GENERATE\_SERIES (Transact-SQL)

Applies to:  SQL Server 2022 (16.x)  Azure SQL Database  Azure SQL Managed Instance

Generates a series of numbers within a given interval. The interval and the step between series values are defined by the user.

Syntax:-

GENERATE\_SERIES ( start, stop [, step ] )

Arguments

*start*

The first value in the interval. *start* is specified as a variable, a literal, or a scalar [expression](https://learn.microsoft.com/en-us/sql/t-sql/language-elements/expressions-transact-sql?view=sql-server-ver16) of type **tinyint**, **smallint**, **int**, **bigint**, **decimal**, or **numeric**.

*stop*

The last value in the interval. *stop* is specified as a variable, a literal, or a scalar [expression](https://learn.microsoft.com/en-us/sql/t-sql/language-elements/expressions-transact-sql?view=sql-server-ver16) of type **tinyint**, **smallint**, **int**, **bigint**, **decimal**, or **numeric**. The series stops once the last generated step value exceeds the *stop* value.

The data type for *stop* **must** match the data type for *start*.

*[ step ]*

Indicates the number of values to increment or decrement between steps in the series. *step* is an [expression](https://learn.microsoft.com/en-us/sql/t-sql/language-elements/expressions-transact-sql?view=sql-server-ver16) of type **tinyint**, **smallint**, **int**, **bigint**, **decimal**, or **numeric**. *step* can be either negative or positive, but can't be zero (0).

This argument is optional. The default value for *step* is 1 if *start* is less than *stop*, otherwise, the default value is -1 if *start* is greater than *stop*.

If *start* is less than *stop* and a negative value is specified for *step*, or if *start* is greater than *stop* and a positive value is specified for *step*, an empty result set is returned.

Return types

Returns a single-column table containing a sequence of values in which each differs from the preceding by *step*. The name of the column is value. The output is the same type as *start* and *stop*.

Permissions

No permissions are required for GENERATE\_SERIES. However, the user needs EXECUTE permission on the database, and permission to query any data that is used as inputs.

Examples

The following examples demonstrate the syntax for calling GENERATE\_SERIES.

A. Generate a series of integer values between 1 and 10 in increments of 1 (default)

SQLCopy

SELECT value

FROM GENERATE\_SERIES(1, 10);

Here's the result set.

OutputCopy

value

-----------

1

2

3

4

5

6

7

8

9

10

B. Generate a series of integer values between 1 and 50 in increments of 5

SQLCopy

SELECT value

FROM GENERATE\_SERIES(1, 50, 5);

Here's the result set.

OutputCopy

value

-----------

1

6

11

16

21

26

31

36

41

46

C. Generate a series of decimal values between 0.0 and 1.0 in increments of 0.1

SQLCopy

DECLARE @start decimal(2, 1) = 0.0;

DECLARE @stop decimal(2, 1) = 1.0;

DECLARE @step decimal(2, 1) = 0.1;

SELECT value

FROM GENERATE\_SERIES(@start, @stop, @step);

Here's the result set.

OutputCopy

value

---------------------------------------

0.0

0.1

0.2

0.3

0.4

0.5

0.6

0.7

0.8

0.9

1.0