Error Handling in SQL Server:

CREATE PROCEDURE DivideTwoNumber

@Number1 INT,

@Number2 INT

AS

BEGIN

DECLARE @Result INT

SET @Result = 0

SET @Result = @Number1 / @Number2

PRINT 'RESULT IS :'+CAST(@Result AS VARCHAR)

END

exec DivideTwoNumber 20,2

exec DivideTwoNumber 10,0

Handling the Error using Global variable:

ALTER PROCEDURE DivideTwoNumber

@Number1 INT,

@Number2 INT

AS

BEGIN

DECLARE @Result INT

SET @Result = 0

IF(@Number2 = 0)

RAISERROR('Second Number Cannot be zero',16,1)

ELSE

BEGIN

SET @Result = @Number1 / @Number2

PRINT 'RESULT IS :'+CAST(@Result AS VARCHAR)

END

END

exec DivideTwoNumber 10,0

**ERROR NUMBER:** The Error number is a unique identifier given for each and every error that occurs in the program. This value will be below 50,000 for predefined errors and must be above 50,000 for error defined by the user.

**ERROR MESSAGE:** It is a brief information describing the error occurred which should be maxing from 2047 characters.

**SEVERITY LEVEL:**

This tells about the importance of the error which can be ranging between 0 to 24. In which

**0 to 9:** are not serves which can be considered as information or status messages.

**11 to 16:**  Indicates these errors can be created by the user.

**17 to 19:** Indicates these are software errors cannot be corrected by the user must be reported to the system administrator.

**20 to 24:** Indicates fatal errors and if these errors occur they can damage the system or database. So here the connection immediately terminates with the database.

**STATE:** It is an arbitrary value which is not that important can be ranging between 0 to 127. We use this whenever the same error has to occur in multiple places.

Exemption Handling Using TRY CATCH

CREATE PROCEDURE DivideTwoNumbers

@Number1 INT,

@Number2 INT

AS

BEGIN

DECLARE @Result INT

SET @Result = 0

BEGIN TRY

SET @Result = @Number1 / @Number2

PRINT 'RESULT IS : '+CAST(@Result AS VARCHAR)

END TRY

BEGIN CATCH

PRINT 'SECOND NUMBER SHOULD NOT BE ZERO'

END CATCH

END

DivideTwoNumbers 100,0

Capturing the Error Message:

GO

ALTER PROCEDURE [dbo].[DivideTwoNumbers]

@Number1 INT,

@Number2 INT

AS

BEGIN

DECLARE @Result INT

SET @Result = 0

BEGIN TRY

SET @Result = @Number1 / @Number2

PRINT 'RESULT IS : '+CAST(@Result AS VARCHAR)

END TRY

BEGIN CATCH

PRINT ERROR\_MESSAGE()

END CATCH

END

Error Handling and Transaction

BEGIN

BEGIN TRY

--Example showing implementation of transaction

BEGIN TRANSACTION

update employee

SET dept\_no ='D1'

WHERE emp\_no = 15000/0; ---FORCED AN ERROR BY DIVIDING BY ZERO

update works\_on

SET project\_no ='p3'

Where emp\_no =15000

and project\_no ='p1'

COMMIT TRANSACTION

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION

SELECT ERROR\_NUMBER() as ErrorNumber,

ERROR\_MESSAGE() as ErrorMessage,

ERROR\_PROCEDURE() as ErrorProcedure,

ERROR\_STATE() as ErrorState,

ERROR\_SEVERITY() as ErrorSeverity,

ERROR\_LINE() as ErrorLine

END CATCH

END