<u>FINAL REPORT</u>

ISYS 864

Healthcare Management System:

A John Hopkins Case Study

<u>REPORT BY</u>:

GROUP 3

Bhakti Kate
Vivid Liu
Anushka Mondal
Ananyaa Shahi

26 May 2024

Data Managaement for Analytics

Professor Guillame Faddoul

Lam Family College of Business

San Francisco State University

INTRODUCTION

In today's fast-paced healthcare environment, efficient management of hospitals and pharmacy operations is critical to ensuring high-quality patient care and services. The Johns Hopkins Hospital, a non-profit academic medical center in Baltimore, Maryland, sets healthcare standards in patient care, research, and education. We decided to work with Johns Hopkins Hospital due to its prominence in the healthcare industry and the complexity of pharmacy operations within such a renowned academic medical center.

Our area of focus is on patient care and pharmacy operations. In Patient Care, we have entities including Patients, Medical Records, Departments, Employees, Physicians, Nurses, Billing & Insurance. Meanwhile, the Pharmacy Operations within the hospital will encompass a wide range of functions, including Medicine, Prescriptions, Inventory, and Transactions. The database and data we created centers around optimizing patient care and pharmacy operations, enhancing patient satisfaction, and ensuring regulatory compliance within Johns Hopkins Hospital.

OBJECTIVE

This project's objective is to design and implement a comprehensive database management system that streamlines patient care and pharmacy operations. The database will facilitate efficient management of patients, medical records, billing, insurance, pharmacy transactions, and more. By working on a Healthcare Management System project, we can gain exposure to a critical practical, real-world application. Additionally, working with pharmacies provides us access to diverse pharmacy workflows, patient populations, and clinical settings, allowing us to design a comprehensive and realistic database solution. Moreover, the project will involve creating a set of SQL queries to build tables, load data, and perform essential data retrieval tasks, thereby providing a practical and scalable solution to the hospital's data management needs.

ENTITIES AND THEIR DESCRIPTION

Patients - The "patients" entity stores information about individuals receiving medical care. It includes details like a unique patient identifier (PatientID), their first and last name, date of birth, contact information (address and phone number). Below is the list of their attributes:

- PatientID
- PatientName(FirstName, LastName)
- DateOfBirth
- Address
- Phone

Medical Records - The "medical records" entity holds detailed medical information for patients. It links to a specific patient using the PatientID and stores crucial data like Allergies (Any substances the patient has a negative reaction to), Medications (Current and past medications prescribed to the patient), Immunizations (Record of received vaccinations and their dates), Medical History (Past illnesses, surgeries, and other relevant medical events) & Family History (Health conditions that run in the patient's family). Below is the list of attributes for this entity:

- MedicalRecordID
- Allergies
- Medications
- Immunizations
- MedicalHistory
- FamilyHistory

Departments - The "departments" entity represents functional divisions within a healthcare organization. It uses a unique identifier (DepartmentID) for reference and stores details about the department's function (DepartmentName), its physical location (Location), its size (DepartmentSize), its allocated budget (DepartmentBudget), and a contact number (ContactNo) for communication purposes**.** It connects to employees

- DepartmentID
- DepartmentName
- Location
- Department Size
- Department Budget
- ContactNo

Employees- The "employee" entity represents general staff within a healthcare organization. It captures core details like a unique employee identifier (EmployeeID), full name (FirstName and LastName), their area of specialization (Specialization), and their salary (Salary). Also, the employee type (EmployeeType) acts as a subtype, allowing for categorization as either "physician" or "nurse". Below are the attributes for "Employee":

- EmployeeID
- EmployeeName (FirstName, LastName)
- Specialization
- Salary
- EmployeeType

Physicians - The "physicians" entity inherits all attributes from its supertype "employee". This means it includes a unique identifier (PhysicianID), full name (FirstName and LastName), area of specialization (Specialization), and salary (Salary)**.** Here is the full list of its attributes:

- PhysicianID
- EmployeeName (FirstName,LastName)
- Specialty
- Salary

Nurses - The "nurses" entity inherits all attributes from the "employee" supertype, like EmployeeID, names, salary and speciality.This entity has its own unique attribute namely LicenseNumber. Below is the list of its attributes:

- NurseID
- EmployeeName (FirstName,LastName)
- Specialization
- Salary
- LicenseNumber

Billing & Insurance - The "billing & insurance" entity tracks financial transactions for patient care. It uses a unique identifier (BillingInsuranceID) and links to a specific patient. It then captures details about the service rendered, including the procedure code (ServiceProcedureCode) and the date of service (ServiceDate). Financial information includes the total charge for the service (ChargeAmount) and the insurance provider involved (InsuranceProvider). Finally, it tracks the claim status with the insurer (InsuranceClaimStatus), indicating if it's pending, approved, denied, or requires further action. Below is the list of its attributes:

- BillingInsuranceID
- ServiceProcedureCode
- ServiceDate
- ChargeAmount
- InsuranceProvider
- InsuranceClaimStatus

Prescriptions - The "prescriptions" entity manages medication orders within a healthcare system. It uses a unique identifier (PrescriptionID) and links to the specific patient, the medication that they are receiving and the transcations involved for a prescription. The record also includes the doctor who prescribed the medication (PrescribingPhysician) along with dosage instructions (DosageInstructions) and the date the prescription was written (PrescriptionDate). This entity provides a vital link between patients, physicians, and dispensed medications. Below is the list of its attributes:

- PrescriptionID
- DosageInstructions
- PrescriptionDate
- PrescribingPhysicianID

Medical Supplies - The "medicine" entity forms the core of a pharmacy management system. It holds information about individual medications using a unique identifier (MedicineID). Each record details the medication name (MedicationName), its strength (concentration), and the dosage form (e.g., tablet, capsule, syrup). Information about the manufacturer (Manufacturer) is also stored. Most importantly, it tracks the medication's expiration date (ExpirationDate) to ensure patient safety and proper inventory management. It has the following attributes:

- MedicineID
- SupplyName
- Strength
- Type
- Manufacturer
- ExpiryDate

Inventory - The "inventory" entity serves as an important part of pharmacy stock management, featuring unique identifiers (InventoryID) linked to specific medications. It meticulously tracks vital inventory details such as the current quantity of medication units available (QuantityOnHand), unit price (UnitPrice), minimum stock level (ReorderLevel) prompting reorders when inventory falls below, and the date of the last medication reorder. This information allows for informed decisions about restocking medications, preventing stockouts and ensuring patients can receive their prescriptions without delays. It has the following attributes:

- InventoryID
- QuantityOnHand
- UnitPrice
- ReorderLevel
- LastReorderDate

Transactions - The "transactions" entity captures details about each medication dispensed at the pharmacy. It uses a unique identifier (TransactionID) and links to the specific prescription being filled. It also records the pharmacist who dispensed the medication and the date of the transaction (TransactionDate). Additionally, it tracks the quantity of medication dispensed (QuantityDispensed) and the total price paid (TotalPrice). This entity provides an audit trail of dispensed medications, ensuring accurate inventory control, billing, and patient medication adherence. It has the following attributes:

- TransactionID
- PrescriptionID
- PharmacistID
- TransactionDate
- QuantityDispensed
- TotalPrice

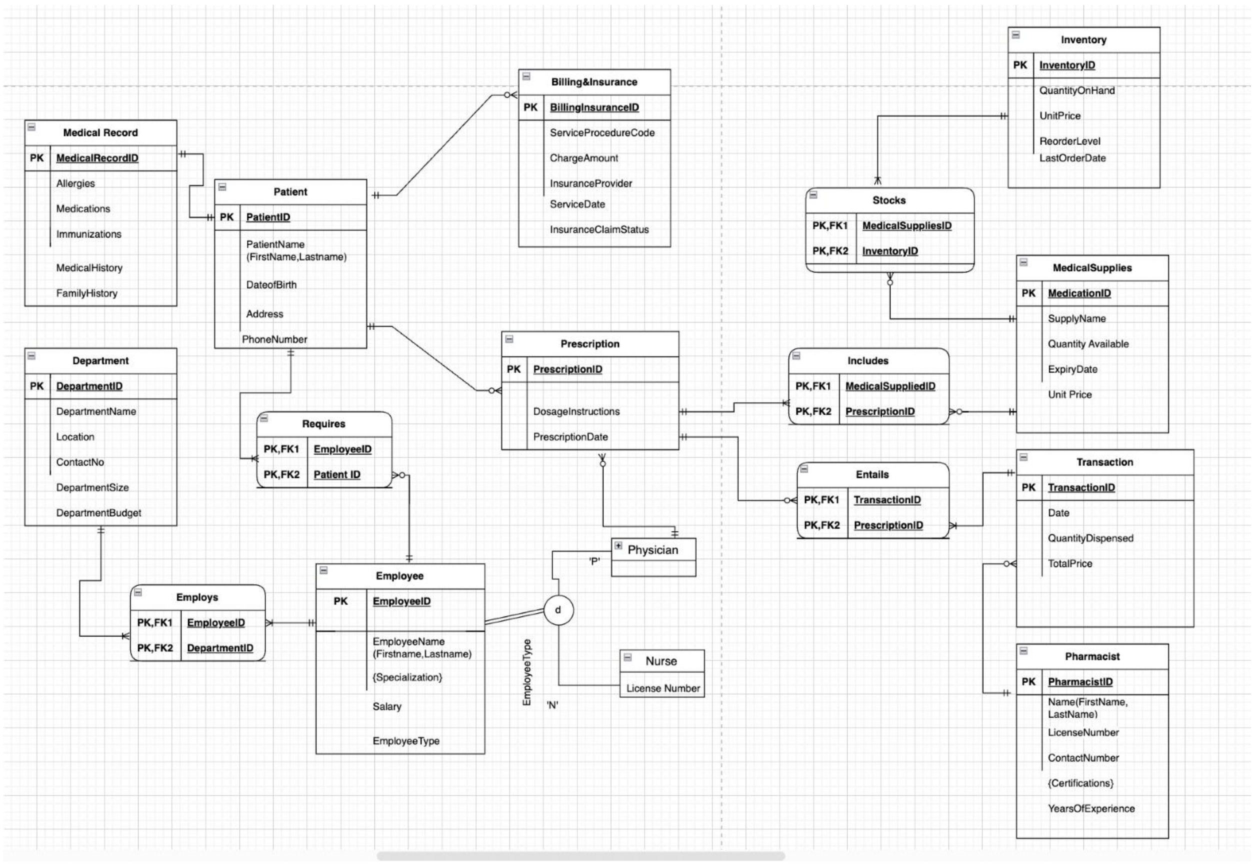
Pharmacist- The "pharmacist" entity stores information about licensed pharmacy professionals. Each record has a unique identifier (PharmacistID) and details about the pharmacist's name (FirstName, LastName). It also captures critical credentials like their license number (LicenseNumber) to ensure they are authorized to dispense medications. For communication purposes, the entity stores the pharmacist's contact number (ContactNumber). Additionally, it includes details about their experience level (YearsOfExperience) and any certifications (Certifications) they hold that demonstrate their expertise in specific areas of pharmacy practice. It has the following attributes:

- PharmacistID
- PharmacistName(FirstName, LastName)
- License Number
- ContactNumber
- YearsOfExperience
- Certifications

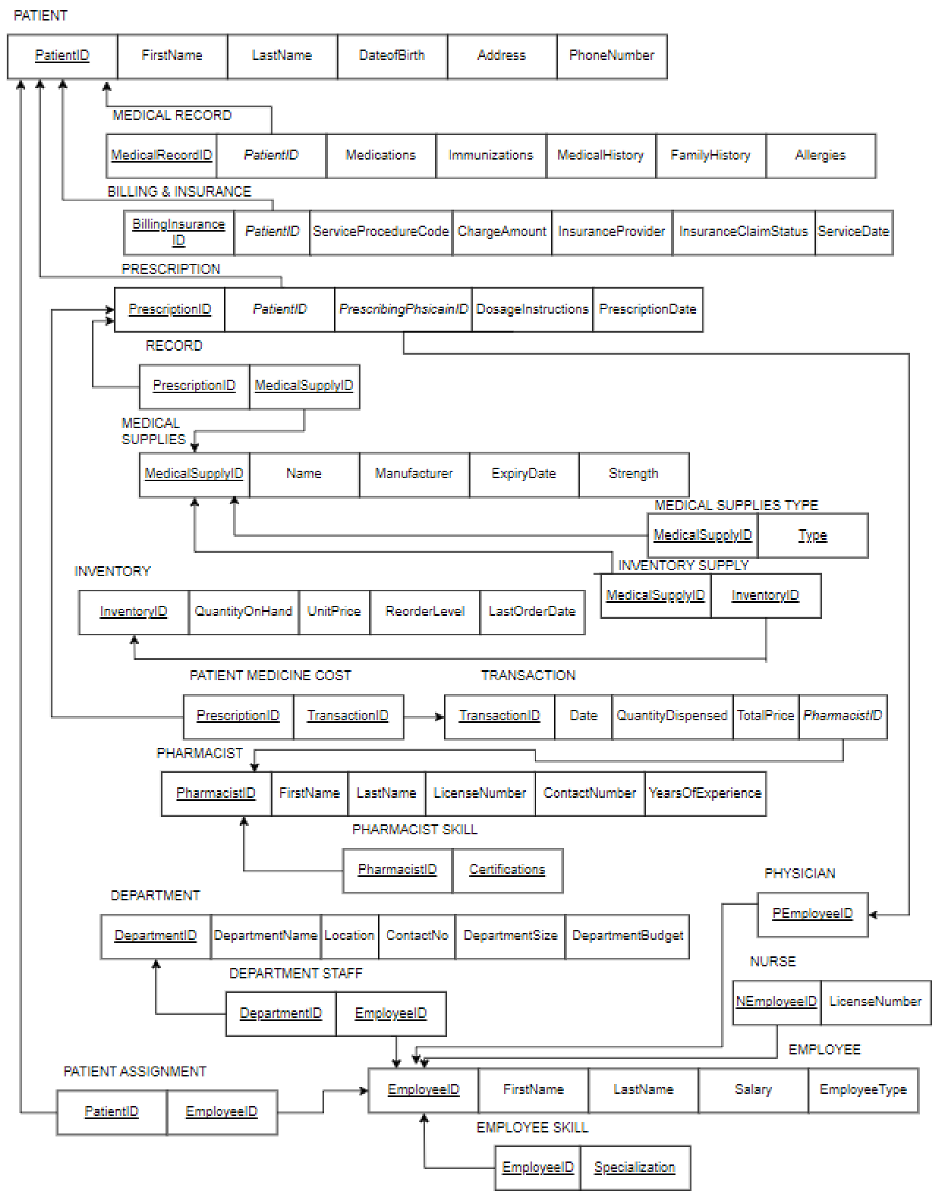
BUSINESS RULES:

1. A Patient must have one and only one Medical Record (1:1).
2. A Medical Record must belong to one and only one Patient (1:1).
3. A Patient can have zero or more Billing & Insurance entries (1:M).
4. A Billing & Insurance entry must belong to one and only one Patient (1:1).
5. A Physician must belong to at least one Department (1:M).
6. A Department must have one or more Physicians (1:M).
7. A Department must have one or more Nurses (1:M).
8. A Nurse must belong to at least one Department (1:M).
9. A Prescription must be assigned to one and only Patient (1:M).
10. A Patient can have zero or more Prescriptions (1:M).
11. A Prescription must be written by one Physician (1:1).
12. A Physician can write zero or more Prescriptions (1:M).
13. A Prescription must have one or more MedicalSupplies (1:M).
14. A MedicalSupply can be associated with zero or more Prescriptions (1:M).
15. A Prescription can have zero or more Transactions (1:M).
16. A transaction must be associated with at least one prescription(1:M)
17. An inventory must have one or more medical supplies (1:M).
18. A medical supply can be in zero or more inventory(1:M)
19. A Transaction must be processed by only one Pharmacist (1:1).
20. A pharmacist can process zero or more transactions(1:M)
21. A physician can have zero or more patients (1:M)
22. A patient must be associated with at least one physician (1:M)
23. A nurse can have zero or more patients (1:M)
24. A patient must be associated with at least one nurse (1:M)

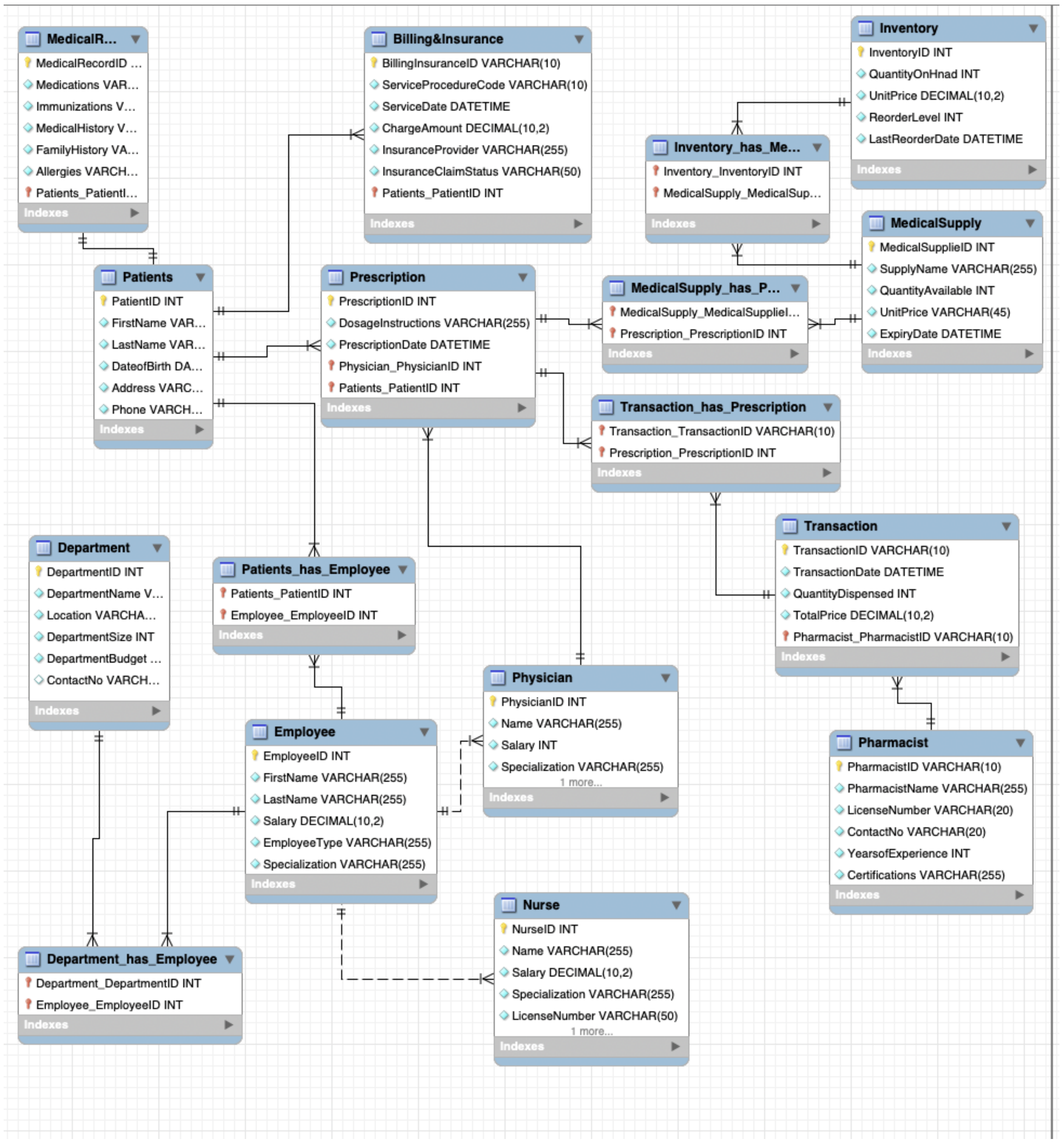
EER DIAGRAM


Inventory
PK InventoryID
QuantityOnHand
UnitPrice
ReorderLevel
LastOrderDate
Billing&Insurance
PK BillingInsuranceID
ServiceProcedureCode
ChargeAmount
InsuranceProvider
ServiceDate
InsuranceClaimStatus
Medical Record
PK MedicalRecordID
Allergies
Medications
Immunizations
MedicalHistory
FamilyHistory
Patient
PK PatientID
PatientName (FirstName,Lastname)
DateofBirth
Address
PhoneNumber
Stocks
PK,FK1 MedicalSuppliesID
PK,FK2 InventoryID
MedicalSupplies
PK MedicationID
SupplyName
Quantity Available
ExpiryDate
Unit Price
Department
PK DepartmentID
DepartmentName
Location
ContactNo
DepartmentSize
DepartmentBudget
Prescription
PK PrescriptionID
DosageInstructions
PrescriptionDate
Includes
PK,FK1 MedicalSuppliedID
PK,FK2 PrescriptionID
Requires
PK,FK1 EmployeeID
PK,FK2 Patient ID
Entails
PK,FK1 TransactionID
PK,FK2 PrescriptionID
Transaction
PK TransactionID
Date
QuantityDispensed
TotalPrice
Physician
'P'
Employee
PK EmployeeID
EmployeeName (Firstname,Lastname)
{Specialization}
Salary
EmployeeType
Employs
PK,FK1 EmployeeID
PK,FK2 DepartmentID
d
EmployeeType
Nurse
License Number
'N'
Pharmacist
PK PharmacistID
Name(FirstName, LastName)
LicenseNumber
ContactNumber
{Certifications}
YearsOfExperience

3NF RELATIONAL MODEL


PATIENT
PatientID | FirstName | LastName | DateofBirth | Address | PhoneNumber
MEDICAL RECORD
MedicalRecordID | PatientID | Medications | Immunizations | MedicalHistory | FamilyHistory | Allergies
BILLING & INSURANCE
BillingInsurance ID | PatientID | ServiceProcedureCode | ChargeAmount | InsuranceProvider | InsuranceClaimStatus | ServiceDate
PRESCRIPTION
PrescriptionID | PatientID | PrescribingPhsicainID | DosageInstructions | PrescriptionDate
RECORD
PrescriptionID | MedicalSupplyID
MEDICAL SUPPLIES
MedicalSupplyID | Name | Manufacturer | ExpiryDate | Strength
MEDICAL SUPPLIES TYPE
MedicalSupplyID | Type
INVENTORY SUPPLY
MedicalSupplyID | InventoryID
INVENTORY
InventoryID | QuantityOnHand | UnitPrice | ReorderLevel | LastOrderDate
PATIENT MEDICINE COST
PrescriptionID | TransactionID
TRANSACTION
TransactionID | Date | QuantityDispensed | TotalPrice | PharmacistID
PHARMACIST
PharmacistID | FirstName | LastName | LicenseNumber | ContactNumber | YearsOfExperience
PHARMACIST SKILL
PharmacistID | Certifications
PHYSICIAN
PEmployeeID
DEPARTMENT
DepartmentID | DepartmentName | Location | ContactNo | DepartmentSize | DepartmentBudget
DEPARTMENT STAFF
DepartmentID | EmployeeID
NURSE
NEmployeeID | LicenseNumber
EMPLOYEE
EmployeeID | FirstName | LastName | Salary | EmployeeType
PATIENT ASSIGNMENT
PatientID | EmployeeID
EMPLOYEE SKILL
EmployeeID | Specialization

MYSQL ER DIAGRAM



SQL Queries

Creating and loading the data

create schema healthcare_db;

use healthcare_db;

Refer: Appendix A.1

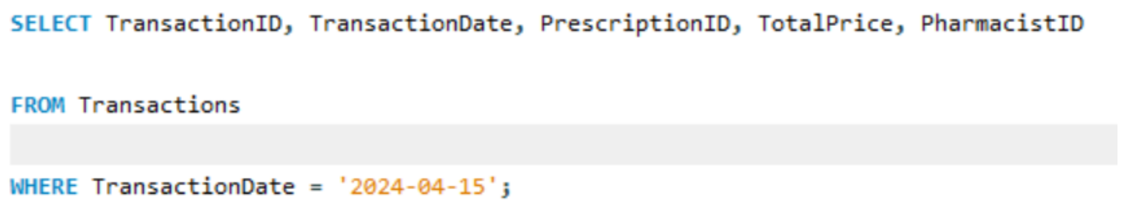
Data Retrieval SQL queries

Query 1.

Problem Statement: Create a report summarizing today's transactions.

Objective: The purpose of this query is to generate a daily report summarizing sales transactions to improve transparency and accountability. It would help in monitoring daily sales, track individual pharmacist performance, provide a financial overview, ensure prescriptions are correctly processed, and enhance operational efficiency. This query aids management in making informed decisions and maintaining a transparent sales process

Query:
SELECT TransactionID, TransactionDate, PrescriptionID, TotalPrice, PharmacistID
FROM Transactions
WHERE TransactionDate = CURRENT_DATE;

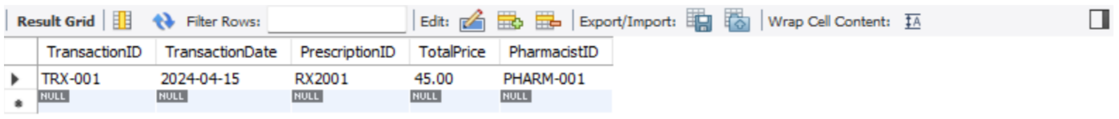


```
SELECT TransactionID, TransactionDate, PrescriptionID, TotalPrice, PharmacistID

FROM Transactions

WHERE TransactionDate = '2024-04-15';
```

Result returned:



TransactionID	TransactionDate	PrescriptionID	TotalPrice	PharmacistID
TRX-001	2024-04-15	RX2001	45.00	PHARM-001
NULL	NULL	NULL	NULL	NULL

Refer: Appendix A.2.

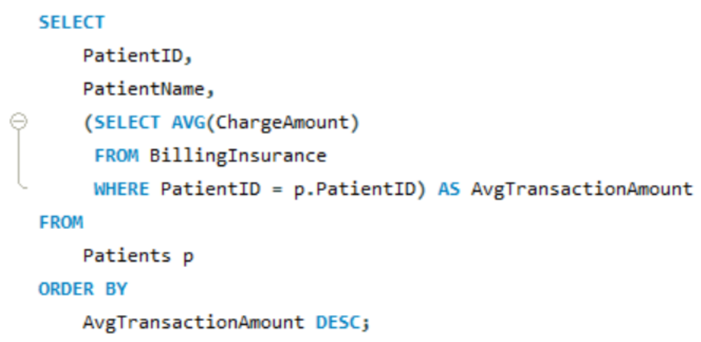
Query 2.

Problem Statement: Calculate the average transaction amount (total price paid) per patient, sorted in descending order

Objective: The purpose of this query is to calculate and rank patients by their average transaction amount in descending order. It aims to identify high-spending patients, support financial analysis, aid in resource allocation, and facilitate patient segmentation based on spending patterns.

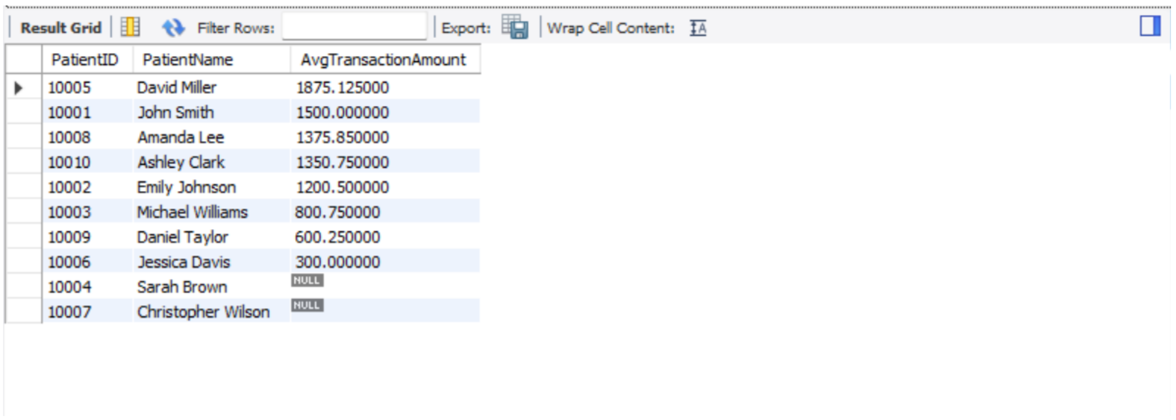
Query:

```
SELECT PatientID, PatientName,
(SELECT AVG(ChargeAmount)
    FROM BillingInsurance
WHERE PatientID = p.PatientID) AS AvgTransactionAmount
FROM Patients p
ORDER BY AvgTransactionAmount DESC;
```



```
SELECT
    PatientID,
    PatientName,
    (SELECT AVG(ChargeAmount)
     FROM BillingInsurance
     WHERE PatientID = p.PatientID) AS AvgTransactionAmount
FROM
    Patients p
ORDER BY
    AvgTransactionAmount DESC;
```

Result returned:



PatientID	PatientName	AvgTransactionAmount
10005	David Miller	1875.125000
10001	John Smith	1500.000000
10008	Amanda Lee	1375.850000
10010	Ashley Clark	1350.750000
10002	Emily Johnson	1200.500000
10003	Michael Williams	800.750000
10009	Daniel Taylor	600.250000
10006	Jessica Davis	300.000000
10004	Sarah Brown	NULL
10007	Christopher Wilson	NULL

<u>Refer:</u> Appendix A.2.1, A.2.12

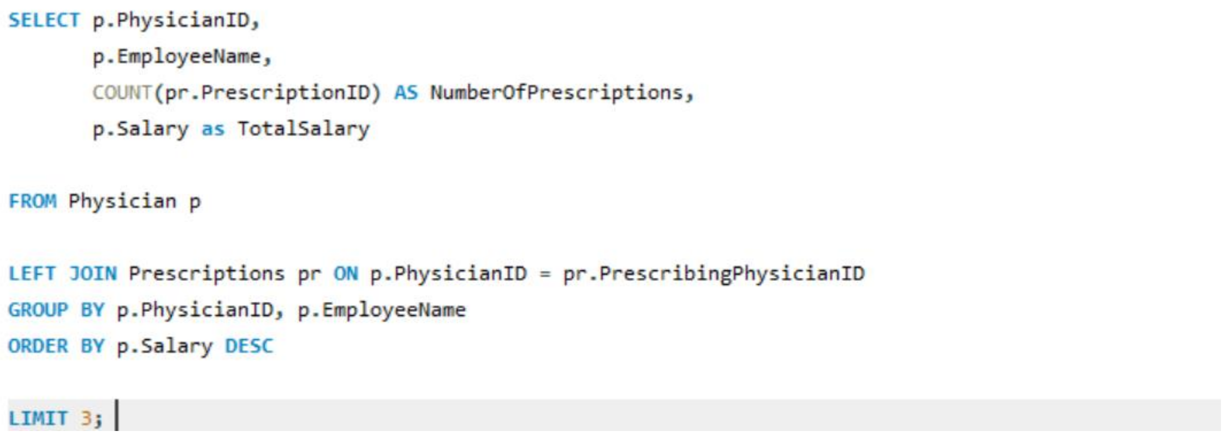
Query 3.

Problem Statement: Retrieve the top 3 physicians with the highest total salary, along with the number of prescriptions they have written.

Objective: The objective of this query is to identify the top 3 highest-paid physicians and the number of prescriptions they have written, providing insights into their earnings and productivity.

Query:

```
SELECT p.PhysicianID, p.EmployeeName,
COUNT(pr.PrescriptionID) AS NumberOfPrescriptions,
 p.Salary as TotalSalary
FROM Physician p
LEFT JOIN Prescriptions pr ON p.PhysicianID = pr.PrescribingPhysicianID
GROUP BY p.PhysicianID, p.EmployeeName
ORDER BY p.Salary DESC
LIMIT 3;
```



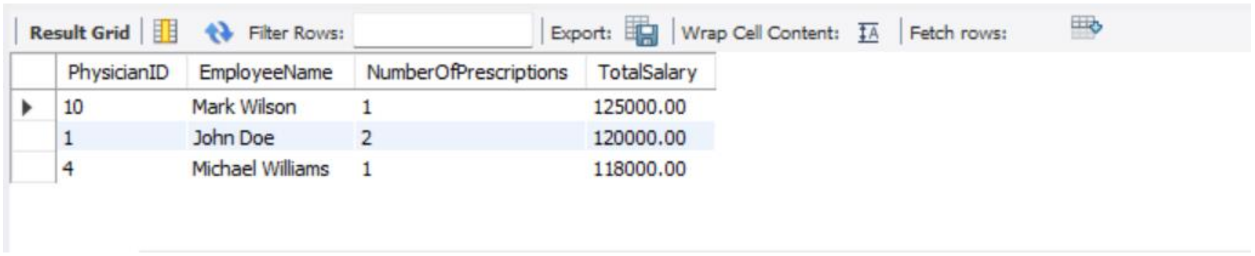
```
SELECT p.PhysicianID,
       p.EmployeeName,
       COUNT(pr.PrescriptionID) AS NumberOfPrescriptions,
       p.Salary as TotalSalary

FROM Physician p

LEFT JOIN Prescriptions pr ON p.PhysicianID = pr.PrescribingPhysicianID
GROUP BY p.PhysicianID, p.EmployeeName
ORDER BY p.Salary DESC

LIMIT 3;
```

Result returned:



PhysicianID	EmployeeName	NumberOfPrescriptions	TotalSalary
10	Mark Wilson	1	125000.00
1	John Doe	2	120000.00
4	Michael Williams	1	118000.00

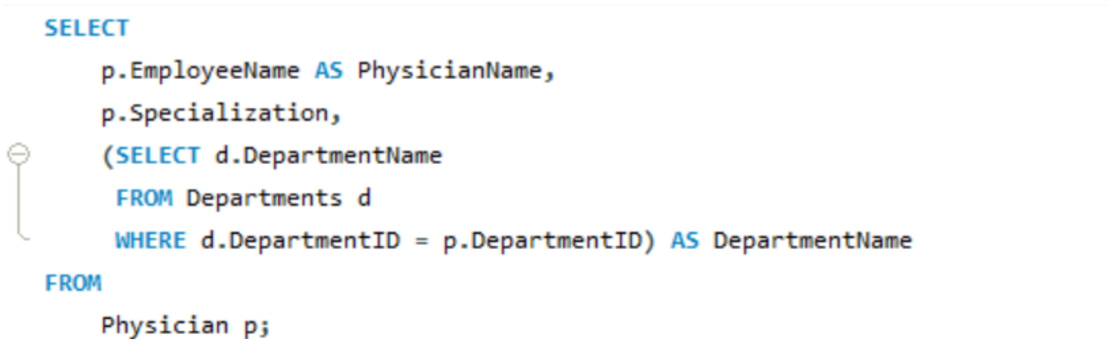
<u>**Refer:**</u> Appendix A.2.6, A.2.7

Query 4.

Problem Statement: Retrieve the names and specialties of physicians along with the department they belong to.

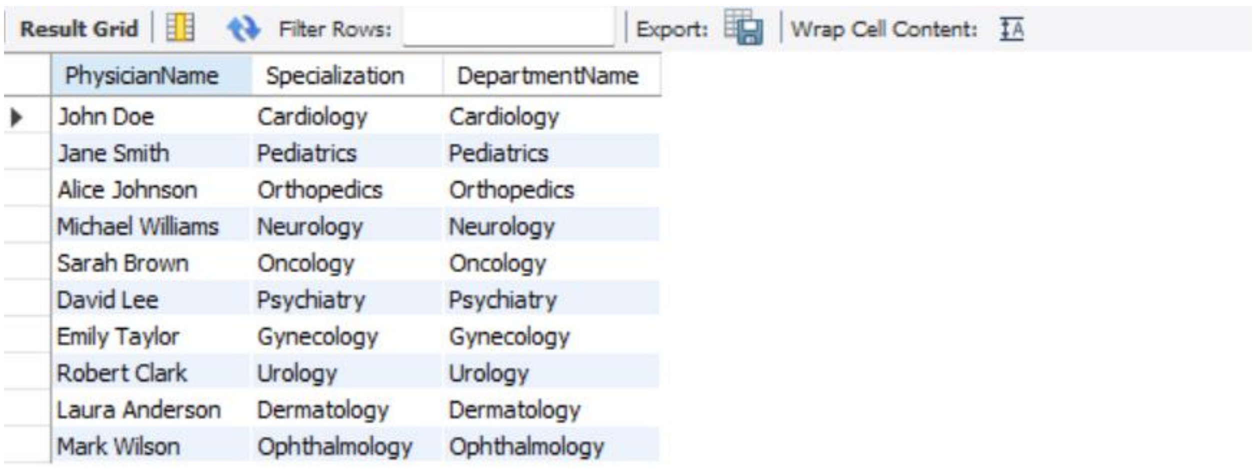
Objective: The objective of this query is to retrieve the names, specialties, and departments of physicians, providing a comprehensive overview of their roles and departmental affiliations.

Query:
SELECT p.EmployeeName AS PhysicianName, p.Specialization,
(SELECT d.DepartmentName
FROM Departments d
WHERE d.DepartmentID = p.DepartmentID) AS DepartmentName
FROM Physician p;



```
SELECT
    p.EmployeeName AS PhysicianName,
    p.Specialization,
    (SELECT d.DepartmentName
     FROM Departments d
     WHERE d.DepartmentID = p.DepartmentID) AS DepartmentName
FROM
    Physician p;
```

Result returned:



PhysicianName	Specialization	DepartmentName
John Doe	Cardiology	Cardiology
Jane Smith	Pediatrics	Pediatrics
Alice Johnson	Orthopedics	Orthopedics
Michael Williams	Neurology	Neurology
Sarah Brown	Oncology	Oncology
David Lee	Psychiatry	Psychiatry
Emily Taylor	Gynecology	Gynecology
Robert Clark	Urology	Urology
Laura Anderson	Dermatology	Dermatology
Mark Wilson	Ophthalmology	Ophthalmology

<u>Refer:</u> Appendix A.2.3, A.2.6

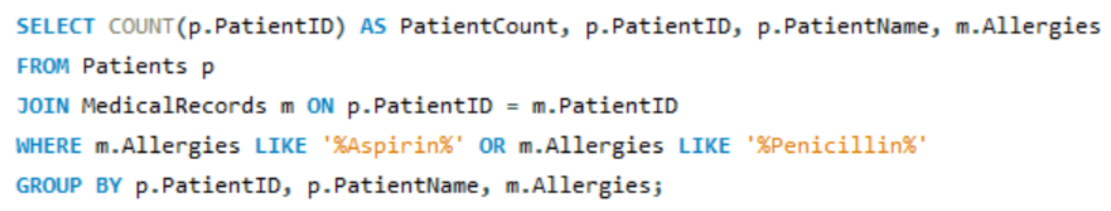
Query 5.

Problem Statement: How many patients have allergies to either Aspirin or Penicillin, and what are their names and corresponding allergies?

Objective: The objective of this query is to identify and count patients with allergies to Aspirin or Penicillin, providing their names and corresponding allergies. This helps improve patient safety and care by ensuring awareness of these specific allergies.

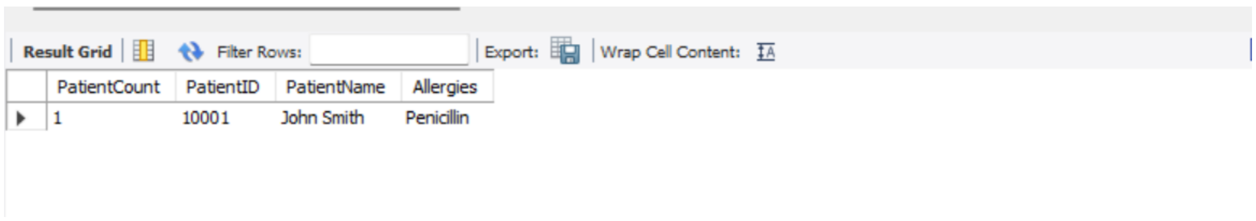
Query:

SELECT COUNT(p.PatientID) AS PatientCount, p.PatientID, p.PatientName, m.Allergies
FROM Patients p
JOIN MedicalRecords m ON p.PatientID = m.PatientID
WHERE m.Allergies LIKE '%Aspirin%' OR m.Allergies LIKE '%Penicillin%'
GROUP BY p.PatientID, p.PatientName, m.Allergies;



```
SELECT COUNT(p.PatientID) AS PatientCount, p.PatientID, p.PatientName, m.Allergies
FROM Patients p
JOIN MedicalRecords m ON p.PatientID = m.PatientID
WHERE m.Allergies LIKE '%Aspirin%' OR m.Allergies LIKE '%Penicillin%'
GROUP BY p.PatientID, p.PatientName, m.Allergies;
```

Result returned:



PatientCount	PatientID	PatientName	Allergies
1	10001	John Smith	Penicillin

<u>**Refer:**</u> Appendix A.2.1, A.2.2

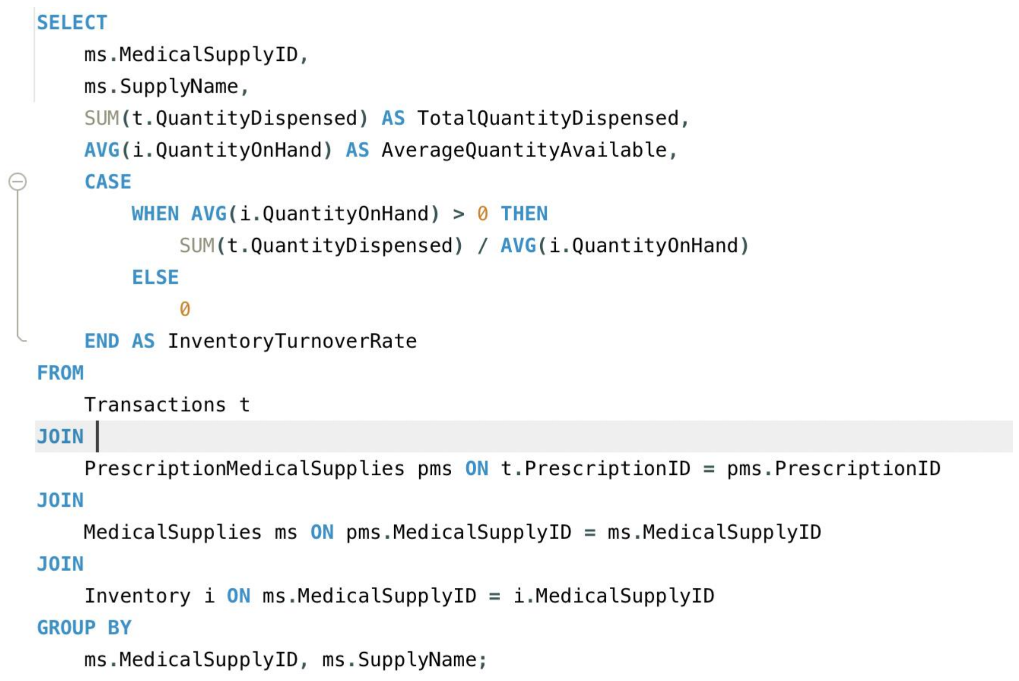
Query 6

Problem Statement: Calculate the inventory turnover rate for each medication based on transactions**.**

Objective: The objective of this query is to calculate key inventory metrics for medical supplies, including total quantity dispensed, average quantity available, and inventory turnover rate, to help manage inventory effectively.

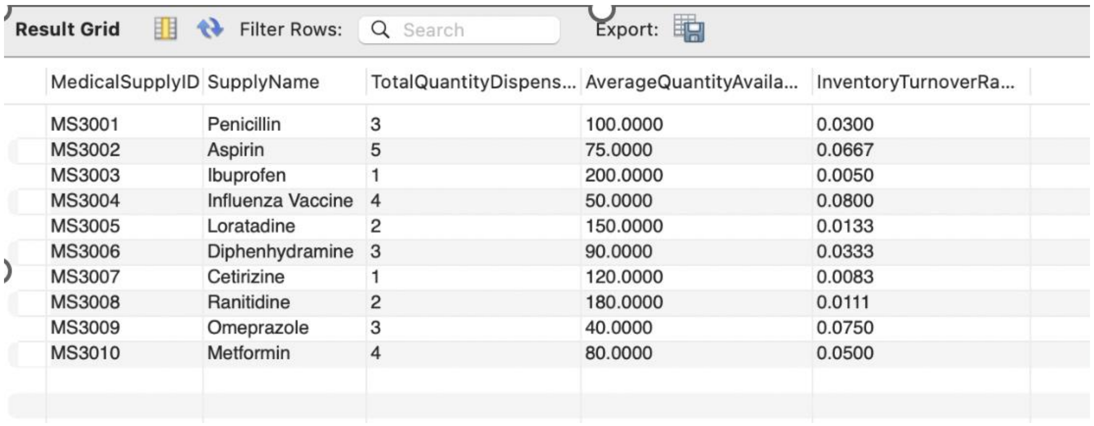
Query:

```
SELECT ms.MedicalSupplyID,ms.SupplyName, SUM(t.QuantityDispensed) AS
TotalQuantityDispensed, AVG(i.QuantityOnHand) AS AverageQuantityAvailable,
CASE
    WHEN AVG(i.QuantityOnHand) > 0 THEN
     SUM(t.QuantityDispensed) / AVG(i.QuantityOnHand)
ELSE 0
END AS InventoryTurnoverRate
FROM Transactions t
JOIN PrescriptionMedicalSupplies pms ON t.PrescriptionID = pms.PrescriptionID
JOIN MedicalSupplies ms ON pms.MedicalSupplyID = ms.MedicalSupplyID
JOIN Inventory i ON ms.MedicalSupplyID = i.MedicalSupplyID
GROUP BY ms.MedicalSupplyID, ms.SupplyName;
```



```
SELECT
    ms.MedicalSupplyID,
    ms.SupplyName,
    SUM(t.QuantityDispensed) AS TotalQuantityDispensed,
    AVG(i.QuantityOnHand) AS AverageQuantityAvailable,
    CASE
        WHEN AVG(i.QuantityOnHand) > 0 THEN
            SUM(t.QuantityDispensed) / AVG(i.QuantityOnHand)
        ELSE
            0
    END AS InventoryTurnoverRate
FROM
    Transactions t
JOIN
    PrescriptionMedicalSupplies pms ON t.PrescriptionID = pms.PrescriptionID
JOIN
    MedicalSupplies ms ON pms.MedicalSupplyID = ms.MedicalSupplyID
JOIN
    Inventory i ON ms.MedicalSupplyID = i.MedicalSupplyID
GROUP BY
    ms.MedicalSupplyID, ms.SupplyName;
```

Result Returned:



Result Grid | Filter Rows: Search | Export:

MedicalSupplyID	SupplyName	TotalQuantityDispens...	AverageQuantityAvaila...	InventoryTurnoverRa...
MS3001	Penicillin	3	100.0000	0.0300
MS3002	Aspirin	5	75.0000	0.0667
MS3003	Ibuprofen	1	200.0000	0.0050
MS3004	Influenza Vaccine	4	50.0000	0.0800
MS3005	Loratadine	2	150.0000	0.0133
MS3006	Diphenhydramine	3	90.0000	0.0333
MS3007	Cetirizine	1	120.0000	0.0083
MS3008	Ranitidine	2	180.0000	0.0111
MS3009	Omeprazole	3	40.0000	0.0750
MS3010	Metformin	4	80.0000	0.0500

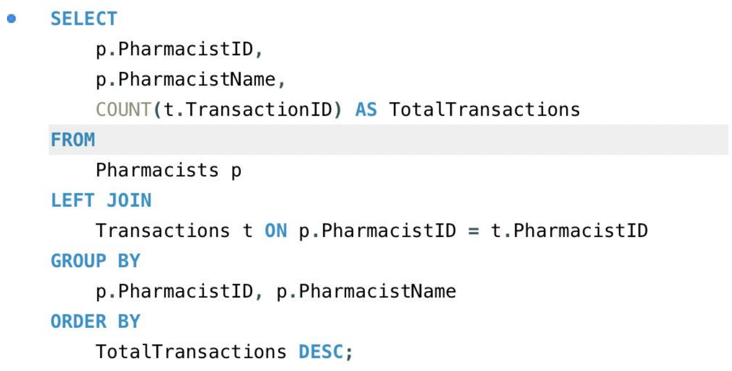
<u>Refer:</u> Appendix A.2.13, A.2.4, A.2.5, A.2.9

Query 7.

Problem Statement: Retrieve the total number of transactions processed by each pharmacist, along with their names.

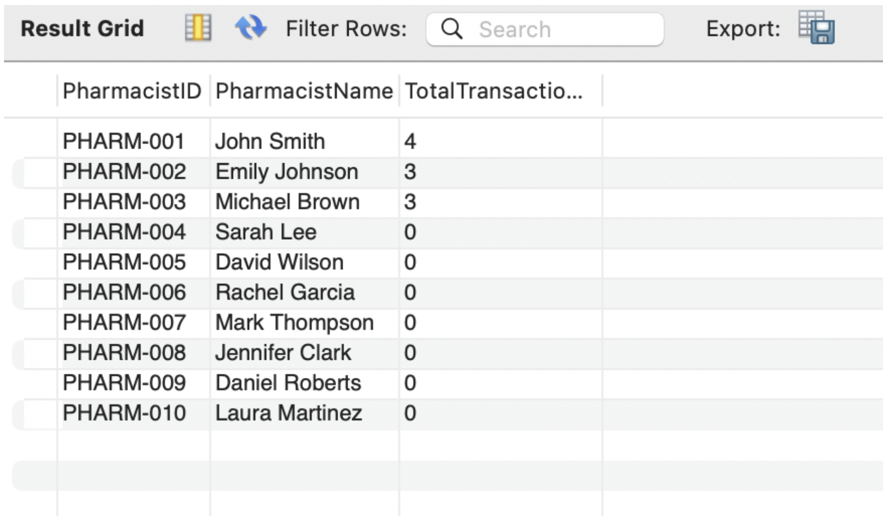
Objective: The objective of this query is to count and rank pharmacists by the number of transactions they have processed, providing insights into their activity and productivity.

Query:
```
SELECT p.PharmacistID, p.PharmacistName,
COUNT(t.TransactionID) AS TotalTransactions
FROM Pharmacists p
LEFT JOIN Transactions t ON p.PharmacistID = t.PharmacistID
GROUP BY p.PharmacistID, p.PharmacistName
ORDER BY TotalTransactions DESC;
```



```
SELECT
    p.PharmacistID,
    p.PharmacistName,
    COUNT(t.TransactionID) AS TotalTransactions
FROM
    Pharmacists p
LEFT JOIN
    Transactions t ON p.PharmacistID = t.PharmacistID
GROUP BY
    p.PharmacistID, p.PharmacistName
ORDER BY
    TotalTransactions DESC;
```

Result returned:



PharmacistID	PharmacistName	TotalTransactio...
PHARM-001	John Smith	4
PHARM-002	Emily Johnson	3
PHARM-003	Michael Brown	3
PHARM-004	Sarah Lee	0
PHARM-005	David Wilson	0
PHARM-006	Rachel Garcia	0
PHARM-007	Mark Thompson	0
PHARM-008	Jennifer Clark	0
PHARM-009	Daniel Roberts	0
PHARM-010	Laura Martinez	0

Refer: Appendix A.2.8, A.2.9

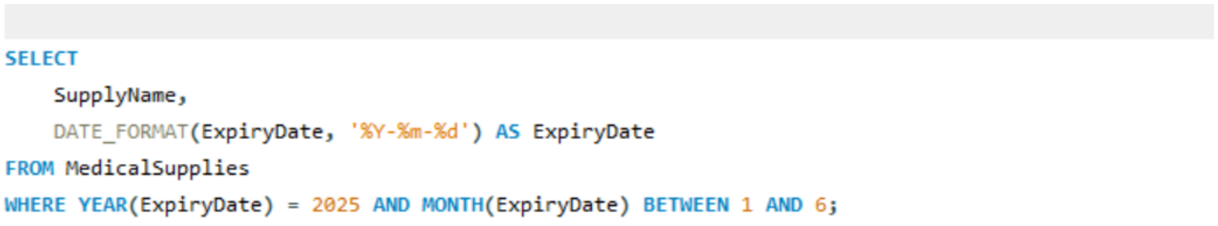
Query 8.

Problem Statement: Retrieve the medication names and expiration dates of medical supplies expiring in the first 6 months of 2025.

Objective: The objective of this query is to identify medical supplies expiring in the first half of 2025, aiding in proactive inventory management and reducing waste.

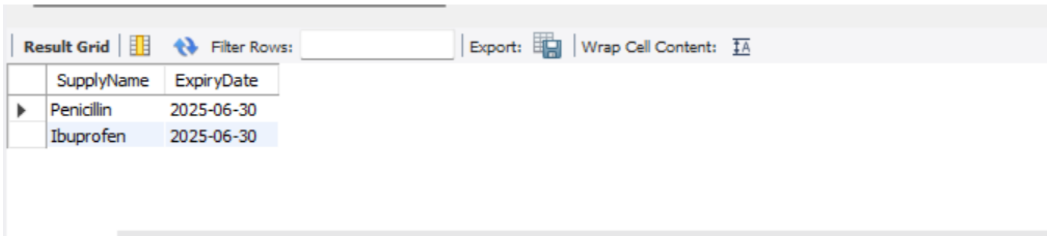
Query:
SELECT SupplyName, DATE_FORMAT(ExpiryDate, '%Y-%m-%d') AS ExpiryDate

FROM MedicalSupplies
WHERE YEAR(ExpiryDate) = 2025 AND MONTH(ExpiryDate) BETWEEN 1 AND 6;



```
SELECT
    SupplyName,
    DATE_FORMAT(ExpiryDate, '%Y-%m-%d') AS ExpiryDate
FROM MedicalSupplies
WHERE YEAR(ExpiryDate) = 2025 AND MONTH(ExpiryDate) BETWEEN 1 AND 6;
```

Results returned:



SupplyName	ExpiryDate
Penicillin	2025-06-30
Ibuprofen	2025-06-30

Refer: Appendix A.2.4

CONCLUSION

This project not only developed our technical understanding of database management but also provided valuable insights into the medical industry. We got a chance to get acquainted with the operations of John Hopkins Hospital ranging from patient care to pharmacy operations. Moreover, this project's lifecycle fostered team collaboration where diverse perspectives helped in problem solving. We understood that to address a specific industry problem we must first learn the industry's working. Furthermore, developing the ER diagram and Queries enhanced our analyzing skills and helped us get a deeper understanding of the subject.

The project did have a few challenges which we would like to mention here. The major was time constraints, with more time on hand we would have explored more queries. Also, limited experience and domain knowledge created hindrance in developing accurate queries. Moreover, our project's scope involved mostly all major hospital operations from patient entry to inventory check for their medicine, increasing the project's complexity. A narrow and department focused approach would have suited better here.

APPENDIX

A.1 Database Creation

```
CREATE SCHEMA healthcare_db;

USE healthcare_db;
```

A.2 Table Definitions and Data Insertion

A.2.1 Patients Table

```
CREATE TABLE Patients (
    PatientID INT PRIMARY KEY,
    PatientName VARCHAR(255),
    DateOfBirth DATE,
    Address VARCHAR(255),
    Phone VARCHAR(20)
);
INSERT INTO Patients (PatientID, PatientName, DateOfBirth, Address, Phone)
VALUES
    (10001, 'John Smith', '1980-05-12', '123 Main St, Cityville, CA', '723-452-5789'),
    (10002, 'Emily Johnson', '1975-08-20', '456 Oak St, Townsburg, NY', '232-537-9861'),
    (10003, 'Michael Williams', '1990-03-15', '789 Elm St, Villagetown, TX', '828-631-9059'),
    (10004, 'Sarah Brown', '1970-11-10', '321 Pine St, Hamletown, FL', '208-573-1926'),
    (10005, 'David Miller', '1985-09-25', '654 Cedar St, Riverside, WA', '284-305-4646'),
    (10006, 'Jessica Davis', '1988-02-18', '987 Birch St, Mountainville, CO', '956-367-2154'),
    (10007, 'Christopher Wilson', '1992-06-30', '741 Maple St, Baytown, AL', '285-423-7934'),
    (10008, 'Amanda Lee', '1983-04-05', '369 Walnut St, Beachville, MA', '583-056-4945'),
    (10009, 'Daniel Taylor', '1987-07-08', '852 Oak St, Hillside, NV', '705-472-3892'),
    (10010, 'Ashley Clark', '1995-12-23', '147 Pine St, Lakeside, OR', '347-952-3509');
```

A.2.2 Medical Records Table

```
CREATE TABLE MedicalRecords (
    MedicalRecordID INT PRIMARY KEY,
    Allergies VARCHAR(255),
    Medications VARCHAR(255),
    Immunizations VARCHAR(255),
    MedicalHistory VARCHAR(255),
    FamilyHistory VARCHAR(255),
    PatientID INT,
    FOREIGN KEY (PatientID) REFERENCES Patients(PatientID)
);
INSERT INTO MedicalRecords (MedicalRecordID, Allergies, Medications, Immunizations,
MedicalHistory, FamilyHistory, PatientID)
VALUES
    (15235, 'Penicillin', 'Aspirin', 'MMR', 'Hypertension', 'Diabetes', 10001),
    (23405, 'Sulfa Drugs', 'Ibuprofen', 'Influenza', 'Asthma', 'Cancer', 10002),
    (33457, 'Peanuts', 'Acetaminophen', 'Tetanus', 'High Cholesterol', 'Alzheimer''s', 10003),
    (34512, 'Shellfish', 'Naproxen', 'Hepatitis B', 'Diabetes', 'Heart Disease', 10004),
    (52358, 'Pollen', 'Loratadine', 'HPV', 'Arthritis', 'Stroke', 10005),
    (64596, 'Dust', 'Diphenhydramine', 'Varicella', 'Depression', 'Obesity', 10006),
    (73402, 'Cat Hair', 'Cetirizine', 'Pneumococcal', 'Anxiety', 'Parkinson''s', 10007),
    (82409, 'Dog Dander', 'Ranitidine', 'Hepatitis A', 'Migraines', 'Asthma', 10008),
    (93494, 'Mold', 'Omeprazole', 'Polio', 'Thyroid Disorder', 'Rheumatoid Arthritis', 10009),
    (10230, 'Insect Venom', 'Metformin', 'Measles', 'COPD', 'Multiple Sclerosis', 10010);
```

A.2.3 Departments Table

```
CREATE TABLE Departments (
    DepartmentID INT PRIMARY KEY,
    DepartmentName VARCHAR(255),
    Location VARCHAR(255),
    DepartmentSize INT,
    DepartmentBudget DECIMAL(10, 2),
    ContactNo VARCHAR(20)
);

INSERT INTO Departments (DepartmentID, DepartmentName, Location, DepartmentSize, DepartmentBudget, ContactNo)
VALUES
    (101, 'Cardiology', 'Mercy Hospital, Cityville', 50, 500000.00, '234-872-5491'),
    (102, 'Pediatrics', 'Children''s Hospital, Townsburg', 40, 400000.00, '346-823-1487'),
    (103, 'Orthopedics', 'Community Clinic, Villagetown', 30, 300000.00, '859-589-9874'),
    (104, 'Neurology', 'Regional Medical Center, Hamletown', 25, 250000.00, '903-472-7824');

INSERT INTO Departments (DepartmentID, DepartmentName, Location, DepartmentSize, DepartmentBudget, ContactNo)
VALUES
    (105, 'Oncology', 'Cancer Institute, Riverside', 35, '350000.00', '987-654-3210'),
    (106, 'Psychiatry', 'Mental Health Center, Mountainville', 20, '200000.00', '345-678-9012'),
    (107, 'Gynecology', 'Women''s Health Clinic, Baytown', 30, '300000.00', '789-012-3456'),
    (108, 'Urology', 'Urology Center, Beachville', 25, '250000.00', '210-543-6789'),
    (109, 'Dermatology', 'Skin Care Clinic, Hillside', 20, '200000.00', '654-321-0987'),
    (110, 'Ophthalmology', 'Eye Clinic, Lakeside', 15, '150000.00', '876-543-2109');
```

A.2.4 Medical Supplies Table

```
CREATE TABLE MedicalSupplies (
    MedicalSupplyID VARCHAR(10) PRIMARY KEY,
    SupplyName VARCHAR(255),
    QuantityAvailable INT,
    UnitPrice DECIMAL(10, 2),
     ExpiryDate DATE
);

INSERT INTO MedicalSupplies (MedicalSupplyID, SupplyName, QuantityAvailable,
UnitPrice, ExpiryDate)
VALUES
    ('MS3001', 'Penicillin', 100, 15.00, '2025-06-30'),
    ('MS3002', 'Aspirin', 200, 5.00, '2025-07-31'),
    ('MS3003', 'Ibuprofen', 150, 8.00, '2025-06-30'),
    ('MS3004', 'Influenza Vaccine', 50, 25.00, '2025-08-31'),
    ('MS3005', 'Loratadine', 100, 10.00, '2025-09-30'),
    ('MS3006', 'Diphenhydramine', 80, 12.00, '2025-10-31'),
    ('MS3007', 'Cetirizine', 120, 15.00, '2025-11-30'),
    ('MS3008', 'Ranitidine', 180, 8.00, '2025-12-31'),
    ('MS3009', 'Omeprazole', 40, 20.00, '2026-01-31'),
    ('MS3010', 'Metformin', 80, 15.00, '2026-02-28');
```

A.2.5 Inventory Table

```
CREATE TABLE Inventory (
    InventoryID VARCHAR(10) PRIMARY KEY,
    QuantityOnHand INT,
    UnitPrice DECIMAL(10, 2),
    ReorderLevel INT,
    LastReorderDate DATE,
    MedicalSupplyID VARCHAR(10),
    FOREIGN KEY (MedicalSupplyID) REFERENCES MedicalSupplies(MedicalSupplyID)
);

INSERT INTO Inventory (InventoryID, QuantityOnHand, UnitPrice, ReorderLevel,
LastReorderDate, MedicalSupplyID)
VALUES
    ('INV-001', 100, '25.00', 50, '2024-04-25', 'MS3001'),
    ('INV-002', 75, '18.50', 40, '2024-04-22', 'MS3002'),
    ('INV-003', 200, '10.00', 100, '2024-04-20', 'MS3003'),
    ('INV-004', 50, '30.00', 20, '2024-04-28', 'MS3004'),
    ('INV-005', 150, '12.75', 80, '2024-04-18', 'MS3005'),
    ('INV-006', 90, '22.00', 60, '2024-04-24', 'MS3006'),
    ('INV-007', 120, '15.25', 70, '2024-04-26', 'MS3007'),
    ('INV-008', 180, '9.50', 90, '2024-04-21', 'MS3008'),
    ('INV-009', 40, '35.00', 30, '2024-04-23', 'MS3009'),
    ('INV-010', 80, '20.50', 45, '2024-04-27', 'MS3010');
```

A.2.6 Physician Table

```
CREATE TABLE Physician (
    PhysicianID INT PRIMARY KEY,
    DepartmentID INT,
    EmployeeName VARCHAR(255),
    Specialization VARCHAR(255),
    Salary DECIMAL(10, 2),
    FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)
);

INSERT INTO Physician (PhysicianID, DepartmentID, EmployeeName, Specialization, Salary)
VALUES
    (1, 101, 'John Doe', 'Cardiology', 120000.00),
    (2, 102, 'Jane Smith', 'Pediatrics', 110000.00),
    (3, 103, 'Alice Johnson', 'Orthopedics', 115000.00),
    (4, 104, 'Michael Williams', 'Neurology', 118000.00),
    (5, 105, 'Sarah Brown', 'Oncology', 112000.00),
    (6, 106, 'David Lee', 'Psychiatry', 105000.00),
    (7, 107, 'Emily Taylor', 'Gynecology', 108000.00),
    (8, 108, 'Robert Clark', 'Urology', 113000.00),
    (9, 109, 'Laura Anderson', 'Dermatology', 106000.00),
    (10, 110, 'Mark Wilson', 'Ophthalmology', 125000.00);
```

A.2.7 Prescriptions Table

```
CREATE TABLE Prescriptions (
    PrescriptionID VARCHAR(10) PRIMARY KEY,
    DosageInstructions VARCHAR(255),
    PrescriptionDate DATE,
    PrescribingPhysicianID INT,
    PatientID INT,
    FOREIGN KEY (PrescribingPhysicianID) REFERENCES Physician(PhysicianID),
    FOREIGN KEY (PatientID) REFERENCES Patients(PatientID)
);

INSERT INTO Prescriptions (PrescriptionID, DosageInstructions, PrescriptionDate,
PrescribingPhysicianID, PatientID)
VALUES
    ('RX2001', 'Take twice daily', '2024-04-10', 1, 10001),
    ('RX2002', 'Take as needed', '2024-04-12', 1, 10002),
    ('RX2003', 'Take once daily', '2024-04-14', 7, 10002),
    ('RX2004', 'Take with meals', '2024-04-16', 2, 10004),
    ('RX2005', 'Take before bed', '2024-04-18', 2, 10005),
    ('RX2006', 'Take every 4 hours', '2024-04-20', 10, 10006),
    ('RX2007', 'Take in the morning', '2024-04-22', 4, 10003),
    ('RX2008', 'Take after meals', '2024-04-24', 3, 10008),
    ('RX2009', 'Take with water', '2024-04-26', 5, 10009),
    ('RX2010', 'Take with food', '2024-04-28', 6, 10007);
```

A.2.8 Pharmacists Table

```
CREATE TABLE Pharmacists (
    PharmacistID VARCHAR(10) PRIMARY KEY,
    PharmacistName VARCHAR(255),
    LicenseNumber VARCHAR(20),
    ContactNo VARCHAR(20),
    YearsOfExperience INT,
    Certifications VARCHAR(255)
);
INSERT INTO Pharmacists (PharmacistID, PharmacistName, LicenseNumber, ContactNo,
YearsOfExperience, Certifications)
```

```
VALUES
    ('PHARM-001', 'John Smith', 'PH123456', '555-123-4567', 10, 'BPS Certification, CPR'),
    ('PHARM-002', 'Emily Johnson', 'PH987654', '555-987-6543', 8, 'Immunization, Diabetes
Care'),
    ('PHARM-003', 'Michael Brown', 'PH246810', '555-246-8101', 12, 'MTM, Pharmacotherapy'),
    ('PHARM-004', 'Sarah Lee', 'PH135792', '555-135-7924', 5, 'BLS, Anticoagulation'),
    ('PHARM-005', 'David Wilson', 'PH864209', '555-864-2095', 15, 'Geriatric Pharmacy, APh'),
    ('PHARM-006', 'Rachel Garcia', 'PH357951', '555-357-9512', 7, 'Ambulatory Care, Asthma
Educator'),
    ('PHARM-007', 'Mark Thompson', 'PH682430', '555-682-4308', 9, 'Oncology, Palliative
Care'),
    ('PHARM-008', 'Jennifer Clark', 'PH209763', '555-209-7630', 11, 'Critical Care, Pain
Management'),
    ('PHARM-009', 'Daniel Roberts', 'PH578241', '555-578-2419', 6, 'Pediatrics, Compounding'),
    ('PHARM-010', 'Laura Martinez', 'PH934561', '555-934-5617', 14, 'Infectious Diseases,
HIV/AIDS');
```

A.2.9 Transactions Table

```
CREATE TABLE Transactions (
    TransactionID VARCHAR(10) PRIMARY KEY,
    TransactionDate DATE,
    QuantityDispensed INT,
    TotalPrice DECIMAL(10,2),
    PrescriptionID VARCHAR(10),
    PharmacistID VARCHAR(10),
    FOREIGN KEY (PrescriptionID) REFERENCES Prescriptions(PrescriptionID),
    FOREIGN KEY (PharmacistID) REFERENCES Pharmacists(PharmacistID)
);
```

```
INSERT INTO Transactions (TransactionID, TransactionDate, QuantityDispensed, TotalPrice,
PrescriptionID, PharmacistID)
VALUES
    ('TRX-001', '2024-04-15', 3, 45.00, 'RX2001', 'PHARM-001'),
    ('TRX-002', '2024-04-16', 2, 30.00, 'RX2002', 'PHARM-002'),
    ('TRX-003', '2024-04-17', 1, 15.50, 'RX2003', 'PHARM-003'),
    ('TRX-004', '2024-04-18', 4, 60.00, 'RX2004', 'PHARM-001'),
    ('TRX-005', '2024-04-19', 2, 25.00, 'RX2005', 'PHARM-002'),
    ('TRX-006', '2024-04-20', 3, 45.75, 'RX2006', 'PHARM-003'),
    ('TRX-007', '2024-04-21', 1, 12.00, 'RX2007', 'PHARM-001'),
    ('TRX-008', '2024-04-22', 2, 24.50, 'RX2008', 'PHARM-002'),
    ('TRX-009', '2024-04-23', 3, 35.25, 'RX2009', 'PHARM-003'),
    ('TRX-010', '2024-04-24', 4, 48.00, 'RX2010', 'PHARM-001');
```

A.2.10 Employee Table

```
CREATE TABLE Employee (
    EmployeeID INT PRIMARY KEY,
    DepartmentID INT,
    EmployeeName VARCHAR(255),
    Specialization VARCHAR(255),
    Salary DECIMAL(10, 2),
    EmployeeType VARCHAR(50),
    FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)
);

INSERT INTO Employee (EmployeeID, DepartmentID, EmployeeName, Specialization, Salary,
EmployeeType)
VALUES
```

```
    (1, 101, 'John Doe', 'Cardiology', 120000.00, 'Physician'),
    (2, 102, 'Jane Smith', 'Pediatrics', 110000.00, 'Physician'),
    (3, 103, 'Alice Johnson', 'Orthopedics', 115000.00, 'Physician'),
    (4, 104, 'Michael Williams', 'Neurology', 118000.00, 'Physician'),
    (5, 105, 'Sarah Brown', 'Oncology', 112000.00, 'Physician'),
    (6, 106, 'David Lee', 'Psychiatry', 105000.00, 'Physician'),
    (7, 107, 'Emily Taylor', 'Gynecology', 108000.00, 'Physician'),
    (8, 108, 'Robert Clark', 'Urology', 113000.00, 'Physician'),
    (9, 109, 'Laura Anderson', 'Dermatology', 106000.00, 'Physician'),
    (10, 110, 'Mark Wilson', 'Ophthalmology', 125000.00, 'Physician'),
    (11, 101, 'Maria Garcia', 'Cardiology', 90000.00, 'Nurse'),
    (12, 102, 'James Rodriguez', 'Pediatrics', 85000.00, 'Nurse'),
    (13, 103, 'Emma Martinez', 'Orthopedics', 82000.00, 'Nurse'),
    (14, 104, 'William Lopez', 'Neurology', 80000.00, 'Nurse'),
    (15, 105, 'Olivia Hernandez', 'Oncology', 83000.00, 'Nurse'),
    (16, 106, 'Sophia Nguyen', 'Psychiatry', 87000.00, 'Nurse'),
    (17, 107, 'Ethan Baker', 'Gynecology', 84000.00, 'Nurse'),
    (18, 108, 'Ava Green', 'Urology', 86000.00, 'Nurse'),
    (19, 109, 'Noah Carter', 'Dermatology', 82500.00, 'Nurse'),
    (20, 110, 'Lily Adams', 'Ophthalmology', 81000.00, 'Nurse'),
    (21, NULL, 'Liam Stewart', NULL, 82000.00, 'Nurse');
```

A.2.11 Nurse Table

```
CREATE TABLE Nurse (
    NurseID INT PRIMARY KEY,
    DepartmentID INT,
    EmployeeName VARCHAR(255),
    Specialization VARCHAR(255),
    Salary DECIMAL(10, 2),
    LicenseNumber VARCHAR(50),
    FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)
);

INSERT INTO Nurse (NurseID, DepartmentID, EmployeeName, Specialization, Salary,
LicenseNumber)
VALUES
    (11, 101, 'Maria Garcia', 'Cardiology', 90000.00, 'LN123456'),
    (12, 102, 'James Rodriguez', 'Pediatrics', 85000.00, 'LN234567'),
    (13, 103, 'Emma Martinez', 'Orthopedics', 82000.00, 'LN345678'),
    (14, 104, 'William Lopez', 'Neurology', 80000.00, 'LN456789'),
    (15, 105, 'Olivia Hernandez', 'Oncology', 83000.00, 'LN567890'),
    (16, 106, 'Sophia Nguyen', 'Psychiatry', 87000.00, 'LN678901'),
    (17, 107, 'Ethan Baker', 'Gynecology', 84000.00, 'LN789012'),
    (18, 108, 'Ava Green', 'Urology', 86000.00, 'LN890123'),
    (19, 109, 'Noah Carter', 'Dermatology', 82500.00, 'LN901234'),
    (20, 110, 'Lily Adams', 'Ophthalmology', 81000.00, 'LN012345'),
    (21, NULL, 'Liam Stewart', NULL, 82000.00, 'LN123456');
```

A.2.12 Billing Insurance Table

```
CREATE TABLE BillingInsurance (
    BillingInsuranceID VARCHAR(10) PRIMARY KEY,
    ServiceProcedureCode VARCHAR(10),
    ServiceDate DATE,
    ChargeAmount DECIMAL(10, 2),
    InsuranceProvider VARCHAR(255),
    InsuranceClaimStatus VARCHAR(50),
    PatientID INT,
    FOREIGN KEY (PatientID) REFERENCES Patients(PatientID)
);

INSERT INTO BillingInsurance (BillingInsuranceID, ServiceProcedureCode, ServiceDate,
ChargeAmount, InsuranceProvider, InsuranceClaimStatus, PatientID)
VALUES
    ('BI1001', 'PROC001', '2024-04-15', 1500.00, 'ABC Insurance', 'Pending', 10001),
    ('BI1002', 'PROC002', '2024-04-18', 1200.50, 'XYZ Insurance', 'Approved', 10002),
    ('BI1003', 'PROC003', '2024-04-20', 800.75, 'DEF Insurance', 'Denied', 10003),
    ('BI1004', 'PROC004', '2024-04-22', 2000.25, 'GHI Insurance', 'Approved', 10005),
    ('BI1005', 'PROC005', '2024-04-25', 1750.00, 'JKL Insurance', 'Pending', 10005),
    ('BI1006', 'PROC006', '2024-04-27', 300.00, 'MNO Insurance', 'Denied', 10006),
    ('BI1007', 'PROC007', '2024-04-29', 950.80, 'PQR Insurance', 'Approved', 10008),
    ('BI1008', 'PROC008', '2024-05-01', 1800.90, 'STU Insurance', 'Pending', 10008),
    ('BI1009', 'PROC009', '2024-05-03', 600.25, 'VWX Insurance', 'Denied', 10009),
    ('BI1010', 'PROC010', '2024-05-05', 1350.75, 'YZA Insurance', 'Approved', 10010);
```

A.2.13 Prescription Medical Supplies Table

```
CREATE TABLE PrescriptionMedicalSupplies (
    PrescriptionID VARCHAR(10),
    MedicalSupplyID VARCHAR(10),
    PRIMARY KEY (PrescriptionID, MedicalSupplyID),
    FOREIGN KEY (PrescriptionID) REFERENCES Prescriptions(PrescriptionID),
    FOREIGN KEY (MedicalSupplyID) REFERENCES MedicalSupplies(MedicalSupplyID)
);

INSERT INTO PrescriptionMedicalSupplies (PrescriptionID, MedicalSupplyID)
VALUES
    ('RX2001', 'MS3001'),
    ('RX2001', 'MS3002'),
    ('RX2002', 'MS3002'),
    ('RX2003', 'MS3003'),
    ('RX2004', 'MS3004'),
    ('RX2005', 'MS3005'),
    ('RX2006', 'MS3006'),
    ('RX2007', 'MS3007'),
    ('RX2008', 'MS3008'),
    ('RX2009', 'MS3009'),
    ('RX2010', 'MS3010');
```

NOTE: As per the business rules, A Prescription must have one or more MedicalSupplies (1:M). A MedicalSupply can be associated with zero or more Prescriptions (1:M). To represent a many-to-many relationship between prescriptions and medical supplies, you typically use a junction table. This junction table connects the two entities by storing pairs of primary keys from each table. **PrescriptionMedicalSupplies** junction table:

a. PrescriptionID (Foreign Key referencing Prescriptions)
b. MedicationID (Foreign Key referencing MedicalSupplies)

This junction table will allow you to link each prescription with multiple medical supplies and vice versa. **PrescriptionMedicalSupplies** table satisfies the logic of an associative entity.