**GROUP 8**

**Health Care Management System**

**Yash Badani**

**Shrey Patel**

**Raveena Patil**

**Rutuja Patil**

**Anushka Paradkar**

**Database:**  
Create Database HealthCareManagementSystem

**Tables:**

CREATE TABLE Patient (

PatientID VARCHAR(50) PRIMARY KEY,

SSN VARCHAR(20),

FirstName VARCHAR(50),

LastName VARCHAR(50),

DateOfBirth DATE,

Gender VARCHAR(10) CHECK (Gender IN ('Other', 'Female', 'Male')),

Email VARCHAR(100),

Address VARCHAR(255),

PhoneNumber VARCHAR(20),

EmergencyContact VARCHAR(50),

BloodType VARCHAR(10) CHECK (BloodType IN ('O-', 'O+', 'AB-', 'AB+', 'B-', 'B+', 'A-', 'A+'))

);

CREATE TABLE Doctor (

DoctorID VARCHAR(50) PRIMARY KEY,  
 DoctorName Nvarchar(100),

LicenseNumber VARCHAR(20),

YearsOfExperience INT,

Education VARCHAR(100),

AvailabilitySchedule VARCHAR(100),

Specialization VARCHAR(50)

);

CREATE TABLE Visit (

VisitID VARCHAR(50) PRIMARY KEY,

VisitDate DATETIME,

Purpose VARCHAR(255),

Status VARCHAR(20) CHECK (Status IN ('Cancelled', 'Completed', 'Scheduled')),

DoctorID VARCHAR(50) FOREIGN KEY REFERENCES Doctor(DoctorID),

PatientID VARCHAR(50) FOREIGN KEY REFERENCES Patient(PatientID)

);

CREATE TABLE Treatment (

TreatmentID VARCHAR(50) PRIMARY KEY,

TreatmentName VARCHAR(255),

TreatmentFees DECIMAL(10, 2),

VisitID VARCHAR(50) FOREIGN KEY REFERENCES Visit(VisitID),

Admit\_Flag CHAR(1) CHECK (Admit\_Flag IN ('N', 'Y',’Hold’)),  
 Notes Nvarchar(100)

);

CREATE TABLE Prescription (

PrescriptionID VARCHAR(50) NOT NULL,

MedicineName VARCHAR(50) NOT NULL,

TreatmentID VARCHAR(50) NULL,

PrescriptionDate DATETIME NULL,

Dosage VARCHAR(255) NULL,

PRIMARY KEY CLUSTERED (PrescriptionID ASC, MedicineName ASC),

FOREIGN KEY (TreatmentID) REFERENCES Treatment(TreatmentID)

);

CREATE TABLE Room (

RoomID VARCHAR(50) NOT NULL PRIMARY KEY,

RoomType VARCHAR(20) NULL CHECK (RoomType IN ('Operating Room', 'Suite', 'General Ward', 'ICU')),

RoomStatus VARCHAR(20) NULL CHECK (RoomStatus IN ('Maintenance', 'Available', 'Occupied')),

RoomCharge DECIMAL(10, 2) NULL

);

CREATE TABLE Admission (

AdmissionID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

TreatmentID VARCHAR(50) NULL,

RoomID VARCHAR(50) NULL,

AdmissionDate DATETIME NULL,

DischargeDate DATETIME NULL,

AdmissionReason VARCHAR(255) NULL,

Status VARCHAR(20) NULL,

FOREIGN KEY (RoomID) REFERENCES Room(RoomID),

FOREIGN KEY (TreatmentID) REFERENCES Treatment(TreatmentID)

);

CREATE TABLE Bill (

BillID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

BillingAmount DECIMAL(10, 2) NULL,

DateIssued DATETIME NULL,

Status VARCHAR(20) NULL,

VisitID VARCHAR(50) NULL,

FOREIGN KEY (VisitID) REFERENCES Visit(VisitID)

);

CREATE TABLE InsuranceProvider (

ProviderID VARCHAR(50) NOT NULL PRIMARY KEY,

Name VARCHAR(255) NOT NULL,

Address VARCHAR(255) NULL,

PhoneNumber VARCHAR(15) NULL,

ContactEmail VARCHAR(255) NULL,

Website VARCHAR(255) NULL

);

CREATE TABLE CoveragePlan (

PlanID VARCHAR(50) NOT NULL PRIMARY KEY,

PlanName VARCHAR(255) NOT NULL,

CoverageDetails VARCHAR(255) NULL,

Premium DECIMAL(10, 2) NULL,

Deductible DECIMAL(10, 2) NULL,

CoverageStartDate DATE NULL,

CoverageEndDate DATE NULL,

ProviderID VARCHAR(5) NULL,

FOREIGN KEY (ProviderID) REFERENCES InsuranceProvider (ProviderID)

);

CREATE TABLE PatientCoveragePlan

(

PatientID VARCHAR(50),

PlanID VARCHAR(5),

FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),

FOREIGN KEY (PlanID) REFERENCES CoveragePlan(PlanID),

PRIMARY KEY (PatientID, PlanID)

);

CREATE TABLE TreatmentHistory (

TreatmentID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

DateAdministered DATETIME NULL,

TreatmentDescription NVARCHAR(50) NULL,

VisitID VARCHAR(50) NULL,

Cost DECIMAL(18, 2) NULL,

DoctorID VARCHAR(50) NULL,

PatientID VARCHAR(50) NULL, FOREIGN KEY (DoctorID) REFERENCES Doctor (DoctorID),

FOREIGN KEY (PatientID) REFERENCES Patient (PatientID),

FOREIGN KEY (VisitID) REFERENCES Visit (VisitID)

);

**USER DEFINED FUNCTIONS**

CREATE FUNCTION dbo.CalculateAmountToBePaid (

@BillID INT,

@PlanID VARCHAR(5)

)

RETURNS DECIMAL(10, 2)

AS

BEGIN

DECLARE @Deductible DECIMAL(10, 2);

DECLARE @AmountToBePaid DECIMAL(10, 2);

-- Retrieve deductible based on PlanID

SELECT @Deductible = Deductible

FROM CoveragePlan

WHERE PlanID = @PlanID;

-- Retrieve the total amount from the Bill table based on BillID

SELECT @AmountToBePaid = BillingAmount

FROM Bill

WHERE BillID = @BillID;

-- Calculate the amount to be paid by subtracting the deductible and coverage amount from the total amount

SET @AmountToBePaid = @AmountToBePaid - ISNULL(dbo.AmountCoveredByInsurance(@BillID, @PlanID),0);

-- Ensure the result is non-negative (no negative amounts to be paid)

SET @AmountToBePaid = CASE WHEN @AmountToBePaid < 0 THEN 0 ELSE @AmountToBePaid END;

RETURN @AmountToBePaid;

END;

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Create FUNCTION dbo.AmountCoveredByInsurance (

@BillID INT,

@PlanID VARCHAR(5)

)

RETURNS DECIMAL(10, 2)

AS

BEGIN

DECLARE @CoverageAmount DECIMAL(10, 2);

DECLARE @BillAmount DECIMAL(10, 2);

-- Retrieve coverage amount based on BillID and PlanID

SELECT @CoverageAmount = cp.deductible,

@BillAmount = b.BillingAmount

FROM Bill b

inner join visit v on b.VisitID=v.VisitID

Join PatientCoveragePlan pcp on pcp.patientid=v.patientid

JOIN CoveragePlan cp ON pcp.planid=cp.planid

WHERE b.BillID = @BillID AND cp.PlanID = @PlanID;

-- Return the minimum of BillAmount and CoverageAmount to handle the case when BillAmount is less

RETURN CASE WHEN @BillAmount < @CoverageAmount THEN @BillAmount ELSE @CoverageAmount END;

END;

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***CREATE FUNCTION dbo.GetTotalTreatmentCostForPatient(@PatientID VARCHAR(50))

RETURNS DECIMAL(10, 2)

AS

BEGIN

DECLARE @TotalAmount DECIMAL(10, 2);

WITH TreatmentData AS (

SELECT

t.TreatmentID,

t.Admit\_Flag,

a.AdmissionDate,

a.DischargeDate,

r.RoomCharge,

t.TreatmentFees

FROM

Treatment t

LEFT JOIN Admission a ON t.TreatmentID = a.TreatmentID

LEFT JOIN Room r ON a.RoomID = r.RoomID

LEFT JOIN Visit v ON t.VisitID = v.VisitID

WHERE

v.PatientID = @PatientID

)

SELECT @TotalAmount = COALESCE(

SUM(CASE

WHEN td.Admit\_Flag = 'Y' THEN

DATEDIFF(DAY, td.AdmissionDate, COALESCE(td.DischargeDate, GETDATE())) \* td.RoomCharge +

COALESCE(td.TreatmentFees, 0)

ELSE

COALESCE(td.TreatmentFees, 0)

END),

0

)

FROM TreatmentData td;

RETURN @TotalAmount;

END;

– UDF based column

CREATE TABLE Payment (

PaymentID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

BillID INT NULL,

PlanID VARCHAR(50) NULL,

TotalAmount DECIMAL(10, 2) NULL,

PaymentDate DATE NULL,

AmountCoveredByInsurance AS ([dbo].[AmountCoveredByInsurance](BillID, PlanID)),

AmountToBePaid AS ([dbo].[CalculateAmountToBePaid](BillID, PlanID)),

FOREIGN KEY (BillID) REFERENCES Bill(BillID),

FOREIGN KEY (PlanID) REFERENCES CoveragePlan(PlanID)

);

**DDL Commands**

**Patient:**

INSERT INTO Patient (PatientID, SSN, FirstName, LastName, DateOfBirth, Gender, Email, Address, PhoneNumber, EmergencyContact, BloodType)

VALUES

('PAT001', '123-45-6789', 'John', 'Doe', '1990-01-01', 'Male', 'john.doe@email.com', '123 Main St', '555-1234', 'Jane Doe', 'A+'),

('PAT002', '987-65-4321', 'Jane', 'Smith', '1985-05-15', 'Female', 'jane.smith@email.com', '456 Oak St', '555-5678', 'John Smith', 'B-'),

('PAT003', '543-21-9876', 'Alice', 'Johnson', '1995-08-20', 'Female', 'alice.johnson@email.com', '789 Pine St', '555-8765', 'Bob Johnson', 'O+'),

('PAT004', '654-32-1098', 'Michael', 'Williams', '1980-03-10', 'Male', 'michael.williams@email.com', '456 Maple St', '555-4321', 'Laura Williams', 'AB-'),

('PAT005', '876-54-3210', 'Emily', 'Brown', '1992-11-05', 'Female', 'emily.brown@email.com', '789 Cedar St', '555-9876', 'Chris Brown', 'A-'),

('PAT006', '234-56-7890', 'David', 'Jones', '1988-07-15', 'Male', 'david.jones@email.com', '123 Birch St', '555-2345', 'Sarah Jones', 'B+'),

('PAT007', '567-89-0123', 'Sophia', 'Miller', '1975-04-25', 'Female', 'sophia.miller@email.com', '789 Elm St', '555-6789', 'Daniel Miller', 'O-'),

('PAT008', '789-01-2345', 'Matthew', 'Davis', '1997-09-30', 'Male', 'matthew.davis@email.com', '456 Oak St', '555-8901', 'Olivia Davis', 'A+'),

('PAT009', '210-98-7654', 'Olivia', 'Taylor', '1982-12-12', 'Female', 'olivia.taylor@email.com', '123 Pine St', '555-2109', 'William Taylor', 'AB+'),

('PAT010', '432-10-9876', 'Daniel', 'Wilson', '1998-02-18', 'Male', 'daniel.wilson@email.com', '789 Maple St', '555-5432', 'Emma Wilson', 'O-'),

('PAT011', '876-54-3210', 'Emma', 'Anderson', '1986-06-22', 'Female', 'emma.anderson@email.com', '456 Cedar St', '555-8765', 'James Anderson', 'B-'),

('PAT012', '345-67-8901', 'Christopher', 'Harris', '1993-04-09', 'Male', 'christopher.harris@email.com', '789 Birch St', '555-3456', 'Jessica Harris', 'A+'),

('PAT013', '987-65-4321', 'Mia', 'Moore', '1984-10-14', 'Female', 'mia.moore@email.com', '123 Elm St', '555-9876', 'Christopher Moore', 'O-'),

('PAT014', '123-45-6789', 'William', 'Clark', '1991-08-07', 'Male', 'william.clark@email.com', '456 Pine St', '555-1234', 'Sophia Clark', 'A-'),

('PAT015', '567-89-0123', 'Ava', 'White', '1980-01-30', 'Female', 'ava.white@email.com', '789 Maple St', '555-5678', 'Ethan White', 'AB+'),

('PAT016', '876-54-3210', 'Ethan', 'Thomas', '1995-03-28', 'Male', 'ethan.thomas@email.com', '123 Cedar St', '555-8765', 'Lily Thomas', 'B-'),

('PAT017', '210-98-7654', 'Lily', 'Brown', '1987-07-05', 'Female', 'lily.brown@email.com', '456 Elm St', '555-2109', 'Noah Brown', 'A+'),

('PAT018', '432-10-9876', 'Noah', 'Jones', '1994-09-17', 'Male', 'noah.jones@email.com', '789 Oak St', '555-5432', 'Emily Jones', 'O-'),

('PAT019', '654-32-1098', 'Grace', 'Miller', '1983-05-03', 'Female', 'grace.miller@email.com', '123 Birch St', '555-6543', 'Samuel Miller', 'AB-'),

('PAT020', '876-54-3210', 'Samuel', 'Roberts', '1996-11-12', 'Male', 'samuel.roberts@email.com', '456 Cedar St', '555-8765', 'Olivia Roberts', 'B+');

**Doctor:**INSERT INTO Doctor (DoctorID, LicenseNumber, YearsOfExperience, Education, AvailabilitySchedule, Specialization, DoctorName)

VALUES

('DOC001', 12345, 10, 'MD', 'Monday-Friday', 'Cardiology', 'John Smith'),

('DOC002', 67890, 8, 'DO', 'Tuesday-Saturday', 'Orthopedics', 'Jane Doe'),

('DOC003', 54321, 15, 'MBBS', 'Monday-Wednesday', 'Pediatrics', 'Michael Johnson'),

('DOC004', 98765, 12, 'MD', 'Thursday-Sunday', 'Ophthalmology', 'Emily Davis'),

('DOC005', 23456, 10, 'MBBS', 'Monday-Friday', 'Dermatology', 'Robert Miller'),

('DOC006', 87654, 18, 'DO', 'Wednesday-Saturday', 'Neurology', 'Olivia Wilson'),

('DOC007', 34567, 20, 'MD', 'Tuesday-Thursday', 'Cardiothoracic Surgery', 'William Brown'),

('DOC008', 76543, 8, 'MBBS', 'Monday-Friday', 'Gastroenterology', 'Sophia Taylor'),

('DOC009', 45678, 10, 'DO', 'Monday-Sunday', 'Orthopedics', 'Daniel Anderson'),

('DOC010', 65432, 14, 'MD', 'Thursday-Saturday', 'Urology', 'Ava Martinez'),

('DOC011', 87654, 12, 'MBBS', 'Tuesday-Sunday', 'Endocrinology', 'Ethan Jackson'),

('DOC012', 23456, 15, 'DO', 'Monday-Wednesday', 'Ophthalmology', 'Isabella White'),

('DOC013', 56789, 10, 'MD', 'Friday-Sunday', 'Pulmonology', 'Christopher Harris'),

('DOC014', 34567, 8, 'MBBS', 'Monday-Thursday', 'Rheumatology', 'Amelia Nelson'),

('DOC015', 78901, 10, 'MD', 'Monday-Wednesday', 'Cardiology', 'Matthew Garcia'),

('DOC016', 43210, 12, 'DO', 'Thursday-Saturday', 'Nephrology', 'Emma Smith'),

('DOC017', 89012, 15, 'MD', 'Monday-Sunday', 'Dermatology', 'Andrew Johnson'),

('DOC018', 54321, 8, 'MBBS', 'Wednesday-Friday', 'Neurology', 'Sophie Clark'),

('DOC019', 21098, 10, 'DO', 'Tuesday-Sunday', 'Oncology', 'Nathan Baker'),

('DOC020', 90123, 14, 'MD', 'Monday-Thursday', 'Gynecology', 'Grace Turner')

**Visit:**INSERT INTO Visit (VisitID, VisitDate, Purpose, Status, DoctorID, PatientID)

VALUES

('VIS001', '2023-01-05 10:00:00', 'Heart Checkup', 'Completed', 'DOC001', 'PAT001'),

('VIS002', '2023-02-10 14:30:00', 'Severe Knee Joint Pain', 'Scheduled', 'DOC002', 'PAT002'),

('VIS003', '2023-03-15 11:15:00', 'Fever', 'Completed', 'DOC003', 'PAT003'),

('VIS004', '2023-04-20 09:45:00', 'Blurry Eye check up', 'Scheduled', 'DOC004', 'PAT004'),

('VIS005', '2023-05-25 13:30:00', 'Skin rashes', 'Completed', 'DOC005', 'PAT005'),

('VIS006', '2023-06-30 15:00:00', 'Seizure', 'Scheduled', 'DOC006', 'PAT006'),

('VIS007', '2023-07-05 08:45:00', 'General Heart Checkup', 'Completed', 'DOC007', 'PAT007'),

('VIS008', '2023-08-10 12:15:00', 'Heavy Stomach Pain', 'Scheduled', 'DOC008', 'PAT008'),

('VIS009', '2023-09-15 14:30:00', 'Orthopedic Consultation', 'Completed', 'DOC009', 'PAT009'),

('VIS010', '2023-10-20 10:30:00', 'Urine Tract Checkup', 'Scheduled', 'DOC010', 'PAT010'),

('VIS011', '2023-11-25 09:00:00', 'Endocrine Assessment', 'Completed', 'DOC011', 'PAT011'),

('VIS012', '2023-12-30 11:45:00', 'Eye care Follow-up', 'Scheduled', 'DOC012', 'PAT012'),

('VIS013', '2024-01-04 13:15:00', 'Pulmonology Consultation', 'Completed', 'DOC013', 'PAT013'),

('VIS014', '2024-02-09 16:00:00', 'Rheumatology Checkup', 'Scheduled', 'DOC014', 'PAT014'),

('VIS015', '2024-03-15 10:45:00', 'Heart Check up', 'Completed', 'DOC015', 'PAT015'),

('VIS016', '2024-04-20 14:30:00', 'Kidney function assessment', 'Scheduled', 'DOC016', 'PAT016'),

('VIS017', '2024-05-25 09:30:00', 'Rosacea condition evaluation', 'Completed', 'DOC017', 'PAT017'),

('VIS018', '2024-06-30 11:15:00', 'Memory loss checkup', 'Scheduled', 'DOC018', 'PAT018'),

('VIS019', '2024-07-05 13:45:00', 'Cancer Chemotherapy', 'Completed', 'DOC019', 'PAT019'),

('VIS020', '2024-08-10 15:30:00', 'Pregancy Checkup', 'Scheduled', 'DOC020', 'PAT020');

**Treatment**

INSERT INTO Treatment (TreatmentID, TreatmentName, TreatmentFees, VisitID, Admit\_Flag)

VALUES

('TRT001', 'Cardiac Angiography', 500.00, 'VIS001', 'N'),

('TRT002', 'Knee Joint Operation', 400.00, 'VIS002', 'Y'),

('TRT003', 'Antibiotics for Fever', 50.00, 'VIS003', 'N'),

('TRT004', 'Eye Prescription Glasses', 300.00, 'VIS004', 'N'),

('TRT005', 'Topical Cream for Skin Rashes', 20.00, 'VIS005', 'N'),

('TRT006', 'Anti-seizure Medication', 150.00, 'VIS006', 'N'),

('TRT007', 'Cardiac Stress Test', 200.00, 'VIS007', 'N'),

('TRT008', 'Gastrointestinal Endoscopy', 500.00, 'VIS008', 'Y'),

('TRT009', 'Orthopedic Physical Therapy', 400.00, 'VIS009', 'Y'),

('TRT010', 'Urological Tests', 80.00, 'VIS010', 'N'),

('TRT011', 'Hormone Replacement Therapy', 600.00, 'VIS011', 'N'),

('TRT012', 'Eye Care Follow-up Consultation', 50.00, 'VIS012', 'N'),

('TRT013', 'Pulmonary Function Tests', 300.00, 'VIS013', 'N'),

('TRT014', 'Rheumatoid Arthritis Medication', 200.00, 'VIS014', 'N'),

('TRT015', 'Cardiac Medication', 100.00, 'VIS015', 'N'),

('TRT016', 'Nephrology Consultation', 50.00, 'VIS016', 'N'),

('TRT017', 'Dermatology Cream for Rosacea', 25.00, 'VIS017', 'N'),

('TRT018', 'Neurological Memory Tests', 150.00, 'VIS018', 'N'),

('TRT019', 'Chemotherapy Session', 500.00, 'VIS019', 'Y'),

('TRT020', 'Obstetric Ultrasound', 150.00, 'VIS020', 'N');

**Prescription:**INSERT INTO Prescription (PrescriptionID, MedicineName, TreatmentID, PrescriptionDate, Dosage)

VALUES

('PRE001', 'Aspirin', 'TRT001', Getdate(), 'once every 3 months'),

('PRE001', 'Lisinopril', 'TRT001', Getdate(), '2 times daily for 10 days'),

('PRE002', 'Advil', 'TRT002', Getdate(), 'Thrice every day for 3 months'),

('PRE002', 'Motrin', 'TRT002', Getdate(), 'twice per month'),

('PRE003', 'Crocin', 'TRT003', Getdate(), '4 times every week'),

('PRE004', 'Eye Drops', 'TRT004', Getdate(), 'once daily for 2 weeks'),

('PRE005', 'Azelic Acid Cream', 'TRT005', Getdate(), 'thrice for 5 weeks'),

('PRE006', 'Dilantin', 'TRT006', Getdate(), '2 times a daily for 3 months'),

('PRE007', 'Atenolol', 'TRT007', Getdate(), '3 times a day for 3 weeks'),

('PRE007', 'Carvetilol', 'TRT007', Getdate(), 'once daily for 3months'),

('PRE008', 'Lomeramide', 'TRT008', Getdate(), 'Twice a day for 4 days'),

('PRE009', 'Moov', 'TRT009', Getdate(), 'Daily for a week'),

('PRE010', 'Nitrofuraentoin', 'TRT010', Getdate(), '3 times a day'),

('PRE011', 'Hormone Replacement Drug', 'TRT011', Getdate(), 'twice a day for 5 days '),

('PRE012', 'Eye Drops', 'TRT012', Getdate(), 'thrice a day for 3 days'),

('PRE013', 'Bronchodilator', 'TRT013', Getdate(), 'twice daily for 2 months'),

('PRE013', 'Nilgiri', 'TRT013', Getdate(), 'thrice daily for 2 months'),

('PRE014', 'Methotrexate', 'TRT014', Getdate(), 'thrice daily for 1 month'),

('PRE015', 'Cardiac Medication', 'TRT015', Getdate(), '3 times a day for 3 months'),

('PRE016', 'Diuretic', 'TRT016', Getdate(), '2 times daily for 1 week'),

('PRE017', 'Topical Cream', 'TRT017', Getdate(), 'twice daily for 3 months'),

('PRE018', 'Ritalin', 'TRT018', Getdate(), 'once daily for 1 month'),

('PRE019', 'Chemotherapy Drugs', 'TRT019', Getdate(), 'As directed by the physician during admission'),

('PRE020', 'Omega3 Vitamins', 'TRT020', Getdate(), 'Once daily for 1 month'),

('PRE020', 'Estrace', 'TRT020', Getdate(), 'Once daily for 1 month'),

('PRE020', 'Azithromycin', 'TRT020', Getdate(), 'Once daily for 1 month');

**Room:**  
INSERT INTO Room (RoomID, RoomType, RoomStatus, RoomCharge)

VALUES

('Room001', 'ICU', 'Available', 500.00),

('Room002', 'General Ward', 'Occupied', 200.00),

('Room003', 'Suite', 'Available', 800.00),

('Room004', 'Operating Room', 'Maintenance', 1000.00),

('Room005', 'ICU', 'Occupied', 550.00),

('Room006', 'General Ward', 'Available', 250.00),

('Room007', 'Suite', 'Occupied', 900.00),

('Room008', 'Operating Room', 'Maintenance', 1100.00),

('Room009', 'ICU', 'Available', 520.00),

('Room010', 'General Ward', 'Occupied', 210.00),

('Room011', 'Suite', 'Available', 850.00),

('Room012', 'Operating Room', 'Maintenance', 1050.00),

('Room013', 'ICU', 'Occupied', 530.00),

('Room014', 'General Ward', 'Available', 220.00),

('Room015', 'Suite', 'Occupied', 920.00),

('Room016', 'Operating Room', 'Maintenance', 1150.00),

('Room017', 'ICU', 'Available', 510.00),

('Room018', 'General Ward', 'Occupied', 230.00),

('Room019', 'Suite', 'Available', 880.00),

('Room020', 'Operating Room', 'Maintenance', 1020.00);

**InsuranceProvider :**

INSERT INTO InsuranceProvider (ProviderID, Name, Address, PhoneNumber, ContactEmail, Website)

VALUES

('PRO01', 'HealthGuard Insurance', '123 Insurance Ave', '800-123-4567', 'info@healthguard.com', 'www.healthguard.com'),

('PRO02', 'SecureCare Insurance', '456 Coverage St', '800-987-6543', 'info@securecare.com', 'www.securecare.com'),

('PRO03', 'QuickHealth Insurance', '789 Coverage Blvd', '800-567-8901', 'info@quickhealth.com', 'www.quickhealth.com'),

('PRO04', 'LifeInsure Health', '101 Policy Lane', '800-234-5678', 'info@lifeinsure.com', 'www.lifeinsure.com'),

('PRO05', 'SafeHealth Assurance', '202 Security St', '800-345-6789', 'info@safehealth.com', 'www.safehealth.com'),

('PRO06', 'NewProvider Insurance', '789 NewCoverage Blvd', '800-111-2222', 'info@newprovider.com', 'www.newprovider.com'),

('PRO07', 'AnotherProvider Insurance', '456 AnotherCoverage St', '800-333-4444', 'info@anotherprovider.com', 'www.anotherprovider.com'),

('PRO08', 'LastProvider Insurance', '123 LastCoverage Ave', '800-555-6666', 'info@lastprovider.com', 'www.lastprovider.com');

**CoveragePlan:**

INSERT INTO CoveragePlan (PlanID, PlanName, CoverageDetails, Premium, Deductible, CoverageStartDate, CoverageEndDate, ProviderID)

VALUES

('PLA01', 'Basic Plan', 'Basic coverage for common medical services', '60.00', '550.00', '2025-02-01', '2023-12-31', 'PRO01'),

('PLA02', 'Premium Plan', 'Comprehensive coverage including specialist visits', '120.00', '200.00', '2025-03-01', '2023-12-31', 'PRO02'),

('PLA03', 'QuickHealth Standard', 'Standard coverage for medical needs', '80.00', '400.00', '2025-04-01', '2023-12-31', 'PRO03'),

('PLA04', 'LifeInsure Comprehensive', 'Extensive coverage for health and wellness', '140.00', '180.00', '2025-05-01', '2023-12-31', 'PRO04'),

('PLA05', 'SafeHealth Plus', 'Enhanced coverage for preventive care', '100.00', '250.00', '2025-06-01', '2023-12-31', 'PRO05'),

('PLA06', 'Guardian Gold Plan', 'Gold-level coverage for comprehensive health needs', '120.00', '200.00', '2025-07-01', '2023-12-31', 'PRO01'),

('PLA07', 'EverWell Assurance', 'Assurance plan for health and well-being', '105.00', '300.00', '2025-08-01', '2023-12-31', 'PRO02'),

('PLA08', 'FamilyCare Family Plan', 'Family-focused coverage for healthcare needs', '150.00', '180.00', '2025-09-01', '2023-12-31', 'PRO03'),

('PLA09', 'PremierHealth Executive', 'Executive-level coverage for comprehensive healthcare', '170.00', '150.00', '2025-10-01', '2023-12-31', 'PRO04'),

('PLA10', 'EliteCare Platinum', 'Platinum-level coverage for elite health services', '200.00', '100.00', '2025-11-01', '2023-12-31', 'PRO05'),

('PLA11', 'OptimaHealth Elite', 'Elite coverage for advanced medical needs', '145.00', '170.00', '2025-05-15', '2023-12-31', 'PRO06'),

('PLA12', 'TotalCare Premium', 'Premium coverage including wellness programs', '110.00', '220.00', '2025-06-10', '2023-12-31', 'PRO07'),

('PLA13', 'MediSecure Plus', 'Extended coverage for specialized medical services', '75.00', '500.00', '2025-02-10', '2023-12-31', 'PRO08');

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**STORED PROCEDURES**

--To Insert records into Admission Table  
Create PROCEDURE InsertAdmissionFromTreatment

AS

BEGIN

SET NOCOUNT ON;

-- Declare variables

DECLARE @RoomID VARCHAR(50);

DECLARE @AdmissionDate DATETIME;

DECLARE @TreatmentID VARCHAR(50);

DECLARE @AdmissionReason NVARCHAR(255);

DECLARE @TreatmentCount INT;

-- Get the count of eligible TreatmentID

SELECT @TreatmentCount = COUNT(\*)

FROM Treatment

WHERE Admit\_Flag = 'Y'

AND TreatmentID NOT IN (SELECT TreatmentID FROM Admission);

-- Loop through eligible treatments

WHILE @TreatmentCount > 0

BEGIN

-- Get the available room

SELECT TOP 1 @RoomID = RoomID

FROM Room

WHERE RoomStatus = 'Available'

ORDER BY RoomID;  
 SET @AdmissionDate = GETDATE();

-- Get the next eligible TreatmentID

SELECT TOP 1 @TreatmentID = TreatmentID

FROM Treatment

WHERE Admit\_Flag = 'Y'

AND TreatmentID NOT IN (SELECT TreatmentID FROM Admission)

ORDER BY TreatmentID;

-- Get the AdmissionReason from TreatmentName

SELECT @AdmissionReason = TreatmentName

FROM Treatment

WHERE TreatmentID = @TreatmentID;

-- Insert into Admission table

INSERT INTO Admission (TreatmentID, RoomID, AdmissionDate, DischargeDate, AdmissionReason, Status)

VALUES (

@TreatmentID,

@RoomID,

@AdmissionDate,

NULL,

@AdmissionReason,

'Admitted'

);

-- Update RoomStatus to 'Occupied' for the assigned room

UPDATE Room

SET RoomStatus = 'Occupied'

WHERE RoomID = @RoomID;

-- Decrement the count

SET @TreatmentCount = @TreatmentCount - 1;

END;

END;

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***– To calculate billing amount and insert data into billing table

Create PROCEDURE InsertBillingData

AS

BEGIN

SET NOCOUNT ON;

DECLARE @BillingAmount DECIMAL(10, 2);

DECLARE @DateIssued DATETIME = GETDATE();

DECLARE @Status VARCHAR(20) = 'Pending';

DECLARE @VisitID VARCHAR(50);

DECLARE @DateDifference INT;

-- Create a temporary table to store unique VisitIDs not present in the Bill table

CREATE TABLE #UniqueVisitIDs (VisitID VARCHAR(50));

-- Insert unique VisitIDs into the temporary table

INSERT INTO #UniqueVisitIDs (VisitID)

SELECT t.visitid

FROM treatment t

WHERE NOT EXISTS (SELECT 1 FROM Bill b WHERE b.VisitID = t.VisitID)

and t.treatmentid not in (select treatmentid from admission where dischargedate is null)

--select \* from treatment

-- Iterate through each unique VisitID

WHILE EXISTS (SELECT TOP 1 1 FROM #UniqueVisitIDs)

BEGIN

-- Get the next unique VisitID

SELECT TOP 1 @VisitID = VisitID FROM #UniqueVisitIDs;

if exists(Select 1 from treatment t inner join admission a on t.treatmentid=a.treatmentid and t.visitid=@VisitID and a.dischargedate is null)

begin

continue;

end

-- Calculate billing amount for admitted patients

SELECT @DateDifference = DATEDIFF(

DAY,

a.AdmissionDate,

a.DischargeDate)

FROM Admission a

WHERE a.TreatmentID IN (SELECT TreatmentID FROM Treatment WHERE VisitID = @VisitID);

SELECT @BillingAmount = (isnull(@DateDifference,0) \* isnull(r.RoomCharge,0) + (t.TreatmentFees))

FROM treatment t

left join Admission a ON a.TreatmentID = t.TreatmentID

left JOIN Room r ON a.RoomID = r.RoomID

WHERE t.VisitID = @VisitID

-- Insert into the Bill table without specifying BillID (assuming it's an identity column)

INSERT INTO Bill (BillingAmount, DateIssued, Status, VisitID)

VALUES (

@BillingAmount,

@DateIssued,

@Status,

@VisitID

);

-- Delete the processed VisitID from the temporary table

DELETE FROM #UniqueVisitIDs WHERE VisitID = @VisitID;

END;

-- Drop the temporary table

DROP TABLE #UniqueVisitIDs;

END;

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***– To insert data into treatmenthistory table

create procedure inserttreatmenthistory

as

begin

insert into treatmenthistory

select Visitdate,treatmentname,v.visitid,b.billingamount,v.doctorid,v.patientid from visit v

inner join treatment t on v.visitid=t.visitid

inner join bill b on v.visitid=b.visitid

where v.VisitDate < DATEADD(MONTH, -1, GETDATE()) and v.visitid not in (select visitid from TreatmentHistory)

end

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

— To insert record in payment

create PROCEDURE InsertPaymentData

AS

BEGIN

SET NOCOUNT ON;

-- Insert data into Payment table

INSERT INTO Payment (BillID, PlanID,TotalAmount , PaymentDate)

SELECT distinct

b.BillID,

cp.PlanID,

b.BillingAmount,

GETDATE() AS PaymentDate

FROM

Bill b

inner join visit v on b.VisitID=v.VisitID

LEFT JOIN CoveragePlan cp ON v.PatientID = cp.PatientID

where b.BillID not in (select billid from payment);

END;

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

– To schedule visit between doctor and patient

Create PROCEDURE ScheduleVisit

@PatientID VARCHAR(50),

@DoctorID VARCHAR(50),

@VisitDate DATETIME,

@Purpose VARCHAR(255)

AS

BEGIN

DECLARE @VisitID VARCHAR(50);

Declare @MaxID VARCHAR(50);

SELECT @MaxID = MAX(VisitID) FROM Visit;

-- Set the new VisitID by incrementing the numeric part

DECLARE @NewNumericID INT = ISNULL(CAST(SUBSTRING(@MaxID, 4, LEN(@MaxID) - 3) AS INT), 0) + 1;

DECLARE @NewVisitID VARCHAR(50) = 'VIS' + CAST(@NewNumericID AS VARCHAR);

INSERT INTO Visit (VisitID, PatientID, DoctorID, VisitDate, Purpose, Status)

VALUES ( @NewVisitID, @PatientID, @DoctorID, @VisitDate, @Purpose, 'Scheduled');

-- Retrieving the generated VisitID

SET @VisitID = (SELECT VisitID FROM Visit WHERE PatientID = @PatientID AND DoctorID = @DoctorID AND VisitDate = @VisitDate);

-- Returning the VisitID for further reference

SELECT @VisitID AS NewVisitID;

END;

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
–**for generating primary keyCREATE FUNCTION dbo.GenerateNewPrimaryKey

(

@PrimaryKey NVARCHAR(50),

@Prefix NVARCHAR(10)

)

RETURNS NVARCHAR(50)

AS

BEGIN

DECLARE @NewNumericID INT;

DECLARE @NewPrimaryKey NVARCHAR(50);

-- Set the new numeric ID by incrementing the numeric part

DECLARE @NumericPart NVARCHAR(50) = RIGHT(@PrimaryKey, LEN(@PrimaryKey) - LEN(@Prefix));

SET @NewNumericID = ISNULL(CAST(@NumericPart AS INT), 0) + 1;

-- Concatenate the prefix and the new numeric ID to get the new primary key

SET @NewPrimaryKey = @Prefix + CAST(@NewNumericID AS NVARCHAR);

RETURN @NewPrimaryKey;

END;

– To create treatment on the given visitid:

CREATE PROCEDURE InsertTreatment

@VisitID VARCHAR(50),

@Admit\_Flag CHAR(1),

@TreatmentFees DECIMAL(10,2)

AS

BEGIN

DECLARE @TreatmentID VARCHAR(50);

DECLARE @Note NVARCHAR(255) = '';

DECLARE @RoomID VARCHAR(50);

-- Generate a new TreatmentID

SELECT @TreatmentID = 'TRT0' + CAST(ISNULL(MAX(CAST(SUBSTRING(TreatmentID, 4, LEN(TreatmentID) - 3) AS INT)), 0) + 1 AS VARCHAR)

FROM Treatment;

DECLARE @TreatmentName NVARCHAR(50);

SELECT @TreatmentName = Purpose FROM Visit WHERE VisitID = @VisitID;

SET @TreatmentName = @TreatmentName + ' Treatment';

-- Check for room availability if Admit\_Flag is 'Y'

IF @Admit\_Flag = 'Y'

BEGIN

-- Get the available room

SELECT TOP 1 @RoomID = RoomID

FROM Room

WHERE RoomStatus = 'Available'

ORDER BY RoomID;

-- If a room is available, insert into the Treatment table

IF @RoomID IS NULL

BEGIN

SET @Note = 'Room not available.';

INSERT INTO Treatment (TreatmentID, TreatmentName, TreatmentFees, VisitID, Admit\_Flag, Note)

VALUES (@TreatmentID, @TreatmentName, @TreatmentFees, @VisitID, @Admit\_Flag, @Note);

END

END

ELSE

BEGIN

-- If Admit\_Flag is 'N', simply insert into the Treatment table

INSERT INTO Treatment (TreatmentID, TreatmentName, TreatmentFees, VisitID, Admit\_Flag,Note)

VALUES (@TreatmentID, @TreatmentName, @TreatmentFees, @VisitID, @Admit\_Flag,'');

END

END;

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

–To check availability of room and if available then insert in admission table

CREATE PROCEDURE CheckRoomAvailability

@VisitID VARCHAR(50)

AS

BEGIN

-- Get admission information based on the provided VisitID

DECLARE @RoomCount INT;

SELECT @RoomCount = COUNT(1)

FROM Room

WHERE RoomStatus = 'Available';

IF @RoomCount > 0

BEGIN

-- Room is available, you can proceed with the admission

EXEC InsertAdmissionFromTreatment;

PRINT 'Room is available.Available room is assigned.';

END

ELSE

BEGIN

THROW 50001, 'Room is not available during the specified period.', 1;

END

END;

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**VIEWS**

CREATE VIEW MonthlySummaryView AS

SELECT

Year,

Month,

SUM(InpatientCount) AS TotalInpatient,

SUM(OutpatientCount) AS TotalOutpatient,

SUM(TotalAdmissionDays) AS TotalAdmissionDays,

SUM(TotalRoomCharges) AS TotalRoomCharges,

SUM(TotalTreatmentFees) AS TotalTreatmentFees,

SUM(TotalRevenue) AS TotalRevenue

FROM (

-- Subquery for Inpatient Admissions

SELECT

YEAR(Admission.AdmissionDate) AS Year,

MONTH(Admission.AdmissionDate) AS Month,

COUNT(1) AS InpatientCount,

0 AS OutpatientCount,

SUM(DATEDIFF(DAY, Admission.AdmissionDate, Admission.DischargeDate)) AS TotalAdmissionDays,

SUM(Room.RoomCharge \* DATEDIFF(DAY, Admission.AdmissionDate, Admission.DischargeDate)) AS TotalRoomCharges,

SUM(Treatment.TreatmentFees) AS TotalTreatmentFees,

SUM(Room.RoomCharge \* DATEDIFF(DAY, Admission.AdmissionDate, Admission.DischargeDate) + Treatment.TreatmentFees) AS TotalRevenue

FROM

Admission

INNER JOIN

Room ON Admission.RoomID = Room.RoomID

INNER JOIN

Treatment ON Admission.TreatmentID = Treatment.TreatmentID

GROUP BY

YEAR(Admission.AdmissionDate),

MONTH(Admission.AdmissionDate)

-- Union with Subquery for Outpatient Visits

UNION

-- Subquery for Outpatient Visits

SELECT

YEAR(Visit.VisitDate) AS Year,

MONTH(Visit.VisitDate) AS Month,

0 AS InpatientCount,

COUNT(1) AS OutpatientCount,

0 AS TotalAdmissionDays,

0 AS TotalRoomCharges,

SUM(Treatment.TreatmentFees) AS TotalTreatmentFees,

SUM(Treatment.TreatmentFees) AS TotalRevenue

FROM

Visit

INNER JOIN

Treatment ON Visit.VisitID = Treatment.VisitID

GROUP BY

YEAR(Visit.VisitDate),

MONTH(Visit.VisitDate)

) AS MonthlyData

GROUP BY

Year,

Month

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

CREATE VIEW PatientTreatment AS

SELECT

v.VisitID,

v.PatientID,

p.FirstName+''+p.LastName Patient\_Name,

p.Gender,

d.DoctorID,

t.TreatmentName,

t.TreatmentFees,

t.Admit\_Flag,

b.BillingAmount,

c.PlanName AS CoveragePlan,

py.AmountCoveredByInsurance AS CoverageAmount,

py.AmountToBePaid

FROM

Visit v

JOIN Patient p ON v.PatientID = p.PatientID

JOIN Doctor d ON v.DoctorID = d.DoctorID

LEFT JOIN Treatment t ON v.VisitID = t.VisitID

LEFT JOIN Admission a ON t.TreatmentID = a.TreatmentID

LEFT JOIN Bill b ON v.VisitID = b.VisitID

LEFT JOIN CoveragePlan c ON c.PatientID=v.PatientID

LEFT JOIN Payment py ON b.BillID = py.BillID

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

CREATE VIEW PatientPrescriptionTreatmentView

AS

SELECT

p.PatientID,

p.FirstName + ' ' + p.LastName AS PatientName,

d.DoctorID,

pr.PrescriptionID,

pr.PrescriptionDate,

tr.TreatmentName,

ROW\_NUMBER() OVER (ORDER BY pr.PrescriptionID) AS MedicineLine,

MedicineName

FROM

Patient p

JOIN Visit d ON p.PatientID = d.PatientID

JOIN Treatment tr ON tr.VisitID = d.VisitID

JOIN Prescription pr ON tr.TreatmentID = pr.TreatmentID;

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**DML TRIGGERS**

CREATE TRIGGER UpdateBillStatus

ON Payment

AFTER INSERT

AS

BEGIN

SET NOCOUNT ON;

UPDATE Bill

SET Status = 'Paid'

FROM Bill b

INNER JOIN inserted i ON b.BillID = i.BillID;

END;

CREATE TRIGGER UpdateBillingOnDischarge

ON Admission

AFTER UPDATE

AS

BEGIN

-- Check if DischargeDate is updated

IF UPDATE(DischargeDate)

BEGIN

-- Call the InsertBillingData stored procedure

EXEC InsertBillingData;

END

END;

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**NON CLUSTERED INDEXES**

CREATE NONCLUSTERED INDEX IX\_Visit\_DoctorID

ON Visit (DoctorID);

CREATE NONCLUSTERED INDEX IX\_Visit\_PatientID

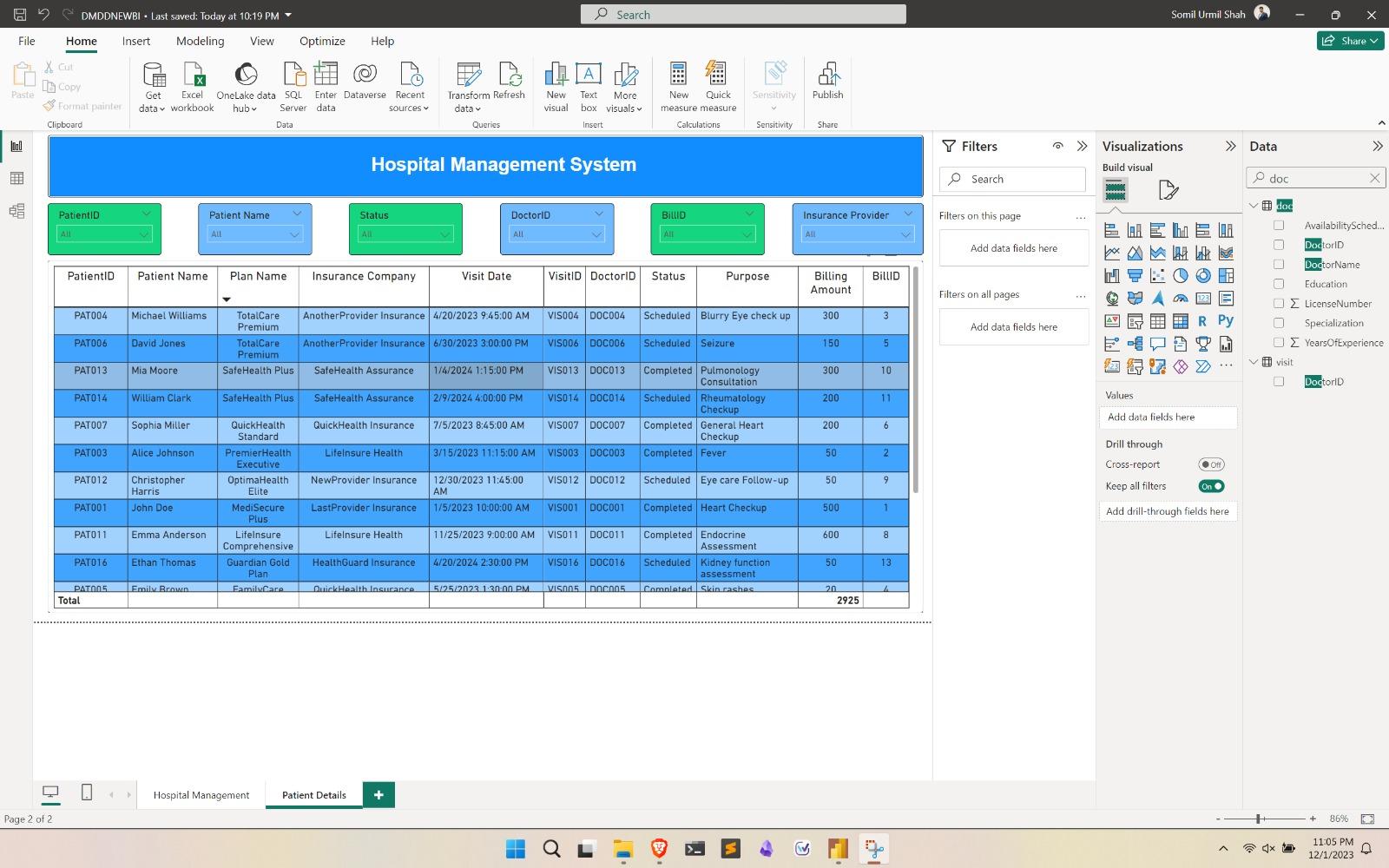
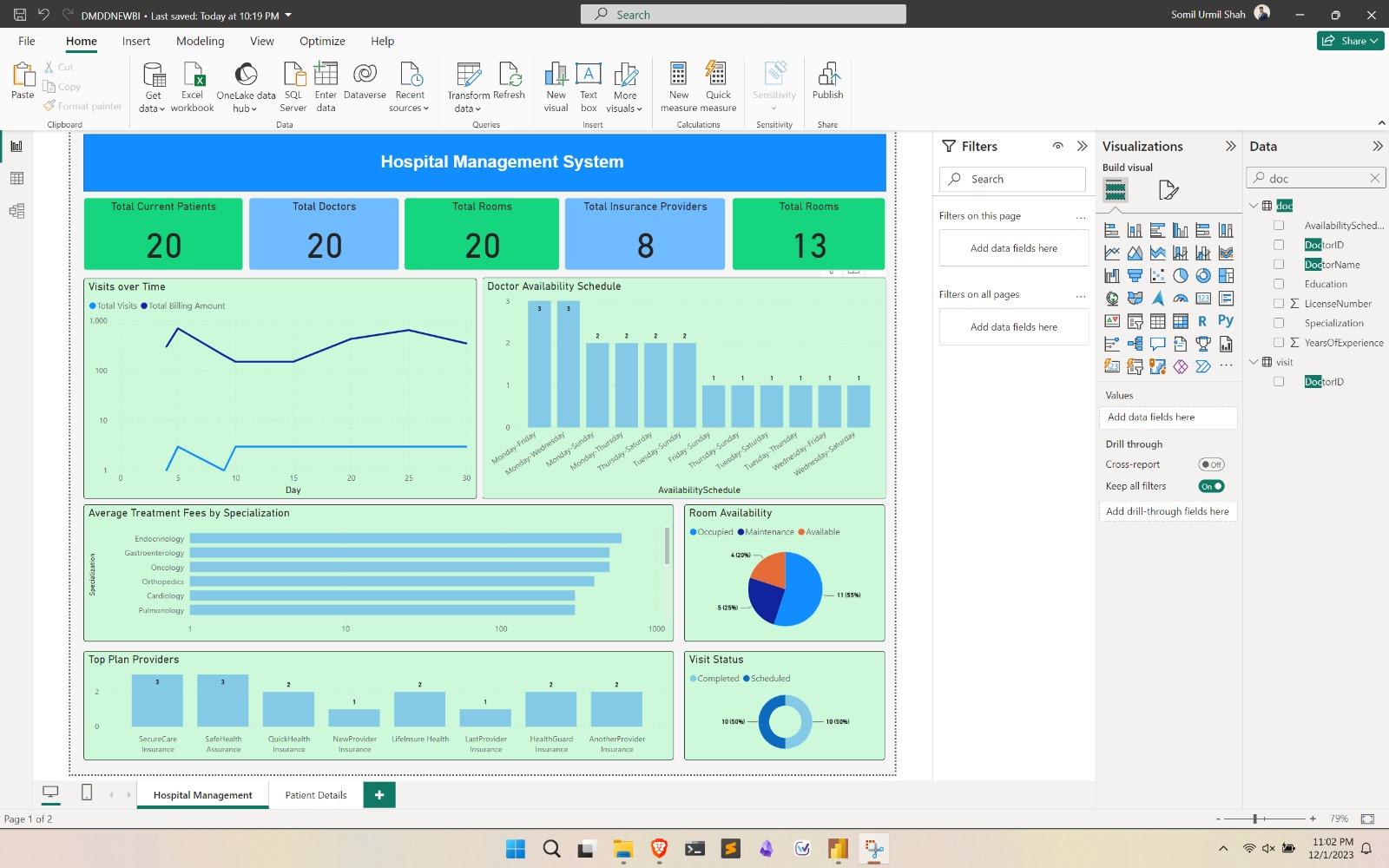
ON Visit (PatientID);

CREATE NONCLUSTERED INDEX IX\_Bill\_VisitID

ON Bill (VisitID);

CREATE NONCLUSTERED INDEX IX\_Admission\_TreatmentID

ON Admission (TreatmentID);



**For GUI further reference check GIT solution for “Visual Studio”**