What is a table-valued function in SQL Server

A table-valued function is a [user-defined function](https://www.sqlservertutorial.net/sql-server-user-defined-functions/) that returns data of a table type. The return type of a table-valued function is a table, therefore, you can use the table-valued function just like you would use a table.

Creating a table-valued function

The following statement example creates a table-valued function that returns a list of products including product name, model year and the list price for a specific model year:

CREATE FUNCTION udfProductInYear (

@model\_year INT

)

RETURNS TABLE

AS

RETURN

SELECT

product\_name,

model\_year,

list\_price

FROM

production.products

WHERE

model\_year = @model\_year;

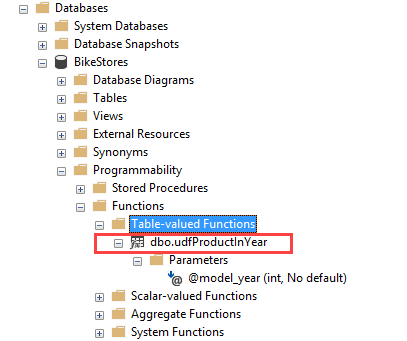
Code language: SQL (Structured Query Language) (sql)

The syntax is similar to the one that creates a user-defined function.

The RETURNS TABLE specifies that the function will return a table. As you can see, there is no BEGIN...END statement. The statement simply queries data from the production.products table.

The udfProductInYear function accepts one parameter named @model\_year of type [INT](https://www.sqlservertutorial.net/sql-server-basics/sql-server-int/). It returns the products whose model years equal @model\_year parameter.

Once the table-valued function is created, you can find it under **Programmability > Functions > Table-valued Functions** as shown in the following picture:



The function above returns the result set of a single [SELECT](https://www.sqlservertutorial.net/sql-server-basics/sql-server-select/) statement, therefore, it is also known as an inline table-valued function.

Executing a table-valued function

To execute a table-valued function, you use it in the FROM clause of the SELECT statement:

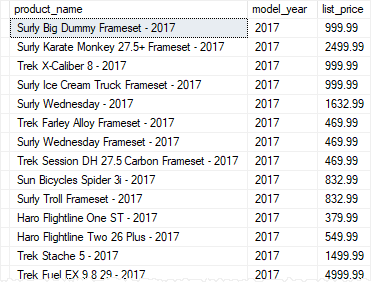
SELECT

\*

FROM

udfProductInYear(2017);

Code language: SQL (Structured Query Language) (sql)



In this example, we selected the products whose model year is 2017.

You can also specify which columns to be returned from the table-valued function as follows:

SELECT

product\_name,

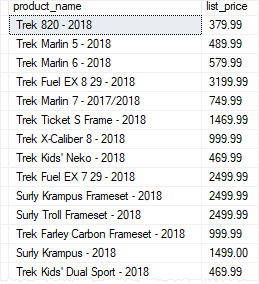
list\_price

FROM

udfProductInYear(2018);

Code language: SQL (Structured Query Language) (sql)

Here is the partial output:



Modifying a table-valued function

To modify a table-valued function, you use the ALTER instead of CREATE keyword. The rest of the script is the same.

For example, the following statement modifies the udfProductInYear by changing the existing parameter and adding one more parameter:

ALTER FUNCTION udfProductInYear (

@start\_year INT,

@end\_year INT

)

RETURNS TABLE

AS

RETURN

SELECT

product\_name,

model\_year,

list\_price

FROM

production.products

WHERE

model\_year BETWEEN @start\_year AND @end\_year

Code language: SQL (Structured Query Language) (sql)

The udfProductInYear function now returns products whose model year between a starting year and an ending year.

The following statement calls the udfProductInYear function to get the products whose model years are between 2017 and 2018:

SELECT

product\_name,

model\_year,

list\_price

FROM

udfProductInYear(2017,2018)

ORDER BY

product\_name;

Code language: SQL (Structured Query Language) (sql)

Here is the partial output:



Multi-statement table-valued functions (MSTVF)

A multi-statement table-valued function or MSTVF is a table-valued function that returns the result of multiple statements.

The multi-statement-table-valued function is very useful because you can execute multiple queries within the function and aggregate results into the returned table.

To define a multi-statement table-valued function, you use a table variable as the return value. Inside the function, you execute one or more queries and insert data into this table variable.

The following udfContacts() function combines staffs and customers into a single contact list:

CREATE FUNCTION udfContacts()

RETURNS @contacts TABLE (

first\_name VARCHAR(50),

last\_name VARCHAR(50),

email VARCHAR(255),

phone VARCHAR(25),

contact\_type VARCHAR(20)

)

AS

BEGIN

INSERT INTO @contacts

SELECT

first\_name,

last\_name,

email,

phone,

'Staff'

FROM

sales.staffs;

INSERT INTO @contacts

SELECT

first\_name,

last\_name,

email,

phone,

'Customer'

FROM

sales.customers;

RETURN;

END;

Code language: SQL (Structured Query Language) (sql)

The following statement illustrates how to execute a multi-statement table-valued function udfContacts:

SELECT

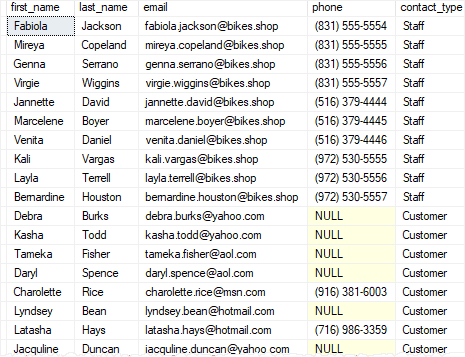
\*

FROM

udfContacts();

Code language: SQL (Structured Query Language) (sql)

Output:



When to use table-valued functions

We typically use table-valued functions as parameterized [views](https://www.sqlservertutorial.net/sql-server-views/). In comparison with [stored procedures](https://www.sqlservertutorial.net/sql-server-stored-procedures/), the table-valued functions are more flexible because we can use them wherever tables are used.