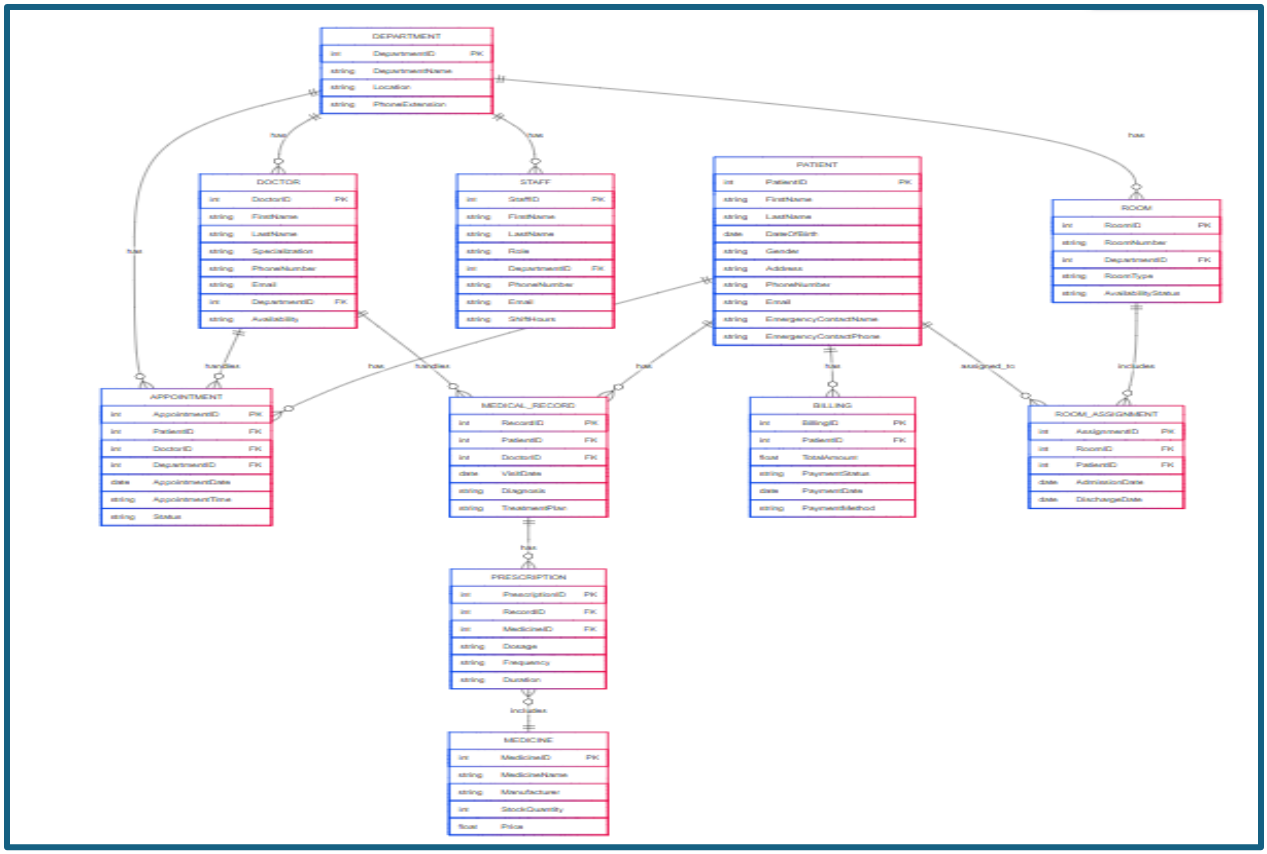
Methodology

The following steps outline the methodology used to complete the hospital database system project:

1. **Requirement Analysis**
    - Studied the project brief and identified required entities, attributes, and relationships.
    - Outlined functional and non-functional requirements of the hospital system.
2. **Database Design**
    - Created a conceptual ER diagram representing entities like Patient, Doctor, Department, etc.



    - Developed a physical schema with attribute details, data types, keys, and constraints.



3. Table Structure and Relationships:

- One-to-Many Relationships:
    - A patient can have many appointments and medical records.
    - A doctor can handle multiple appointments and multiple medical records.
    - A department can have many doctors, appointments, and rooms.
- Many-to-Many Relationships:
    - Patients and doctors interact via appointments.
    - Doctors prescribe multiple medicines to patients via medical records.
- One-to-One Relationship:
    - A medical record can have a corresponding prescription entry.

SQL Constraints Applied:

- **Primary Key Constraints:** Enforced on unique identifiers like Patient ID, Doctor ID, Department ID, Appointment ID.

- **Foreign Key Constraints:** Established for key relationships, e.g., Patient ID in Appointment Table references Patient ID in Patient Table.
- **Not Null Constraints:** Applied on essential attributes like FirstName, LastName, etc.
- **Unique Constraints:** Applied on unique attributes such as Email, Phone Number, and Room Number.

Table Name	Primary Key	Foreign Keys	Purpose
Patients	PatientID		Holds patient demographic information
Doctors	DoctorID	DepartmentID	Contains doctor information and department link
Departments	DepartmentID		Stores hospital department details
Appointments	AppointmentID	PatientID, DoctorID, DepartmentID	Records patient-doctor appointments
MedicalRecords	RecordID	PatientID, DoctorID, AppointmentID	Stores diagnosis and treatment plans
Prescriptions	PrescriptionID	RecordID, MedicineID	Links medicines to medical records
Medicines	MedicineID		Contains medicine details and stock information
Billing	BillingID	PatientID	Manages billing and payment status

Staff	StaffID	DepartmentID	Stores non-doctor hospital staff details
Rooms	RoomID	DepartmentID	Manages room details and availability
Rooms Assignment	AssignmentID	PatientID, RoomID	Tracks patient room assignments and date

4. **SQL Script Development**
    - Wrote DDL (Data Definition Language) scripts to create tables and enforce relationships and constraints.
    - Applied primary keys, foreign keys, unique and not-null constraints.

```sql
-- Hospital Database DDL Script

-- 1. Patient Table
CREATE TABLE Patient (
    PatientID INT PRIMARY KEY,
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    DateOfBirth DATE NOT NULL,
    Gender VARCHAR(10),
    Address TEXT,
    PhoneNumber VARCHAR(15) UNIQUE,
    Email VARCHAR(100) UNIQUE,
    EmergencyContactName VARCHAR(100),
    EmergencyContactPhone VARCHAR(15)
);

-- 2. Department Table
CREATE TABLE Department (
    DepartmentID INT PRIMARY KEY,
    DepartmentName VARCHAR(100) UNIQUE NOT NULL,
    Location VARCHAR(100),
    PhoneExtension VARCHAR(10)
);

-- 3. Doctor Table
CREATE TABLE Doctor (
    DoctorID INT PRIMARY KEY,
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    Specialization VARCHAR(100),
    PhoneNumber VARCHAR(15) UNIQUE,
    Email VARCHAR(100) UNIQUE,
    DepartmentID INT,
    Availability VARCHAR(100),
    FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)
);
```

5. **Synthetic Data Generation**
    - Used GPT-4 to generate synthetic data for all tables in the hospital database.
    - Designed structured prompts to simulate realistic records for Patients, Doctors, Appointments, etc.

- Performed data quality checks to ensure logical consistency and accuracy.
- Validated data against schema constraints to enforce relational integrity

6. **Data Insertion Process**
    - Performed direct CSV imports using the BULK INSERT command in SQL Server for large tables such as Appointment to enhance performance.

```
--Bulk CSV Import Using T-SQL
BULK INSERT appointment
FROM 'E:\Neeti\BISI\Data Analytics\Project\Appointment.csv'
with (
     FIELDTERMINATOR = ',',
     ROWTERMINATOR = '\n',
     FIRSTROW = 2,
     TABLOCK
);

BULK INSERT Billing
FROM 'E:\Neeti\BISI\Data Analytics\Project\Billing.csv'
with (
     FIELDTERMINATOR = ',',
     ROWTERMINATOR = '\n',
     FIRSTROW = 2,
     TABLOCK
);

BULK INSERT Department
FROM 'E:\Neeti\BISI\Data Analytics\Project\Departments.csv'
with (
     FIELDTERMINATOR = ',',
     ROWTERMINATOR = '\n',
     FIRSTROW = 2,
     TABLOCK
);
```

7. **Business Intelligence Questions & Queries**
    - **Patient and Appointment Analysis:**
        1. What is the distribution of appointments by age group and gender?
        2. What is the average number of appointments per patient in the last 6 months?
    - **Doctor Availability and Workload:**
        1. Which doctors have the highest patient load, and how does it compare to their average working hours per day?

2. Are there any doctors consistently overbooked beyond their declared availability hours?
3. Which doctors have the highest number of completed appointments, and in which departments?

- **Financial Overview:**
    1. What is the total revenue generated from different departments over the past quarter?
    2. What percentage of total revenue remains unpaid, and which departments have the highest outstanding balances?
    3. What is the average billing amount per patient by department in the last quarter?
    4. Which payment methods are most used, and how does payment method impact payment status (paid/unpaid)?
- **Prescription Patterns:**
    1. Which are the top 5 most prescribed medicines in the last 3 months?
    2. Which departments prescribe the highest volume of medications, and how does that compare across specializations?
    3. How often do prescriptions lead to stock shortages (Stock Quantity in Medicine Table)?
- **Room Utilization:**
    1. What percentage of rooms (ICU, General, Private) were occupied in the last month?
    2. What is the average length of stay for patients in each room type over the last month?
    3. Which departments had the highest demand for rooms, and what was the average wait time for admission?
    4. How many rooms were unoccupied for more than 5 consecutive days in the last month?

Developed SQL queries to fetch insights by joining multiple tables and applying aggregation functions.

```sql
-- Patient and Appointment Analysis

-- 1. Distribution of appointments by age group and gender
SELECT
    CASE
        WHEN DATEDIFF(YEAR, P.DateOfBirth, GETDATE()) < 18 THEN '0-17'
        WHEN DATEDIFF(YEAR, P.DateOfBirth, GETDATE()) BETWEEN 18 AND 35 THEN '18-35'
        WHEN DATEDIFF(YEAR, P.DateOfBirth, GETDATE()) BETWEEN 36 AND 60 THEN '36-60'
        ELSE '60+'
    END AS AgeGroup,
    P.Gender,
    COUNT(A.AppointmentID) AS AppointmentCount
FROM Patient P
JOIN Appointment A ON P.PatientID = A.PatientID
GROUP BY
    CASE
        WHEN DATEDIFF(YEAR, P.DateOfBirth, GETDATE()) < 18 THEN '0-17'
        WHEN DATEDIFF(YEAR, P.DateOfBirth, GETDATE()) BETWEEN 18 AND 35 THEN '18-35'
        WHEN DATEDIFF(YEAR, P.DateOfBirth, GETDATE()) BETWEEN 36 AND 60 THEN '36-60'
        ELSE '60+'
    END, P.Gender;

-- 2. Average number of appointments per patient in the last 6 months
SELECT
    AVG(AppointmentsPerPatient) AS AvgAppointmentsLast6Months
FROM (
    SELECT
        PatientID,
        COUNT(*) AS AppointmentsPerPatient
    FROM Appointment
    WHERE AppointmentDate >= DATEADD(MONTH, -6, GETDATE())
    GROUP BY PatientID
) AS PatientAppointments;
```

8. **User Roles and Access Control**
    - Created one admin user with full privileges.
    - Created five normal users with read-only access to ensure data security.

```sql
--Admin Login

Create Login Admin_1
with Password = '123';

use hospital_database_fp;
Create user admin_1 for login admin_1;

--User Login

Create Login user_1
with Password = '1234';

use hospital_database_fp;
Create user user_1 for login user_1;

Create Login user_2
with Password = '1234';

use hospital_database_fp;
Create user user_2 for login user_2;

Create Login user_3
with Password = '1234';

use hospital_database_fp;
Create user user_3 for login user_3;

Create Login user_4
with Password = '1234';

use hospital_database_fp;
Create user user_4 for login user_4;

Create Login user_5
with Password = '1234';

use hospital_database_fp;
Create user user_5 for login user_5;
```

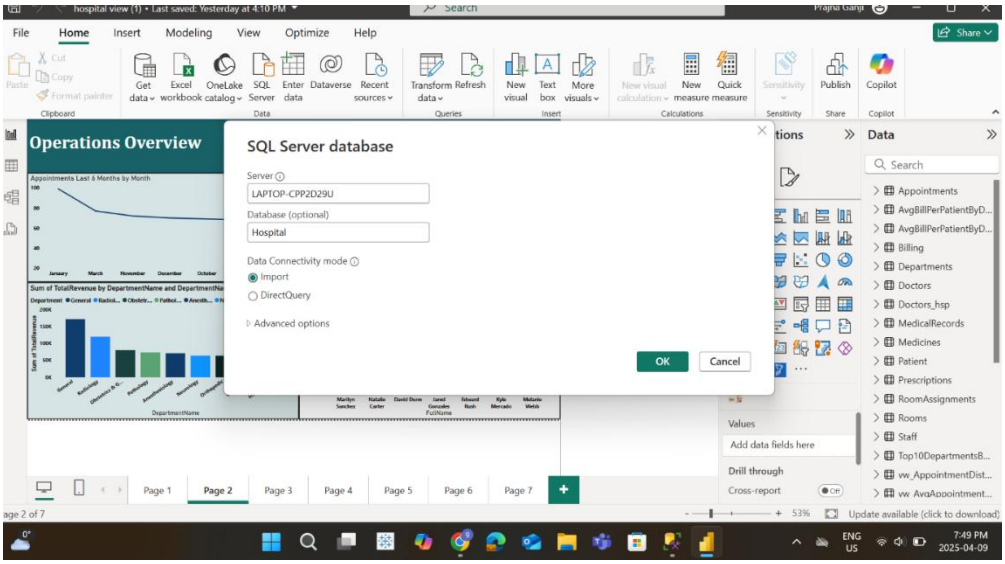
9. Created views to connect to power BI Created

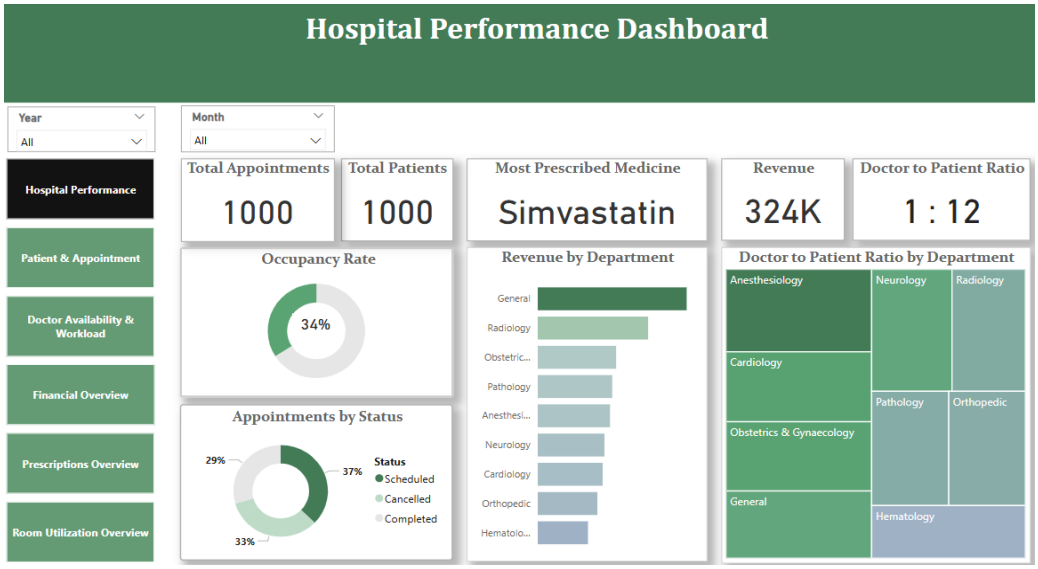
hospital_views_fix...PP2D29U\T490 (60)) | hospital_views.sql...PP2D29U\T490 (52))

```sql
-- 7. View: Room Availability Summary
CREATE VIEW vw_RoomAvailability AS
SELECT
    RoomType,
    COUNT(*) AS TotalRooms,
    SUM(CASE WHEN AvailabilityStatus = 'Available' THEN 1 ELSE 0 END) AS AvailableRooms,
    SUM(CASE WHEN AvailabilityStatus = 'Occupied' THEN 1 ELSE 0 END) AS OccupiedRooms
FROM Room
GROUP BY RoomType;

-- 8. View: Staff Count by Department
CREATE VIEW vw_StaffByDepartment AS
SELECT
    d.DepartmentName,
    COUNT(s.StaffID) AS TotalStaff
FROM Department d
JOIN Staff s ON d.DepartmentID = s.DepartmentID
GROUP BY d.DepartmentName;

-- 9. View: Daily Appointment Trends
CREATE VIEW vw_DailyAppointmentTrends AS
SELECT
    CAST(AppointmentDate AS DATE) AS Date,
    COUNT(*) AS TotalAppointments
FROM Appointments
GROUP BY CAST(AppointmentDate AS DATE)
```

10. Power BI Dashboard Development

- Imported data from the SQL database by creating view for queries.



- Designed visualizations showing revenue, patient statistics, appointment trends, and room usage.
- 

11. **Documentation and Testing**
    - Compiled a detailed technical report.
    - Tested data integrity, query accuracy, and user permissions.

Challenges

1. **Data Generation & Referential Integrity Challenges**
    - Generating large volumes of realistic yet relational synthetic data was time-consuming
    - Early GPT-4 outputs had random, disconnected values, breaking referential integrity.
    - Required multiple prompt refinements and iterations to generate consistent, rule-based data
2. **SQL to Power BI Integration**
    - Needed to write custom SQL queries and views to pre-aggregate and clean data before import
    - Power BI sometimes inferred incorrect relationships requiring manual adjustment

Learning & Takeaways

1. **Technical Learnings**
    - Gained hands-on experience in designing Entity-Relationship (ER) diagrams.
    - Improved SQL skills for data generation, validation, and analytical querying.

    - Learned to use prompt engineering with GPT-4 to generate high-quality synthetic data.
    - Developed data transformation workflows connecting SQL Server with Power BI
2. **Analytical & Modeling Skills**
    - Built a relational data model with real-world hospital operations in mind.
    - Understood the importance of referential integrity and data consistency for meaningful analysis.
    - Created and optimized DAX measures and visualizations in Power BI.
3. **Problem Solving & Iteration**
    - Discovered how iterative refinement (in prompts and design) leads to better outcomes.
    - Adapted quickly to challenges in data inconsistencies and modeling logic.

Discussion

The normalized schema improved data accuracy and enabled more complex joins and aggregations. Patterns such as uneven doctor workloads and underused rooms point to resource optimization opportunities. Data quality from the staging table posed some challenges due to missing and inconsistent entries, which were handled with data cleaning rules and constraints. The integrated Power BI dashboard enhanced stakeholder engagement by making insights more accessible.

Conclusion

This project successfully established a normalized hospital database and leveraged it for insightful analysis. The combination of structured SQL queries and dynamic Power BI visualizations allowed for comprehensive hospital monitoring. The model can be extended to accommodate real-time data and more advanced analytics in future implementations.

References

- Microsoft Docs. SQL Server Documentation. https://docs.microsoft.com/en-us/sql/sql-server/
- Power BI Documentation. Microsoft Learn. https://learn.microsoft.com/en-us/power-bi/
- Course Material and Lecture Notes.