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);

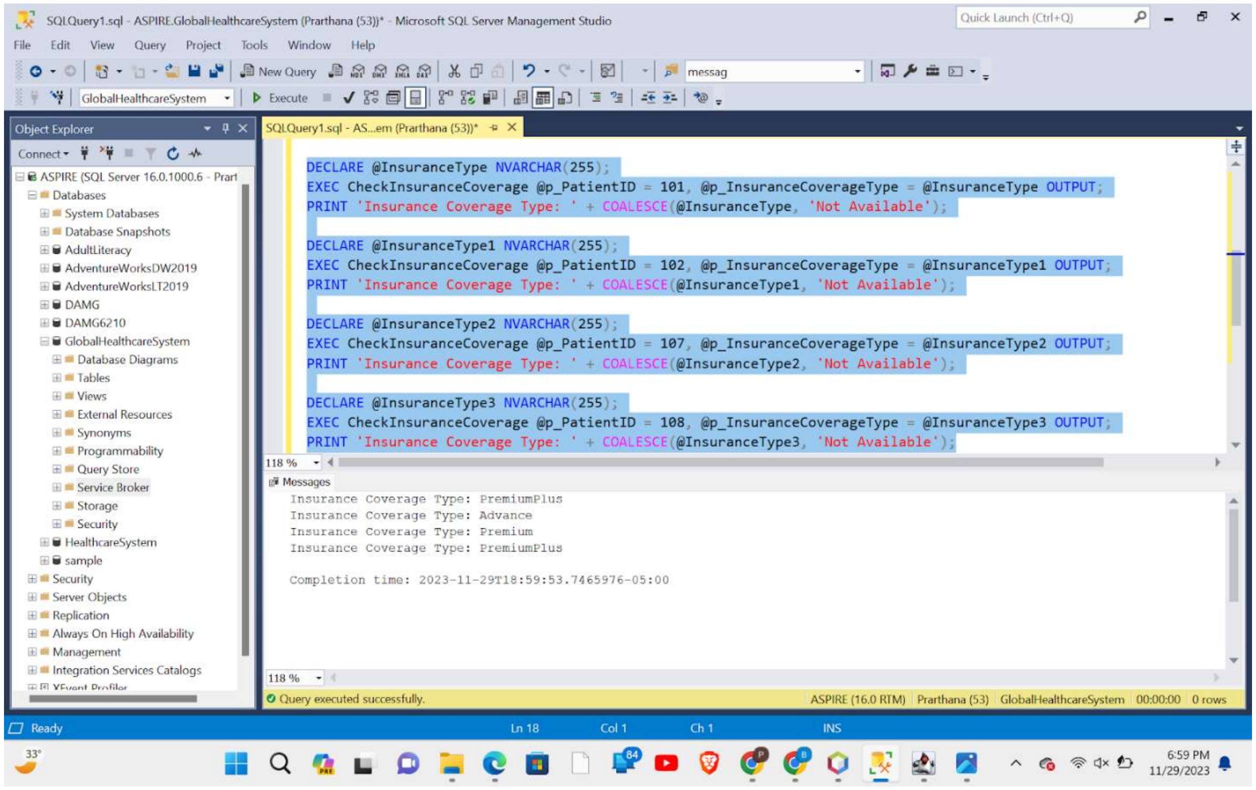
EXEC CheckInsuranceCoverage @p_PatientID = 107, @p_InsuranceCoverageType = @InsuranceType2 OUTPUT;

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');


SQLQuery1.sql - ASPIRE.GlobalHealthcareSystem (Prarthana (53))* - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
Object Explorer
SQLQuery1.sql - AS...em (Prarthana (53))*
Messages
Query executed successfully.
ASPIRE (16.0 RTM) | Prarthana (53) | GlobalHealthcareSystem | 00:00:00 | 0 rows
Ready Ln 18 Col 1 Ch 1 INS

```
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');
```

```
Insurance Coverage Type: PremiumPlus
Insurance Coverage Type: Advance
Insurance Coverage Type: Premium
Insurance Coverage Type: PremiumPlus

Completion time: 2023-11-29T18:59:53.7465976-05:00
```

---- Stored Procedure 2 ----

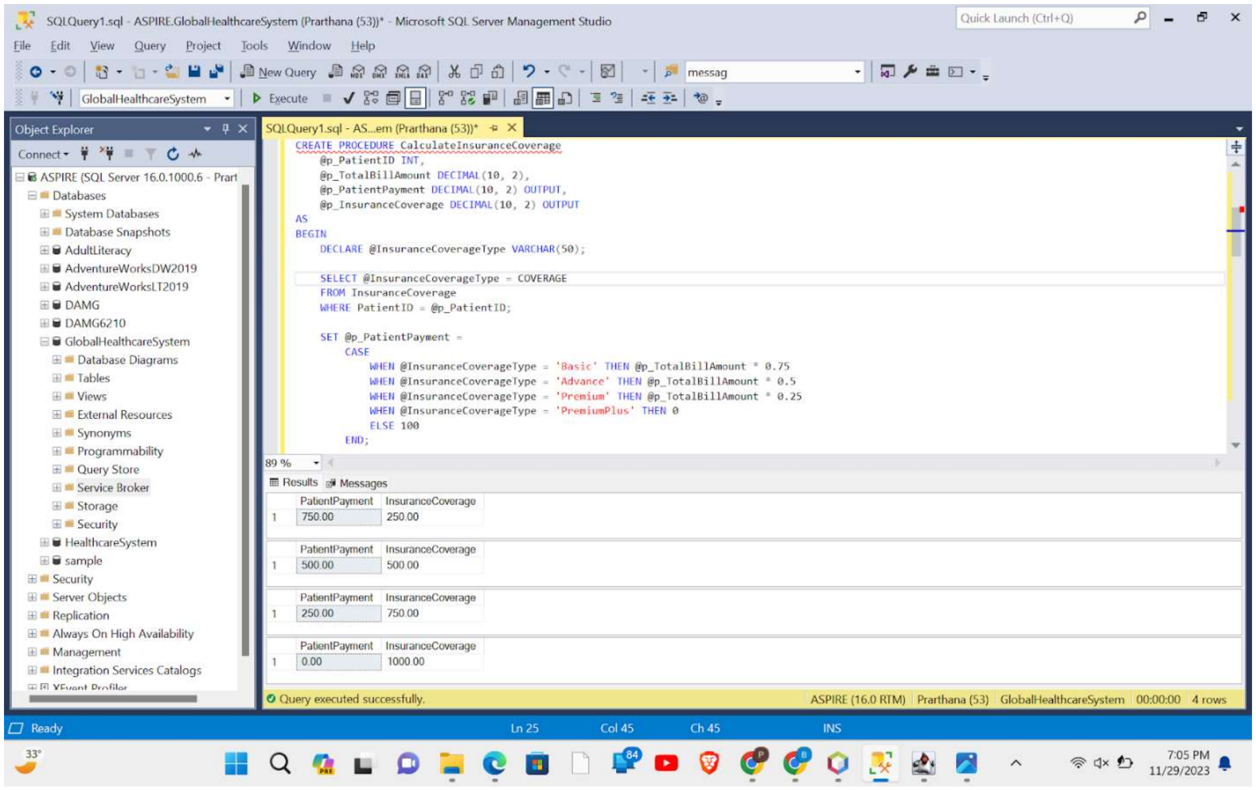
CREATE PROCEDURE CalculateInsuranceCoverage

@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
SQLQuery1.sql - AS...em (Prarthana (53))*
Object Explorer
ASPIRE (SQL Server 16.0.1000.6 - Prart
Databases
GlobalHealthcareSystem
Results
Messages

```
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;
```

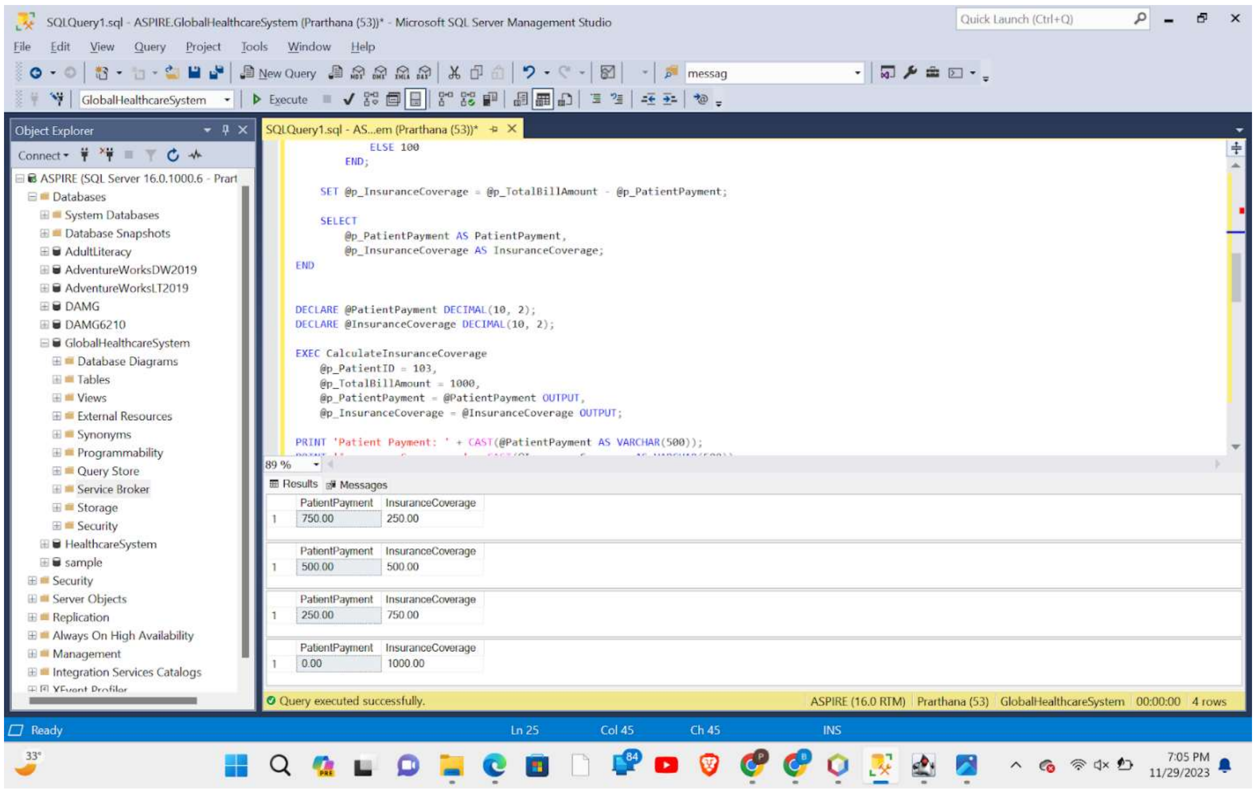
	PatientPayment	InsuranceCoverage
1	750.00	250.00

	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



SQLQuery1.sql - ASPIRE.GlobalHealthcareSystem (Prarthana (53))* - Microsoft SQL Server Management Studio

```sql
            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));
```

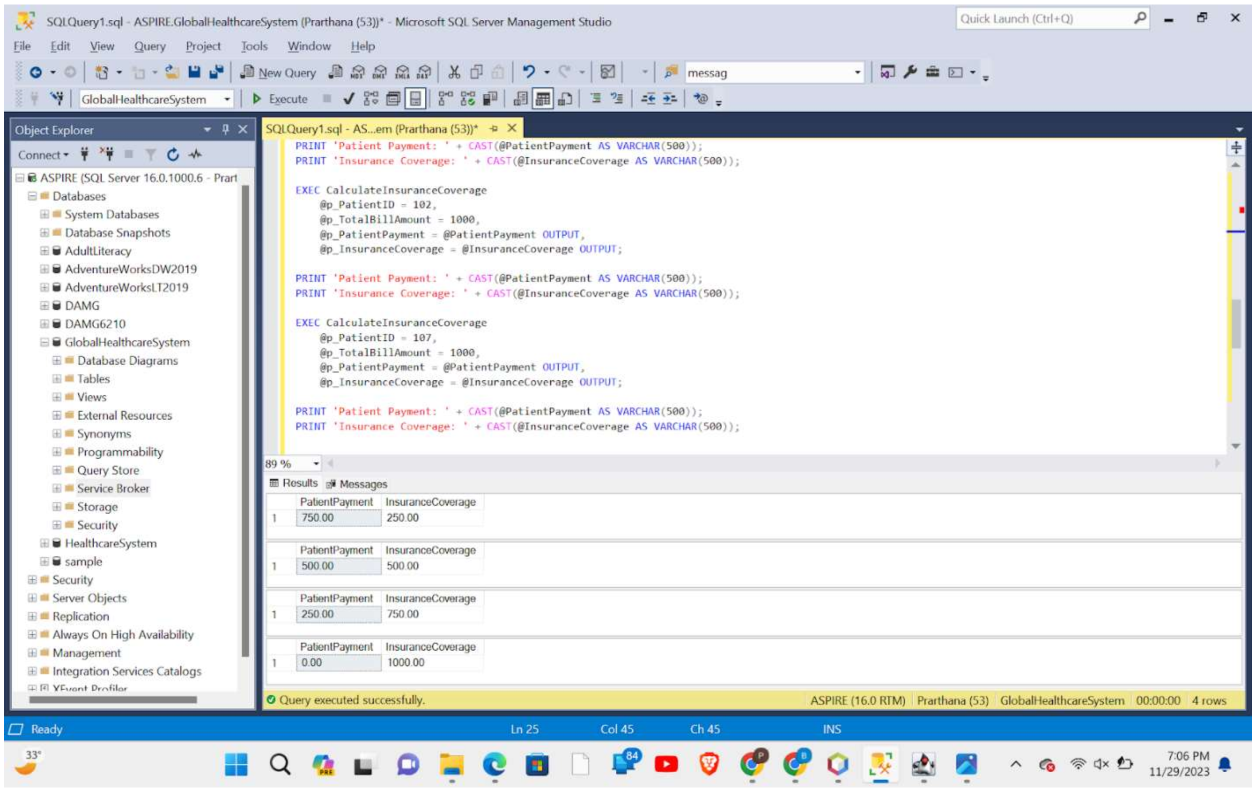
	PatientPayment	InsuranceCoverage
1	750.00	250.00

	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



SQLQuery1.sql - ASPIRE.GlobalHealthcareSystem (Prarthana (53))* - Microsoft SQL Server Management Studio

```sql
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));
```

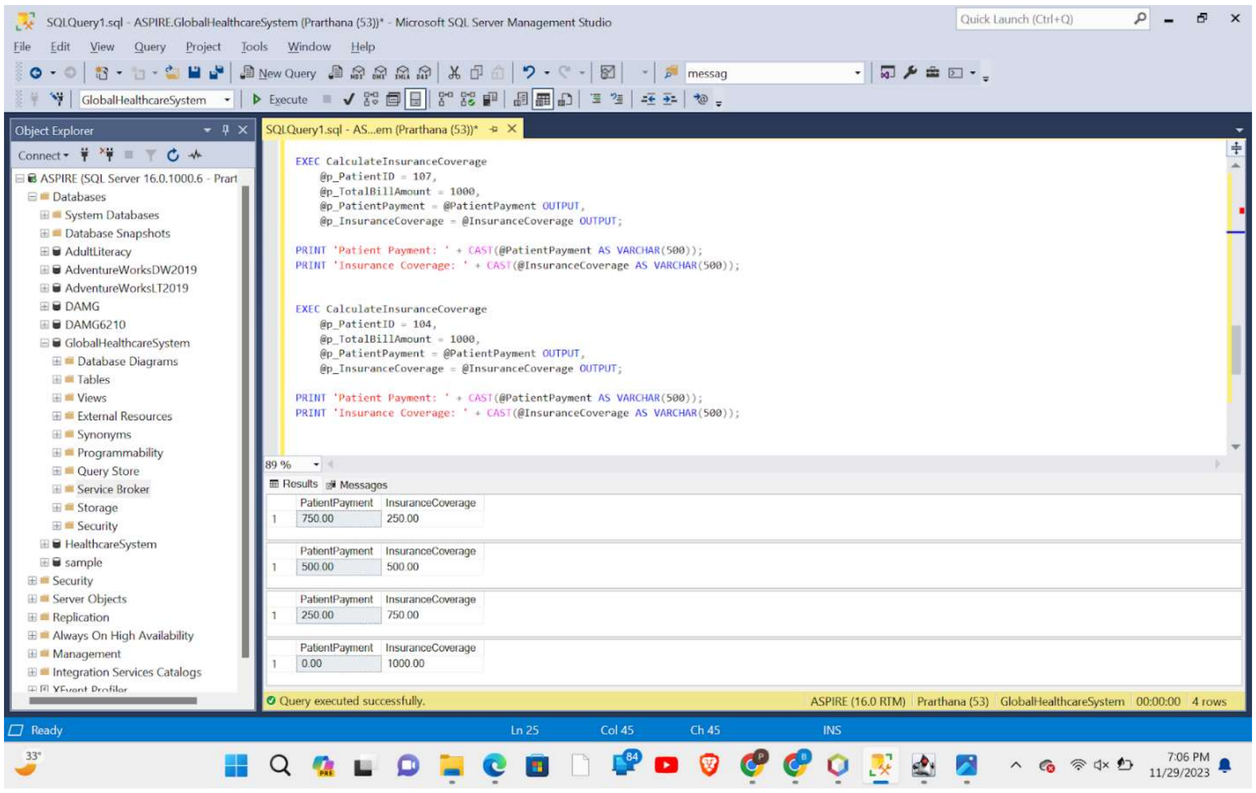
	PatientPayment	InsuranceCoverage
1	750.00	250.00

	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



```
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));
```

	PatientPayment	InsuranceCoverage
1	750.00	250.00

	PatientPayment	InsuranceCoverage
1	500.00	500.00

	PatientPayment	InsuranceCoverage
1	250.00	750.00

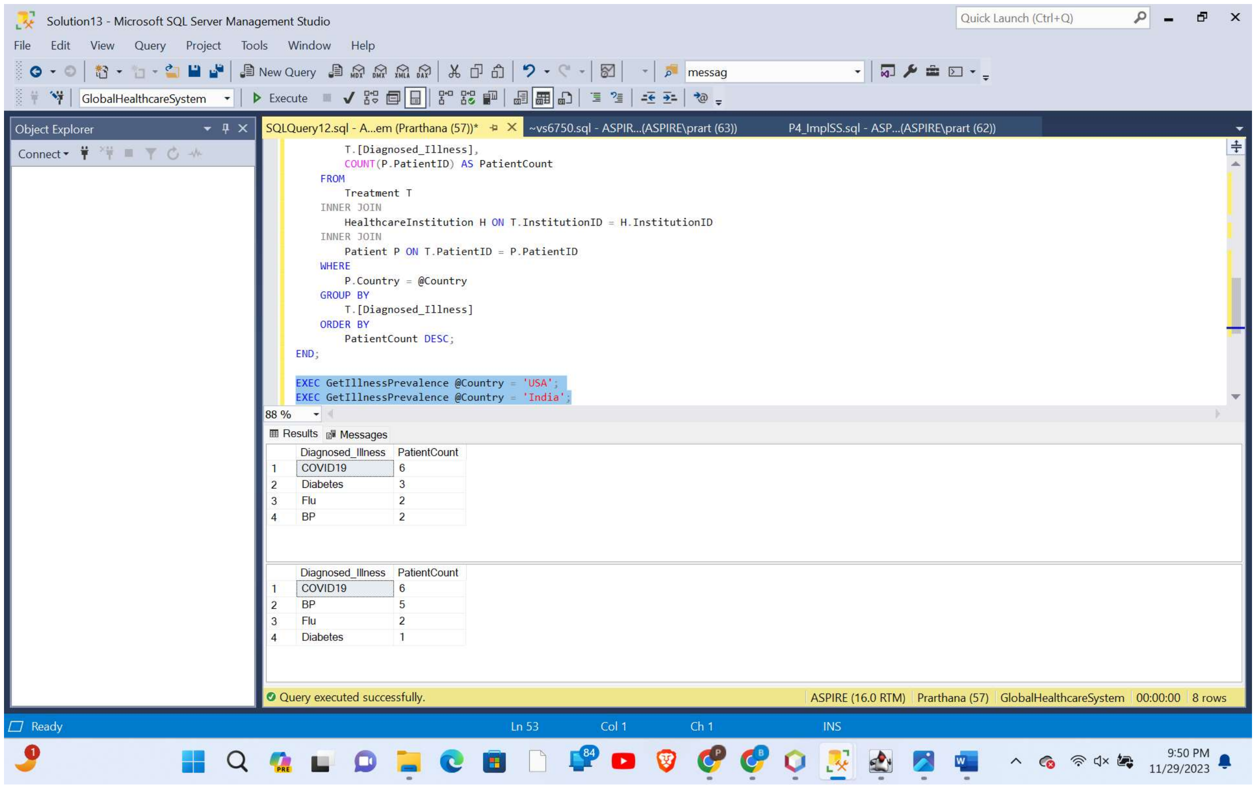
	PatientPayment	InsuranceCoverage
1	0.00	1000.00

Query executed successfully.

---- 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';
```



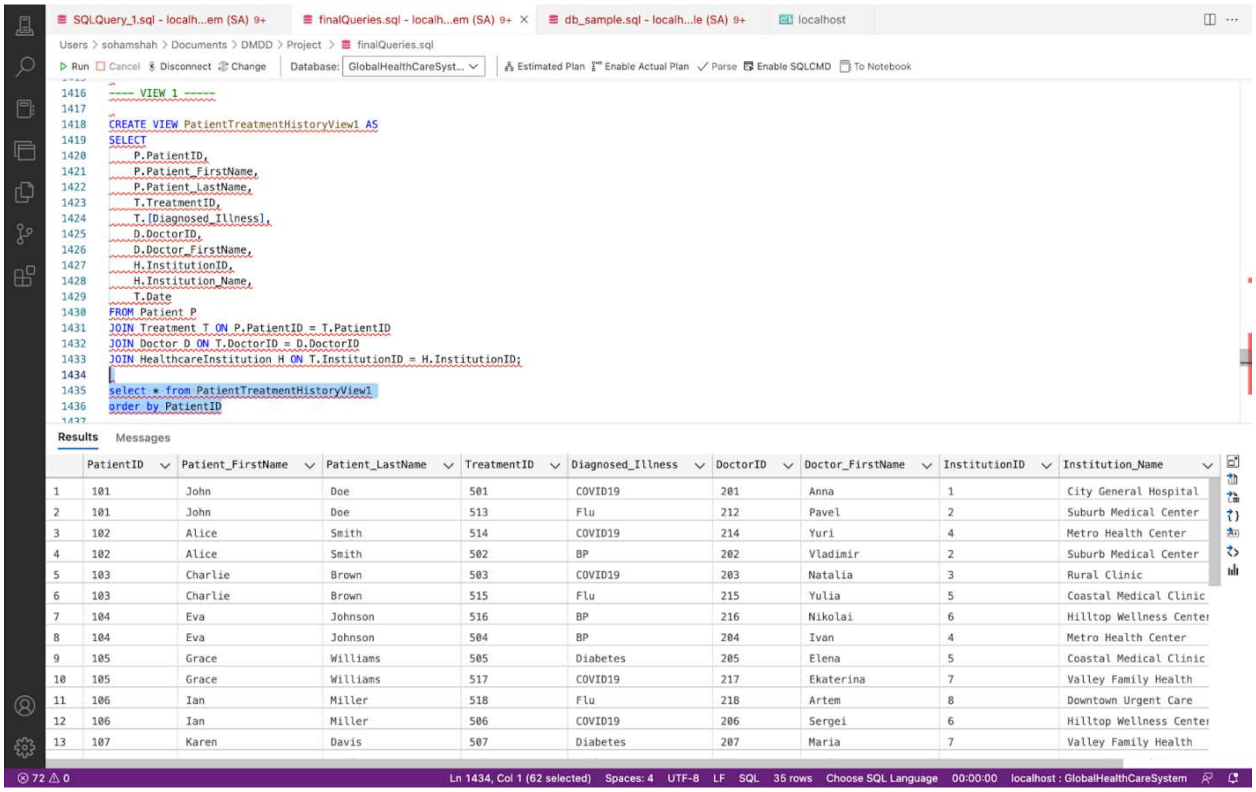
```
        T.[Diagnosed_Illness],
        COUNT(P.PatientID) AS PatientCount
    FROM
        Treatment T
    INNER JOIN
        HealthcareInstitution H ON T.InstitutionID = H.InstitutionID
    INNER JOIN
        Patient P ON T.PatientID = P.PatientID
    WHERE
        P.Country = @Country
    GROUP BY
        T.[Diagnosed_Illness]
    ORDER BY
        PatientCount DESC;
END;

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

	Diagnosed_Illness	PatientCount
1	COVID19	6
2	Diabetes	3
3	Flu	2
4	BP	2

	Diagnosed_Illness	PatientCount
1	COVID19	6
2	BP	5
3	Flu	2
4	Diabetes	1

Query executed successfully. ASPIRE (16.0 RTM) | Prarthana (57) | GlobalHealthcareSystem | 00:00:00 | 8 rows

```
---- 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
```



```sql
---- 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
```

	PatientID	Patient_FirstName	Patient_LastName	TreatmentID	Diagnosed_Illness	DoctorID	Doctor_FirstName	InstitutionID	Institution_Name
1	101	John	Doe	501	COVID19	201	Anna	1	City General Hospital
2	101	John	Doe	513	Flu	212	Pavel	2	Suburb Medical Center
3	102	Alice	Smith	514	COVID19	214	Yuri	4	Metro Health Center
4	102	Alice	Smith	502	BP	202	Vladimir	2	Suburb Medical Center
5	103	Charlie	Brown	503	COVID19	203	Natalia	3	Rural Clinic
6	103	Charlie	Brown	515	Flu	215	Yulia	5	Coastal Medical Clinic
7	104	Eva	Johnson	516	BP	216	Nikolai	6	Hilltop Wellness Center
8	104	Eva	Johnson	504	BP	204	Ivan	4	Metro Health Center
9	105	Grace	Williams	505	Diabetes	205	Elena	5	Coastal Medical Clinic
10	105	Grace	Williams	517	COVID19	217	Ekaterina	7	Valley Family Health
11	106	Ian	Miller	518	Flu	218	Artem	8	Downtown Urgent Care
12	106	Ian	Miller	506	COVID19	206	Sergei	6	Hilltop Wellness Center
13	107	Karen	Davis	507	Diabetes	207	Maria	7	Valley Family Health

SQLQuery_1.sql - localh...em (SA) 9+
finalQueries.sql - localh...em (SA) 9+
db_sample.sql - localh...le (SA) 9+
localhost
Users > sohamshah > Documents > DMDD > Project > finalQueries.sql
Run
Cancel
Disconnect
Change
Database: GlobalHealthCareSyst...
Estimated Plan
Enable Actual Plan
Parse
Enable SQLCMD
To Notebook
Results
Messages
Ln 1434, Col 1 (62 selected)
Spaces: 4
UTF-8
LF
SQL
35 rows
Choose SQL Language
00:00:00
localhost : GlobalHealthCareSystem

---- 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

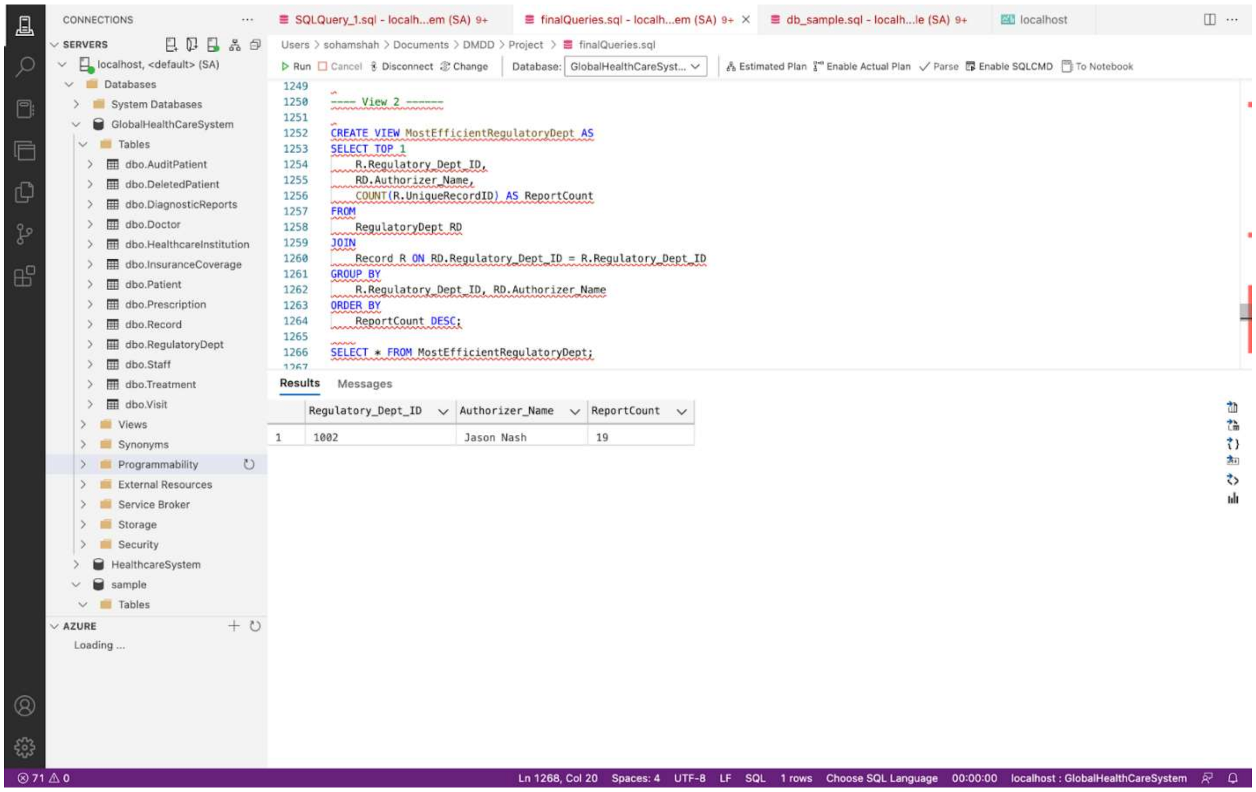
GROUP BY

    R.Regulatory_Dept_ID, RD.Authorizer_Name

ORDER BY

    ReportCount DESC;

SELECT * FROM MostEfficientRegulatoryDept;



```
---- View 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;
```

Results

	Regulatory_Dept_ID	Authorizer_Name	ReportCount
1	1002	Jason Nash	19

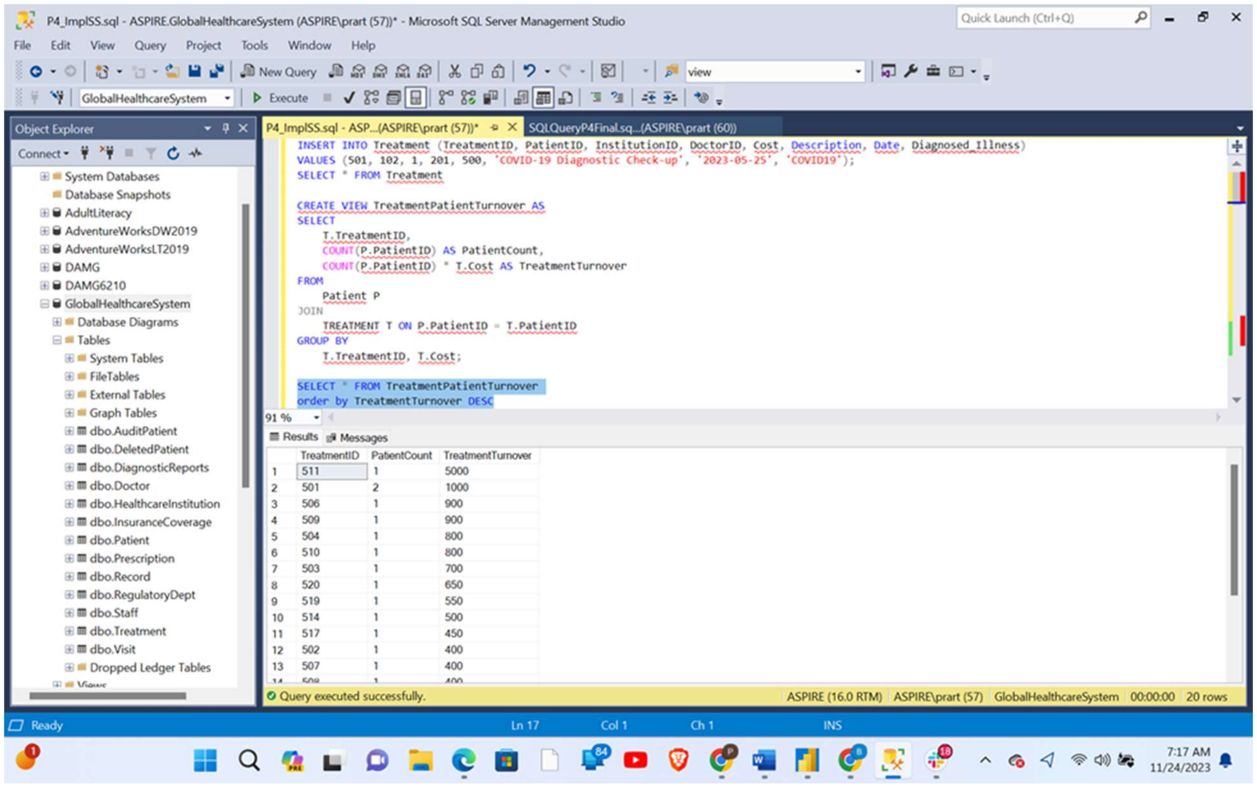
---- VIew 3 -----

```
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
```



```
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

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;

SELECT * FROM TreatmentPatientTurnover
order by TreatmentTurnover DESC
```

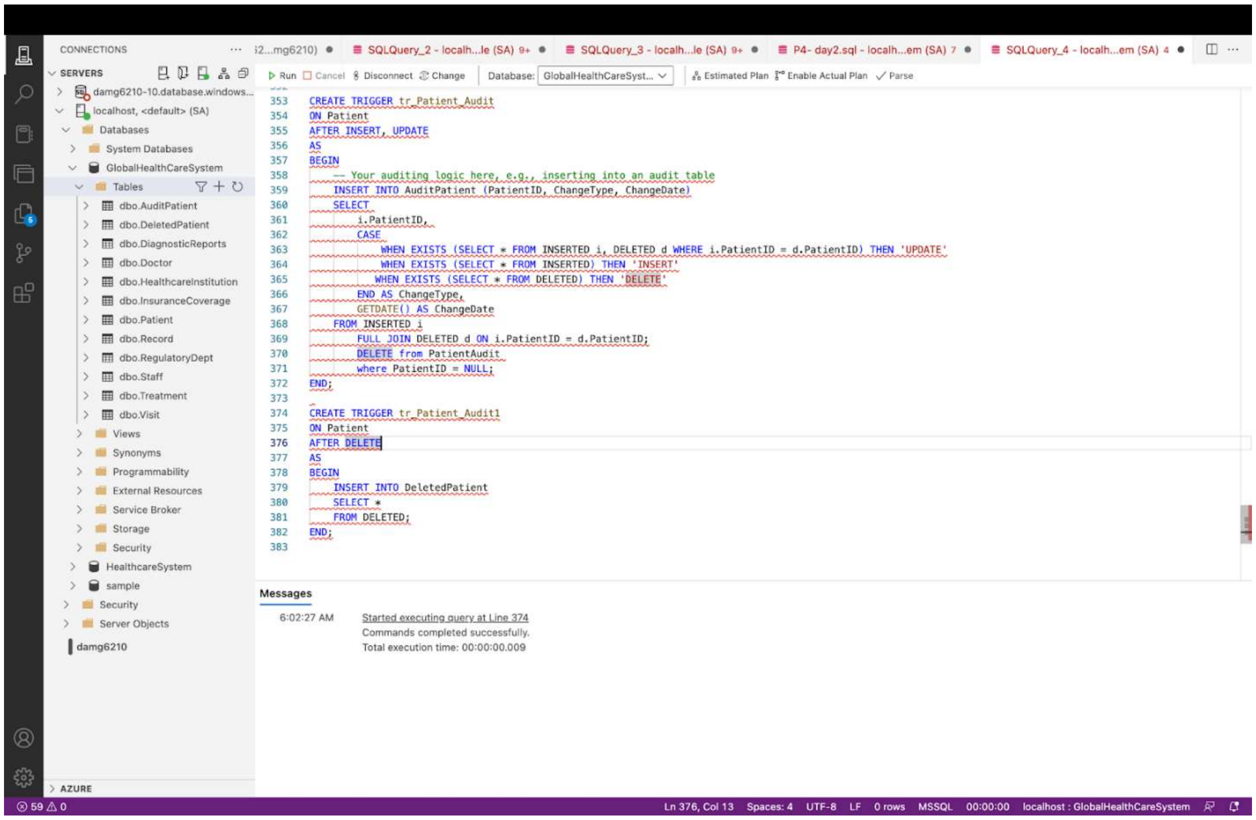
	TreatmentID	PatientCount	TreatmentTurnover
1	511	1	5000
2	501	2	1000
3	506	1	900
4	509	1	900
5	504	1	800
6	510	1	800
7	503	1	700
8	520	1	650
9	519	1	550
10	514	1	500
11	517	1	450
12	502	1	400
13	507	1	400
14	508	1	400

```
Query executed successfully.   ASPIRE (16.0 RTM)   ASPIRE\prart (57)   GlobalHealthcareSystem   00:00:00   20 rows
```

---- 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;
```



```
CREATE TRIGGER tr_Patient_Audit
ON Patient
AFTER INSERT, UPDATE
AS
BEGIN
    -- Your auditing logic here, e.g., inserting into an audit table
    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 PatientAudit
        where PatientID = NULL;
END;

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

Messages

```
6:02:27 AM    Started executing query at Line 374
              Commands completed successfully.
              Total execution time: 00:00:00.009
```

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

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

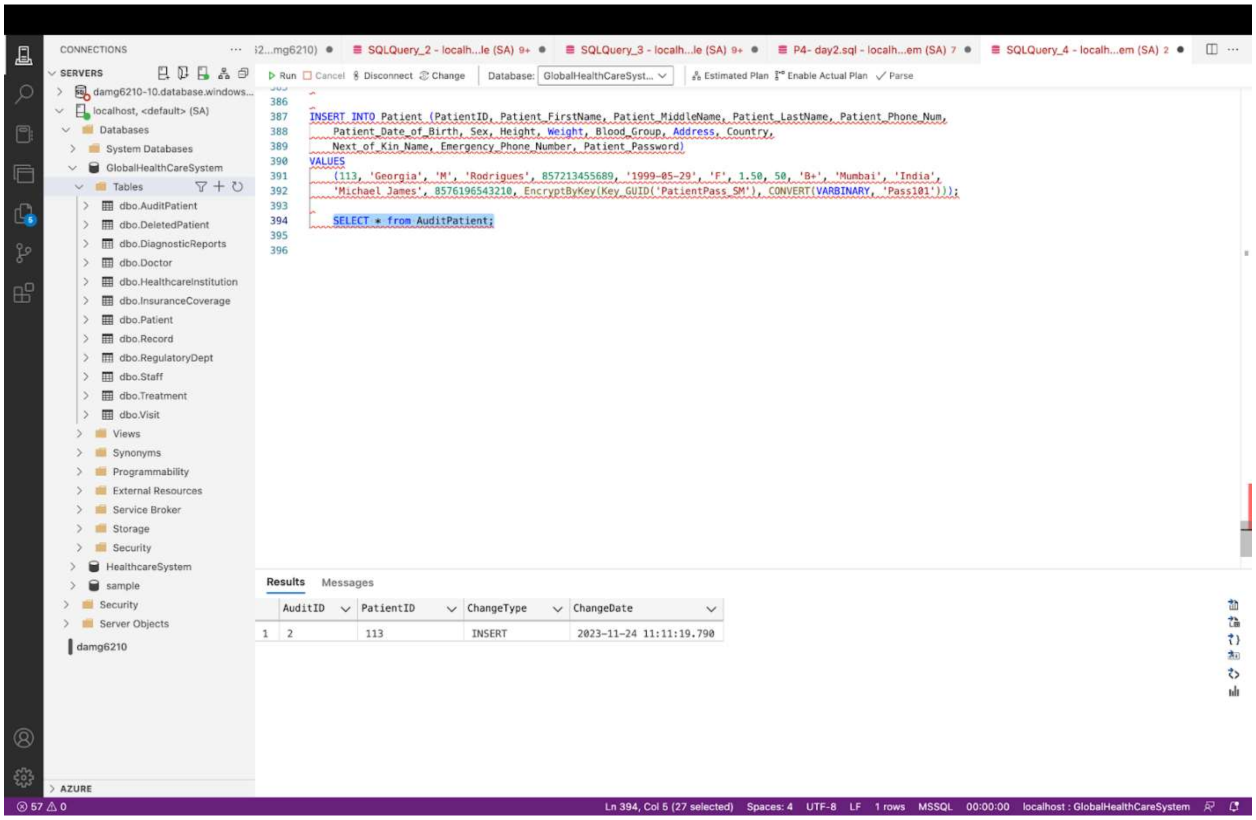
Next_of_Kin_Name, Emergency_Phone_Number, Patient_Password)

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;



```sql
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;
```

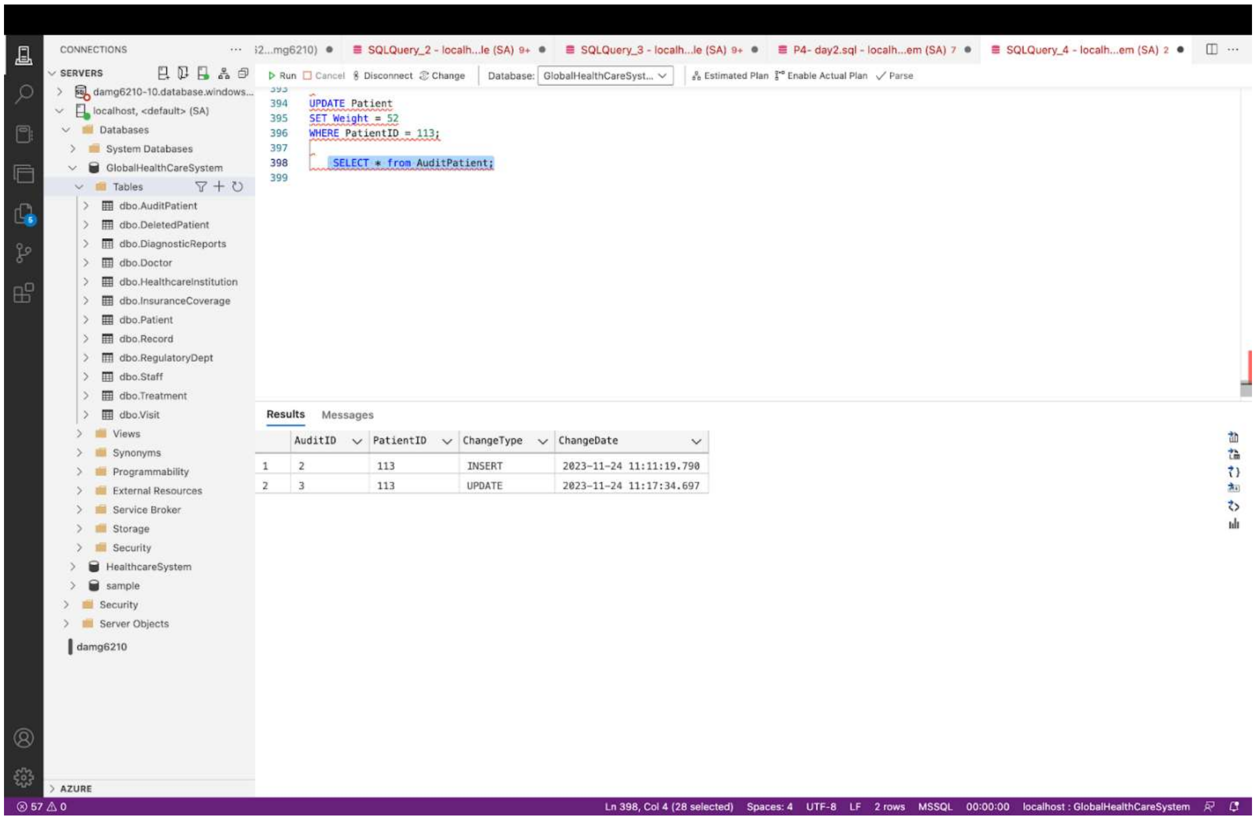
	AuditID	PatientID	ChangeType	ChangeDate
1	2	113	INSERT	2023-11-24 11:11:19.790

UPDATE Patient

SET Weight = 52

WHERE PatientID = 113;

SELECT * from AuditPatient;



```
UPDATE Patient
SET Weight = 52
WHERE PatientID = 113;

    SELECT * from AuditPatient;
```

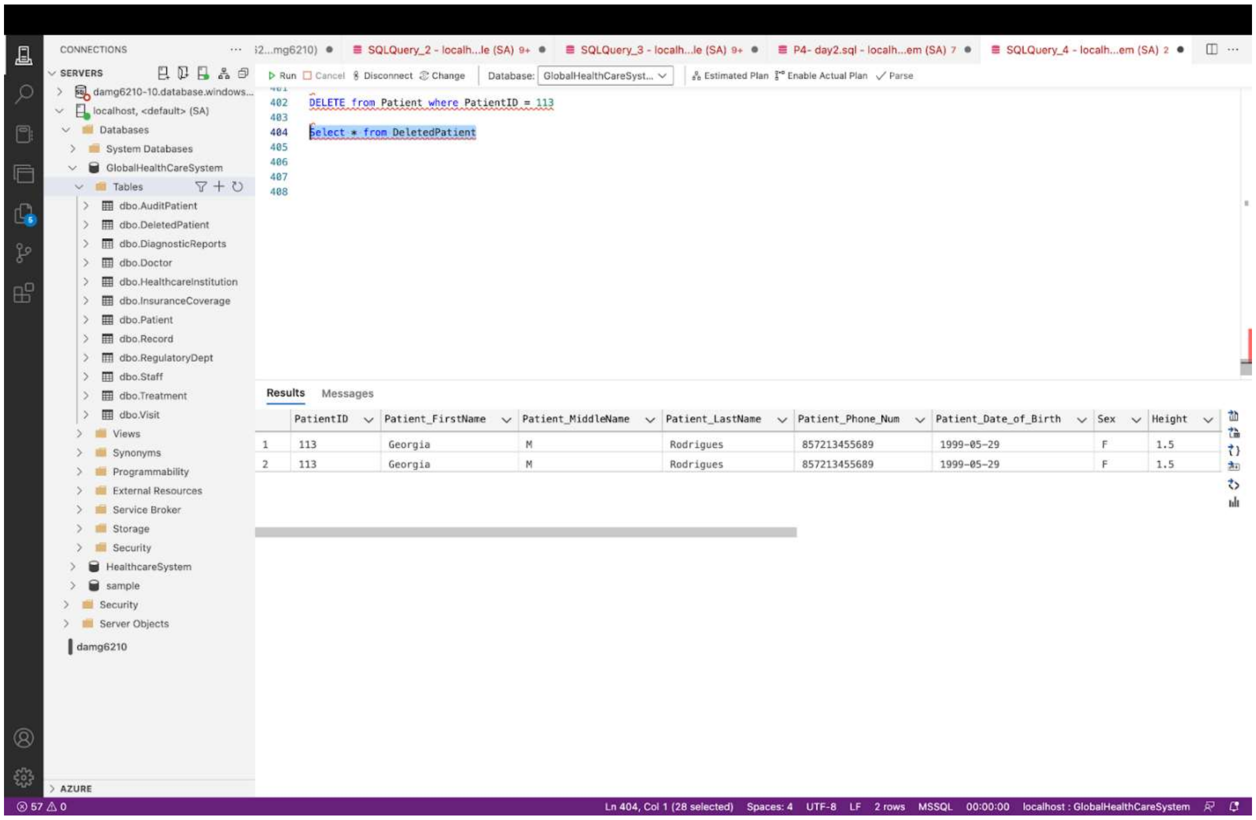
```
   AuditID  PatientID  ChangeType  ChangeDate
1  2        113        INSERT      2023-11-24 11:11:19.790
2  3        113        UPDATE      2023-11-24 11:17:34.697
```

---- 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



```
DELETE from Patient where PatientID = 113

Select * from DeletedPatient
```

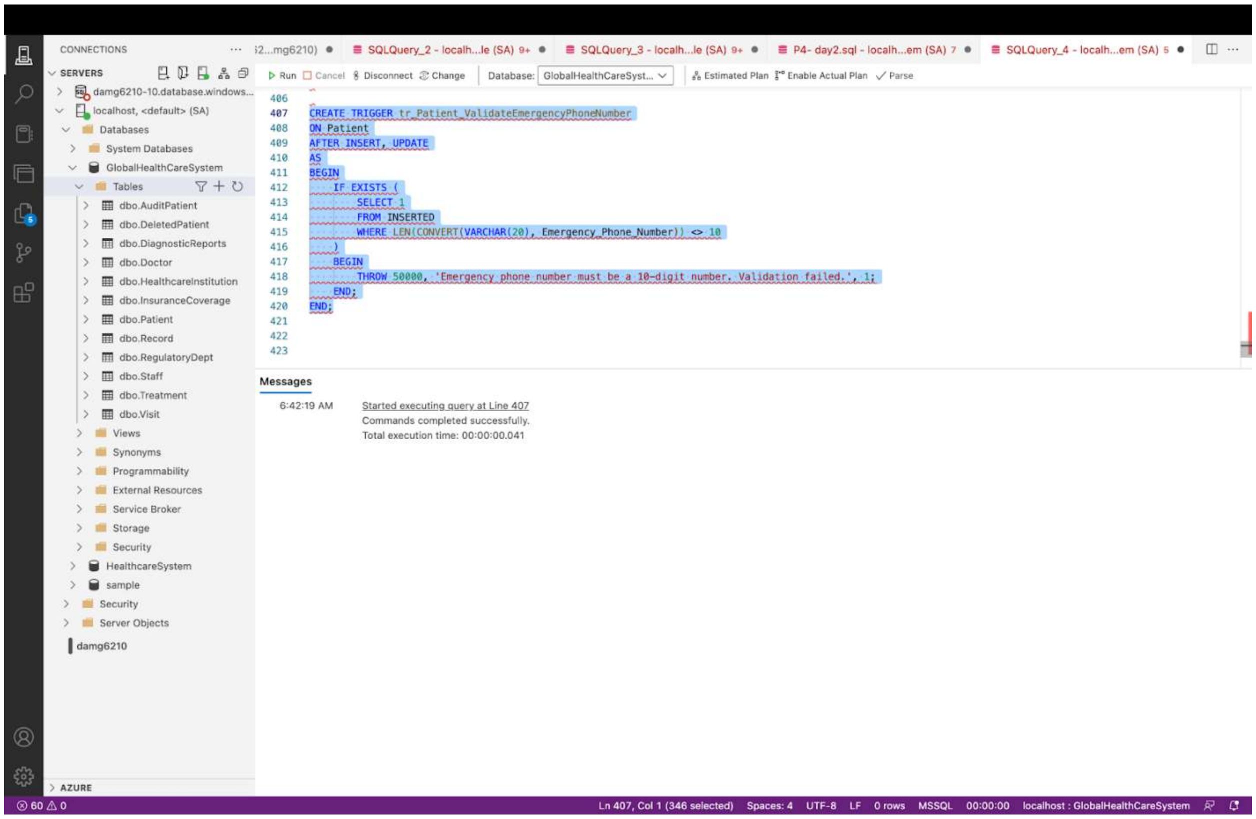
Results Messages

	PatientID	Patient_FirstName	Patient_MiddleName	Patient_LastName	Patient_Phone_Num	Patient_Date_of_Birth	Sex	Height
1	113	Georgia	M	Rodrigues	857213455689	1999-05-29	F	1.5
2	113	Georgia	M	Rodrigues	857213455689	1999-05-29	F	1.5

Ln 404, Col 1 (28 selected) Spaces: 4 UTF-8 LF 2 rows MSSQL 00:00:00 localhost : GlobalHealthCareSystem

---- 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;
```

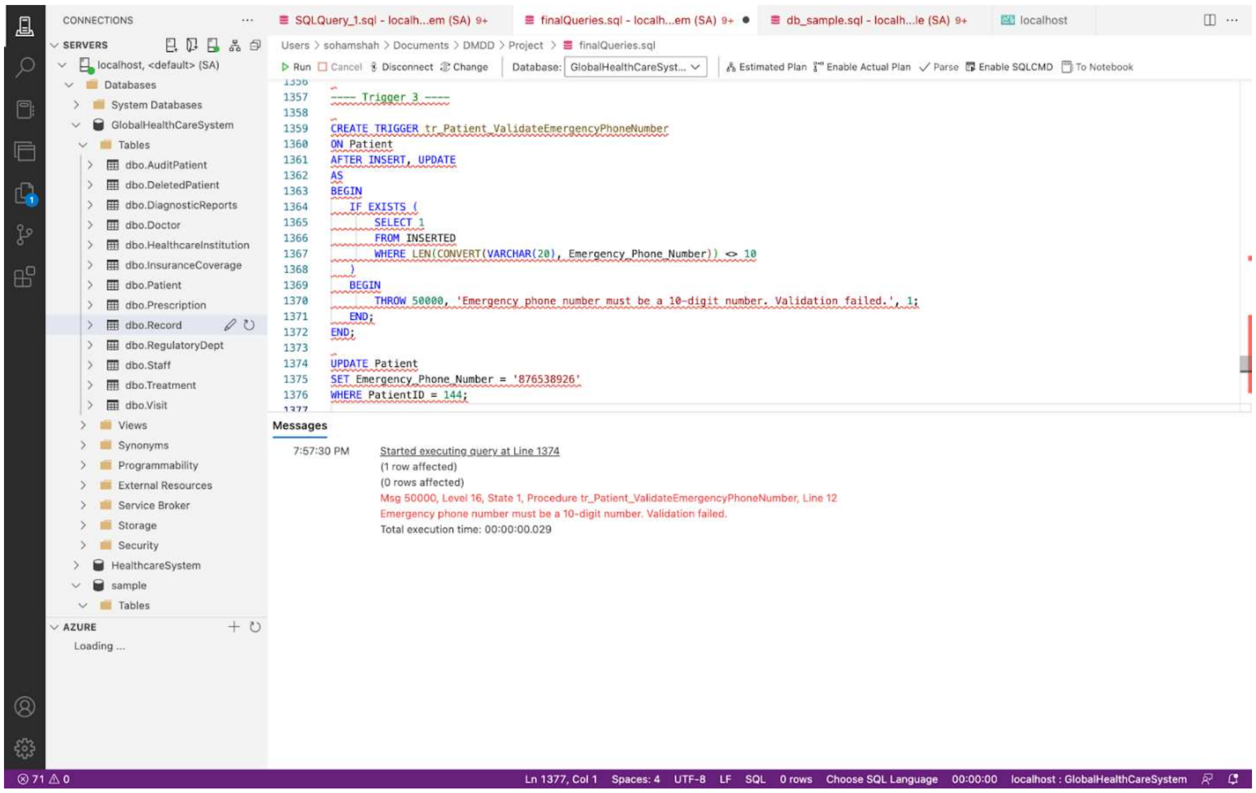


```
CREATE TRIGGER tr_Patient_ValidateEmergencyPhoneNumber
ON Patient
AFTER INSERT, UPDATE
AS
BEGIN
    IF EXISTS (
        SELECT 1
        FROM INSERTED
        WHERE LEN(CONVERT(VARCHAR(20), Emergency_Phone_Number)) <> 10
    )
    BEGIN
        THROW 50000, 'Emergency phone number must be a 10-digit number. Validation failed.', 1;
    END;
END;
```

Messages

```
6:42:19 AM    Started executing query at Line 407
              Commands completed successfully.
              Total execution time: 00:00:00.041
```

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

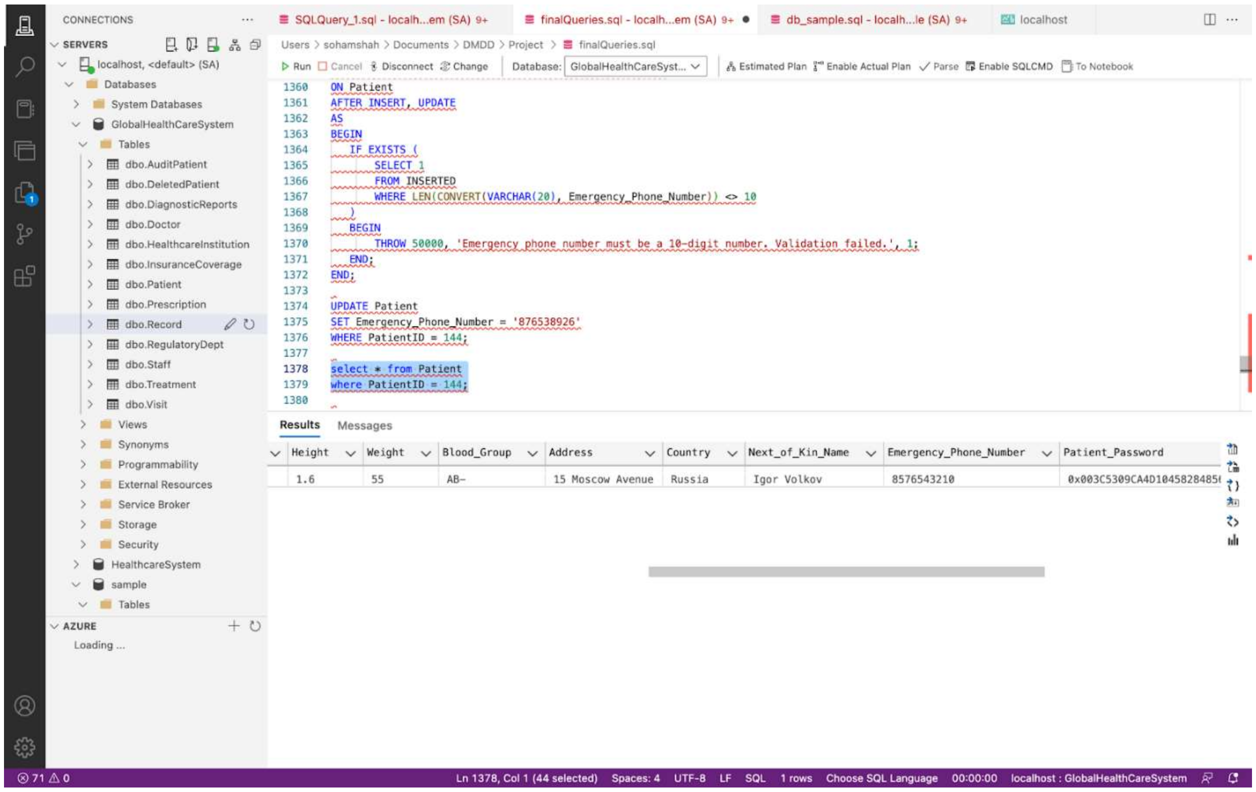


```sql
---- Trigger 3 ----

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;
```

```
7:57:30 PM    Started executing query at Line 1374
              (1 row affected)
              (0 rows affected)
              Msg 50000, Level 16, State 1, Procedure tr_Patient_ValidateEmergencyPhoneNumber, Line 12
              Emergency phone number must be a 10-digit number. Validation failed.
              Total execution time: 00:00:00.029
```

select * from Patient

where PatientID = 144;



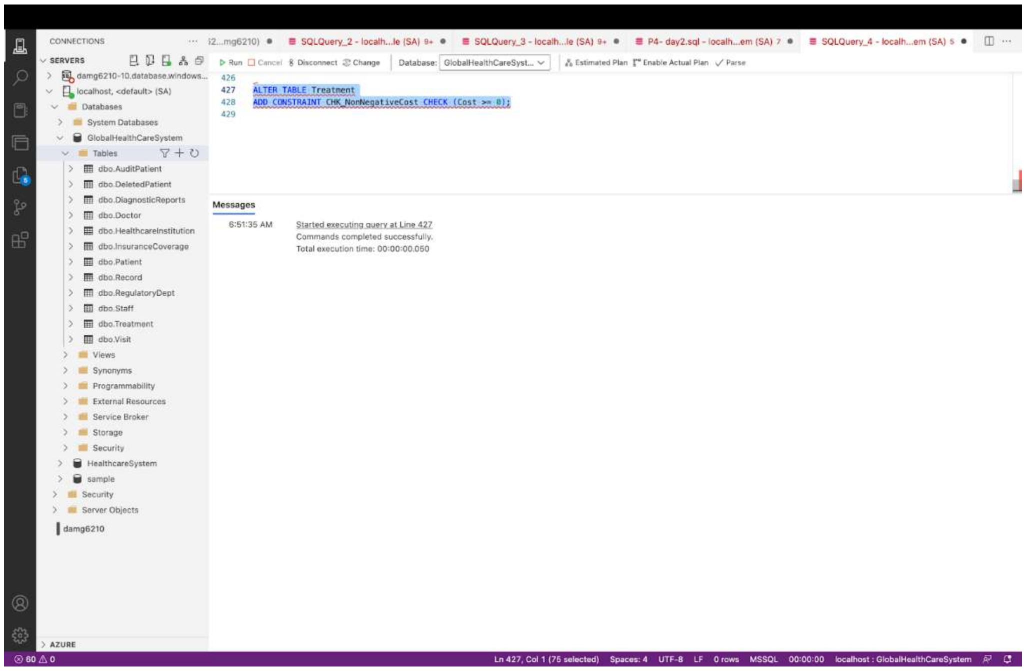
```
ON Patient
AFTER INSERT, UPDATE
AS
BEGIN
    IF EXISTS (
        SELECT 1
        FROM INSERTED
        WHERE LEN(CONVERT(VARCHAR(20), Emergency_Phone_Number)) <> 10
    )
    BEGIN
        THROW 50000, 'Emergency phone number must be a 10-digit number. Validation failed.', 1;
    END;
END;

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

select * from Patient
where PatientID = 144;
```

Height	Weight	Blood_Group	Address	Country	Next_of_Kin_Name	Emergency_Phone_Number	Patient_Password
1.6	55	AB-	15 Moscow Avenue	Russia	Igor Volkov	8576543210	0x003C5309CA4D1045828485

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

ALTER TABLE Treatment

ADD CONSTRAINT CHK_NonNegativeCost CHECK (Cost >= 0);



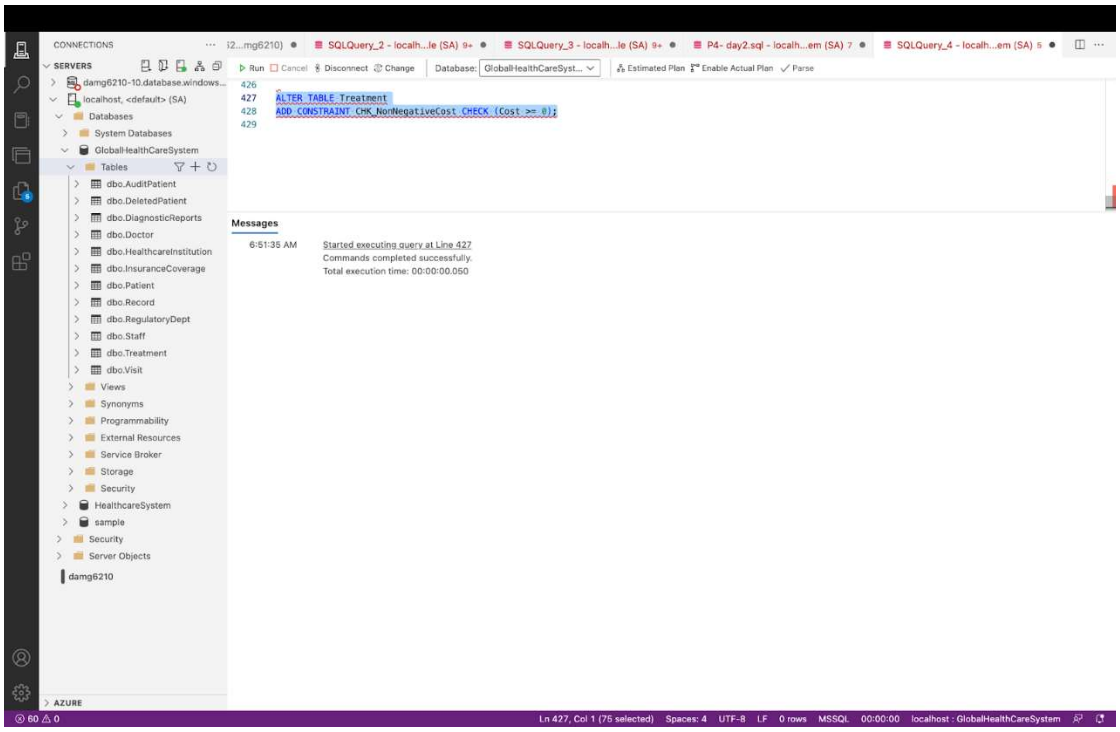
```
426
427  ALTER TABLE Treatment
428  ADD CONSTRAINT CHK_NonNegativeCost CHECK (Cost >= 0);
429
```

Messages

```
6:51:35 AM    Started executing query at Line 427
              Commands completed successfully.
              Total execution time: 00:00:00.050
```

Update Treatment set Cost = -1000

where TreatmentID = 520

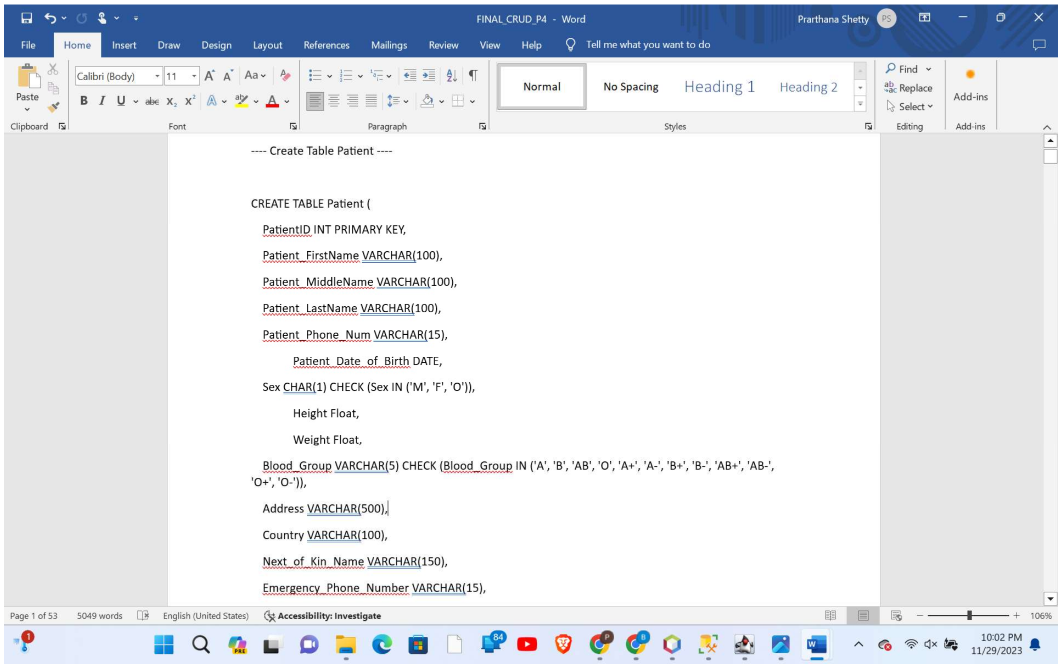


```
426
427  ALTER TABLE Treatment
428  ADD CONSTRAINT CHK_NonNegativeCost CHECK (Cost >= 0);
429
```

Messages

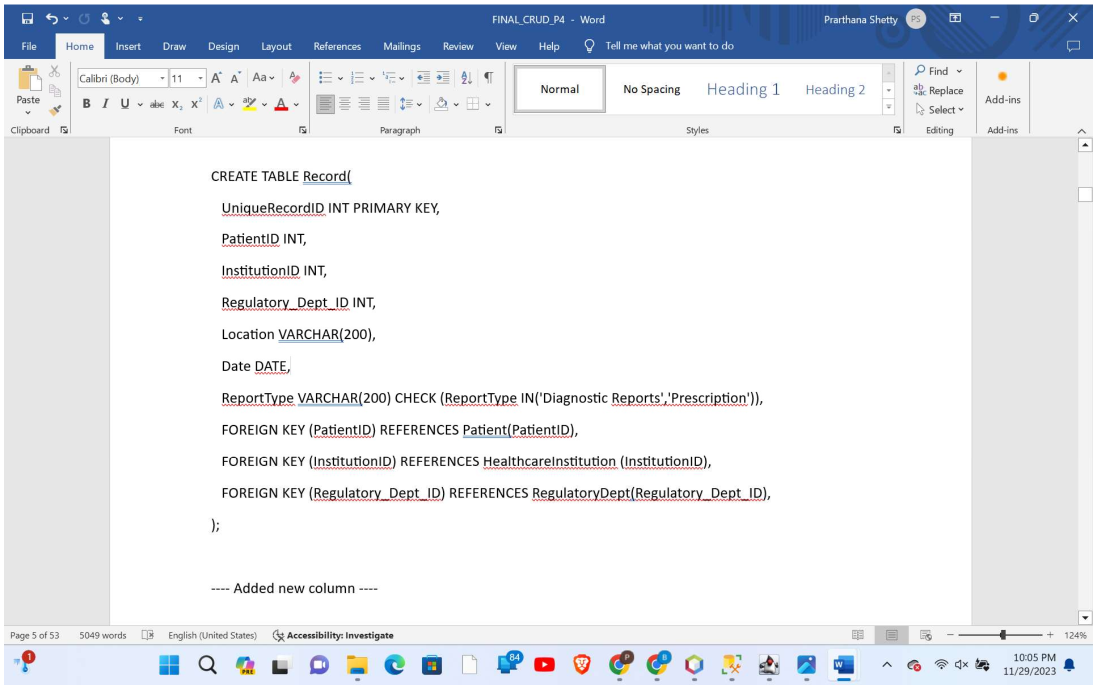
```
6:51:35 AM    Started executing query at Line 427
              Commands completed successfully.
              Total execution time: 00:00:00.050
```

//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),
```

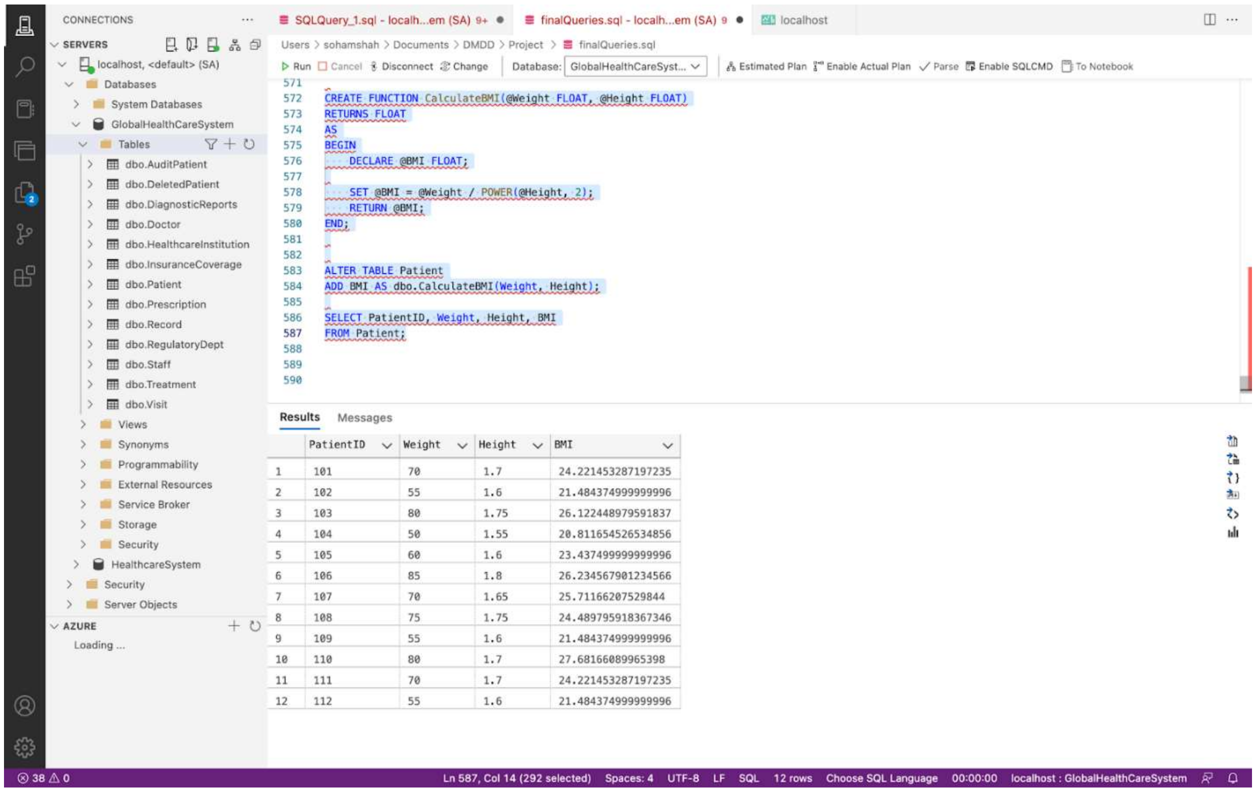
//Record Constraint for ‘ReportType’ which acts as a Discriminator for ‘Record’ table Subtypes ‘Diagnostic Reports’ and ‘Prescription’ (Refer ‘Final_Crud_P4’ pdf file and ‘SQL_CRUD’ file)



```
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)

```sql
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;
```



```sql
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;
```

Results

	PatientID	Weight	Height	BMI
1	101	70	1.7	24.221453287197235
2	102	55	1.6	21.484374999999996
3	103	80	1.75	26.122448979591837
4	104	50	1.55	20.811654526534856
5	105	60	1.6	23.437499999999996
6	106	85	1.8	26.234567901234566
7	107	70	1.65	25.71166207529844
8	108	75	1.75	24.489795918367346
9	109	55	1.6	21.484374999999996
10	110	80	1.7	27.68166089965398
11	111	70	1.7	24.221453287197235
12	112	55	1.6	21.484374999999996

Ln 587, Col 14 (292 selected)   Spaces: 4   UTF-8   LF   SQL   12 rows   00:00:00   localhost : GlobalHealthCareSystem

```
---- 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',

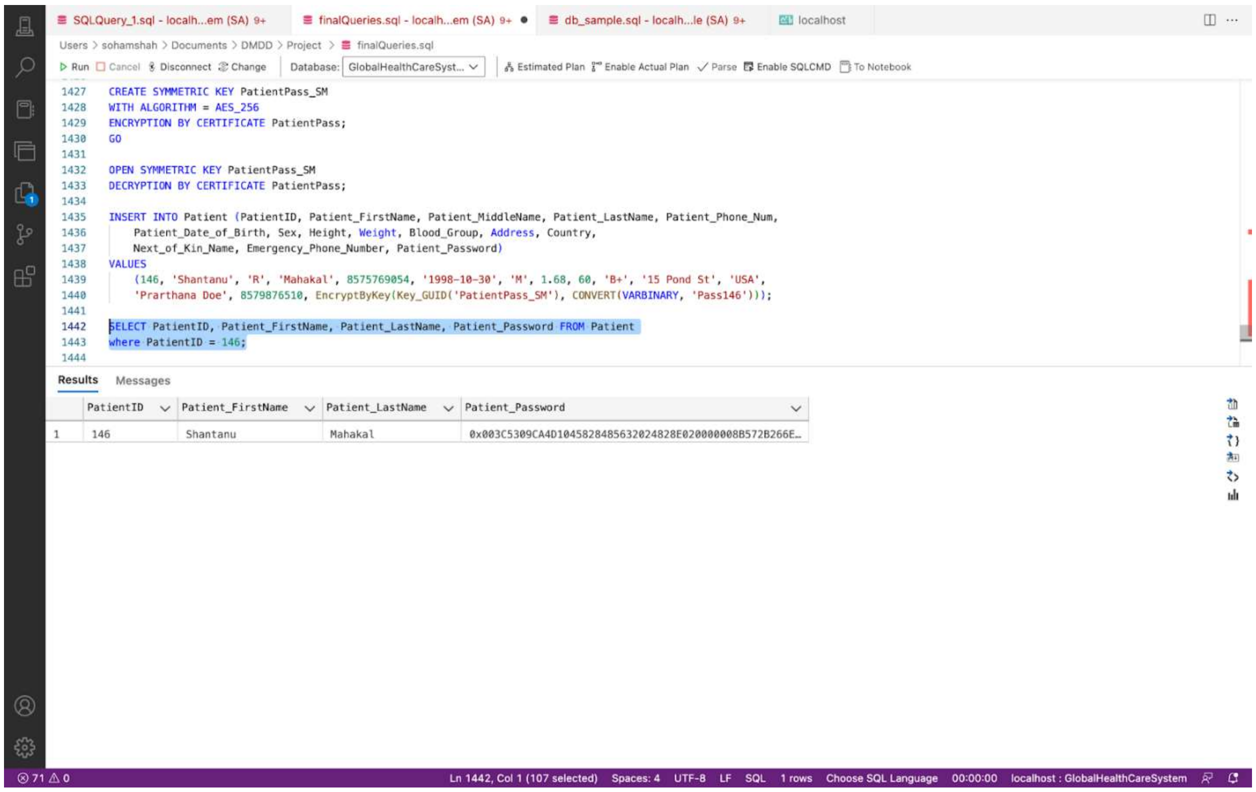
    'Prarthana Doe', 8579876510, EncryptByKey(Key_GUID('PatientPass_SM'), CONVERT(VARBINARY, 'Pass146')));

SELECT PatientID, Patient_FirstName, Patient_LastName, Patient_Password FROM Patient

where PatientID = 146;

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

SELECT * FROM Patient



```
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;
```

	PatientID	Patient_FirstName	Patient_LastName	Patient_Password
1	146	Shantanu	Mahakal	0x003C5309CA4D1045828485632024828E020000008B572B266E…

--- to decrypt ---

OPEN SYMMETRIC KEY PatientPass_SM

DECRYPTION BY CERTIFICATE PatientPass;

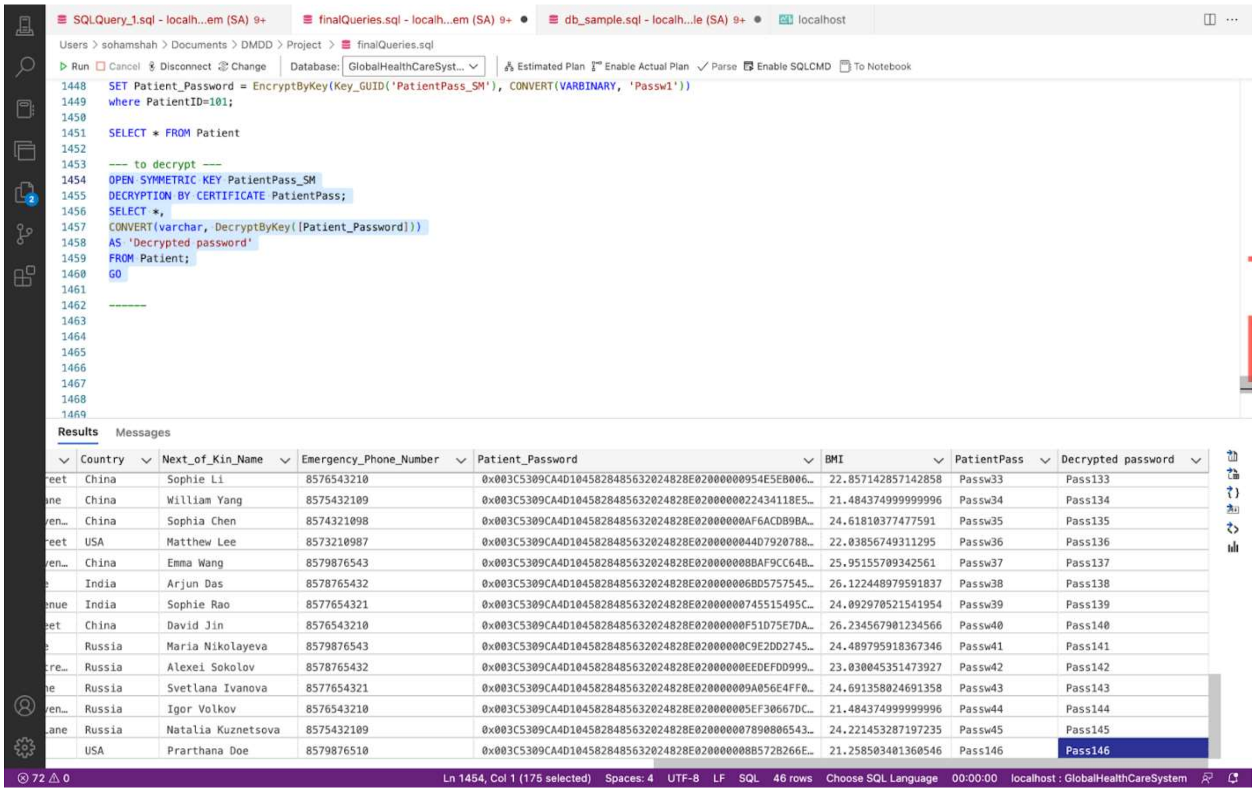
SELECT *,

CONVERT(varchar, DecryptByKey([Patient_Password]))

AS 'Decrypted password'

FROM Patient;

GO



```
SET Patient_Password = EncryptByKey(Key_GUID('PatientPass_SM'), CONVERT(VARBINARY, 'Passw1'))
where PatientID=101;

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

------
```

Results Messages

	Country	Next_of_Kin_Name	Emergency_Phone_Number	Patient_Password	BMI	PatientPass	Decrypted password
reet	China	Sophie Li	8576543210	0x003C5309CA4D1045828485632024828E02000000954E5EB006…	22.857142857142858	Passw33	Pass133
ane	China	William Yang	8575432109	0x003C5309CA4D1045828485632024828E020000002243411 8E5…	21.484374999999996	Passw34	Pass134
ven…	China	Sophia Chen	8574321098	0x003C5309CA4D1045828485632024828E02000000AF6ACDB9BA…	24.61810377477591	Passw35	Pass135
reet	USA	Matthew Lee	8573210987	0x003C5309CA4D1045828485632024828E0200000044D7920788…	22.03856749311295	Passw36	Pass136
ven…	China	Emma Wang	8579876543	0x003C5309CA4D1045828485632024828E020000008BAF9CC64B…	25.95155709342561	Passw37	Pass137
e	India	Arjun Das	8578765432	0x003C5309CA4D1045828485632024828E020000006BD5757545…	26.122448979591837	Passw38	Pass138
enue	India	Sophie Rao	8577654321	0x003C5309CA4D1045828485632024828E0200000074551549 5C…	24.092970521541954	Passw39	Pass139
eet	China	David Jin	8576543210	0x003C5309CA4D1045828485632024828E02000000F51D75E7DA…	26.234567901234566	Passw40	Pass140
e	Russia	Maria Nikolayeva	8579876543	0x003C5309CA4D1045828485632024828E02000000C9E2DD2745…	24.489795918367346	Passw41	Pass141
tre…	Russia	Alexei Sokolov	8578765432	0x003C5309CA4D1045828485632024828E02000000EEDEFDD999…	23.030045351473927	Passw42	Pass142
he	Russia	Svetlana Ivanova	8577654321	0x003C5309CA4D1045828485632024828E020000009A056E4FF0…	24.691358024691358	Passw43	Pass143
ven…	Russia	Igor Volkov	8576543210	0x003C5309CA4D1045828485632024828E020000005EF30667DC…	21.484374999999996	Passw44	Pass144
Lane	Russia	Natalia Kuznetsova	8575432109	0x003C5309CA4D1045828485632024828E0200000078908065 43…	24.221453287197235	Passw45	Pass145
	USA	Prarthana Doe	8579876510	0x003C5309CA4D1045828485632024828E020000008B572B266E…	21.258503401360546	Pass146	Pass146

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

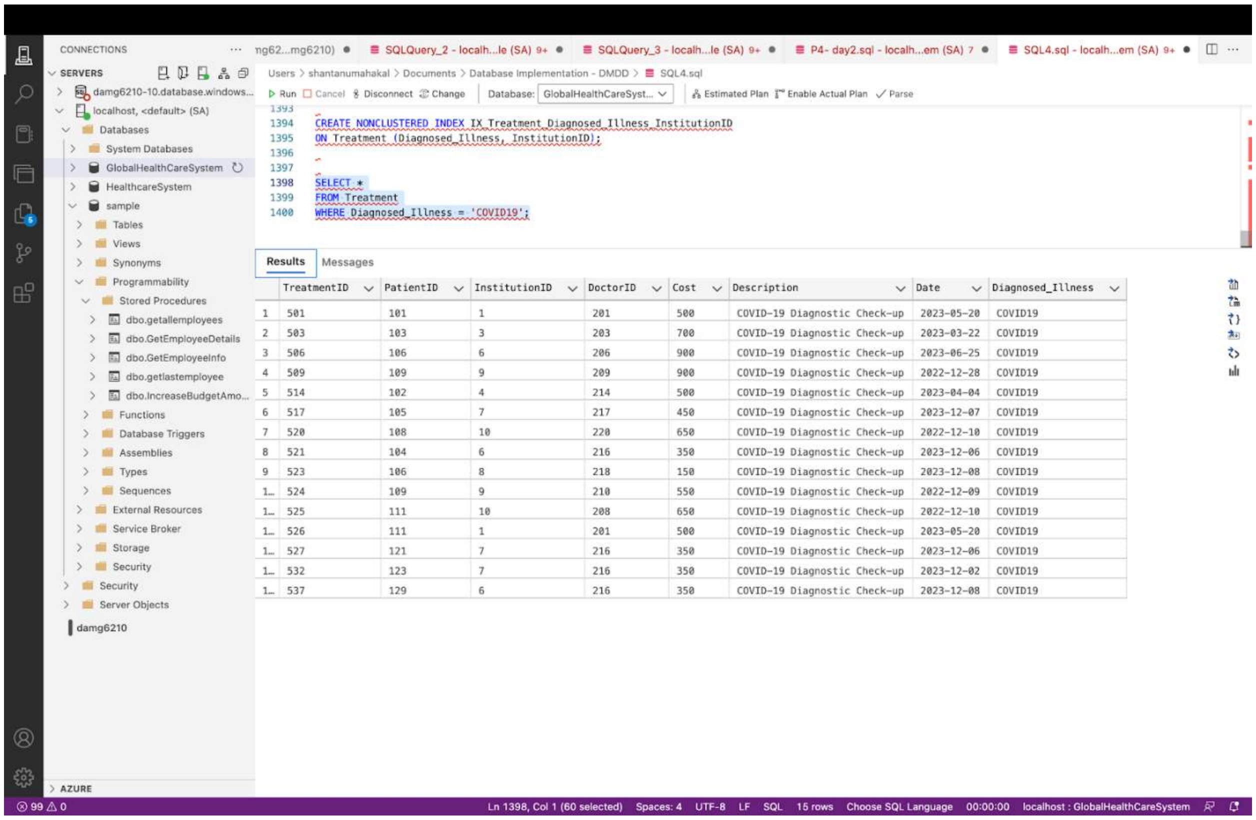
CREATE NONCLUSTERED INDEX IX_Treatment_Diagnosed_Illness_InstitutionID

ON Treatment (Diagnosed_Illness, InstitutionID);

SELECT *

FROM Treatment

WHERE Diagnosed_Illness = 'COVID19';

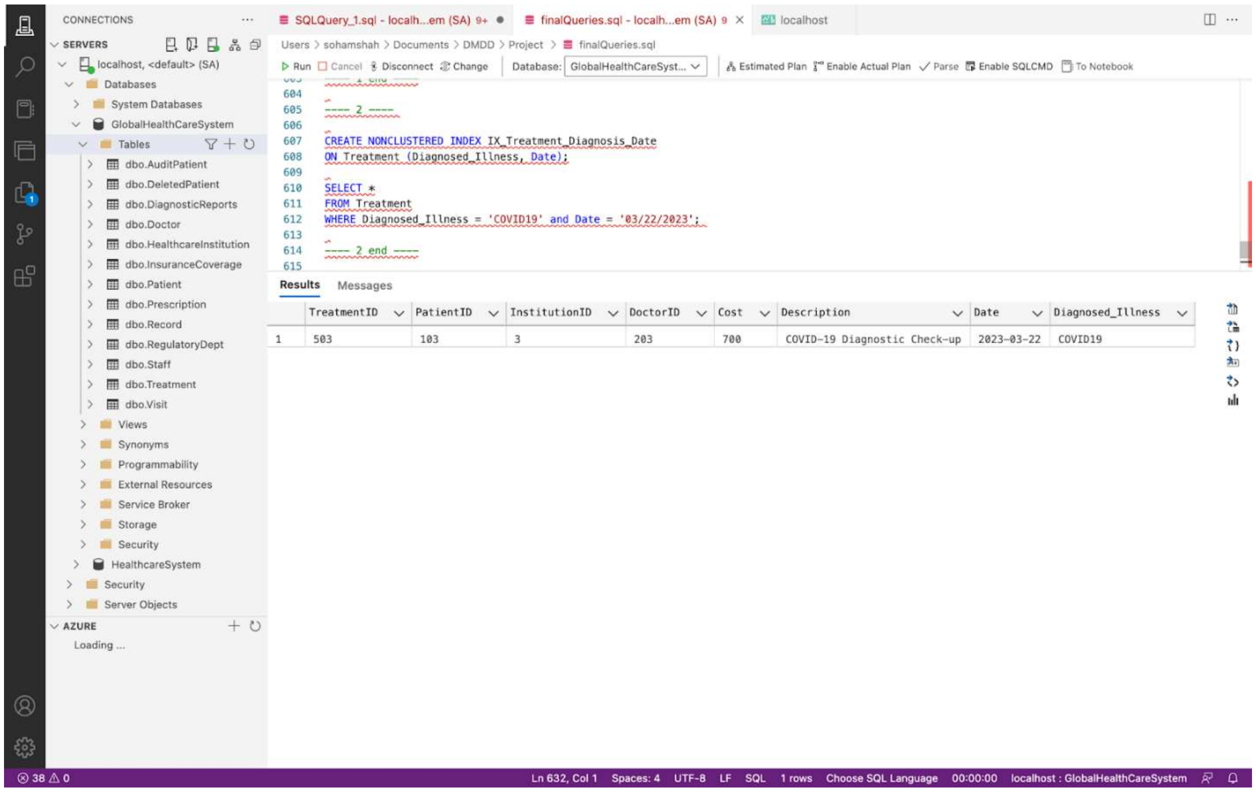


```
CREATE NONCLUSTERED INDEX IX_Treatment_Diagnosed_Illness_InstitutionID
ON Treatment (Diagnosed_Illness, InstitutionID);


SELECT *
FROM Treatment
WHERE Diagnosed_Illness = 'COVID19';
```

	TreatmentID	PatientID	InstitutionID	DoctorID	Cost	Description	Date	Diagnosed_Illness
1	501	101	1	201	500	COVID-19 Diagnostic Check-up	2023-05-20	COVID19
2	503	103	3	203	700	COVID-19 Diagnostic Check-up	2023-03-22	COVID19
3	506	106	6	206	900	COVID-19 Diagnostic Check-up	2023-06-25	COVID19
4	509	109	9	209	900	COVID-19 Diagnostic Check-up	2022-12-28	COVID19
5	514	102	4	214	500	COVID-19 Diagnostic Check-up	2023-04-04	COVID19
6	517	105	7	217	450	COVID-19 Diagnostic Check-up	2023-12-07	COVID19
7	520	108	10	220	650	COVID-19 Diagnostic Check-up	2022-12-10	COVID19
8	521	104	6	216	350	COVID-19 Diagnostic Check-up	2023-12-06	COVID19
9	523	106	8	218	150	COVID-19 Diagnostic Check-up	2023-12-08	COVID19
1...	524	109	9	210	550	COVID-19 Diagnostic Check-up	2022-12-09	COVID19
1...	525	111	10	208	650	COVID-19 Diagnostic Check-up	2022-12-10	COVID19
1...	526	111	1	201	500	COVID-19 Diagnostic Check-up	2023-05-20	COVID19
1...	527	121	7	216	350	COVID-19 Diagnostic Check-up	2023-12-06	COVID19
1...	532	123	7	216	350	COVID-19 Diagnostic Check-up	2023-12-02	COVID19
1...	537	129	6	216	350	COVID-19 Diagnostic Check-up	2023-12-08	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';
```



```sql
---- 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';

---- 2 end ----
```

	TreatmentID	PatientID	InstitutionID	DoctorID	Cost	Description	Date	Diagnosed_Illness
1	503	103	3	203	700	COVID-19 Diagnostic Check-up	2023-03-22	COVID19

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

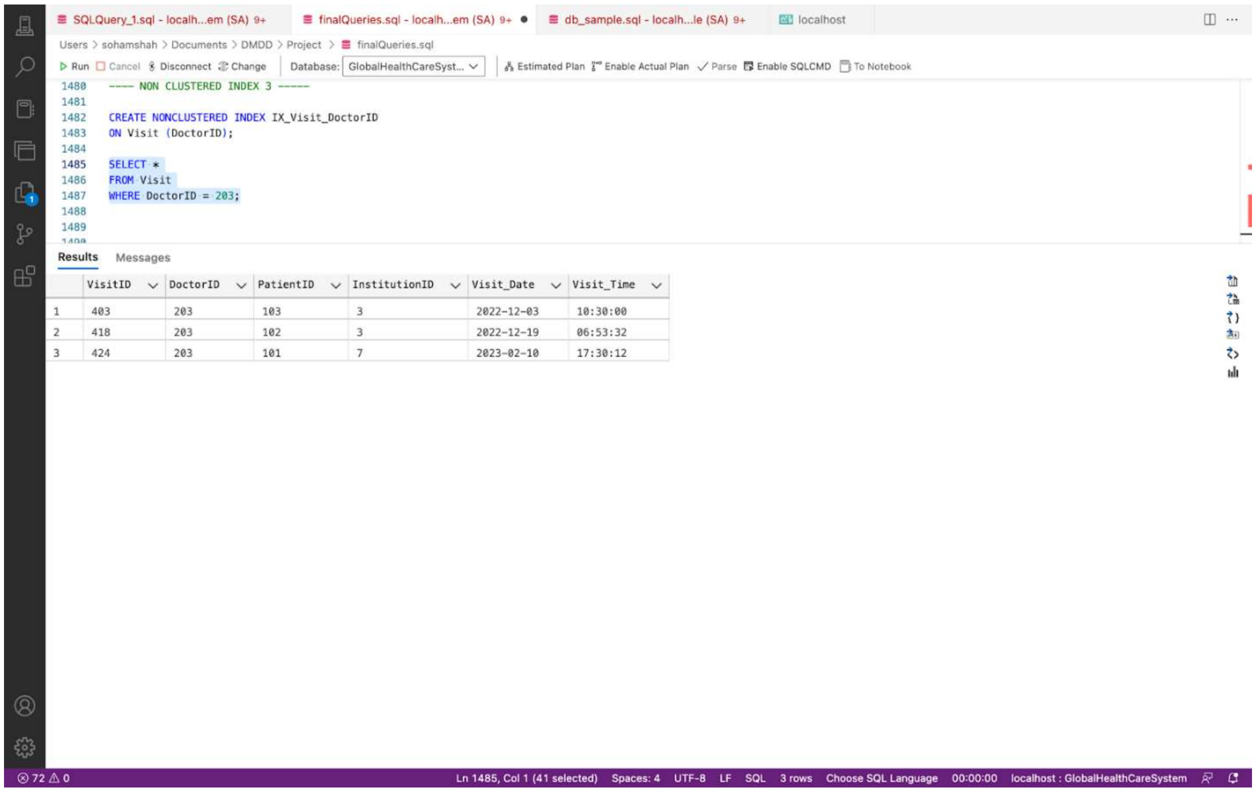
CREATE NONCLUSTERED INDEX IX_Visit_DoctorID

ON Visit (DoctorID);

SELECT *

FROM Visit

WHERE DoctorID = 203;



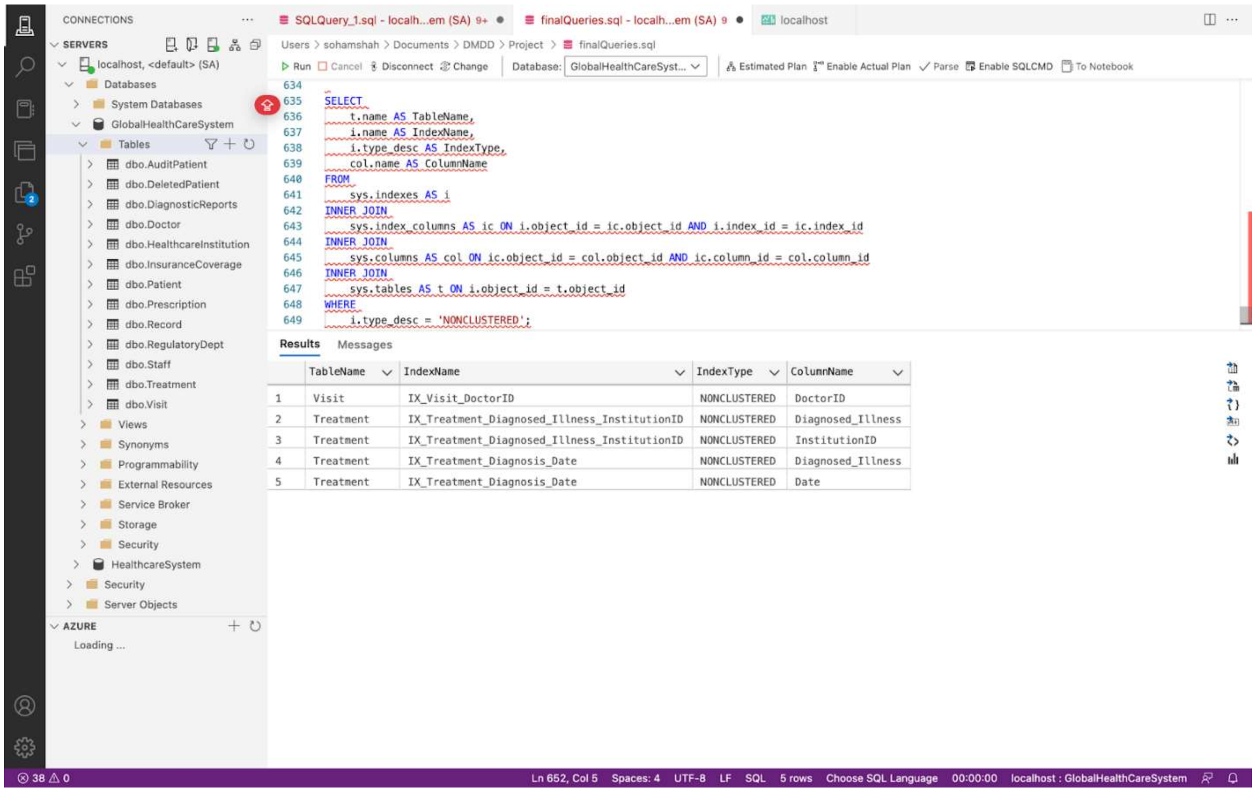
```
---- NON CLUSTERED INDEX 3 -----

CREATE NONCLUSTERED INDEX IX_Visit_DoctorID
ON Visit (DoctorID);

SELECT *
FROM Visit
WHERE DoctorID = 203;
```

	VisitID	DoctorID	PatientID	InstitutionID	Visit_Date	Visit_Time
1	403	203	103	3	2022-12-03	10:30:00
2	418	203	102	3	2022-12-19	06:53:32
3	424	203	101	7	2023-02-10	17:30:12

---- 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';
```



```
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';
```

	TableName	IndexName	IndexType	ColumnName
1	Visit	IX_Visit_DoctorID	NONCLUSTERED	DoctorID
2	Treatment	IX_Treatment_Diagnosed_Illness_InstitutionID	NONCLUSTERED	Diagnosed_Illness
3	Treatment	IX_Treatment_Diagnosed_Illness_InstitutionID	NONCLUSTERED	InstitutionID
4	Treatment	IX_Treatment_Diagnosis_Date	NONCLUSTERED	Diagnosed_Illness
5	Treatment	IX_Treatment_Diagnosis_Date	NONCLUSTERED	Date