Programming SQL Server Database Triggers and Functions

REUSING CODE WITH FUNCTIONS

Ryan Booz
AUTHOR AND SPEAKER

@ryanbooz https://www.softwareandbooz.com

Overview

What is a Function?

Why are they useful?

Deterministic vs. non-deterministic

Multi-statement vs. Inline Table-Valued

What Are SQL Server Functions?

Functions

Reusable code for querying information or performing repetitive actions on data

More like functional programming than reusable, composable sub-routine methods in Object Oriented languages


MONTH()
2019-01-01
SQL
January

SQL Server Functions

Built-In Server Functions

User-Defined Functions (UDF)

Built-in SQL Server Functions

@@SERVERNAME

CAST()

CURRENT_USER()

DAY()

GETDATE()

SUM()

MONTH()

SQRT()

YEAR()

ROUND()

ABS()

EVENTDATA()

EVENTDATA()

LOWER()

@@MAX_CONNECTIONS

@@CURSOR_ROWS

USER_ID()

CONVERT()

FLOOR()

CEILING()

UPPER()

@@SPID

ORIGINAL_LOGIN()

User-Defined Functions

Created for application specific purposes

Can return single (scalar) value OR table data

They must adhere to specific rules

User-Defined Function Rules

Functions must return a value (scalar or table)

Functions CAN...

- Only take input parameters
- Be used in SELECT statements
- Be used in a JOIN if returns table
- Use table variables
- Use Schema Binding

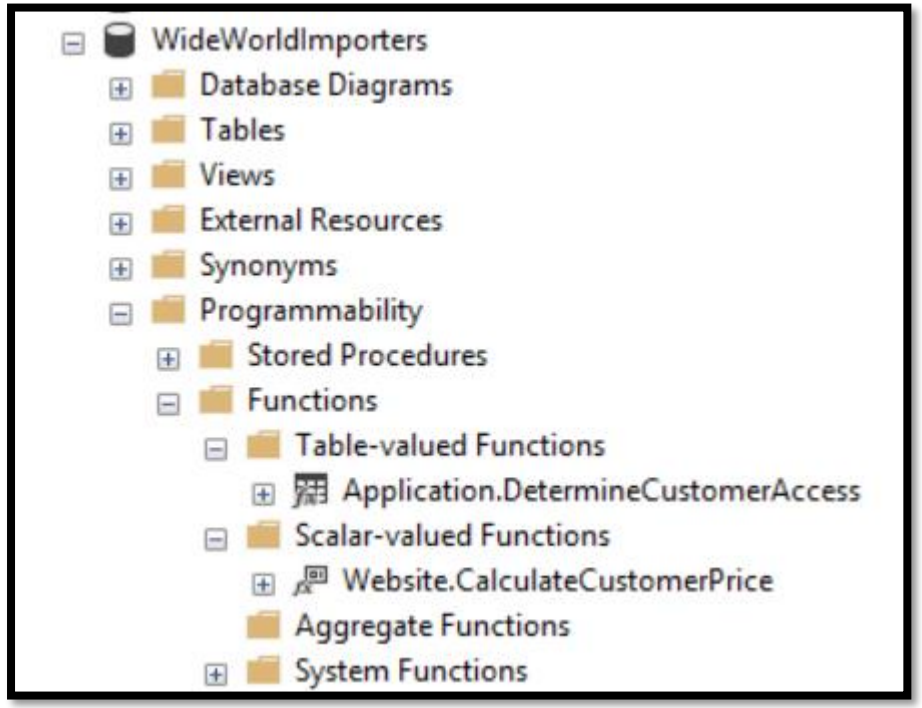
Functions CANNOT...

- Alter the state of the database
- Alter the transaction state
- Use Temporary Tables
- Execute stored procedure
- Use Try/Catch blocks

Functions are NOT catch-all ‘stored procedures’

Located inside “Programmability -> Functions”

Individual folders for each type of function



Why Are Functions Useful?


MONTH()
2019-01-01
SQL
January

FISCALYEAR()

2018-12-31 ----> 

----> 2019

Functions Provide...

Reusable query time logic

Predictable query optimization

Consistent usage and best practices

Functions can generally be used anywhere a scalar value or table would be used

Functions Can Be Used In...

SELECT statements and WHERE clauses

Functions Can Be Used In...

SELECT statements and WHERE clauses

Constraints on a table

Functions Can Be Used In...

SELECT statements and WHERE clauses

Constraints on a table

Computed Columns

Functions Can Be Used In...

SELECT statements and WHERE clauses

Constraints on a table

Computed Columns

JOINs

Functions Can Be Used In...

SELECT statements and WHERE clauses

Constraints on a table

Computed Columns

JOINs

Stored Procedures

Functions Can Be Used In...

SELECT statements and WHERE clauses	Constraints on a table	Computed Columns
JOINs	Stored Procedures	Other Functions


SILVER

Deterministic VS. Non-Deterministic Functions

```sql
SELECT YEAR('2019-01-01'); -- Returns ‘2019’

SELECT YEAR('2019-01-01'); -- Returns ‘2019’

SELECT YEAR('2019-01-01'); -- Returns ‘2019’

SELECT YEAR('2019-01-01'); -- Returns ‘2019’

SELECT YEAR('2019-01-01'); -- Returns ‘2019’
```

Deterministic Functions

Given the same input, deterministic functions will always return the same result

```
SELECT NEWID(); --'9A7C5C73-8EA0-4CAE-8D24-89960EB41072'

SELECT NEWID(); --'03B8B4E6-1B63-482C-AD71-E64ED2F7E212'

SELECT NEWID(); --'7F7DDD36-E96D-4C3F-A15D-F12A70A4C019'

SELECT NEWID(); --'D7124348-075A-4E6F-B313-78349451ED82'

SELECT NEWID(); --'32F7F8D0-5DA7-4FA0-B544-E0D5BBC541F9'
```

Non-Deterministic Functions

Given the same input, non-deterministic functions do not guarantee the same result each time

When creating UDF's, aim to create Deterministic Functions

WITH SCHEMABINDING

Prevents modification of the underlying schema and referenced modules

Cannot call or reference other Non-Deterministic modules

Multi-Statement VS. Inline Table-Valued Functions

Types of Functions

Scaler

Multi-statement Table-Valued

Single-Statement Table-Valued

Simplified Function Types

Multi-Statement Functions

Inline Table-Valued Functions

```
CREATE OR ALTER FUNCTION dbo.AddDemo(@a INT, @b INT)
RETURNS { INT, DATETIME, VARCHAR, Etc. }
AS

BEGIN

    RETURN @a + @b;

END;
```

Multi-Statement Scaler Functions

Any function that has T-SQL between a BEGIN and END, regardless of how many statements there are

Returns a single value of the declared type

```
CREATE OR ALTER FUNCTION dbo.AddDemo(@a INT, @b INT)
RETURNS @SumValueTable TABLE
  ( SumValue INT )
AS
BEGIN
    INSERT INTO @SumValueTable ( SumValue )
        SELECT @a + @b;
    RETURN;
END;
```

Multi-Statement Table Functions

Any function that has T-SQL between a BEGIN and END, regardless of how many statements there are

Returns a table of the specified columns

Use Caution With Multi-Statement Functions

Scalar Functions

Cannot be “in-lined” prior to SQL 2019

Executed Row By Agonizing Row (RBAR)

When the function does non-trivial computation, try to avoid Scalar functions prior to SQL 2019

Table-Valued Functions

Cannot be “in-lined” prior to SQL 2017

Executed once per request

Query Plan Optimizer does not attempt to use any statistics from internal queries

Always estimated at 100 rows (SQL 2014+)

When the function does non-trivial computation, try to avoid Table-Valued functions prior to SQL 2017

```
CREATE OR ALTER FUNCTION dbo.AddDemo(@a INT, @b INT)
RETURNS TABLE
AS
RETURN
(
    SELECT @a + @b AS SumValue
)
```

Inline Table-Valued Functions

Functions that return a table of data from a single T-SQL query

Single-Statement Table-Valued Functions can be directly “in-lined” into the calling query and included in optimization

“Does it really matter if I use one type of function over the other?”

It depends...

```
SELECT orderId
FROM sales.Orders
WHERE dbo.MyMod(orderId,10)=1
```

```
SQL Server Execution Times:
  CPU time = 187 ms,  elapsed time = 298 ms.
```

- Multi-Statement UDF
- Easier to understand

```
SELECT orderId
FROM sales.Orders
CROSS APPLY
dbo.MyMod_tvf(orderId,10) udf
WHERE udf.IsMod = 1;
```

```
SQL Server Execution Times:
  CPU time = 15 ms,  elapsed time = 52 ms.
```

- Inline Table-Valued UDF
- 12X better CPU
- 6X better total time

Comparing Each Function Type

Scalar and Table Multi-Statement

Often easier to implement

Individual plans from queries inside the functions can be cached

Can have very complex logic that's easier to read and understand

Can apply 'SCHEMABINDING' for safety and control

Inline Table-Valued

Usually takes more creative thought to implement similar logic

T-SQL is included ("in-lined") as part of the query optimization

Predictable batch workflow

Can apply 'SCHEMABINDING' for safety and control

SQL Server 2017 and 2019 address most performance concerns pertaining to functions

Summary

- Built-in and User-Defined
- When and why Functions are useful
- Basic rules that Functions must follow
- Deterministic VS. Non-Deterministic
- Scalar, Multi-Statement and Single-statement
- Multi-Statement VS. Inline Table-Valued