Hospital ER Analytics Database

Complete SQL-Based Healthcare Data System
Created by: *Amy Sauden*
Technologies: SQL Server, ERD, Data Modelling, Analytics Views, Role-Based Security

1. Executive Summary

This project is a complete SQL-based data system designed to model how an Emergency Room (ER) manages day-to-day clinical and operational information. The system captures patient details, insurance providers, staff members, diagnoses, and ER visit activity. It provides an organized and analytics-ready database capable of supporting real-world use cases such as patient history lookup, billing and charge tracking, staff performance reporting, and insurance-based analytics.

The solution includes a fully normalized relational schema, realistic synthetic data, analytical SQL views, and role-based access controls that simulate real HIPAA-style environments.

Purpose of the Project

The purpose of this project is to model real healthcare data operations in a structured SQL environment. The system enables consistent tracking of ER visits, supports billing workflows, improves access to patient and diagnosis information, and provides a strong foundation for analytics dashboards, cost reporting, and healthcare research. This project mimics the type of relational structure used in hospitals, pharmacy benefit systems, and insurance claim pipelines.

2. System Overview

Emergency Rooms generate structured data from many different sources: patient admissions, diagnosis codes, staff assignments, visit timestamps, and insurance billing. This project organizes these elements into a clean, connected relational database with one goal: **to make ER data easy to query, analyze, and understand.**

The central object is the **Visit**, which links together:

- The **Patient** receiving care
- The **Staff** member attending
- The **Diagnosis** assigned
- The **Insurance** provider billed

Each surrounding table supports real analytics:

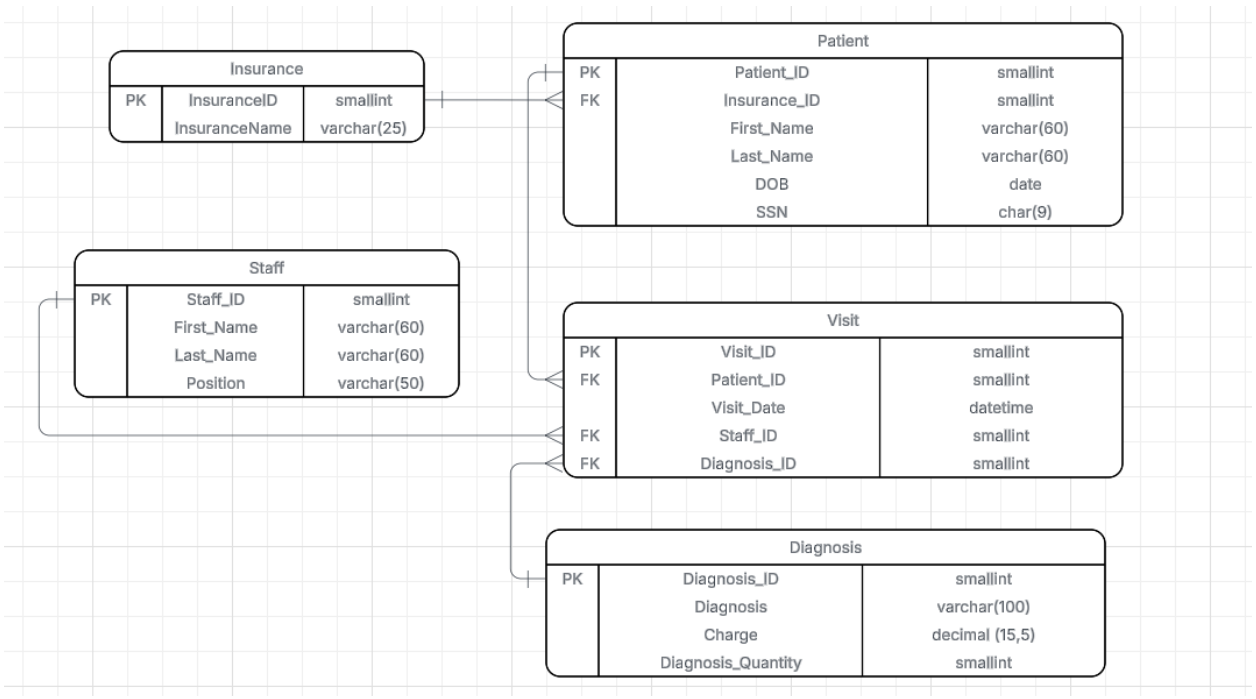
- **Patients:** identify who received care and their demographics
- **Staff:** show who provided care and what roles they hold
- **Diagnosis:** represent medical conditions and their associated costs
- **Insurance:** identify billing sources
- **Visits:** capture clinical events and timestamps

This structure mirrors how electronic health systems (EHRs) and insurance claim systems store data.

Key Features

- **Fully Normalized Healthcare Data Model** – Includes patient, staff, insurance, diagnosis, and visit entities.
- **Analytics-Ready Views** – Provides reporting for patient history, total charges, and staff performance.
- **Realistic Synthetic Data** – Allows meaningful testing and visualization.
- **Role-Based Access Control** – Simulates Admin, Limited, and Read-Only users similar to HIPAA environments.
- **Clean, Scalable Architecture –** Mirrors real EHR and insurance systems used by hospitals.

3. Entity–Relationship Diagram (ERD)



The diagram shows five fully connected tables:

- **Patient** links to **Insurance**.
- **Visit** links to **Patient**, **Staff**, and **Diagnosis**.
- **Diagnosis** contains clinical and financial data.
- **Staff** contain role information.

Visit is the “fact” table that combines all reference data into a single analyzable event.

4. Data Model

Below is the complete data dictionary describing every table and every column in the system.

4.1 Insurance Tabel

Column Name	Data Type	Description
InsuranceID	smallint (PK)	Unique identifier for each provider
InsuranceName	varchar(25)	Name of the insurance company

4.2 Patient Table

Column Name	Data Type	Description
Patient_ID	smallint (PK)	Unique identifier for each patient
Insurance_ID	smallint (FK)	Links patient to the Insurance table
First_Name	varchar(60)	Patient’s first name
Last_Name	varchar(60)	Patient’s last name
DOB	date	Date of birth
SSN	char(9)	Social Security Number

4.3 Staff Table

Column Name	Data Type	Description
Staff_ID	smallint (PK)	Unique staff identifier
First_Name	varchar(60)	Staff first name
Last_Name	varchar(60)	Staff last name
Position	varchar(50)	Role (doctor, nurse, technician, etc.)

4.4 Diagnosis Table

Column Name	**Data Type**	**Description**
Diagnosis_ID	smallint (PK)	Unique medical diagnosis identifier
Diagnosis	varchar(100)	Name of the diagnosis
Charge	decimal(15,5)	Cost associated with the condition
Diagnosis_Quantity	smallint	Quantity (useful for multi-service treatments)

4.5 Visit Table

Column Name	**Data Type**	**Description**
Visit_ID	smallint (PK)	Unique visit ID
Patient_ID	smallint (FK)	Connects visit to a patient
Visit_Date	datetime	Timestamp of ER visit
Staff_ID	smallint (FK)	Staff who attended the visit
Diagnosis_ID	smallint (FK)	Medical condition assigned

5. Technical Implementation

```
CREATE TABLE Insurance (
    InsuranceID smallint PRIMARY KEY,
    InsuranceName varchar(25)
);

CREATE TABLE Patient (
    Patient_ID smallint PRIMARY KEY,
    Insurance_ID smallint FOREIGN KEY REFERENCES
Insurance(InsuranceID),
    First_Name varchar(60),
    Last_Name varchar(60),
    DOB date,
    SSN char(9)
);

CREATE TABLE Staff (
    Staff_ID smallint PRIMARY KEY,
    First_Name varchar(60),
    Last_Name varchar(60),
    Position varchar(50)
);

CREATE TABLE Diagnosis (
    Diagnosis_ID smallint PRIMARY KEY,
    Diagnosis varchar(100),
    Charge decimal(15,5),
    Diagnosis_Quantity smallint
);

CREATE TABLE Visit (
    Visit_ID smallint PRIMARY KEY,
    Patient_ID smallint FOREIGN KEY REFERENCES
Patient(Patient_ID),
    Visit_Date datetime,
    Staff_ID smallint FOREIGN KEY REFERENCES Staff(Staff_ID),
    Diagnosis_ID smallint FOREIGN KEY REFERENCES
Diagnosis(Diagnosis_ID)
);
```

6. Data Population

```
-- Insurance Providers
INSERT INTO Insurance (InsuranceName)
VALUES ('Aetna'), ('Blue Cross'), ('United Health'), ('Cigna'), ('Humana'),
('Medicare'), ('MediAssist'), ('CareSource'), ('HealthPlus'), ('Tricare');

-- Patient Records
INSERT INTO Patient (Insurance_ID, First_Name, Last_Name, DOB, SSN)
VALUES
(1,'John','Smith','1990-04-12','123456789'),
(2,'Maria','Lopez','1985-09-22','234567891'),
(3,'David','Brown','1979-03-10','345678912'),
(4,'Sophia','Taylor','1995-07-15','456789123'),
(5,'Michael','Johnson','1988-11-03','567891234'),
(6,'Emily','Williams','1999-01-09','678912345'),
(7,'James','Davis','1993-05-20','789123456'),
(8,'Olivia','Miller','2000-02-25','891234567'),
(9,'Daniel','Garcia','1982-12-14','912345678'),
(10,'Grace','Martinez','1997-08-07','102345678');

-- Staff
INSERT INTO Staff (First_Name, Last_Name, Position)
VALUES
('Robert','King','Doctor'),
('Anna','Moore','Nurse'),
('William','Clark','Surgeon'),
('Jessica','Hall','Receptionist'),
('Thomas','Allen','Technician'),
('Isabella','Baker','Nurse'),
('Andrew','Nelson','Pharmacist'),
('Megan','Parker','Therapist'),
('Jacob','Adams','Doctor'),
('Lily','Scott','Administrator');

-- Diagnoses
INSERT INTO Diagnosis (Diagnosis, Charge, Diagnosis_Quantity)
VALUES
('Flu',80,1),('Fracture',500,1),('Migraine',120,2),('Allergy',90,1),
('Asthma',250,1),('Infection',200,1),('Back Pain',150,1),
('Sprain',100,1),('Fever',75,1),('Diabetes',300,1);

-- Visit Records
INSERT INTO Visit (Patient_ID, Visit_Date, Staff_ID, Diagnosis_ID)
VALUES
(1,'2025-10-01 09:00:00',1,1),
(2,'2025-09-15 11:00:00',3,2),
(3,'2025-09-10 10:30:00',2,3),
(4,'2025-09-18 14:00:00',5,4),
(5,'2025-08-25 08:45:00',7,5),
(6,'2025-09-05 16:00:00',4,6),
(7,'2025-09-20 12:00:00',8,7),
(8,'2025-10-02 13:15:00',6,8),
(9,'2025-09-29 09:45:00',9,9),
(10,'2025-10-03 15:00:00',10,10);
```

7. Analytics Views

These views simulate real hospital reporting.

7.1 Patient Visit Report View

Combines patient, staff, insurance, diagnosis, and cost into a single analytics record.

```
CREATE VIEW PatientVisitReport AS
SELECT
    p.First_Name, p.Last_Name, p.DOB, p.SSN,
    v.Visit_Date,
    s.First_Name AS Staff_First, s.Last_Name AS Staff_Last,
s.Position,
    d.Diagnosis, d.Charge,
    i.InsuranceName
FROM Visit v
JOIN Patient p ON v.Patient_ID = p.Patient_ID
JOIN Staff s ON v.Staff_ID = s.Staff_ID
JOIN Diagnosis d ON v.Diagnosis_ID = d.Diagnosis_ID
JOIN Insurance i ON p.Insurance_ID = i.InsuranceID;
```

7.2 Staff Activity Summary View

Summarizes workload and total revenue generated by each staff member.

```
CREATE VIEW StaffActivitySummary AS
SELECT
    s.Staff_ID, s.First_Name, s.Last_Name, s.Position,
    COUNT(v.Visit_ID) AS Total_Visits,
    SUM(d.Charge) AS Total_Revenue
FROM Staff s
LEFT JOIN Visit v ON s.Staff_ID = v.Staff_ID
LEFT JOIN Diagnosis d ON v.Diagnosis_ID = d.Diagnosis_ID
GROUP BY s.Staff_ID, s.First_Name, s.Last_Name, s.Position;
```

8. Security & Access Control

Admin User (Full Control)

```
CREATE LOGIN ER_Admin WITH PASSWORD='Admin@123';
CREATE USER ER_Admin_User FOR LOGIN ER_Admin;
ALTER ROLE db_owner ADD MEMBER ER_Admin_User;
```

Limited Access User

```
CREATE LOGIN ER_Limited WITH PASSWORD='Limited@123';
CREATE USER ER_Limited_User FOR LOGIN ER_Limited;
GRANT SELECT, INSERT ON SCHEMA::dbo TO ER_Limited_User;
```

Read-Only User

```
CREATE LOGIN ER_ReadOnly WITH PASSWORD='ReadOnly@123';
CREATE USER ER_ReadOnly_User FOR LOGIN ER_ReadOnly;
EXEC sp_addrolemember 'db_datareader','ER_ReadOnly_User';
```

These roles simulate real healthcare governance.

9. Conclusion

This project demonstrates end-to-end capability in database design, healthcare data modeling, structured SQL development, synthetic data generation, analytics view creation, and security implementation. The structure closely mirrors a real ER or insurance claim environment and provides clear value for analytics, reporting, and operational insight making it highly relevant for data, healthcare, and analytics internships.

10. Future Improvements

To expand this project into a more advanced healthcare analytics model, future enhancements may include:

- **Medication / Treatment Tables**
  Tracking medications administered during ER visits and dosage information.
- **Billing & Claims Module**
  Adding claim status, reimbursement amounts, and denial reasons for insurance workflows.
- **Visit Severity Scores**
  Categorizing ER visits by severity level for triage analytics.
- **Patient Notes / Clinical Documentation**
  Storing short clinical notes or attending staff comments.
- **Power BI / Dashboard Layer**
  Creating interactive dashboards for diagnosis frequency, staff productivity, and cost analysis.
- **Machine Learning Data Prep**
  Preparing the dataset for predictive modeling, such as predicting visit volume or diagnosis risk.

11. How to Run This Project

Step 1 — Create the Database

```
CREATE DATABASE ER_Hospital;
GO
USE ER_Hospital;
```

Step 2 — Run the Schema File
Run **01_database_schema.sql**
(This creates all five tables with relationships.)

Step 3 — Load Sample Data
Run **02_insert_data.sql**
(This populates the database with synthetic patient, diagnosis, staff, and visit records.)

Step 4 — Create Reporting Views
Run **03_analytics_views.sql**
(This builds views used for analytics and reporting.)

Step 5 — Configure Security
Run **04_security_roles.sql**
(This sets up admin, limited-access, and read-only accounts.)

Step 6 — Start Querying
Example:

```
SELECT * FROM PatientVisitReport;
SELECT * FROM StaffActivitySummary;
```