

Introduction

SQL User-Defined Functions (UDFs) allow developers to package logic that needs to be reused throughout a database. Instead of repeating calculations, formatting rules, or filtering expressions in multiple queries, a UDF centralizes that logic in one place. Because they behave like built-in SQL functions, they can be used naturally within SELECT statements and joins. This makes queries easier to read, ensures consistency, and makes the system easier to maintain when business rules change.

1. When You Would Use a SQL UDF

A SQL UDF is useful whenever the same logic appears in multiple parts of a database. This includes repeated calculations (such as KPIs, differences, percentages, or formatted dates), filtering rules that must be applied consistently, or complex expressions that make queries harder to read. By placing this logic into a UDF, changes only need to be made once, instead of editing several views, queries, or reports. UDFs also help standardize business rules so that every part of the system interprets data the same way. Table-valued UDFs are particularly helpful when you want to return a parameterized result set, such as returning only inventory rows that match a specific KPI. In short, UDFs improve clarity, reuse, and consistency across the database.

2. Differences Between Scalar, Inline, and Multi-Statement Functions

SQL Server supports three types of UDFs, each suited for different purposes. A **scalar function** returns a single value, such as a number or string, and is best for simple operations that act on one piece of data. They are easy to use but can slow down large queries because they execute once per row.

An **inline table-valued function (iTVF)** returns a table based on a single SELECT statement. It functions like a parameterized view and is usually the most efficient type of UDF because SQL Server can optimize it fully. Inline TVFs are ideal when you want to return a filtered or computed set of rows, such as returning only the inventory changes with KPI values of 1, 0, or -1.

A **multi-statement table-valued function (mTVF)** also returns a table, but it is built using multiple SQL statements. This type allows for more complex or procedural logic, such as inserting rows in stages or branching based on conditions. However, because these functions rely on internal table variables, they tend to perform slower and are less easily optimized.

Conclusion

User-Defined Functions are an essential part of writing clean, maintainable SQL. They allow developers to consolidate repeated logic, support consistent business rules, and simplify complex queries. Scalar functions work best for individual value calculations, inline table-valued functions provide efficient and reusable table results, and multi-statement functions offer flexibility when multiple steps are required. Choosing the right type of UDF helps ensure a database is efficient, organized, and easier to maintain as requirements evolve.