Module 7

Understanding and handling errors

Module Overview

- Understanding errors and throwing error messages
- Implementing error handling

Lesson: Understanding errors and throwing error messages

- Errors and error messages
- Raising errors using RAISERROR
- Raising errors using THROW
- Creating alerts when errors occurs

Errors and error messages

Elements of SQL Server error messages	
Error number	Unique number identifying the error
Error message	Text describing the error
Severity	Severity of error, ranging from 1 to 25
State	Numerical value between 1 and 255. Read documentation for error to see if or what state means.
Procedure	The name of the stored procedure or trigger in which the error occurred
Line number	Which statement in the batch or procedure generated the error. SSMS and ADS replaces the value returned by database engine to match the line number in your query window.

- You can list the system error messages through sys.messages
- You can add your own errors using sp_add_message

Demo Error messages

Errors and Events Reference (Database Engine)
https://docs.microsoft.com/en-us/sql/relational-databases/errors-events/errors-and-events-reference-database-engine

Raising errors using RAISERROR

- The original way to raise an error
- Can use placeholders
- Pass
  - Message or error number
  - Severity level (typically 16)
  - State (typically 1, but your choice)
  - Optionally WITH LOG (requires sysadmins or ALTER TRACE privileges)
  - Optionally WITH NOWAIT

```
--Basic RAISERROR
RAISERROR('Bad santa', 16, 1)

--Using placeholder
RAISERROR ('Bad %s', 16, 1, 'goblin')
```

Demo RAISERROR

RAISERROR (Transact-SQL)
https://docs.microsoft.com/en-us/sql/t-sql/language-elements/raiserror-transact-sql

SQL Server RAISERROR
https://www.sqlservertutorial.net/sql-server-stored-procedures/sql-server-raiserror/

Raising errors using THROW

- SQL Server provides the THROW statement
  - Successor to the RAISERROR statement
  - Previous command has to be semi-colon terminated
  - Less options than RAISERROR

```
;THROW 50001, 'Something bad happened', 1;
```

Demo THROW

THROW (Transact-SQL)
https://docs.microsoft.com/en-us/sql/t-sql/language-elements/throw-transact-sql

SQL Server THROW
https://www.sqlservertutorial.net/sql-server-stored-procedures/sql-server-throw/

Differences Between RAISERROR and THROW in Sql Server
https://sqlhints.com/tag/raiserror-vs-throw/

Creating alerts when errors occurs

- Alerts can be fired by messages that are stored in the Windows log
- If a message is not normally logged, it can be logged when it is raised with the addition of WITH LOG

Alerts
https://docs.microsoft.com/en-us/sql/ssms/agent/alerts

Agent Alerts Management Pack
https://karaszi.com/agent-alerts-management-pack

Lesson: Implementing error handling

- Using @@ERROR
- TRY/CATCH
- Error handling functions
- Catchable vs. noncatchable errors
- Rethrowing errors using THROW

Using @@Error

- @@ERROR returns error number from last executed T-SQL statement
- Can be captured and stored in a variable
- Very old-fashioned way to perform error handling
- Make sure you grab the value *immediately* after the relevant statement
- There is a better way to do error handling
  - TRY/CATCH

Demo @@ERROR

@@ERROR (Transact-SQL)
https://docs.microsoft.com/en-us/sql/t-sql/functions/error-transact-sql

TRY/CATCH

- TRY block defined by BEGIN TRY...END TRY statements
  - Place all code that might raise an error between them
  - No code may be placed between END TRY and BEGIN CATCH
  - TRY and CATCH blocks may be nested
- CATCH block defined by BEGIN CATCH...END CATCH
  - Execution moves to the CATCH block when catchable errors occur within the TRY block

Demo TRY CATCH

TRY...CATCH (Transact-SQL)
https://docs.microsoft.com/en-us/sql/t-sql/language-elements/try-catch-transact-sql

How to implement error handling in SQL Server
https://www.sqlshack.com/how-to-implement-error-handling-in-sql-server/

Error handling functions

- CATCH blocks make the error-related information available throughout the duration of the CATCH block
- @@ERROR is reset when the next statement is run

Demo Error functions

ERROR_NUMBER (Transact-SQL)
https://docs.microsoft.com/en-us/sql/t-sql/functions/error-number-transact-sql

Catchable vs. noncatchable errors

- TRY/CATCH blocks will only catch errors in the same block
- Common examples of noncatchable errors are:
  - Compile errors, such as syntax errors, that prevent a batch from compiling
  - Statement level recompilation issues that usually relate to deferred name resolution

Error and Transaction Handling in SQL Server
https://www.sommarskog.se/error_handling/Part1.html

Rethrowing errors using THROW

- Use THROW without parameters to re-raise a caught error
- Must be within a CATCH block

```
BEGIN TRY
-- The code we want to execute
END TRY
BEGIN CATCH
EXEC dbo.uspLogError --Whatever we want to do
;THROW;
END CATCH;
```

V1 Lab 7: Implementing Error Handling

- ***Note carefully the folder name in the lab instructions!!!***
- Exercise 1: Redirecting Errors with TRY/CATCH
- Exercise 2: Using THROW to Pass an Error Message Back to a Client

Estimated Time: 30 minutes

V2 Lab 7: Understanding and handling errors

- Ex 1. Understanding an error message
- Ex 2. Throwing an error
- Ex 3. If time permits: catching an error and logging it

Estimated Time: 30 minutes