REQUIREMENTS:

---- Stored Procedure 1 ----

```
CREATE PROCEDURE CheckInsuranceCoverage
   @p_PatientID INT,
   @p_InsuranceCoverageType NVARCHAR(305) OUTPUT
AS
BEGIN
   SET @p_InsuranceCoverageType = NULL;

   SELECT @p_InsuranceCoverageType = Coverage
   FROM InsuranceCoverage
   WHERE PatientID = @p_PatientID;
END
```

```
DECLARE @InsuranceType NVARCHAR(255);
EXEC CheckInsuranceCoverage @p_PatientID = 101, @p_InsuranceCoverageType = @InsuranceType OUTPUT;
PRINT 'Insurance Coverage Type: ' + COALESCE(@InsuranceType, 'Not Available');
```

```
DECLARE @InsuranceType NVARCHAR(255);
EXEC CheckInsuranceCoverage @p_PatientID = 101, @p_InsuranceCoverageType = @InsuranceType OUTPUT;
PRINT 'Insurance Coverage Type: ' + COALESCE(@InsuranceType, 'Not Available');

DECLARE @InsuranceType1 NVARCHAR(255);
EXEC CheckInsuranceCoverage @p_PatientID = 102, @p_InsuranceCoverageType = @InsuranceType1 OUTPUT;
```

PRINT 'Insurance Coverage Type: ' + COALESCE(@InsuranceType1, 'Not Available');

DECLARE @InsuranceType2 NVARCHAR(255);

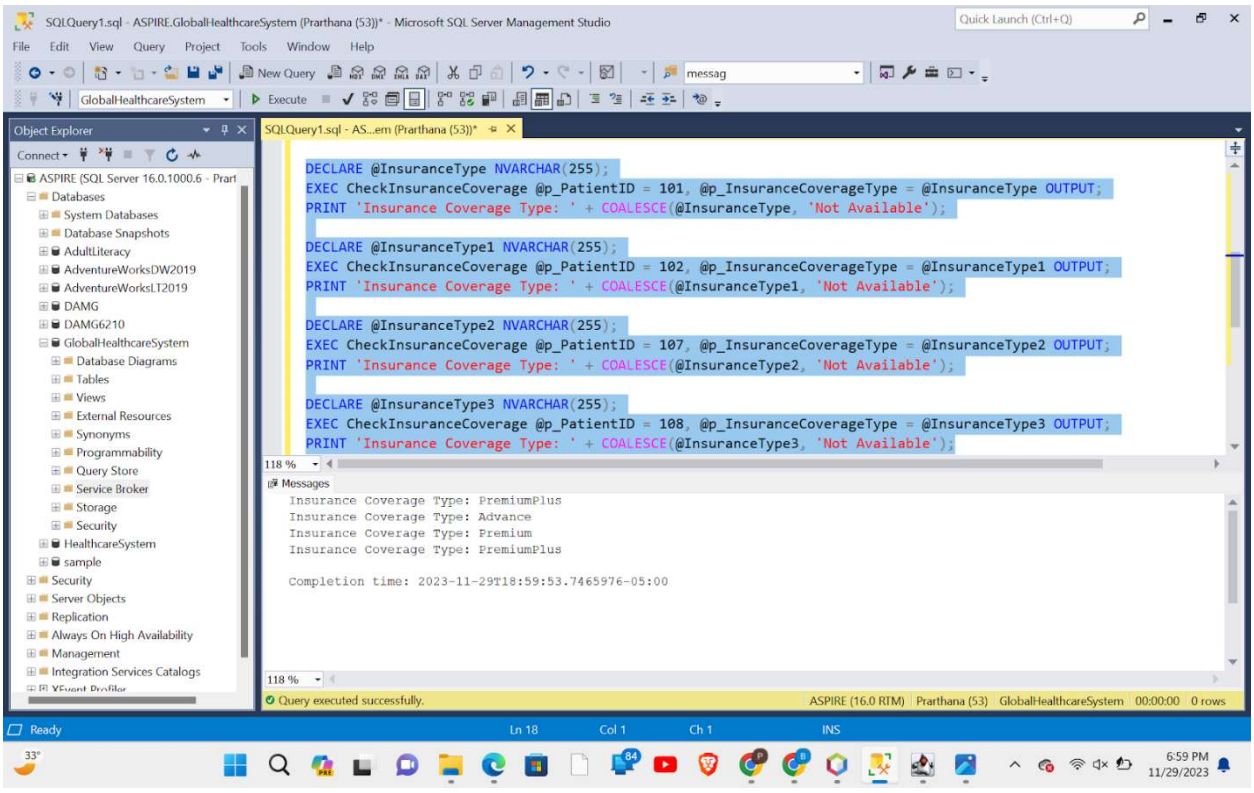
EXEC CheckInsuranceCoverage @p_PatientID = 107, @p_InsuranceCoverageType = @InsuranceType2 OUTPUT;

PRINT 'Insurance Coverage Type: ' + COALESCE(@InsuranceType2, 'Not Available');

DECLARE @InsuranceType3 NVARCHAR(255);

EXEC CheckInsuranceCoverage @p_PatientID = 108, @p_InsuranceCoverageType = @InsuranceType3 OUTPUT;

PRINT 'Insurance Coverage Type: ' + COALESCE(@InsuranceType3, 'Not Available');



---- Stored Procedure 2 ----

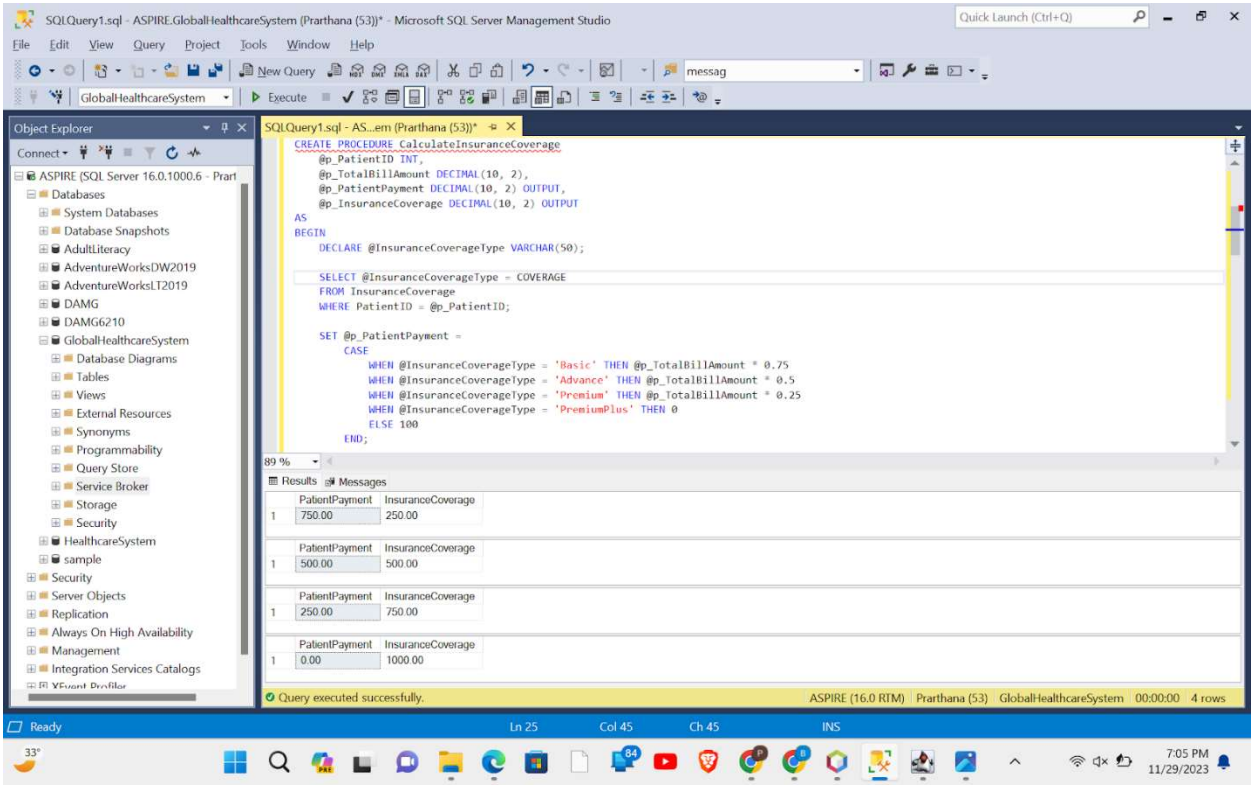
CREATE PROCEDURE CalculateInsuranceCoverage

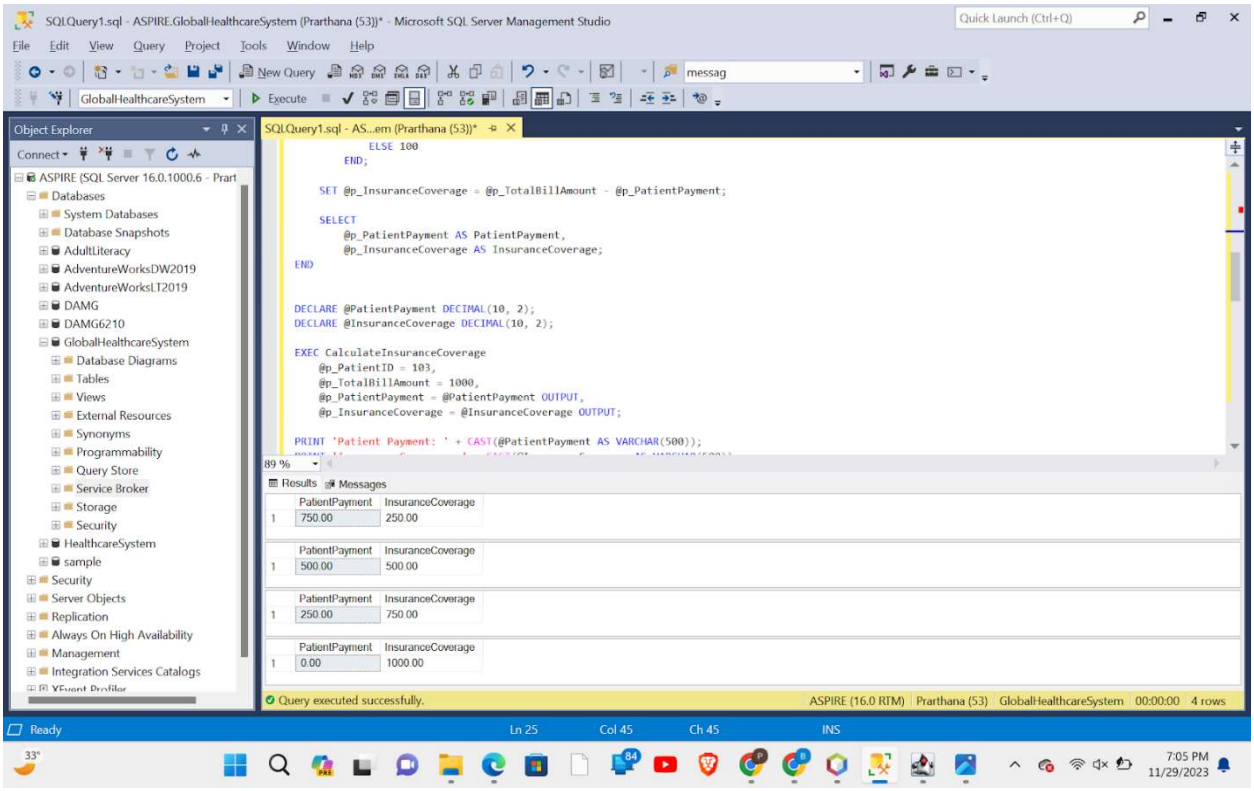
    @p_PatientID INT,

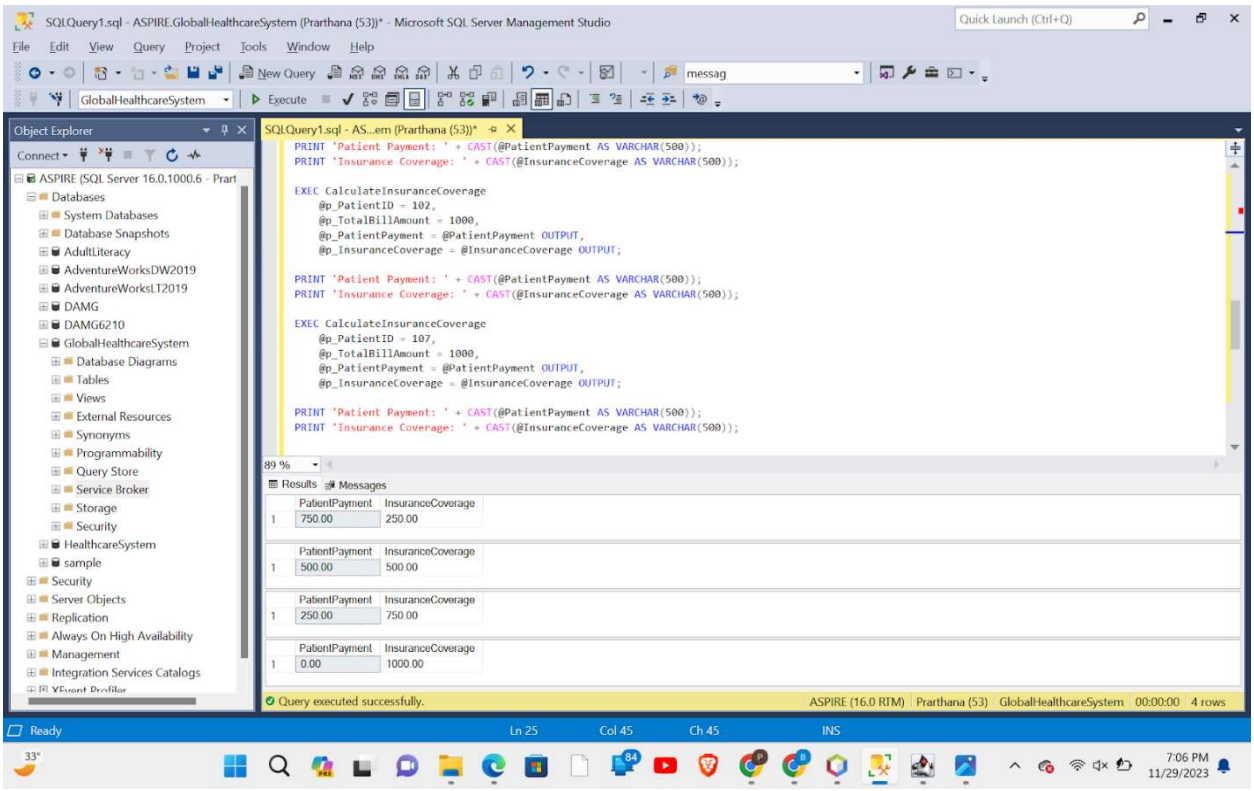
```
    @p_TotalBillAmount DECIMAL(10, 2),
    @p_PatientPayment DECIMAL(10, 2) OUTPUT,
    @p_InsuranceCoverage DECIMAL(10, 2) OUTPUT
AS
BEGIN
    DECLARE @InsuranceCoverageType VARCHAR(50);

    SELECT @InsuranceCoverageType = COVERAGE
    FROM InsuranceCoverage
    WHERE PatientID = @p_PatientID;

    SET @p_PatientPayment =
        CASE
            WHEN @InsuranceCoverageType = 'Basic' THEN @p_TotalBillAmount * 0.75
            WHEN @InsuranceCoverageType = 'Advance' THEN @p_TotalBillAmount * 0.5
            WHEN @InsuranceCoverageType = 'Premium' THEN @p_TotalBillAmount * 0.25
            WHEN @InsuranceCoverageType = 'PremiumPlus' THEN 0
            ELSE 100
        END;

    SET @p_InsuranceCoverage = @p_TotalBillAmount - @p_PatientPayment;

    SELECT
        @p_PatientPayment AS PatientPayment,
        @p_InsuranceCoverage AS InsuranceCoverage;
END

DECLARE @PatientPayment DECIMAL(10, 2);
DECLARE @InsuranceCoverage DECIMAL(10, 2);
```

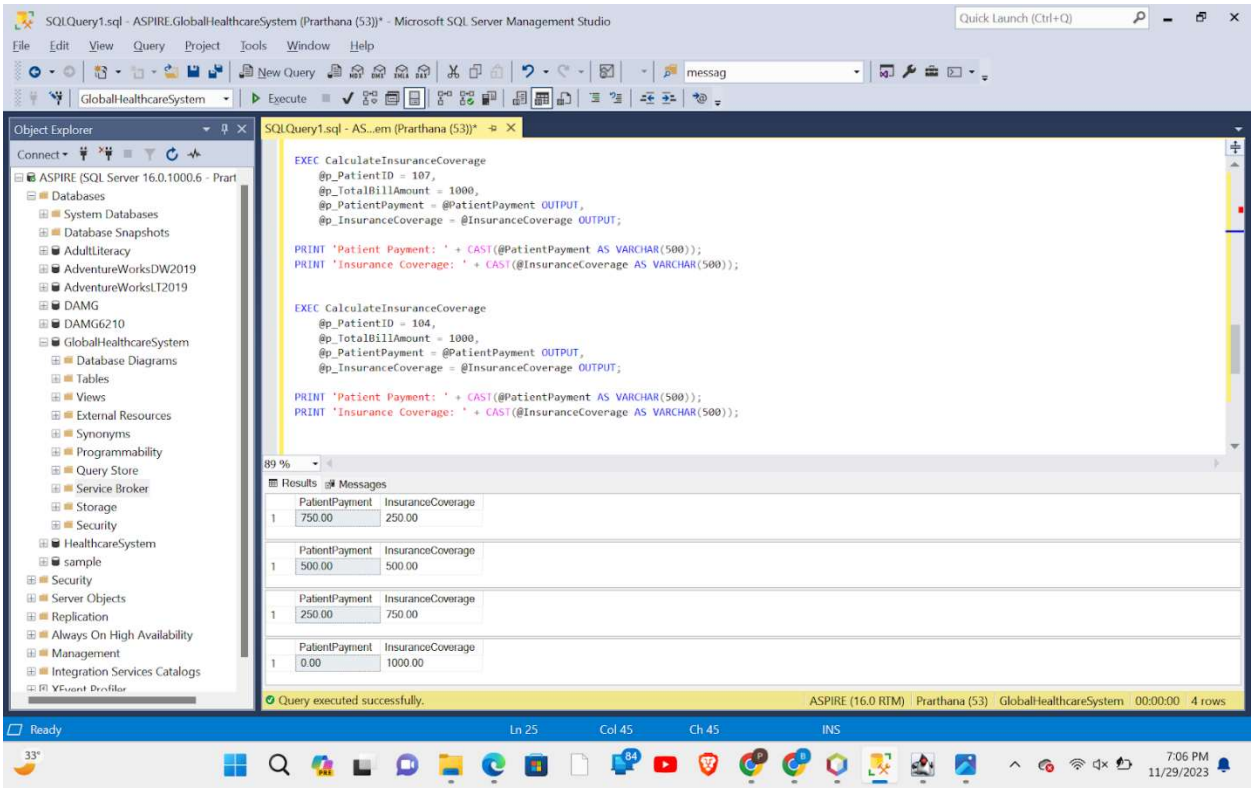
```
EXEC CalculateInsuranceCoverage
    @p_PatientID = 103,
    @p_TotalBillAmount = 1000,
    @p_PatientPayment = @PatientPayment OUTPUT,
    @p_InsuranceCoverage = @InsuranceCoverage OUTPUT;

PRINT 'Patient Payment: ' + CAST(@PatientPayment AS VARCHAR(500));
PRINT 'Insurance Coverage: ' + CAST(@InsuranceCoverage AS VARCHAR(500));

EXEC CalculateInsuranceCoverage
    @p_PatientID = 102,
    @p_TotalBillAmount = 1000,
    @p_PatientPayment = @PatientPayment OUTPUT,
    @p_InsuranceCoverage = @InsuranceCoverage OUTPUT;

PRINT 'Patient Payment: ' + CAST(@PatientPayment AS VARCHAR(500));
PRINT 'Insurance Coverage: ' + CAST(@InsuranceCoverage AS VARCHAR(500));

EXEC CalculateInsuranceCoverage
    @p_PatientID = 107,
    @p_TotalBillAmount = 1000,
    @p_PatientPayment = @PatientPayment OUTPUT,
    @p_InsuranceCoverage = @InsuranceCoverage OUTPUT;

PRINT 'Patient Payment: ' + CAST(@PatientPayment AS VARCHAR(500));
PRINT 'Insurance Coverage: ' + CAST(@InsuranceCoverage AS VARCHAR(500));
```

```
EXEC CalculateInsuranceCoverage
    @p_PatientID = 104,
    @p_TotalBillAmount = 1000,
    @p_PatientPayment = @PatientPayment OUTPUT,
    @p_InsuranceCoverage = @InsuranceCoverage OUTPUT;

PRINT 'Patient Payment: ' + CAST(@PatientPayment AS VARCHAR(500));
PRINT 'Insurance Coverage: ' + CAST(@InsuranceCoverage AS VARCHAR(500));
```




SQLQuery1.sql - ASPIRE.GlobalHealthcareSystem (Prarthana (53))* - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File Edit View Query Project Tools Window Help
New Query
messag
GlobalHealthcareSystem
Execute
Object Explorer
Connect
ASPIRE (SQL Server 16.0.1000.6 - Prart
Databases
System Databases
Database Snapshots
AdultLiteracy
AdventureWorksDW2019
AdventureWorksLT2019
DAMG
DAMG6210
GlobalHealthcareSystem
Database Diagrams
Tables
Views
External Resources
Synonyms
Programmability
Query Store
Service Broker
Storage
Security
HealthcareSystem
sample
Security
Server Objects
Replication
Always On High Availability
Management
Integration Services Catalogs
SQLQuery1.sql - AS...em (Prarthana (53))*
ELSE 100
END;
SET @p_InsuranceCoverage = @p_TotalBillAmount - @p_PatientPayment;
SELECT
@p_PatientPayment AS PatientPayment,
@p_InsuranceCoverage AS InsuranceCoverage;
END
DECLARE @PatientPayment DECIMAL(10, 2);
DECLARE @InsuranceCoverage DECIMAL(10, 2);
EXEC CalculateInsuranceCoverage
@p_PatientID = 103,
@p_TotalBillAmount = 1000,
@p_PatientPayment = @PatientPayment OUTPUT,
@p_InsuranceCoverage = @InsuranceCoverage OUTPUT;
PRINT 'Patient Payment: ' + CAST(@PatientPayment AS VARCHAR(500));
89 %
Results
Messages
PatientPayment InsuranceCoverage
1 750.00 250.00
PatientPayment InsuranceCoverage
1 500.00 500.00
PatientPayment InsuranceCoverage
1 250.00 750.00
PatientPayment InsuranceCoverage
1 0.00 1000.00
Query executed successfully.
ASPIRE (16.0 RTM) Prarthana (53) GlobalHealthcareSystem 00:00:00 4 rows
Ready
Ln 25 Col 45 Ch 45 INS
7:05 PM
11/29/2023

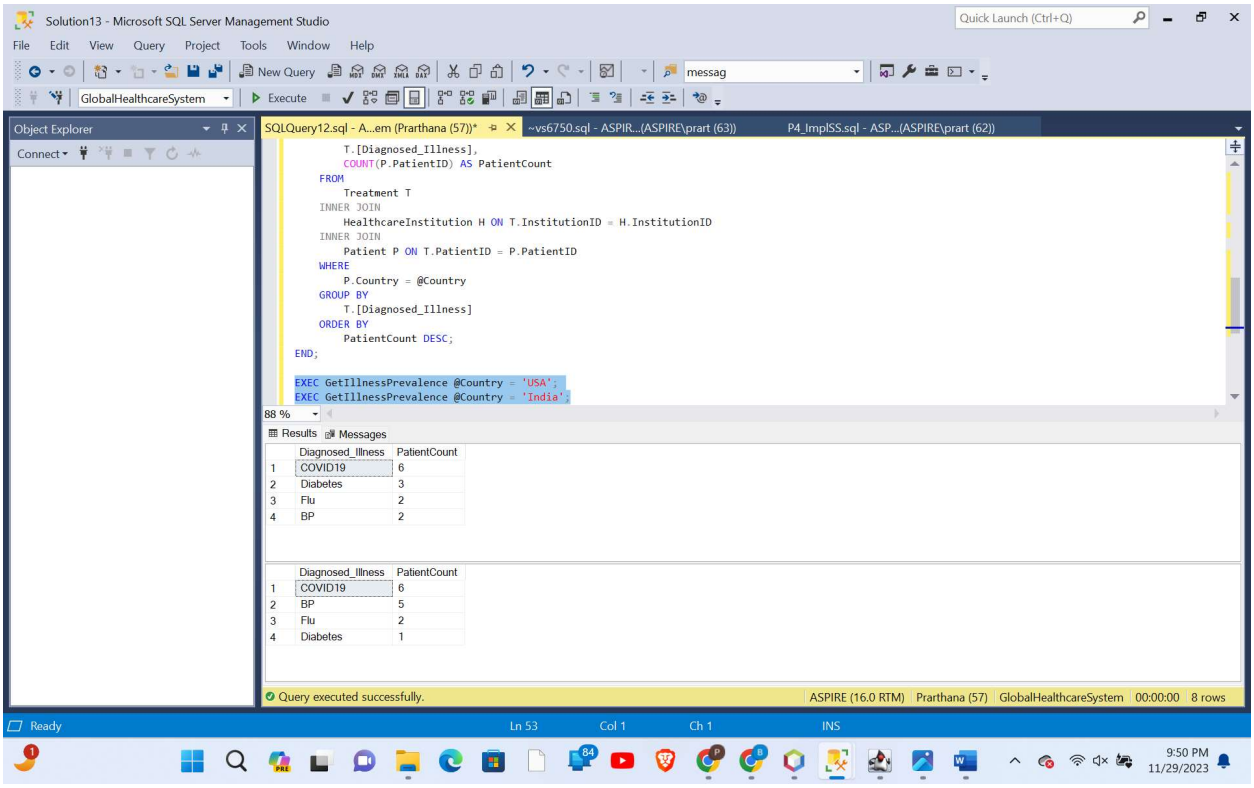

SQLQuery1.sql - ASPIRE.GlobalHealthcareSystem (Prarthana (53))* - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File Edit View Query Project Tools Window Help
New Query
messag
GlobalHealthcareSystem
Execute
Object Explorer
Connect
ASPIRE (SQL Server 16.0.1000.6 - Prart
Databases
System Databases
Database Snapshots
AdultLiteracy
AdventureWorksDW2019
AdventureWorksLT2019
DAMG
DAMG6210
GlobalHealthcareSystem
Database Diagrams
Tables
Views
External Resources
Synonyms
Programmability
Query Store
Service Broker
Storage
Security
HealthcareSystem
sample
Security
Server Objects
Replication
Always On High Availability
Management
Integration Services Catalogs
SQLQuery1.sql - AS...em (Prarthana (53))*
PRINT 'Patient Payment: ' + CAST(@PatientPayment AS VARCHAR(500));
PRINT 'Insurance Coverage: ' + CAST(@InsuranceCoverage AS VARCHAR(500));
EXEC CalculateInsuranceCoverage
@p_PatientID = 102,
@p_TotalBillAmount = 1000,
@p_PatientPayment = @PatientPayment OUTPUT,
@p_InsuranceCoverage = @InsuranceCoverage OUTPUT;
PRINT 'Patient Payment: ' + CAST(@PatientPayment AS VARCHAR(500));
PRINT 'Insurance Coverage: ' + CAST(@InsuranceCoverage AS VARCHAR(500));
EXEC CalculateInsuranceCoverage
@p_PatientID = 107,
@p_TotalBillAmount = 1000,
@p_PatientPayment = @PatientPayment OUTPUT,
@p_InsuranceCoverage = @InsuranceCoverage OUTPUT;
PRINT 'Patient Payment: ' + CAST(@PatientPayment AS VARCHAR(500));
PRINT 'Insurance Coverage: ' + CAST(@InsuranceCoverage AS VARCHAR(500));
89 %
Results
Messages
PatientPayment InsuranceCoverage
1 750.00 250.00
PatientPayment InsuranceCoverage
1 500.00 500.00
PatientPayment InsuranceCoverage
1 250.00 750.00
PatientPayment InsuranceCoverage
1 0.00 1000.00
Query executed successfully.
ASPIRE (16.0 RTM) Prarthana (53) GlobalHealthcareSystem 00:00:00 4 rows
Ready
Ln 25 Col 45 Ch 45 INS
7:06 PM
11/29/2023



---- Stored Procedure 3 ----

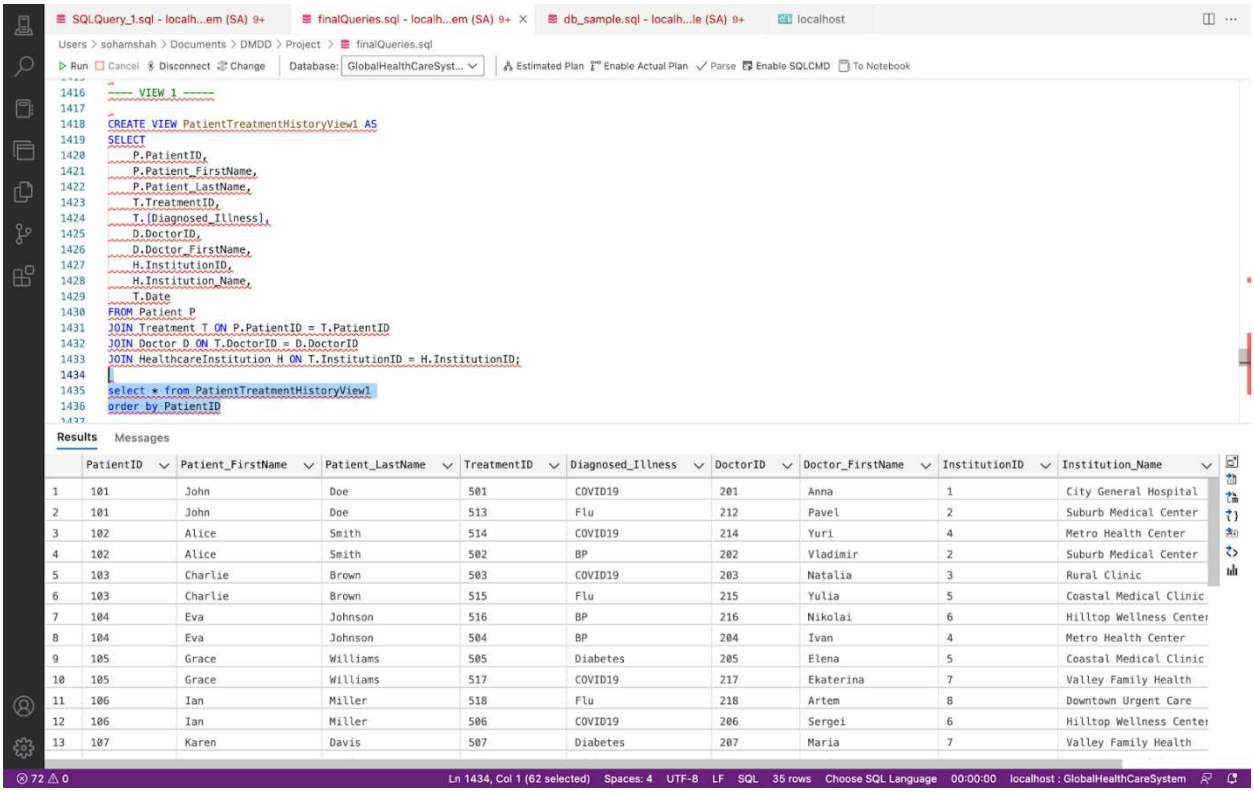
```
CREATE PROCEDURE GetIllnessPrevalence
    @Country VARCHAR(50)
AS
BEGIN
    SELECT
        T.[Diagnosed_Illness],
        COUNT(P.PatientID) AS PatientCount
    FROM
        Treatment T
    INNER JOIN
        HealthcareInstitution H ON T.InstitutionID = H.InstitutionID
    INNER JOIN
```

```
    Patient P ON T.PatientID = P.PatientID
WHERE
    P.Country = @Country
GROUP BY
    T.[Diagnosed_Illness]
ORDER BY
    PatientCount DESC;
END;

EXEC GetIllnessPrevalence @Country = 'USA';
EXEC GetIllnessPrevalence @Country = 'India';
```



---- VIEW 1 -----

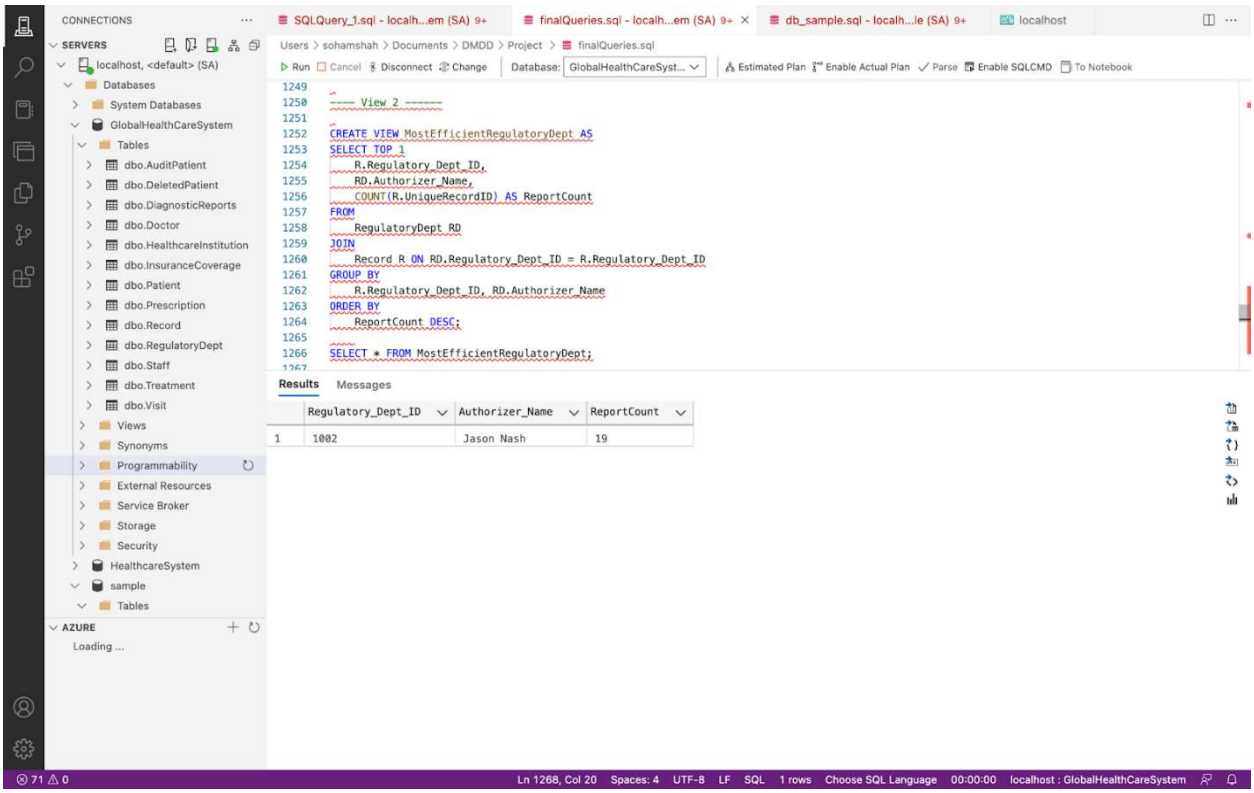
```
CREATE VIEW PatientTreatmentHistoryView1 AS
SELECT
    P.PatientID,
    P.Patient_FirstName,
    P.Patient_LastName,
    T.TreatmentID,
    T.[Diagnosed_Illness],
    D.DoctorID,
    D.Doctor_FirstName,
    H.InstitutionID,
    H.Institution_Name,
    T.Date
FROM Patient P
JOIN Treatment T ON P.PatientID = T.PatientID
JOIN Doctor D ON T.DoctorID = D.DoctorID
JOIN HealthcareInstitution H ON T.InstitutionID = H.InstitutionID;

select * from PatientTreatmentHistoryView1
order by PatientID
```



---- View 2 ------

```
CREATE VIEW MostEfficientRegulatoryDept AS
SELECT TOP 1
    R.Regulatory_Dept_ID,
    RD.Authorizer_Name,
    COUNT(R.UniqueRecordID) AS ReportCount
FROM
    RegulatoryDept RD
JOIN
    Record R ON RD.Regulatory_Dept_ID = R.Regulatory_Dept_ID
GROUP BY
    R.Regulatory_Dept_ID, RD.Authorizer_Name
ORDER BY
    ReportCount DESC;
```

SELECT * FROM MostEfficientRegulatoryDept;



---- VIew 3 -----

CREATE VIEW TreatmentPatientTurnover AS

SELECT

T.TreatmentID,

COUNT(P.PatientID) AS PatientCount,

COUNT(P.PatientID) * T.Cost AS TreatmentTurnover

FROM

Patient P

JOIN

TREATMENT T ON P.PatientID = T.PatientID

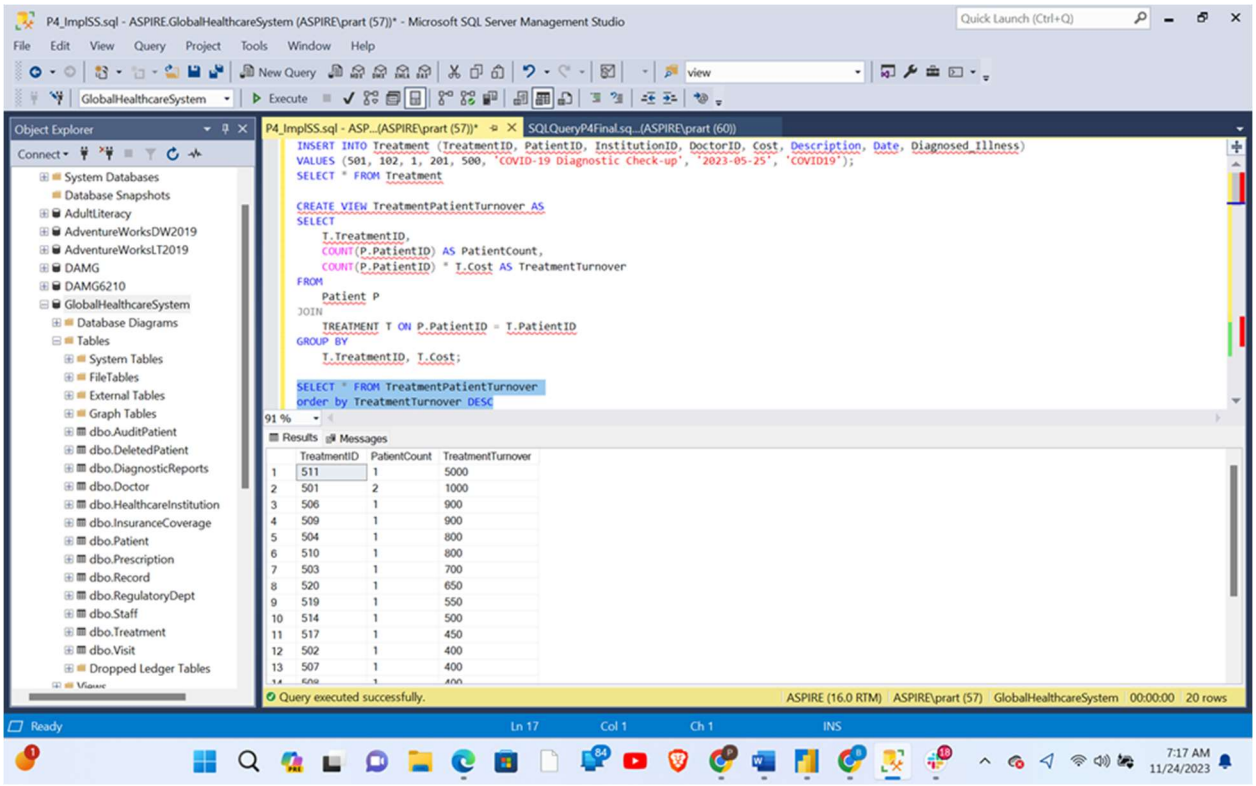
GROUP BY

T.TreatmentID, T.Cost;

(--Inserting a dummy value in treatment 501 to check

```
INSERT INTO Treatment (TreatmentID, PatientID, InstitutionID, DoctorID, Cost, Description, Date, Diagnosed_Illness)

VALUES (501, 102, 1, 201, 500, 'COVID-19 Diagnostic Check-up', '2023-05-25', 'COVID19');

SELECT * FROM Treatment)
```

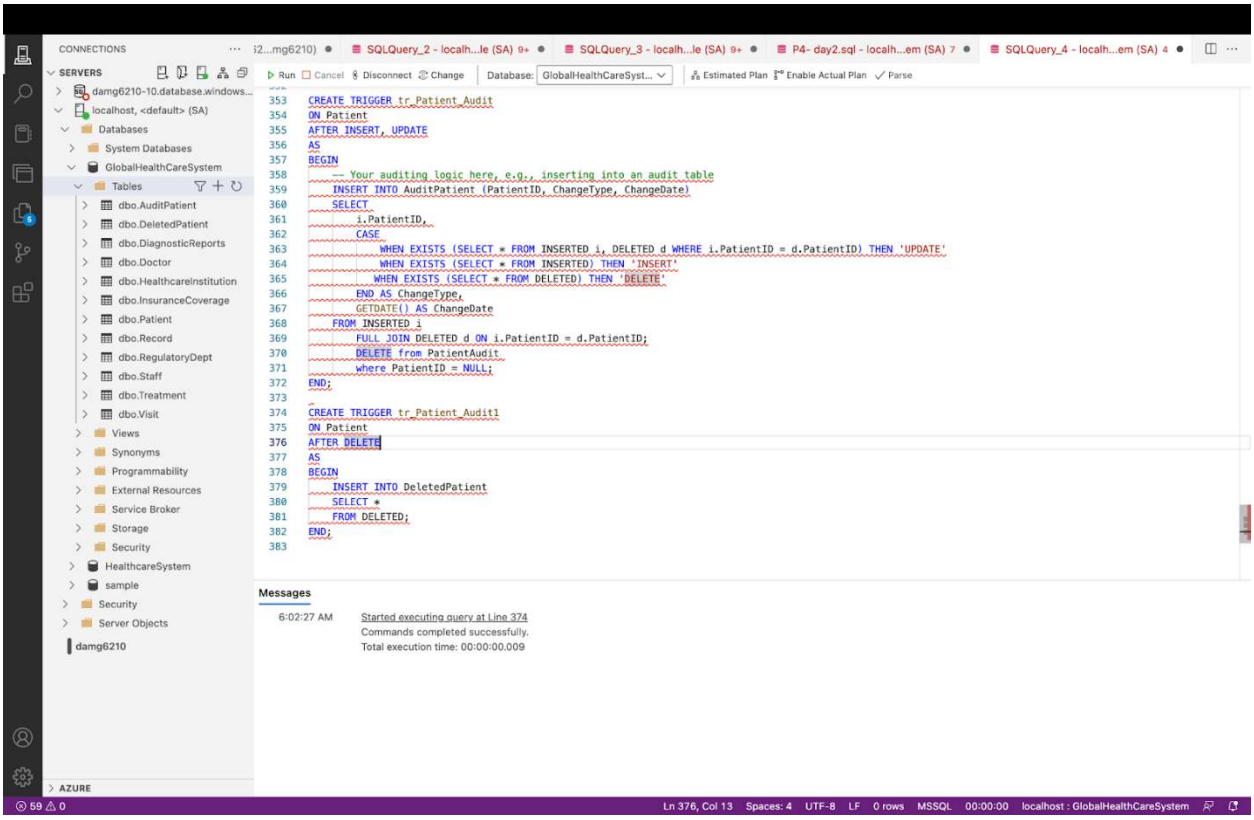
```
SELECT * FROM TreatmentPatientTurnover

order by TreatmentTurnover DESC
```



---- Trigger 1 ----

```
CREATE TRIGGER tr_Patient_Audit

ON Patient

AFTER INSERT, UPDATE

AS

BEGIN
```

```
  INSERT INTO AuditPatient (PatientID, ChangeType, ChangeDate)
  SELECT
      i.PatientID,
      CASE
          WHEN EXISTS (SELECT * FROM INSERTED i, DELETED d WHERE i.PatientID = d.PatientID) THEN
'UPDATE'
          WHEN EXISTS (SELECT * FROM INSERTED) THEN 'INSERT'
         WHEN EXISTS (SELECT * FROM DELETED) THEN 'DELETE'
      END AS ChangeType,
      GETDATE() AS ChangeDate
  FROM INSERTED i
      FULL JOIN DELETED d ON i.PatientID = d.PatientID;
      DELETE from AuditPatient
      where PatientID = NULL;
END;
```



```
END;
```

INSERT INTO Patient (PatientID, Patient_FirstName, Patient_MiddleName, Patient_LastName, Patient_Phone_Num,

Patient_Date_of_Birth, Sex, Height, Weight, Blood_Group, Address, Country,

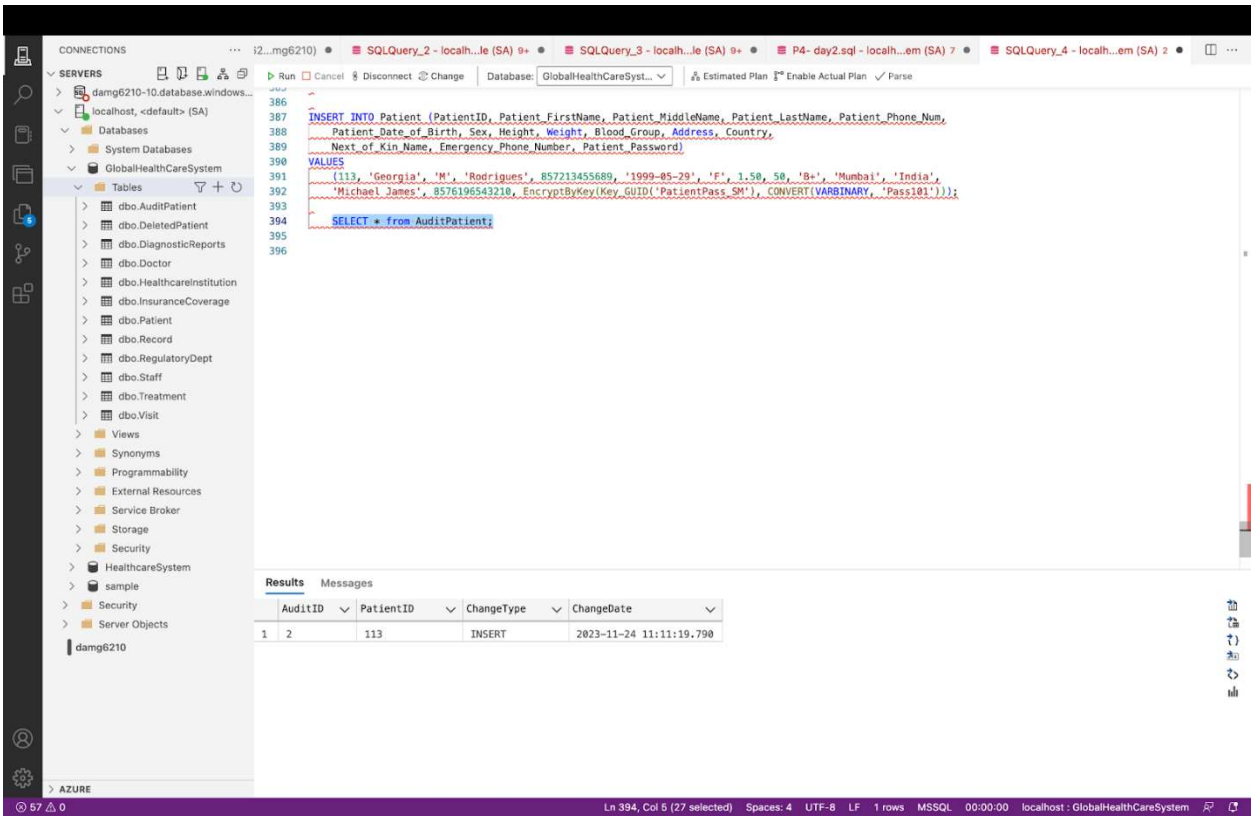
Next_of_Kin_Name, Emergency_Phone_Number, Patient_Password)

VALUES

(113, 'Georgia', 'M', 'Rodrigues', 857213455689, '1999-05-29', 'F', 1.50, 50, 'B+', 'Mumbai', 'India',

'Michael James', 8576196543210, EncryptByKey(Key_GUID('PatientPass_SM'), CONVERT(VARBINARY, 'Pass101')));

SELECT * from AuditPatient;

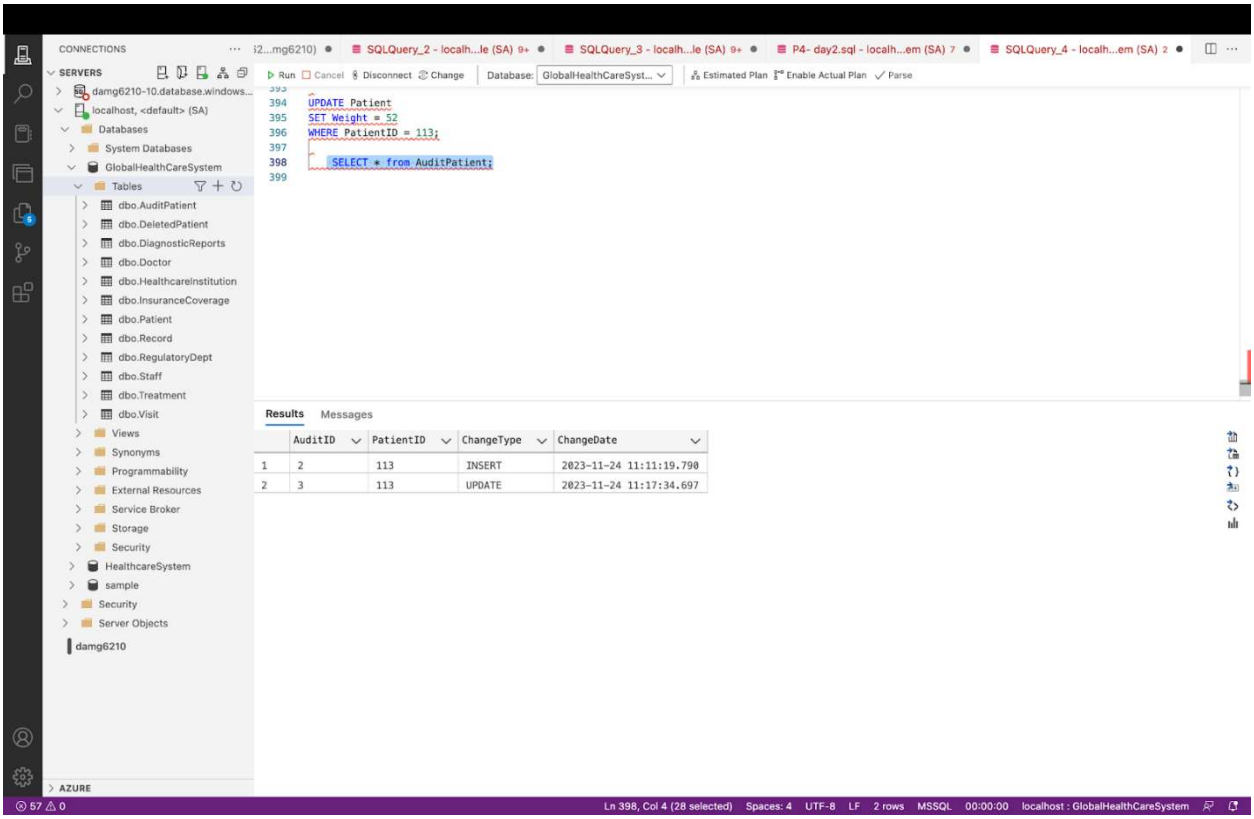


UPDATE Patient

SET Weight = 52

WHERE PatientID = 113;

SELECT * from AuditPatient;

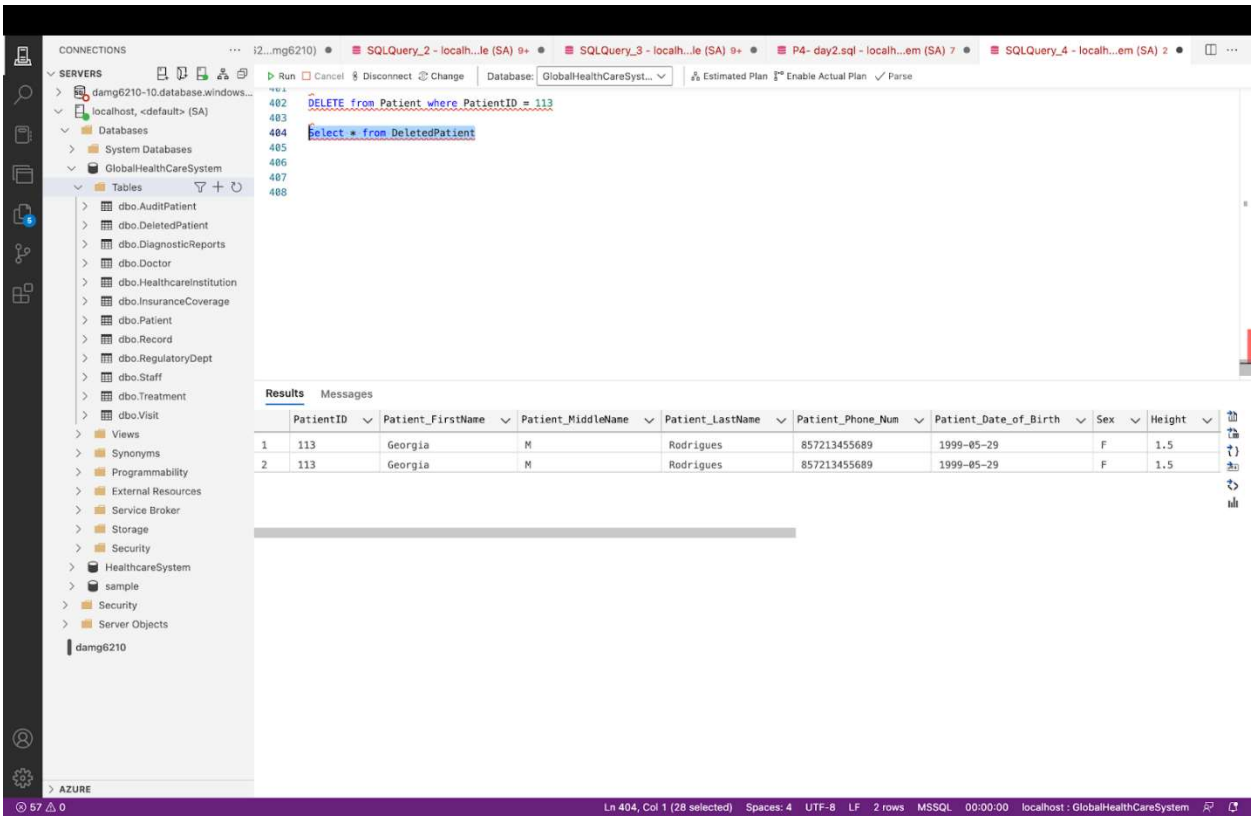


---- Trigger 2 ----

```
CREATE TRIGGER tr_Patient_Audit1
ON Patient
AFTER DELETE
AS
BEGIN
  INSERT INTO DeletedPatient
  SELECT *
  FROM DELETED;
END;
```

DELETE from Patient WHERE PatientID = 113

Select * from DeletedPatient



---- Trigger 3 ----

CREATE TRIGGER tr_Patient_ValidateEmergencyPhoneNumber

ON Patient

AFTER INSERT, UPDATE

AS

BEGIN

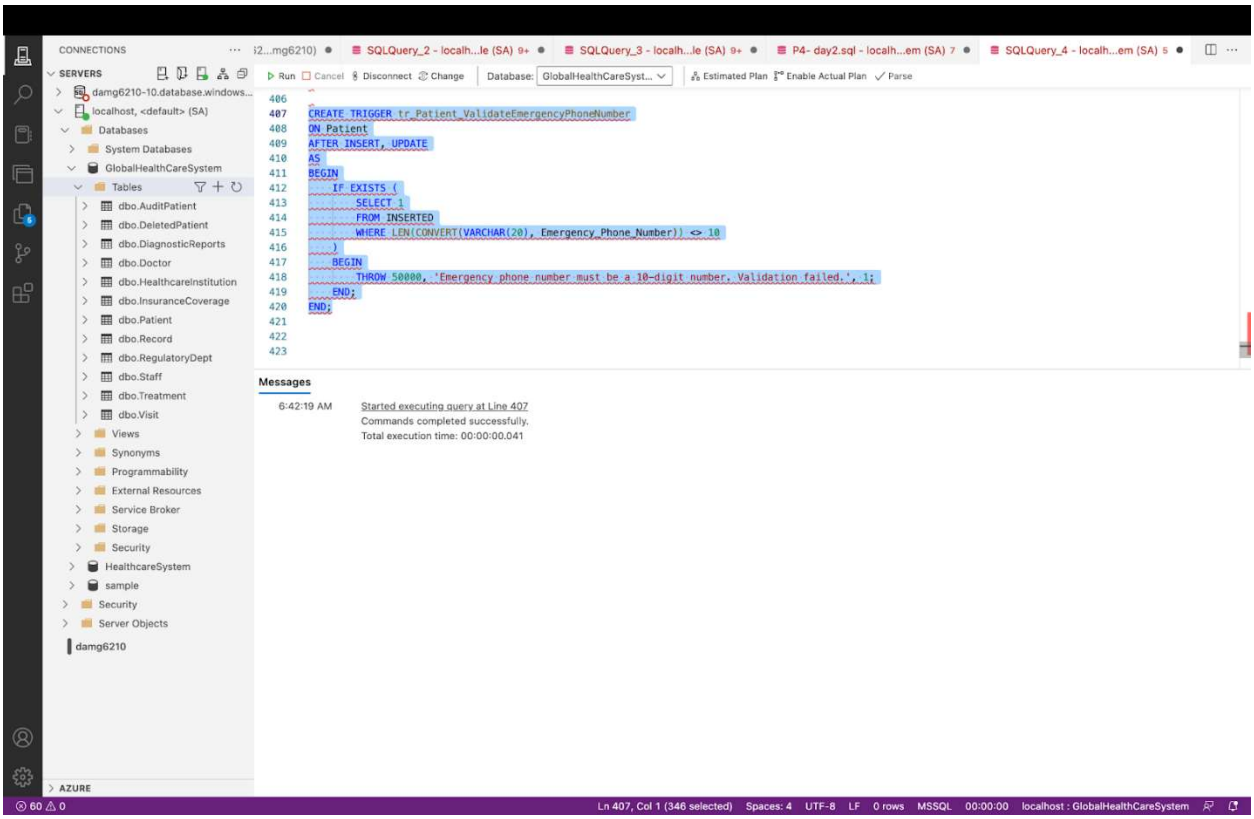
  IF EXISTS (

    SELECT 1

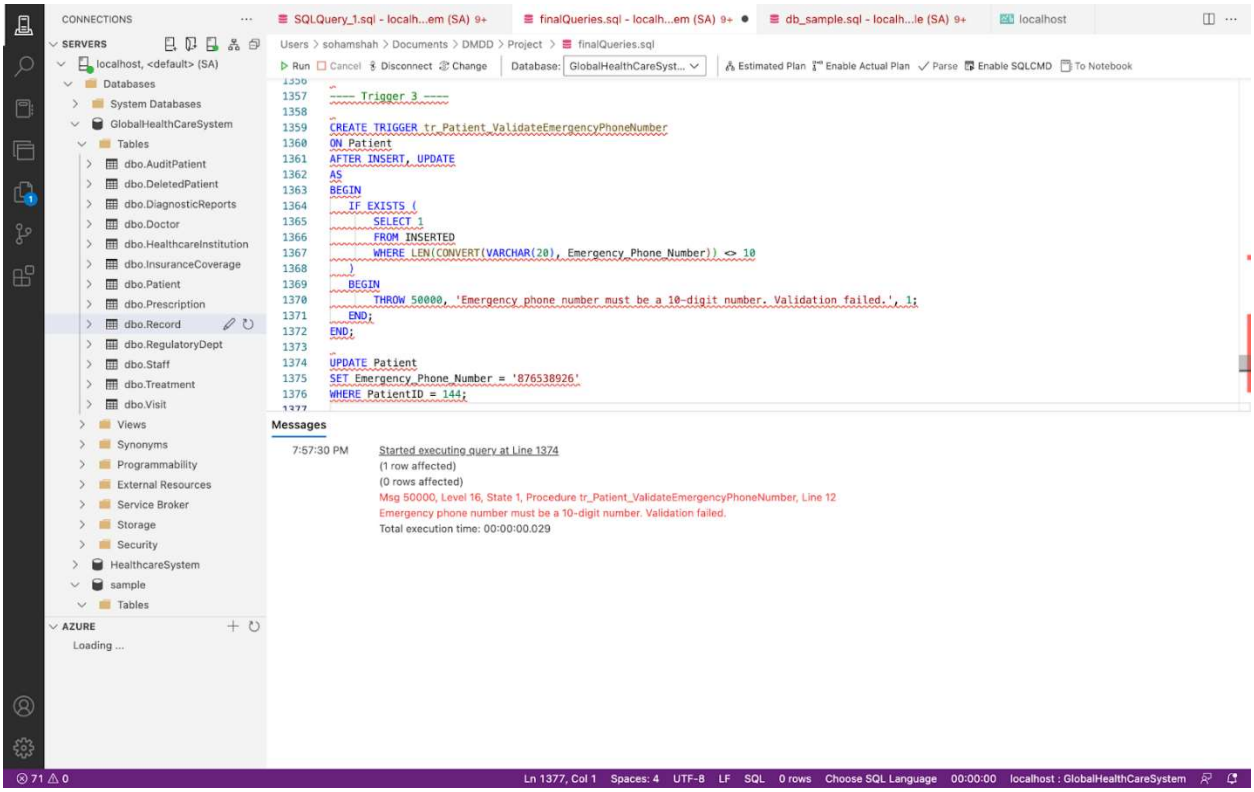
    FROM INSERTED

    WHERE LEN(CONVERT(VARCHAR(20), Emergency_Phone_Number)) <> 10

```
  )
  BEGIN
      THROW 50000, 'Emergency phone number must be a 10-digit number. Validation failed.', 1;
  END;
END;
```

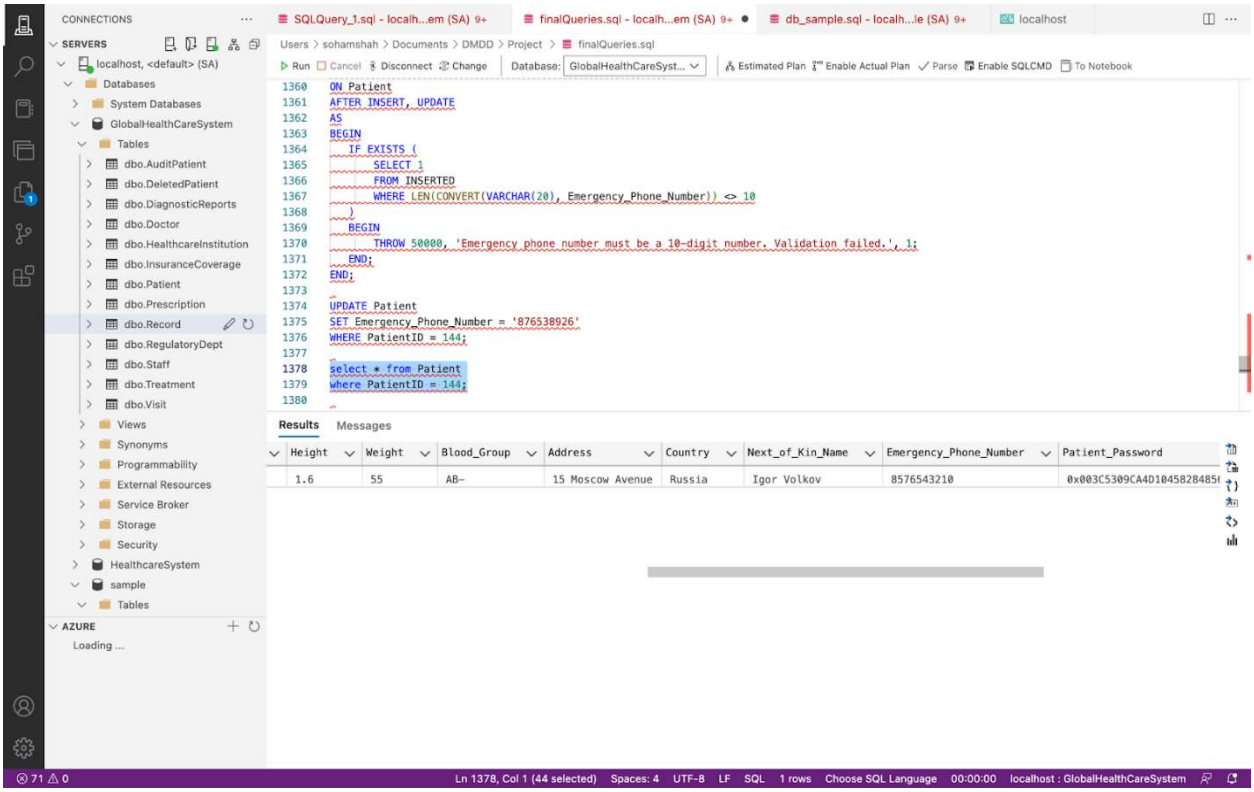



```
UPDATE Patient
SET Emergency_Phone_Number = '876538926'
WHERE PatientID = 144;
```



select * from Patient

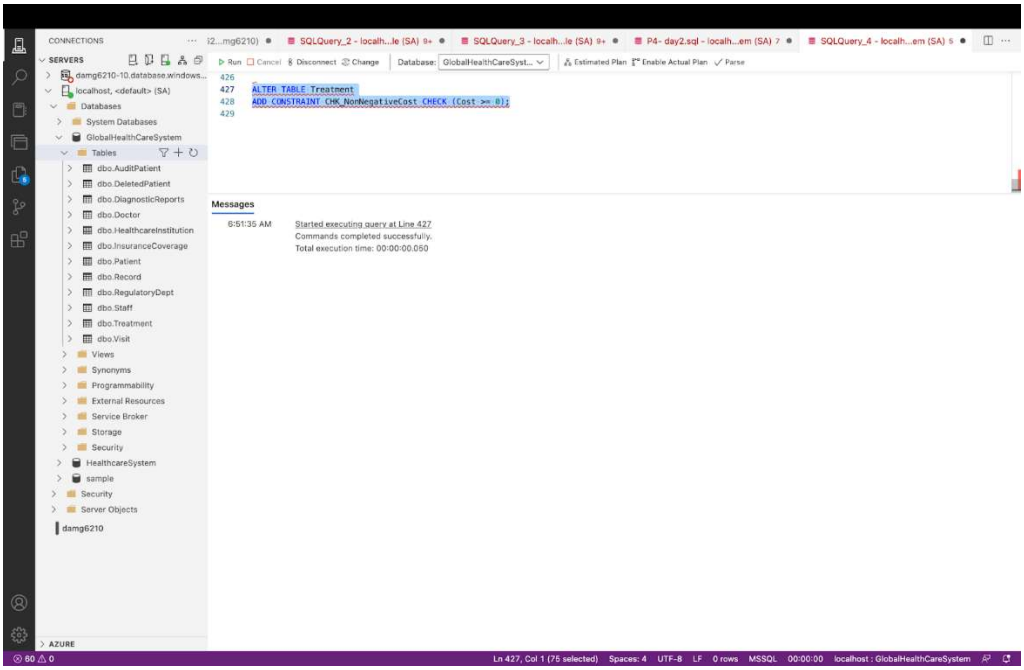
where PatientID = 144;



---- Check Constraint on Cost using Table Treatment ----

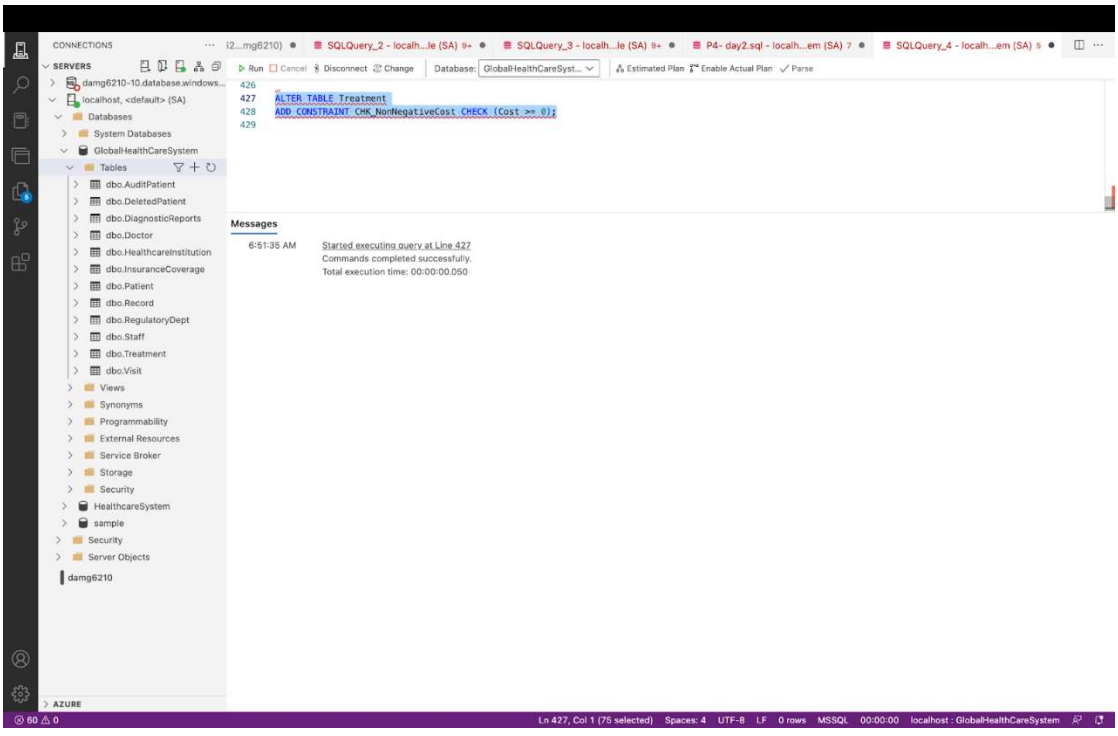
ALTER TABLE Treatment

ADD CONSTRAINT CHK_NonNegativeCost CHECK (Cost >= 0);

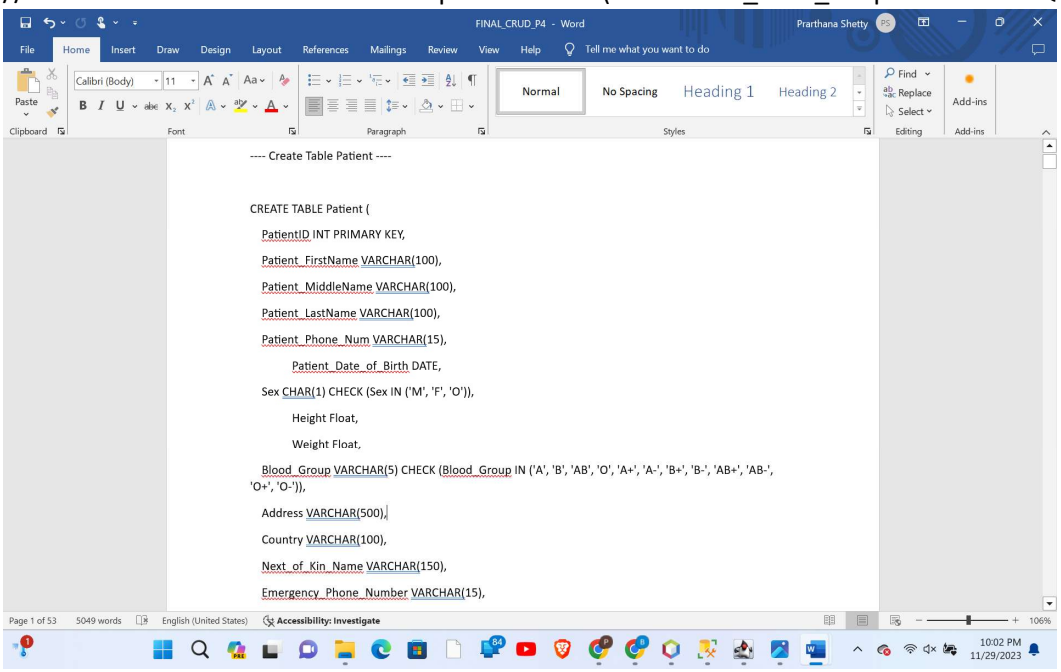


Update Treatment set Cost = -1000

where TreatmentID = 520

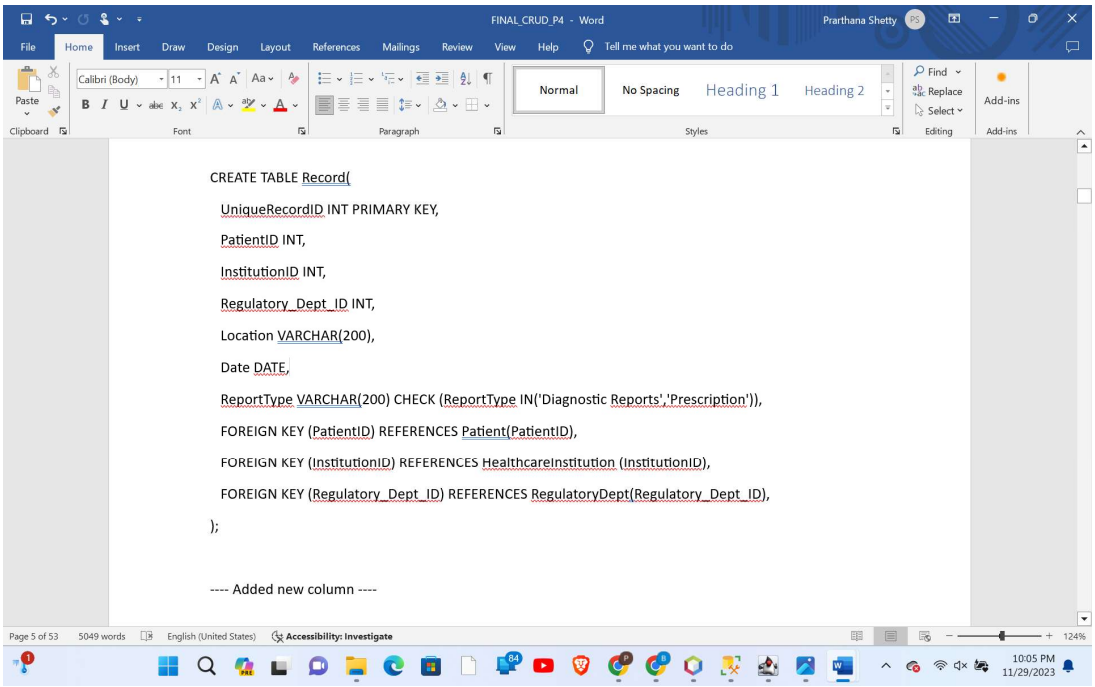


//Patient sex and Patient Blood Group Constraints (Refer ‘Final_Crud_P4’ pdf file and ‘SQL_CRUD’ file)



```
---- Create Table Patient ----

CREATE TABLE Patient (
  PatientID INT PRIMARY KEY,
  Patient_FirstName VARCHAR(100),
  Patient_MiddleName VARCHAR(100),
  Patient_LastName VARCHAR(100),
  Patient_Phone_Num VARCHAR(15),
      Patient_Date_of_Birth DATE,
  Sex CHAR(1) CHECK (Sex IN ('M', 'F', 'O')),
      Height Float,
      Weight Float,
  Blood_Group VARCHAR(5) CHECK (Blood_Group IN ('A', 'B', 'AB', 'O', 'A+', 'A-', 'B+', 'B-', 'AB+', 'AB-',
'O+', 'O-')),
  Address VARCHAR(500),
  Country VARCHAR(100),
  Next_of_Kin_Name VARCHAR(150),
  Emergency_Phone_Number VARCHAR(15),
```

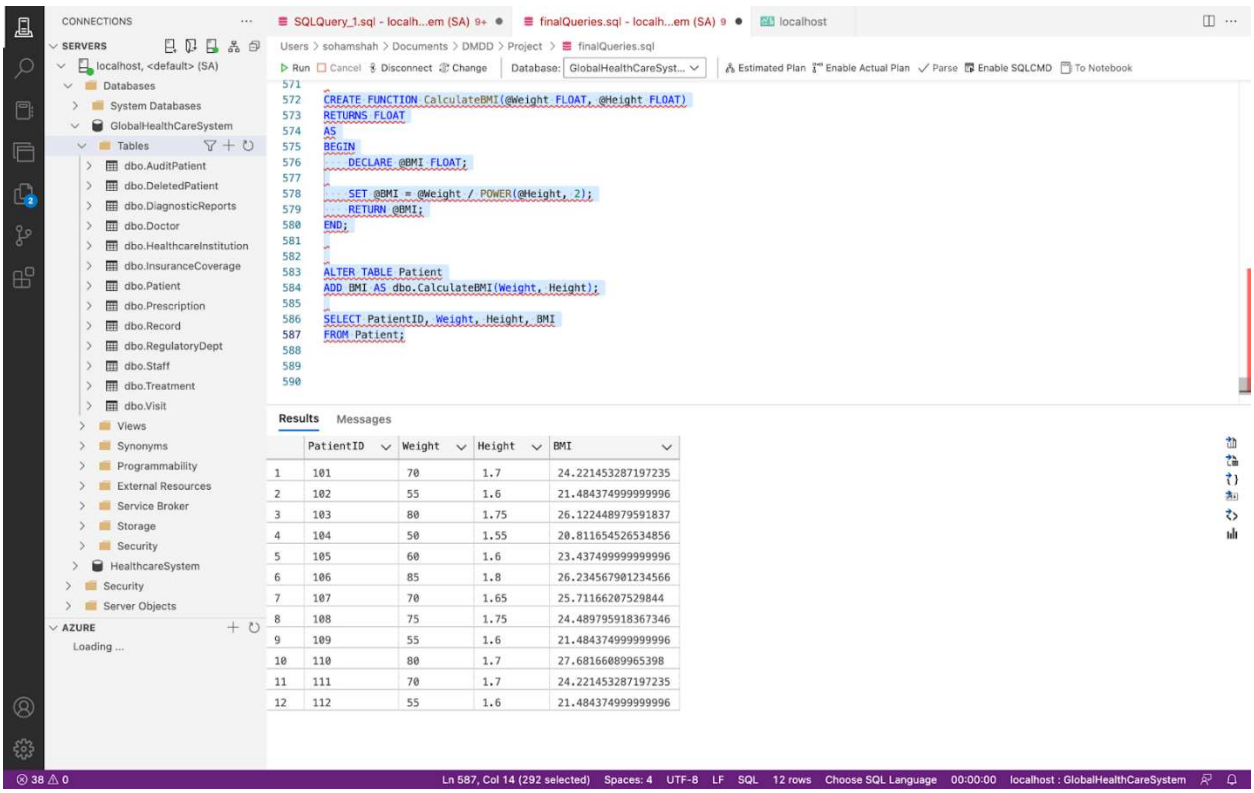
//Record Constraint for ‘ReportType’ which acts as a Discriminator for ‘Record’ table Subtypes ‘Diagnostic Reports’ and ‘Prescription’ (Refer ‘Final_Crud_P4’ pdf file and ‘SQL_CRUD’ file)



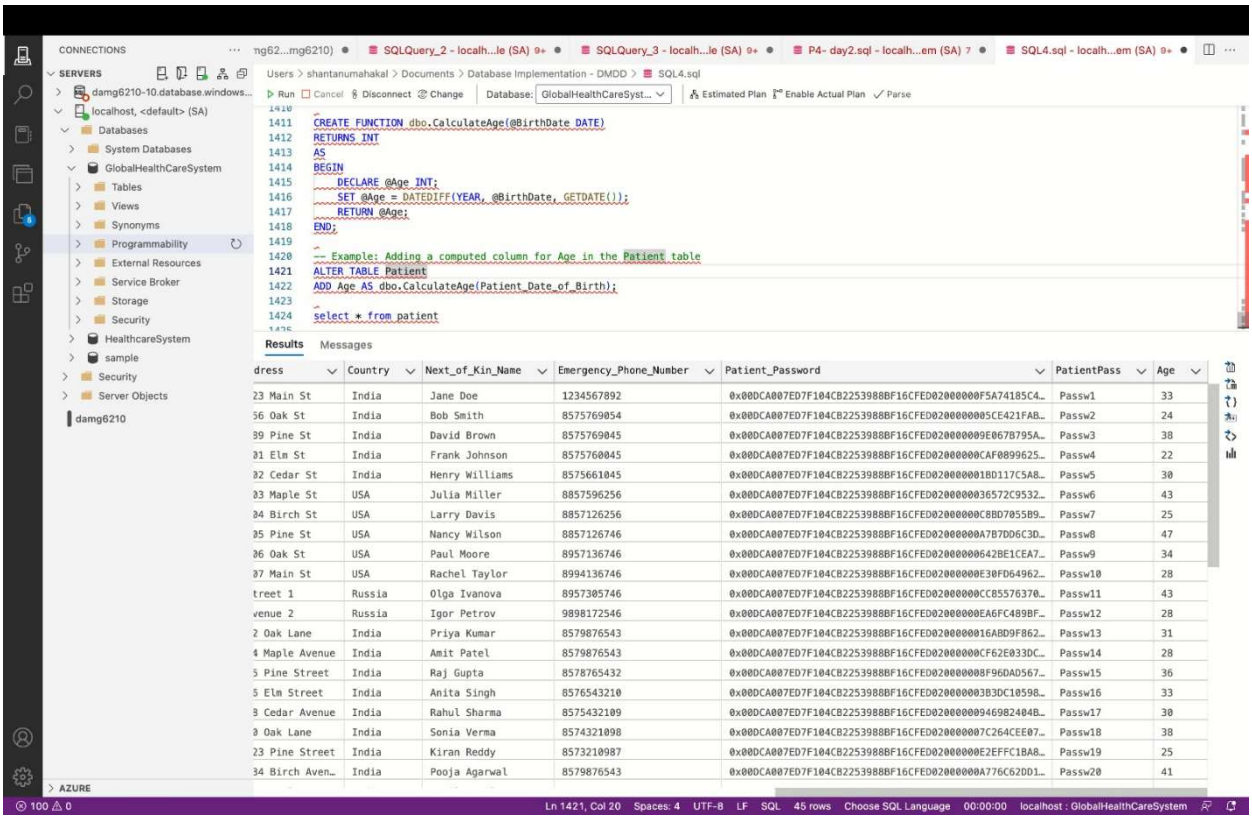
```
CREATE TABLE Record(
  UniqueRecordID INT PRIMARY KEY,
  PatientID INT,
  InstitutionID INT,
  Regulatory_Dept_ID INT,
  Location VARCHAR(200),
  Date DATE,
  ReportType VARCHAR(200) CHECK (ReportType IN('Diagnostic Reports','Prescription')),
  FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),
  FOREIGN KEY (InstitutionID) REFERENCES HealthcareInstitution (InstitutionID),
  FOREIGN KEY (Regulatory_Dept_ID) REFERENCES RegulatoryDept(Regulatory_Dept_ID),
);

---- Added new column ----
```

---- UDF (User Defined Function)

```
CREATE FUNCTION CalculateBMI(@Weight FLOAT, @Height FLOAT)
RETURNS FLOAT
AS
BEGIN
    DECLARE @BMI FLOAT;

    SET @BMI = @Weight / POWER(@Height, 2);
    RETURN @BMI;
END;

ALTER TABLE Patient
ADD BMI AS dbo.CalculateBMI(Weight, Height);

SELECT PatientID, Weight, Height, BMI
FROM Patient;
```



```
CREATE FUNCTION dbo.CalculateAge(@BirthDate DATE)
RETURNS INT
AS
BEGIN
    DECLARE @Age INT;
    SET @Age = DATEDIFF(YEAR, @BirthDate, GETDATE());
    RETURN @Age;
END;

ALTER TABLE Patient
ADD Age AS dbo.CalculateAge(Patient_Date_Of_Birth);

select * from Patient
```



---- Encryption -----

```
create MASTER KEY
ENCRYPTION BY PASSWORD = 'ShreeRam@765';

SELECT NAME KeyName,
symmetric_key_id KeyID,
key_length KeyLength,
algorithm_desc KeyAlgorithm
FROM sys.symmetric_keys;
go

CREATE CERTIFICATE PatientPass
WITH SUBJECT = 'Patient Password';
GO

CREATE SYMMETRIC KEY PatientPass_SM
WITH ALGORITHM = AES_256
ENCRYPTION BY CERTIFICATE PatientPass;
GO

OPEN SYMMETRIC KEY PatientPass_SM
DECRYPTION BY CERTIFICATE PatientPass;

INSERT INTO Patient (PatientID, Patient_FirstName, Patient_MiddleName, Patient_LastName,
Patient_Phone_Num,
   Patient_Date_of_Birth, Sex, Height, Weight, Blood_Group, Address, Country,
```

Next_of_Kin_Name, Emergency_Phone_Number, Patient_Password)

VALUES

(146, 'Shantanu', 'R', 'Mahakal', 8575769054, '1998-10-30', 'M', 1.68, 60, 'B+', '15 Pond St', 'USA',

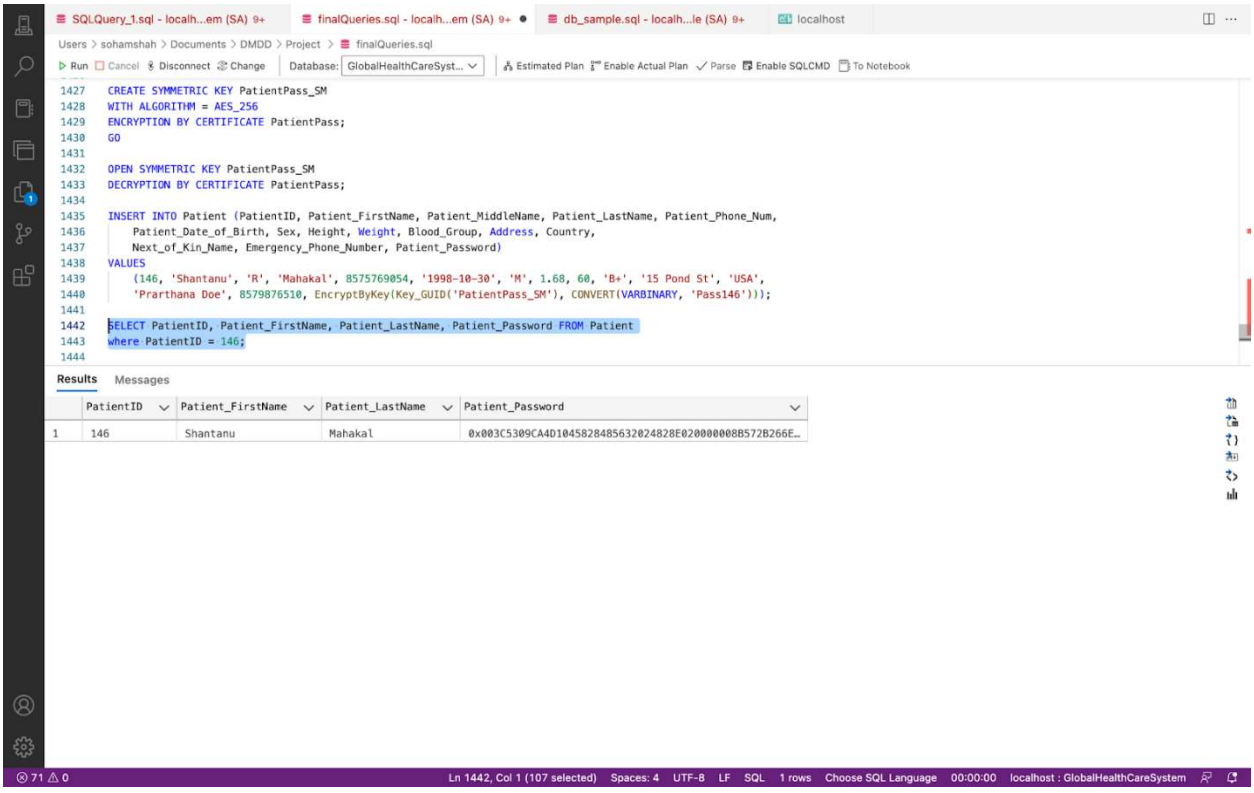
'Prarthana Doe', 8579876510, EncryptByKey(Key_GUID('PatientPass_SM'), CONVERT(VARBINARY, 'Pass146')));

SELECT PatientID, Patient_FirstName, Patient_LastName, Patient_Password FROM Patient

where PatientID = 146;

UPDATE Patient set patientpass = 'Pass146' where PatientID= 146

SELECT * FROM Patient



--- to decrypt ---

OPEN SYMMETRIC KEY PatientPass_SM

DECRYPTION BY CERTIFICATE PatientPass;

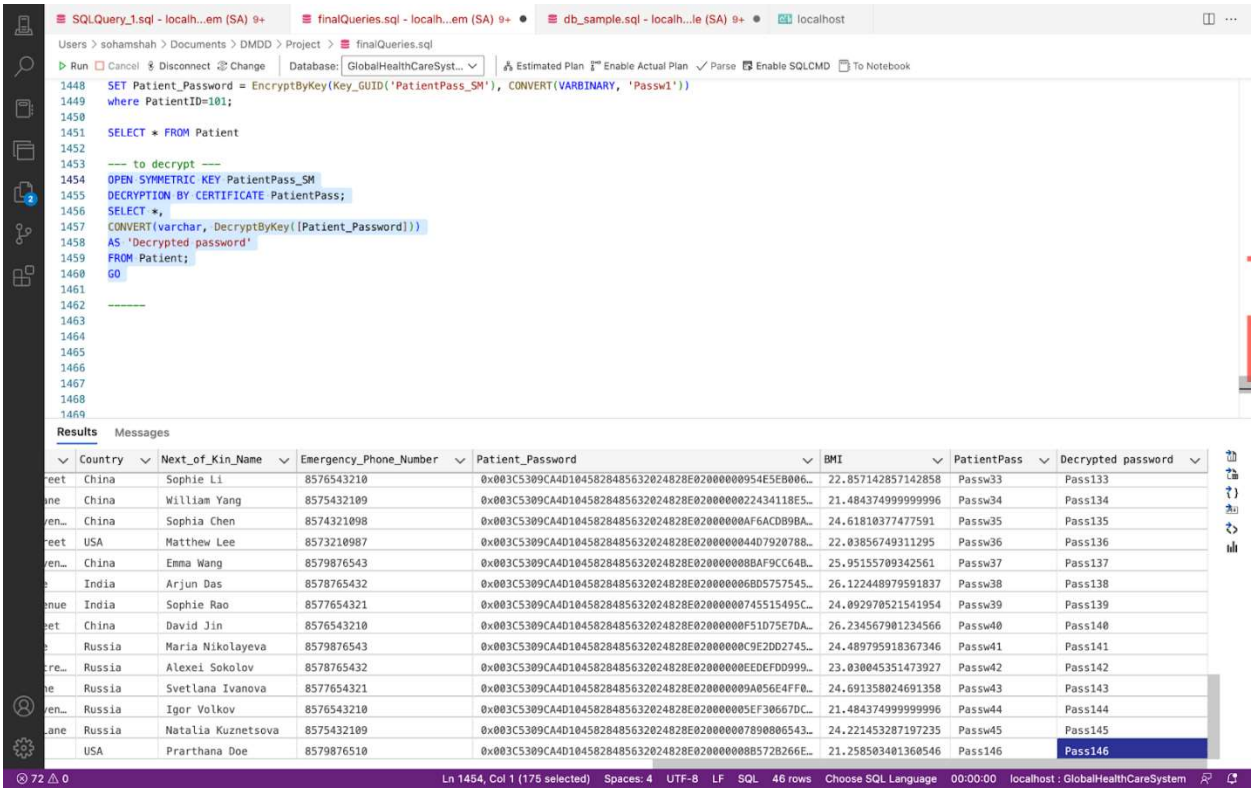
SELECT *,

CONVERT(varchar, DecryptByKey([Patient_Password]))

AS 'Decrypted password'

FROM Patient;

GO



----- NON CLUSTERED INDEX 1 -----

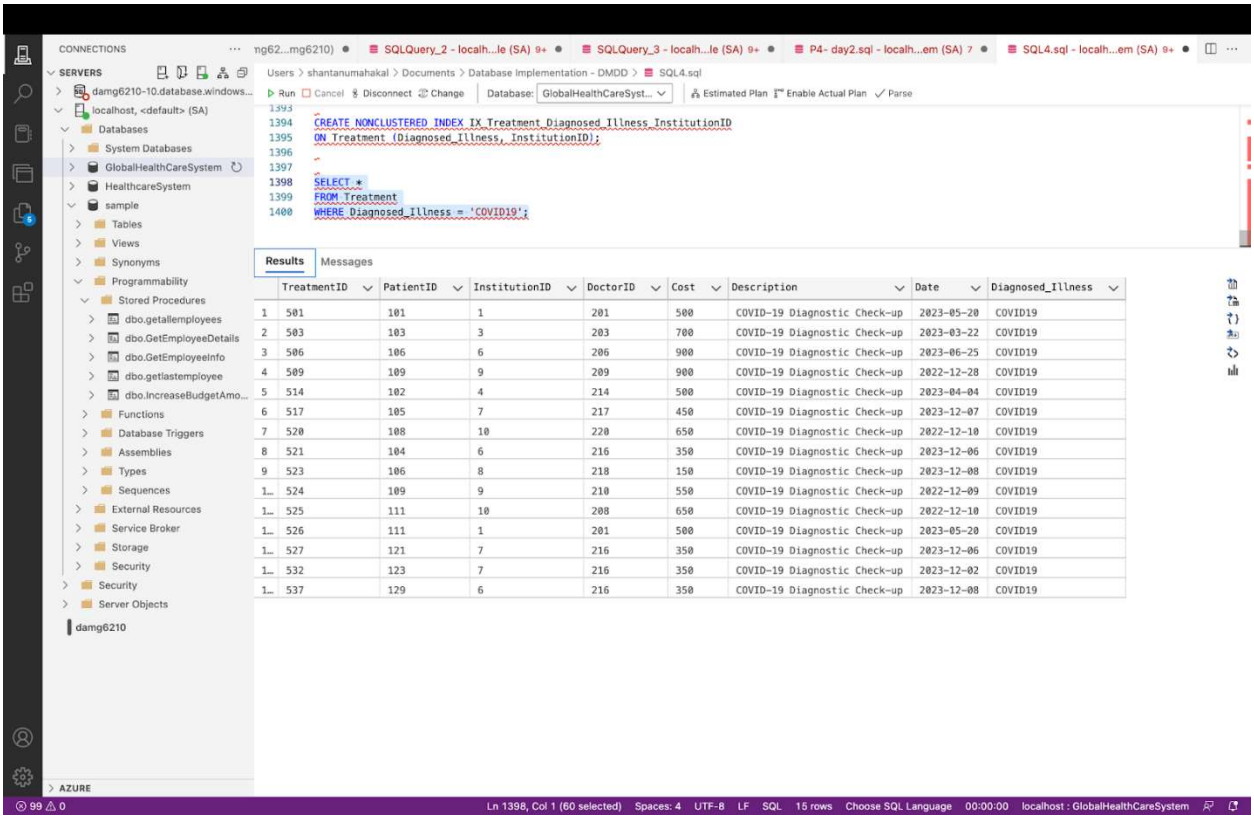
CREATE NONCLUSTERED INDEX IX_Treatment_Diagnosed_Illness_InstitutionID

ON Treatment (Diagnosed_Illness, InstitutionID);

SELECT *

FROM Treatment

WHERE Diagnosed_Illness = 'COVID19';



---- NON CLUSTERED INDEX 2 ----

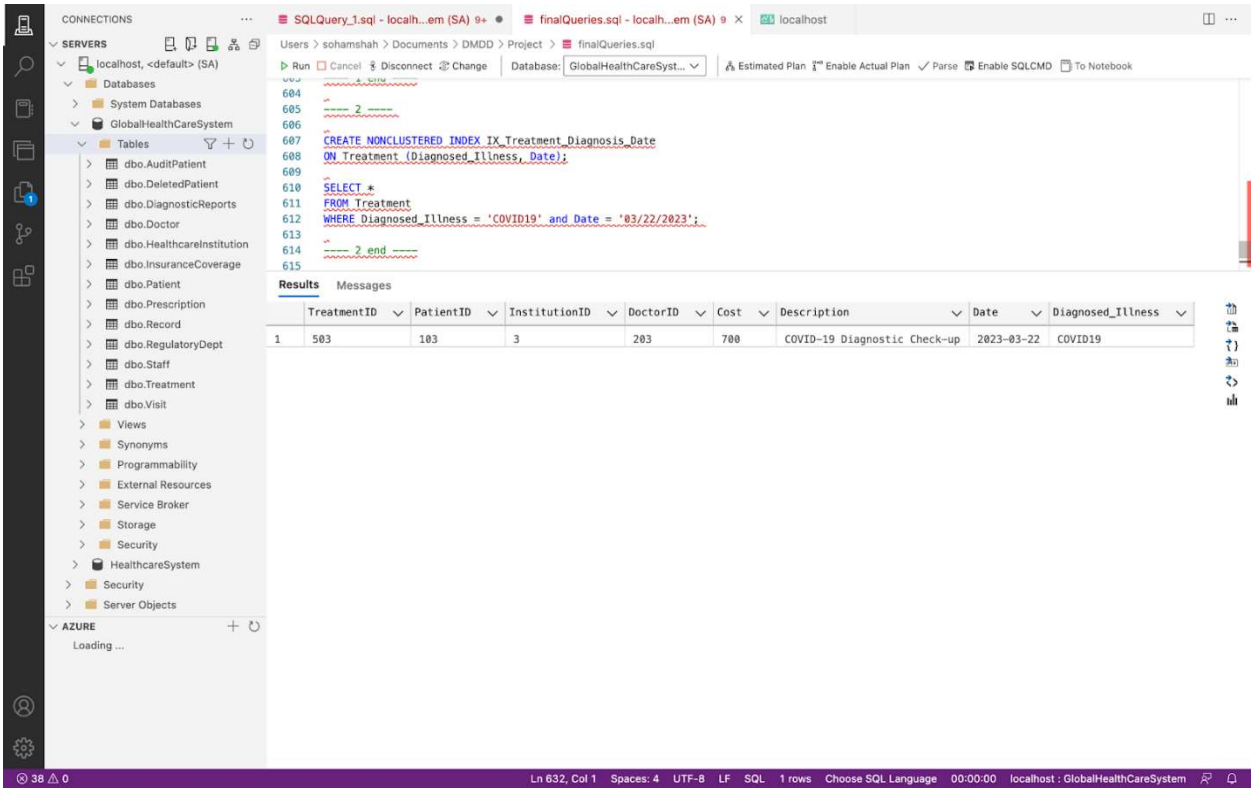
CREATE NONCLUSTERED INDEX IX_Treatment_Diagnosis_Date

ON Treatment (Diagnosed_Illness, Date);

SELECT *

FROM Treatment

WHERE Diagnosed_Illness = 'COVID19' and Date = '03/22/2023';



---- NON CLUSTERED INDEX 3 -----

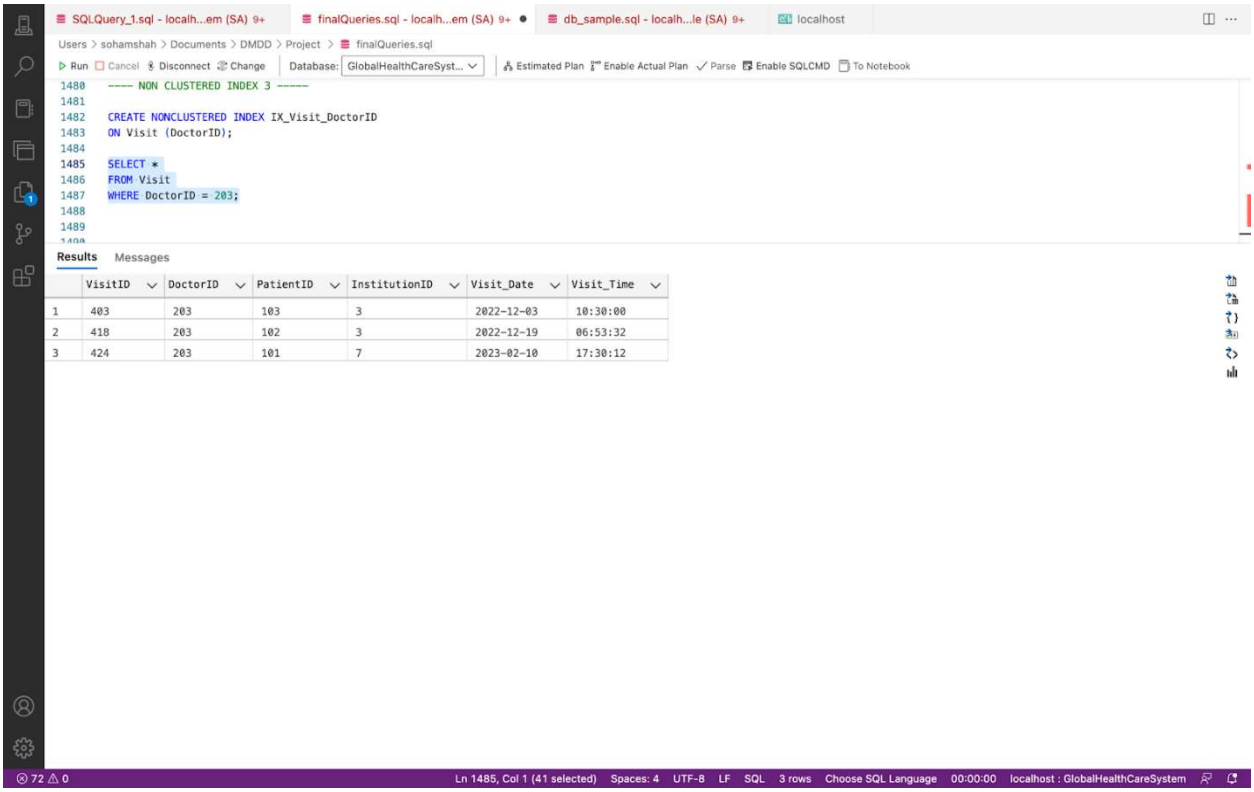
CREATE NONCLUSTERED INDEX IX_Visit_DoctorID

ON Visit (DoctorID);

SELECT *

FROM Visit

WHERE DoctorID = 203;



---- TO VIEW NUMBER OF NON CLUSTERED INDEX ----

```
SELECT
    t.name AS TableName,
    i.name AS IndexName,
    i.type_desc AS IndexType,
    col.name AS ColumnName
FROM
    sys.indexes AS i
INNER JOIN
    sys.index_columns AS ic ON i.object_id = ic.object_id AND i.index_id = ic.index_id
INNER JOIN
    sys.columns AS col ON ic.object_id = col.object_id AND ic.column_id = col.column_id
```

```
INNER JOIN
    sys.tables AS t ON i.object_id = t.object_id
WHERE
    i.type_desc = 'NONCLUSTERED';
```