Name: Brian Moran (bmoran)
Date: 8/27/25
Course: IT Foundations of Database Management
Github Repo: https://github.com/bmoran22/DBFoundations-Module07

Module 7: SQL Functions

Introduction:

In SQL programming, functions play a vital role in simplifying complex queries, enhancing code reusability, and improving data processing efficiency. While built-in functions are commonly used for tasks like date manipulation, aggregation, and string formatting, user-defined functions (UDFs) allow developers to create custom logic tailored to specific business needs. This paper explores the use of SQL UDFs and the distinctions between scalar, inline table-valued, and multi-statement table-valued functions.Explain

When you would use a SQL UDF.

A SQL User-Defined Function (UDF) is ideal when you need to encapsulate reusable logic that can be applied across multiple queries. UDFs are particularly useful in scenarios such as:

- Custom calculations: For example, adding two values using dbo.AddValues() simplifies repetitive arithmetic logic.
- Data validation: Functions like IsDate() or custom constraints can validate inputs, such as ensuring a meeting signup date is before the meeting start time.
- ETL processing: UDFs help extract and transform data, such as separating names or formatting phone numbers.
- Reporting: Wrapping logic into functions makes reporting queries cleaner and more maintainable, especially when using advanced functions like LAG() or LEAD() like was showcased within the homework problems.

Unlike views, UDFs can accept parameters, making them more flexible for dynamic data filtering and transformation.

Explain the differences between Scalar, Inline, and Multi-Statement Functions

<t

SQL UDFs come in three main types, each serving different purposes:

1. **Scalar Functions**
    a. Return a single value (e.g., a number or string).
    b. Syntax includes CREATE FUNCTION with parameters and a RETURNS clause.
    c. Called using SELECT dbo.FunctionName(parameters).
    d. Example: dbo.AddValues(4,5) returns the sum of two floats.
2. **Inline Table-Valued Functions (TVFs)**
    a. Return a table directly from a single SELECT statement
    b. Do not require BEGIN and END.
    c. Syntax includes RETURNS TABLE AS RETURN (SELECT ...).
    d. Example: dbo.FilmsInYear(2001) returns all films released in 2001.
3. **Multi-Statement Table-Valued Functions**
    a. Allow multiple statements to build a result set.
    b. Require a defined table variable (RETURNS @t TABLE) and use BEGIN and END.
    c. Useful for combining data from multiple sources or applying complex logic before returning results.

Each type has its strengths: scalar functions are simple and fast, inline TVFs are efficient for straightforward table queries, and multi-statement TVFs offer flexibility for more complex operations

Summary:

SQL UDFs empower developers to write cleaner, reusable, and more maintainable code. Whether you're performing simple calculations, building dynamic reports, or validating data, choosing the right type of function—scalar, inline, or multi-statement—depends on the complexity and structure of the task. Understanding these distinctions helps streamline SQL development and enhances the overall efficiency of database operations.