Module 10: Implementing User-defined Functions

Overview

- What Is a User-defined Function?
- Defining
- Examples

What Is a User-defined Function?

- **Scalar Functions (do not reference tables)**
  - Similar to a built-in function
- **Multi-Statement Table-valued Functions**
  - Content like a stored procedure
  - Referenced like a view
- **In-Line Table-valued Functions**
  - Similar to a view with parameters
  - Returns a table as the result of single SELECT statement

Defining User-defined Functions

- Creating
- Creating with Schema Binding
- Setting Permissions
- Altering and Dropping

Creating a User-defined Function

- Creating a Function

```
CREATE FUNCTION dbo.fn_NewRegion
    <New function content>
```

```
USE Northwind
CREATE FUNCTION fn_NewRegion
  (@myinput nvarchar(30))
  RETURNS nvarchar(30)
BEGIN
  IF @myinput IS NULL
  SET @myinput = 'Not Applicable'
  RETURN @myinput
END
```

- Restrictions on Functions

Creating a Function with Schema Binding

- Referenced User-defined Functions and Views Are Also Schema Bound
- Objects Are Not Referenced with a Two-Part Name
- Function and Objects Are All in the Same Database
- Have Reference Permission on Required Objects

Setting Permissions for User-defined Functions

- Need CREATE FUNCTION Permission
- Need EXECUTE Permission
- Need REFERENCE Permission on Cited Tables, Views, or Functions
- Must Own the Function to Use in CREATE or ALTER TABLE Statement

Altering and Dropping User-defined Functions

- **Altering Functions**

```
ALTER FUNCTION dbo.fn_NewRegion
   <New function content>
```

  - Retains assigned permissions
  - Causes the new function definition to replace existing definition

- **Dropping Functions**

```
DROP FUNCTION dbo.fn_NewRegion
```

Examples of User-defined Functions

- **Scalar User-defined Function**
  - Usage Example
- **Multi-Statement Table-valued Function**
  - Usage Example
- **In-Line Table-valued Function**
  - Usage Example

Using a Scalar User-defined Function

- RETURNS Clause Specifies Data Type
- Function Is Defined Within a BEGIN and END Block
- Return Type Is Any Data Type Except text, ntext, image, cursor, or timestamp

Example of a Scalar User-defined Function

- **Creating the Function**

```
USE Northwind
CREATE FUNCTION fn_DateFormat
      (@indate datetime, @separator char(1))
  RETURNS Nchar(20)
  AS
  BEGIN
     RETURN
     CONVERT(Nvarchar(20), datepart(mm,@indate))
     + @separator
     + CONVERT(Nvarchar(20), datepart(dd, @indate))
     + @separator
     + CONVERT(Nvarchar(20), datepart(yy, @indate))
  END
```

- **Calling the Function**

```
SELECT dbo.fn_DateFormat(GETDATE(), ':')
```

Using a Multi-Statement Table-valued Function

- BEGIN and END Enclose Multiple Statements
- RETURNS Clause Specifies table Data Type
- RETURNS Clause Names and Defines the Table

Example of a Multi-Statement Table-valued Function

- **Creating the Function**

```
USE Northwind
GO
CREATE FUNCTION fn_Employees (@length nvarchar(9))
RETURNS @fn_Employees table
   (EmployeeID int PRIMARY KEY NOT NULL,
   [Employee Name] nvarchar(61) NOT NULL)
AS
BEGIN
   IF @length = 'ShortName'
      INSERT @fn_Employees SELECT EmployeeID, LastName
      FROM Employees
   ELSE IF @length = 'LongName'
      INSERT @fn_Employees SELECT EmployeeID,
      (FirstName + ' ' + LastName) FROM Employees
RETURN
END
```

- **Calling the Function**

```
SELECT * FROM dbo.fn_Employees('LongName')
Or
SELECT * FROM dbo.fn_Employees('ShortName')
```

Using an In-Line Table-valued Function

- Content of the Function Is a SELECT Statement
- Do Not Use BEGIN and END
- RETURN Specifies table as the Data Type
- Format Is Defined by the Result Set

Example of an In-Line Table-valued Function

- **Creating the Function**

```
USE Northwind
GO
CREATE FUNCTION fn_CustomerNamesInRegion
   ( @RegionParameter nvarchar(30) )
RETURNS table
AS
RETURN (
   SELECT CustomerID, CompanyName
   FROM Northwind.dbo.Customers
   WHERE Region = @RegionParameter
   )
```

- **Calling the Function Using a Parameter**

```
SELECT * FROM fn_CustomerNamesInRegion('WA')
```

Recommended Practices

Use Complex Scalar Functions on Small Result Sets

Use Multi-Statement Functions Instead of Stored Procedures That Return Tables

Use In-Line Functions to Create Parameterized Views

Use In-Line Functions to Filter Indexed Views

Review

- What Is a User-defined Function?
- Defining
- Examples