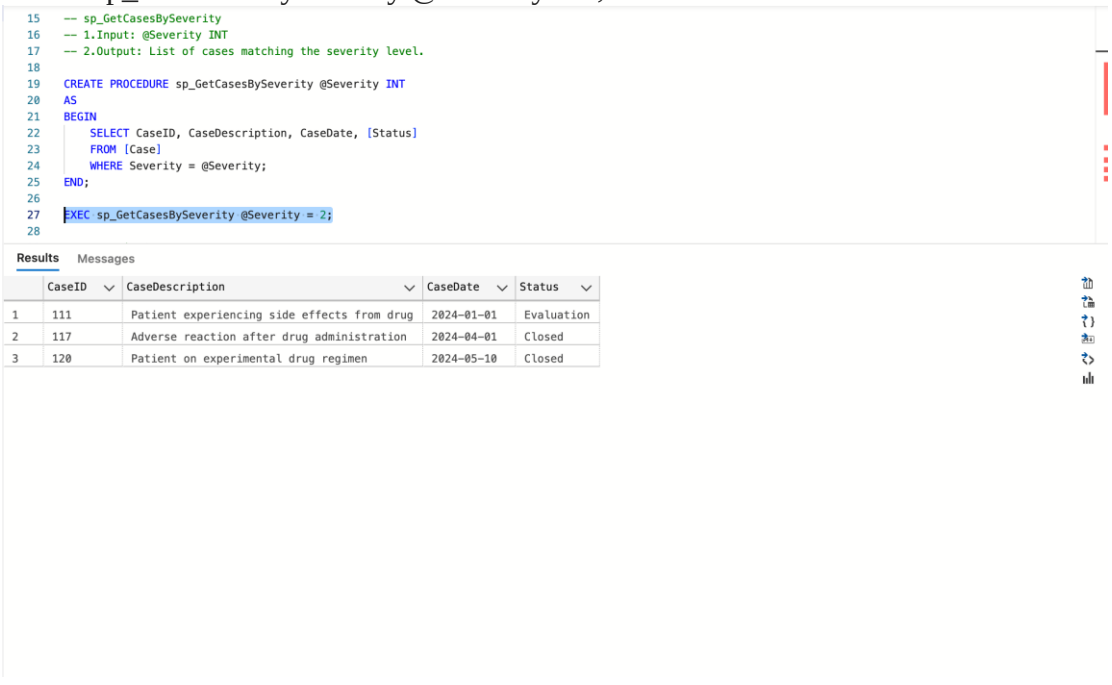
Screenshots/brief explanations of Stored Procedures, Views and User Defined Functions created

STORED PROCEDURES:

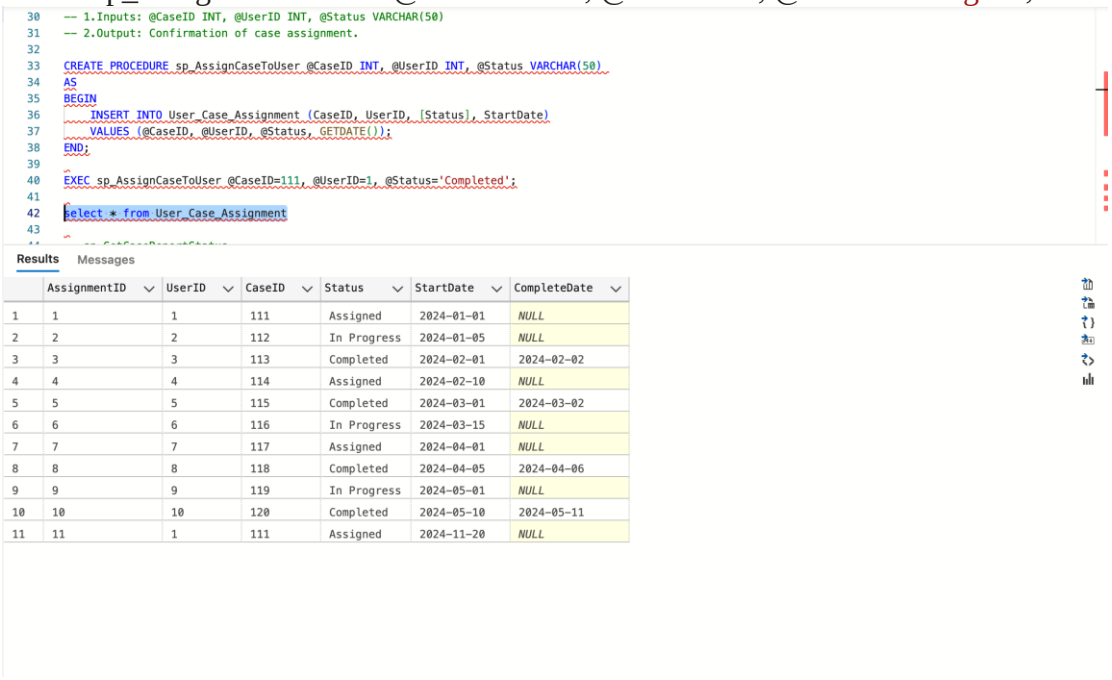
```
-- sp_GetCasesBySeverity
-- 1.Input: @Severity INT
-- 2.Output: List of cases matching the severity level.

CREATE PROCEDURE sp_GetCasesBySeverity @Severity INT
AS
BEGIN
SELECT CaseID, CaseDescription, CaseDate, [Status]
FROM [Case]
WHERE Severity = @Severity;
END;

EXEC sp_GetCasesBySeverity @Severity = 2;
```




```
-- sp_AssignCaseToUser
-- 1.Inputs: @CaseID INT, @UserID INT, @Status VARCHAR(50)
-- 2.Output: Confirmation of case assignment.
```

```
CREATE PROCEDURE sp_AssignCaseToUser @CaseID INT, @UserID INT,
@Status VARCHAR(50)
AS
BEGIN
INSERT INTO User_Case_Assignment (CaseID, UserID, [Status], StartDate)
VALUES (@CaseID, @UserID, @Status, GETDATE());
END;
```

```
EXEC sp_AssignCaseToUser @CaseID=111, @UserID=1, @Status='Assigned;
```

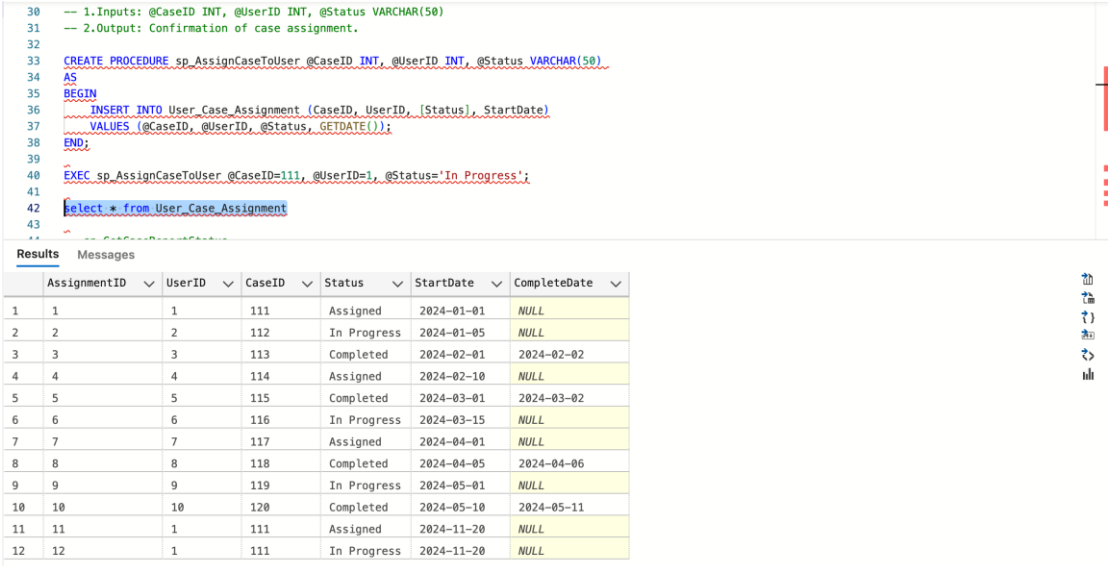


```
-- 1.Inputs: @CaseID INT, @UserID INT, @Status VARCHAR(50)
-- 2.Output: Confirmation of case assignment.

CREATE PROCEDURE sp_AssignCaseToUser @CaseID INT, @UserID INT, @Status VARCHAR(50)
AS
BEGIN
    INSERT INTO User_Case_Assignment (CaseID, UserID, [Status], StartDate)
    VALUES (@CaseID, @UserID, @Status, GETDATE());
END;

EXEC sp_AssignCaseToUser @CaseID=111, @UserID=1, @Status='Completed';

select * from User_Case_Assignment
```

Results | Messages

	AssignmentID	UserID	CaseID	Status	StartDate	CompleteDate
1	1	1	111	Assigned	2024-01-01	NULL
2	2	2	112	In Progress	2024-01-05	NULL
3	3	3	113	Completed	2024-02-01	2024-02-02
4	4	4	114	Assigned	2024-02-10	NULL
5	5	5	115	Completed	2024-03-01	2024-03-02
6	6	6	116	In Progress	2024-03-15	NULL
7	7	7	117	Assigned	2024-04-01	NULL
8	8	8	118	Completed	2024-04-05	2024-04-06
9	9	9	119	In Progress	2024-05-01	NULL
10	10	10	120	Completed	2024-05-10	2024-05-11
11	11	1	111	Assigned	2024-11-20	NULL

```
EXEC sp_AssignCaseToUser @CaseID=111, @UserID=1, @Status='In Progress';
```

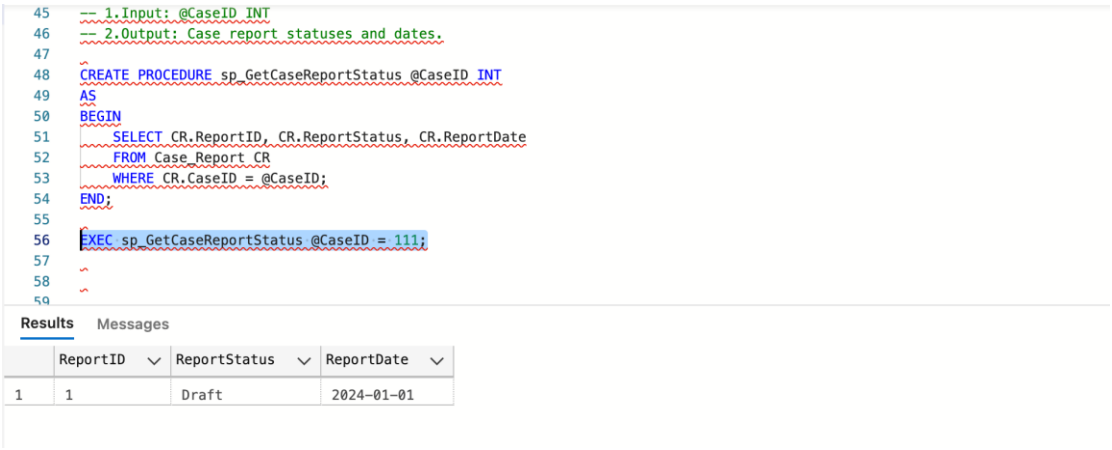


```
-- 1.Inputs: @CaseID INT, @UserID INT, @Status VARCHAR(50)
-- 2.Output: Confirmation of case assignment.

CREATE PROCEDURE sp_AssignCaseToUser @CaseID INT, @UserID INT, @Status VARCHAR(50)
AS
BEGIN
    INSERT INTO User_Case_Assignment (CaseID, UserID, [Status], StartDate)
    VALUES (@CaseID, @UserID, @Status, GETDATE());
END;

EXEC sp_AssignCaseToUser @CaseID=111, @UserID=1, @Status='In Progress';

select * from User_Case_Assignment
```

Results | Messages

	AssignmentID	UserID	CaseID	Status	StartDate	CompleteDate
1	1	1	111	Assigned	2024-01-01	NULL
2	2	2	112	In Progress	2024-01-05	NULL
3	3	3	113	Completed	2024-02-01	2024-02-02
4	4	4	114	Assigned	2024-02-10	NULL
5	5	5	115	Completed	2024-03-01	2024-03-02
6	6	6	116	In Progress	2024-03-15	NULL
7	7	7	117	Assigned	2024-04-01	NULL
8	8	8	118	Completed	2024-04-05	2024-04-06
9	9	9	119	In Progress	2024-05-01	NULL
10	10	10	120	Completed	2024-05-10	2024-05-11
11	11	1	111	Assigned	2024-11-20	NULL
12	12	1	111	In Progress	2024-11-20	NULL

```
-- sp_GetCaseReportStatus
-- 1.Input: @CaseID INT
-- 2.Output: Case report statuses and dates.

CREATE PROCEDURE sp_GetCaseReportStatus @CaseID INT
AS
BEGIN
SELECT CR.ReportID, CR.ReportStatus, CR.ReportDate
FROM Case_Report CR
WHERE CR.CaseID = @CaseID;
END;

EXEC @CaseID = 111;
```



VIEWS:

1. vw_ActiveCases:

- **Purpose**: This view provides a list of active cases, excluding those that are "Closed" or "Case Locked." It can be used for reporting the current state of cases within the system.
- **Use in Decision Making**:
  - **Case Management**: Managers or decision-makers can quickly identify which cases are still open or in progress, helping prioritize actions, resources, or legal actions based on case severity or other factors.
  - **Workflow Monitoring**: It helps track which cases need attention and are still in an active state, aiding in operational efficiency and ensuring that closed or locked cases are excluded from ongoing processes.

2. vw_RegulatoryReports:

- **Purpose**: This view displays the status of regulatory reports submitted to regulatory agencies, showing the tracking number, submission date, and current status.
- **Use in Decision Making**:
  - **Regulatory Compliance**: Organizations can use this view to monitor the status of reports submitted to regulatory bodies, ensuring that all required reports are filed and are up to date. If there are delays or issues, corrective actions can be taken to ensure compliance.

3. vw_PatientOverview:

- **Purpose**: This view provides a detailed summary of patients and their cases, including patient names, medical history, case descriptions, case dates, and severity.
- **Use in Decision Making**:
  - **Patient Care and Treatment**: Healthcare providers or case managers can use this view to get an overview of a patient's medical background and ongoing case severity. It aids in making informed decisions regarding the patient's treatment plan, interventions, or follow-ups.
  - **Care Prioritization**: The view helps prioritize care for patients with more severe cases, allowing healthcare teams to act promptly.

4. vw_ProductDosage:

- **Purpose**: This view shows the products and their corresponding dosage regimens, which includes product names, dosages, and frequencies.
- **Use in Decision Making**:
  - **Treatment Planning**: Healthcare providers can use this view to confirm or adjust the dosage regimens for patients based on the products prescribed. This ensures that treatments are administered as per the prescribed plan, improving patient outcomes.
  - **Inventory and Stocking Decisions**: By reviewing which products are in use and their dosages, hospitals or pharmacies can make decisions about inventory management, ensuring that products are stocked appropriately based on demand.

  Sdf

5. CaseFollowupSummary:

- **Purpose:** This view provides insights into cases, their current status, the patient involved, the most recent follow-up date, and the total number of follow-ups.

```
-- vw_ActiveCases
-- 1.Reports all active cases.

CREATE VIEW vw_ActiveCases AS
SELECT CaseID, CaseDescription, Severity, [Status] FROM [Case]
WHERE [Status] NOT IN ('Closed', 'Case Locked');

select * from vw_ActiveCases;
```

```
74
75  IF OBJECT_ID('vw_ProductDosage', 'V') IS NOT NULL
76      DROP VIEW vw_ProductDosage;
77  GO
78
79  -- vw_ActiveCases
80  -- 1.Reports all active cases.
81
82  CREATE VIEW vw_ActiveCases AS
83  SELECT CaseID, CaseDescription, Severity, [Status] FROM [Case]
84  WHERE [Status] NOT IN ('Closed', 'Case Locked');
85
86  select * from vw_ActiveCases;
87
```

Results | Messages

	CaseID	CaseDescription	Severity	Status
1	111	Patient experiencing side effects from drug	2	Evaluation
2	112	Patient undergoing clinical trial	3	Submitted for MR
3	113	Severe adverse reaction to medication	5	MR In-Progress
4	114	Patient suffering from chronic illness	4	Submitted for Quality Check
5	115	Case of drug overdose	5	Quality Check In-Progress
6	118	Patient with multiple drug interactions	4	Evaluation
7	119	Clinical trial results and analysis	1	Submitted for MR

```
-- vw_RegulatoryReports
-- 1.Shows the status of reports submitted to regulatory agencies.

CREATE VIEW vw_RegulatoryReports AS
SELECT RA.AgencyName, RCCR.TrackingNum, RCCR.[Status],
RCCR.SubmissionDate
FROM Regulatory_Case_Reports RCCR JOIN Regulatory_Agency RA ON
RCCR.AgencyID = RA.AgencyID;

select * from vw_RegulatoryReports;
```

```
87
88  -- vw_RegulatoryReports
89  -- 1.Shows the status of reports submitted to regulatory agencies.
90
91  CREATE VIEW vw_RegulatoryReports AS
92  SELECT RA.AgencyName, RCCR.TrackingNum, RCCR.[Status], RCCR.SubmissionDate
93  FROM Regulatory_Case_Reports RCCR JOIN Regulatory_Agency RA ON RCCR.AgencyID = RA.AgencyID;
94
95  select * from vw_RegulatoryReports;
96
```

Results Messages

	AgencyName	TrackingNum	Status	SubmissionDate
1	FDA	101	Pending	2024-01-01
2	EMA	102	Sent	2024-01-05
3	MHRA	103	Success	2024-02-01
4	TGA	104	Error	2024-02-10
5	HC	105	Pending	2024-03-01
6	PMDA	106	Sent	2024-03-15
7	BfArM	107	Success	2024-04-01
8	SwissMedic	108	Error	2024-04-05
9	ANVISA	109	Pending	2024-05-01
10	NMPA	110	Sent	2024-05-10

```
-- vw_PatientOverview
-- 1.Provides a detailed summary of patients and their cases.
CREATE VIEW vw_PatientOverview AS
SELECT P.PatientName, P.MedicalHistory, C.CaseDescription, C.CaseDate,
C.Severity FROM Patient P JOIN [Case] C ON P.PatientID = C.PatientID;

select * from vw_PatientOverview
```

```
96
97  -- vw_PatientOverview
98  -- 1.Provides a detailed summary of patients and their cases.
99  CREATE VIEW vw_PatientOverview AS
100 SELECT P.PatientName, P.MedicalHistory, C.CaseDescription, C.CaseDate, C.Severity
101 FROM Patient P JOIN [Case] C ON P.PatientID = C.PatientID;
102
103 select * from vw_PatientOverview
104
```

Results Messages

	PatientName	MedicalHistory	CaseDescription	CaseDate	Severity
1	John Doe	No significant history	Patient experiencing side effects from drug	2024-01-01	2
2	Jane Smith	Diabetes	Patient undergoing clinical trial	2024-01-05	3
3	Sam Green	Asthma	Severe adverse reaction to medication	2024-02-01	5
4	Lisa White	Hypertension	Patient suffering from chronic illness	2024-02-10	4
5	Mike Black	No significant history	Case of drug overdose	2024-03-01	5
6	Sophia Blue	Chronic Migraine	Long-term side effects under observation	2024-03-15	3
7	David Grey	No significant history	Adverse reaction after drug administration	2024-04-01	2
8	Isabella Brown	Cancer	Patient with multiple drug interactions	2024-04-05	4
9	Ethan Yellow	No significant history	Clinical trial results and analysis	2024-05-01	1
10	Ava Red	No significant history	Patient on experimental drug regimen	2024-05-10	2

```
-- vw_ProductDosage
-- 1.View to show products and their dosage regimens

CREATE VIEW vw_ProductDosage AS
```

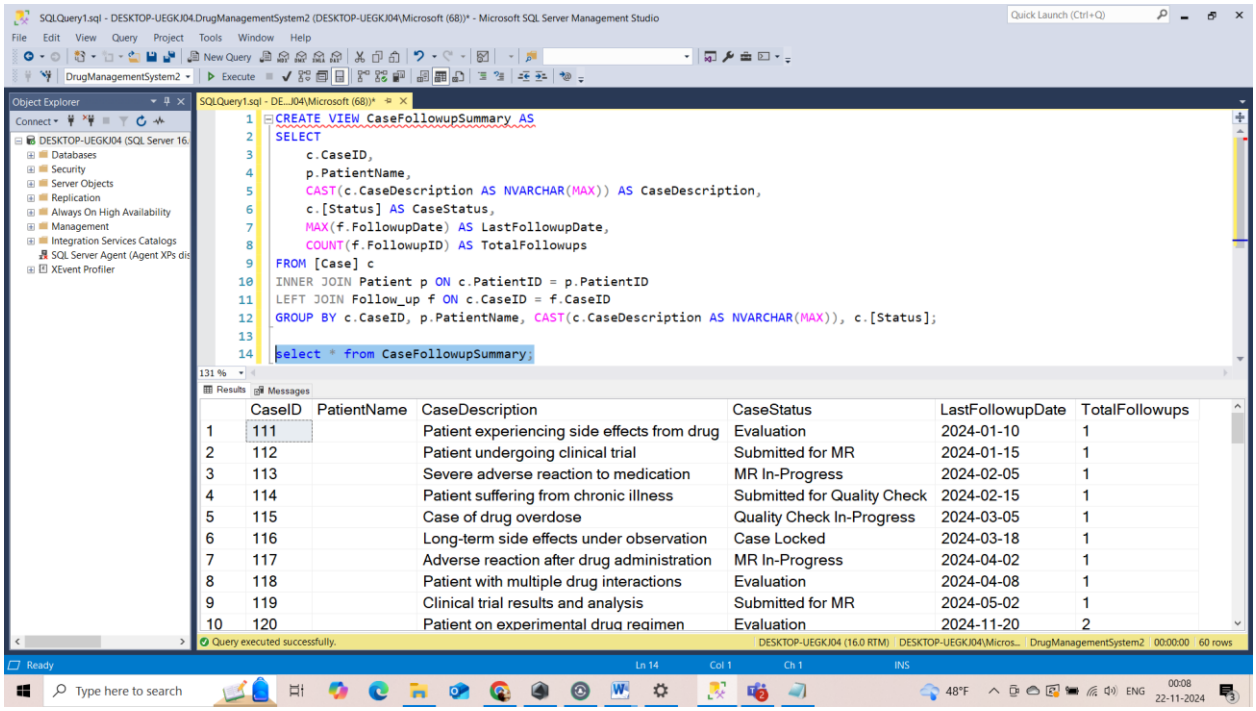
SELECT p.ProductName, dr.Dosage, dr.Frequency FROM Dose_Regimen dr JOIN [Product] p ON dr.ProductID = p.ProductID;

```
104
105    -- vw_ProductDosage
106    -- 1.View to show products and their dosage regimens
107    CREATE VIEW vw_ProductDosage AS
108    SELECT p.ProductName, dr.Dosage, dr.Frequency
109    FROM Dose_Regimen dr JOIN [Product] p ON dr.ProductID = p.ProductID;
110
111    select * from vw_ProductDosage
112
```

Results Messages

	ProductName	Dosage	Frequency
1	Aspirin	500 mg	Once Daily
2	Tylenol	5 ml	Twice a Day
3	Insulin	1 g	Once Weekly
4	Ventolin	200 mg	Every 12 Hours
5	Advil	5 mg	Once Daily
6	Salbutamol	1.5 ml	Twice a Day
7	Glucagon	750 mg	Once Daily
8	Ibuprofen	2 ml	Twice a Day
9	Metformin	10 mg	Once Daily
10	Ciprofloxacin	5 mg	Once Daily

```
--provides insights into cases, their current status, the patient involved, the
most recent follow-up date, and the total number of follow-ups.
CREATE VIEW CaseFollowupSummary AS
SELECT
    c.CaseID,
    p.PatientName,
    CAST(c.CaseDescription AS NVARCHAR(MAX)) AS CaseDescription,
    c.[Status] AS CaseStatus,
    MAX(f.FollowupDate) AS LastFollowupDate,
    COUNT(f.FollowupID) AS TotalFollowups
FROM [Case] c
INNER JOIN Patient p ON c.PatientID = p.PatientID
LEFT JOIN Follow_up f ON c.CaseID = f.CaseID
GROUP BY c.CaseID, p.PatientName, CAST(c.CaseDescription AS NVARCHAR(MAX)),
c.[Status];

select * from CaseFollowupSummary;
```



USER_DEFINED FUNCTIONS

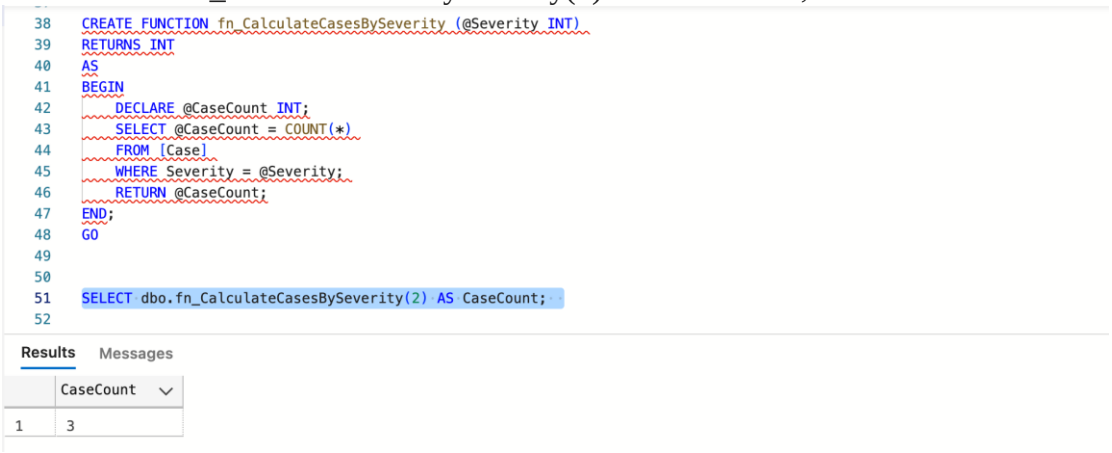
```
-- uf_IsReportApproved
-- 1.Checks if a report is approved.

CREATE FUNCTION uf_IsReportApproved (@ReportID INT) RETURNS BIT
AS
BEGIN
RETURN (SELECT CASE WHEN ReportStatus = 'Approved' THEN 1 ELSE 0 END
FROM Case_Report WHERE ReportID = @ReportID);
END;

SELECT dbo.uf_IsReportApproved(6) AS IsApproved;
```

```
-- 1.Checks if a report is approved.

CREATE FUNCTION uf_IsReportApproved (@ReportID INT) RETURNS BIT
 AS
 BEGIN
     RETURN (SELECT CASE WHEN ReportStatus = 'Approved' THEN 1 ELSE 0 END FROM Case_Report WHERE ReportID = @ReportID);
END;

SELECT dbo.uf_IsReportApproved(6) AS IsApproved;

SELECT dbo.uf_IsReportApproved(1) AS IsApproved;
-- UDF to calculate the number of cases by severity
```

Results Messages

	IsApproved
1	1

```
-- UDF to calculate the number of cases by severity

CREATE FUNCTION fn_CalculateCasesBySeverity (@Severity INT)
RETURNS INT
AS
BEGIN
DECLARE @CaseCount INT;
SELECT @CaseCount = COUNT(*)
FROM [Case]
WHERE Severity = @Severity;
RETURN @CaseCount;
END;
GO


SELECT dbo.fn_CalculateCasesBySeverity(2) AS CaseCount;
```




```
-- UDF to get the full name of a regulatory agency
CREATE FUNCTION fn_GetRegulatoryAgencyName (@AgencyID INT)
RETURNS VARCHAR(100)
AS
BEGIN
DECLARE @AgencyName VARCHAR(100);
SELECT @AgencyName = AgencyName FROM Regulatory_Agency WHERE
AgencyID = @AgencyID;
RETURN @AgencyName;
END;

SELECT dbo.fn_GetRegulatoryAgencyName(1002) AS AgencyName;
```

```sql

-- UDF to get the full name of a regulatory agency
CREATE FUNCTION fn_GetRegulatoryAgencyName (@AgencyID INT) RETURNS VARCHAR(100)
AS
BEGIN
    DECLARE @AgencyName VARCHAR(100);
    SELECT @AgencyName = AgencyName FROM Regulatory_Agency WHERE AgencyID = @AgencyID;
    RETURN @AgencyName;
END;

SELECT dbo.fn_GetRegulatoryAgencyName(1002) AS AgencyName;

```

Results Messages

	AgencyName
1	EMA