[**Are SQL Server Functions Dragging Your Query Down?**](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/)

by [Jes Schultz Borland](http://www.brentozar.com/archive/author/jes/) October 14, 2014

[15 comments](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/#comments)

FROM: <http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/?utm_source=feedburner&utm_medium=feed&utm_campaign=Feed%3A+BrentOzar-SqlServerDba+%28Brent+Ozar+Unlimited%29>

In most coding languages, functions are often-used blocks of code that can be reused from multiple locations, leading to less code – and cleaner code. SQL Server also lets us create functions that can be used the same way. They are reusable blocks of code that can be called from multiple locations. So, if you need to format phone numbers a certain way, or parse for specific characters, you can do so using a function.

The question is, how much work is SQL Server doing when you call a function? If it’s the SELECT clause, is it called once – processing all rows – or once for each row in the result set, regardless if that’s 1 row or 100,00? What if it’s in the WHERE clause?

I’ll let you in on a little secret: if a function is used in the SELECT or WHERE, the function can be called many, many times. If the function is very resource-intensive, it could be causing your query to be very slow – and you would never see the execution of the function within the execution plan of the calling query.

Yep, SQL Server’s execution plans can be a bit vague when it comes to functions – and by “a bit vague”, I mean, “They don’t show up at all”. You need to dig deeper!

I’m going to run a few demos against the [AdventureWorks2012](http://msftdbprodsamples.codeplex.com/releases/view/55330) sample database in a SQL Server 2014 instance to show this!

First, I create a scalar-value function that will return the five left-most letters of a LastName.

CREATE FUNCTION [dbo].[ParseLastName](@LastName VARCHAR(50))

RETURNS VARCHAR(5)

AS

-- Returns the 5 left characters of the last name

BEGIN

DECLARE @ret VARCHAR(5);

SET @ret =

LEFT(@LastName, 5)

RETURN @ret

END;

1. CREATE FUNCTION [dbo].[ParseLastName](@LastName VARCHAR(50))
2. RETURNS VARCHAR(5)
3. AS
4. -- Returns the 5 left characters of the last name
5. BEGIN
6. DECLARE @ret VARCHAR(5);
7. SET @ret =
8. LEFT(@LastName, 5)
9. RETURN @ret
10. END;

CREATE FUNCTION [dbo].[ParseLastName](@LastName VARCHAR(50))  
RETURNS VARCHAR(5)  
AS  
-- Returns the 5 left characters of the last name  
BEGIN  
DECLARE @ret VARCHAR(5);  
  
SET @ret =  
LEFT(@LastName, 5)  
  
RETURN @ret  
END;

Then, I create an Extended Events session to track statement completion. (Note: I have only tested this on SQL Server 2014, no lower versions.) (Using SQL Server 2008 R2 or earlier? You could create a server-side trace to capture sp\_statement\_completed and sql\_statement\_completed, but it won’t give you some functionality I’ll show later.)

CREATE EVENT SESSION [CaptureFunctionExecutions] ON SERVER

ADD EVENT sqlserver.sp\_statement\_completed(

ACTION(sqlserver.sql\_text,sqlserver.tsql\_stack)),

ADD EVENT sqlserver.sql\_statement\_completed(

ACTION(sqlserver.sql\_text,sqlserver.tsql\_stack))

ADD TARGET package0.ring\_buffer

WITH (MAX\_MEMORY=4096 KB,EVENT\_RETENTION\_MODE=ALLOW\_SINGLE\_EVENT\_LOSS,MAX\_DISPATCH\_LATENCY=30 SECONDS,MAX\_EVENT\_SIZE=0 KB,MEMORY\_PARTITION\_MODE=NONE,TRACK\_CAUSALITY=OFF,STARTUP\_STATE=OFF)

GO

1. CREATE EVENT SESSION [CaptureFunctionExecutions] ON SERVER
2. ADD EVENT sqlserver.sp\_statement\_completed(
3. ACTION(sqlserver.sql\_text,sqlserver.tsql\_stack)),
4. ADD EVENT sqlserver.sql\_statement\_completed(
5. ACTION(sqlserver.sql\_text,sqlserver.tsql\_stack))
6. ADD TARGET package0.ring\_buffer
7. WITH (MAX\_MEMORY=4096 KB,EVENT\_RETENTION\_MODE=ALLOW\_SINGLE\_EVENT\_LOSS,MAX\_DISPATCH\_LATENCY=30 SECONDS,MAX\_EVENT\_SIZE=0 KB,MEMORY\_PARTITION\_MODE=NONE,TRACK\_CAUSALITY=OFF,STARTUP\_STATE=OFF)
8. GO

CREATE EVENT SESSION [CaptureFunctionExecutions] ON SERVER  
ADD EVENT sqlserver.sp\_statement\_completed(  
ACTION(sqlserver.sql\_text,sqlserver.tsql\_stack)),  
ADD EVENT sqlserver.sql\_statement\_completed(  
ACTION(sqlserver.sql\_text,sqlserver.tsql\_stack))  
ADD TARGET package0.ring\_buffer  
WITH (MAX\_MEMORY=4096 KB,EVENT\_RETENTION\_MODE=ALLOW\_SINGLE\_EVENT\_LOSS,MAX\_DISPATCH\_LATENCY=30 SECONDS,MAX\_EVENT\_SIZE=0 KB,MEMORY\_PARTITION\_MODE=NONE,TRACK\_CAUSALITY=OFF,STARTUP\_STATE=OFF)  
  
GO

I start the Extended Events session, and then turn on actual execution plans.

I start with a simple query, which returns 19,972 rows.

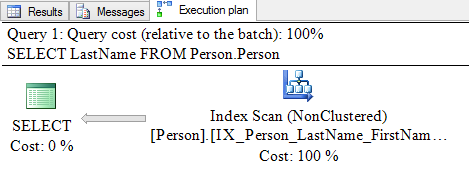
SELECT LastName

FROM Person.Person;

1. SELECT LastName
2. FROM Person.Person;

SELECT LastName  
FROM Person.Person;

The execution plan shows an index scan and has a cost of 0.10451.

[](http://www.brentozar.com/wp-content/uploads/2014/10/function-1.png)

Looking at the details of the index scan, I see Estimated Number of Executions is 1, and Number of Executions is 1.

Let’s look at the same query when it performs the same calculation as the function – LEFT(LastName, 5).

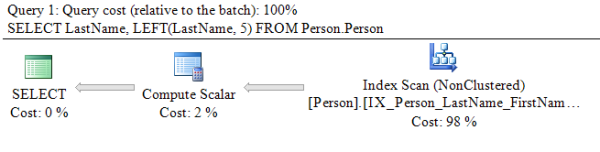
SELECT LastName, LEFT(LastName, 5)

FROM Person.Person;

1. SELECT LastName, LEFT(LastName, 5)
2. FROM Person.Person;

SELECT LastName, LEFT(LastName, 5)  
FROM Person.Person;

There’s now an additional operator – a compute scalar. The cost has risen slightly to 0.106508.

[](http://www.brentozar.com/wp-content/uploads/2014/10/function-2.png)

Now, I will modify the query to call the function from the SELECT clause.

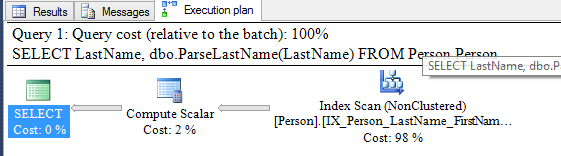
SELECT LastName, dbo.ParseLastName(LastName)

FROM Person.Person;

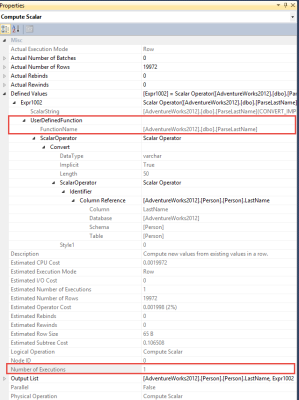
1. SELECT LastName, dbo.ParseLastName(LastName)
2. FROM Person.Person;

SELECT LastName, dbo.ParseLastName(LastName)  
FROM Person.Person;

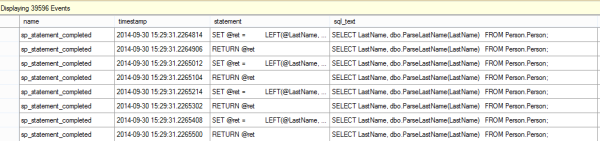
Looking at the execution plan, I see an index scan and a compute scalar. The cost is the same as before –  0.106508.

[](http://www.brentozar.com/wp-content/uploads/2014/10/function-3.png)

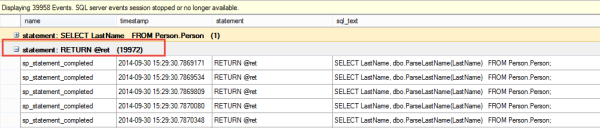
Expanding the properties for the compute scalar, I see the function, but it says there is only one execution.

[](http://www.brentozar.com/wp-content/uploads/2014/10/function-4.png)

A quick glance at my Extended Events live feed tells a different story.

[](http://www.brentozar.com/wp-content/uploads/2014/10/function-5.png)

If I add grouping by statement, I can see the function was actually executed 19,972 times – once for each row in the result set.

[](http://www.brentozar.com/wp-content/uploads/2014/10/function-6.png)

That’s a lot more work than advertised!

Does the same thing happen if the function is in the WHERE clause?

SELECT FirstName, LastName

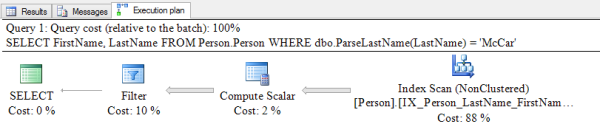
FROM Person.Person

WHERE dbo.ParseLastName(LastName) = 'McCar';

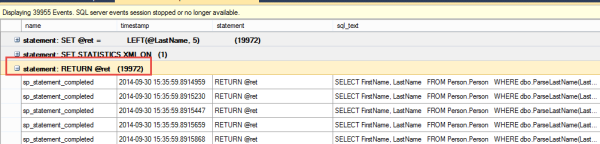
1. SELECT FirstName, LastName
2. FROM Person.Person
3. WHERE dbo.ParseLastName(LastName) = 'McCar';

SELECT FirstName, LastName  
FROM Person.Person  
WHERE dbo.ParseLastName(LastName) = 'McCar';

Two rows are returned. The execution plan now has an index scan, a compute scalar, and a filter. The cost is 0.118091.

[](http://www.brentozar.com/wp-content/uploads/2014/10/function-7.png)

The Extended Events session again shows 19,972 executions – once for each row in the index.

[](http://www.brentozar.com/wp-content/uploads/2014/10/function-8.png)

The data isn’t filtered out until after the function is called, so it is executed once for each row.

**Conclusion**

These examples prove that whether one or many rows are returned as the query result set, if a function is used in the SELECT or WHERE, the function can be called many, many times. It could be one of the top resource-consuming queries in your server!

How can you see if a function is bringing your server’s performance down? Look at the top queries in your plan cache using our [sp\_BlitzCache](http://www.brentozar.com/blitzcache/) tool, by total CPU and by number of executions, to see if this is happening to you.

[↑ Back to top](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/?utm_source=feedburner&utm_medium=feed&utm_campaign=Feed%3A+BrentOzar-SqlServerDba+%28Brent+Ozar+Unlimited%29#top)

15 comments

[Tweet](https://twitter.com/share)

[Leave a comment](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/#comments)

1. http://1.gravatar.com/avatar/a3caa97983786e8c6b7bb03265bf42d6?s=50&d=mm&r=g

*Ranjeet* [October 14, 2014 | 11:39 am](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/#comment-1280387)

I do understand Scalar functions can be very expensive depending on how you use it and how many rows it is returning.

But what if we modify the same function to return as a table and use it OUTER APPLY.

Then how will it perform considering the same resultselt ? I am curious to know your views on this.

EG:  
–Modified Function  
CREATE FUNCTION [dbo].[ParseLastNameT](@LastName VARCHAR(50))  
RETURNS TABLE  
AS  
RETURN  
(  
SELECT ‘LastName’=CAST(LEFT(@LastName, 5) AS VARCHAR(5))  
)

–Query  
SELECT A.LastName,fnParseLastName.LastName  
FROM  
Person.Person A  
OUTER APPLY  
dbo.ParseLastNameT(A.LastName) AS fnParseLastName

[Reply](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/?replytocom=1280387#respond)

* + http://1.gravatar.com/avatar/77f776c2eaf0cc691e8a0880bb8a191f?s=50&d=mm&r=g

[*Brent Ozar*](http://www.brentozar.com) [October 14, 2014 | 11:41 am](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/#comment-1280393)

Ranjeet – I’m a big fan of running experiments. If you wonder what performance of an approach will be, why not try experiments on your own system? You’ll get first-hand experience learning how things work in your own environment. Enjoy!

[Reply](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/?replytocom=1280393#respond)

* + - http://1.gravatar.com/avatar/a3caa97983786e8c6b7bb03265bf42d6?s=50&d=mm&r=g

*Ranjeet C* [October 14, 2014 | 11:45 am](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/#comment-1280410)

Thanks Brent , I will definately try as this has increased my curosity and will find out but I just wanted to know your views. But I do appreciate your articles and it has taught me lot. 

[Reply](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/?replytocom=1280410#respond)

* + - * http://1.gravatar.com/avatar/77f776c2eaf0cc691e8a0880bb8a191f?s=50&d=mm&r=g

[*Brent Ozar*](http://www.brentozar.com) [October 14, 2014 | 11:52 am](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/#comment-1280444)

Great, glad I could help!

[Reply](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/?replytocom=1280444#respond)

1. http://1.gravatar.com/avatar/715c7b833206950dd5cbf19b31bde5d5?s=50&d=mm&r=g

[*Dave Clary*](http://thesqlnomad.com) [October 14, 2014 | 1:38 pm](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/#comment-1280921)

Jess, will there be a followup to this article? Specifically: alternatives to the inline scalar function, or tips and tricks for performance tuning them?

[Reply](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/?replytocom=1280921#respond)

* + http://1.gravatar.com/avatar/d2c1da1e2fb5d1324bcb9c452400c49a?s=50&d=mm&r=g

*Jes Schultz Borland* [October 14, 2014 | 2:49 pm](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/#comment-1281210)

Dave, if you’re looking for more info on functions, you should check out this excellent article Jeremiah wrote for simple-talk a while back: <https://www.simple-talk.com/sql/t-sql-programming/sql-server-functions-the-basics/>.

[Reply](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/?replytocom=1281210#respond)

1. http://2.gravatar.com/avatar/e42682123ebf6e51ac9620ccf914e27f?s=50&d=mm&r=g

*Daniil S* [October 14, 2014 | 4:21 pm](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/#comment-1281596)

I’m not surprised by this at all. In fact, to me this is as obvious a result as it can be. Now if SQL called your function more times than you had rows via WHERE, then it’d be more interesting.

[Reply](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/?replytocom=1281596#respond)

1. http://0.gravatar.com/avatar/3f94bdab0183bd3250f99ff1ebda03ad?s=50&d=mm&r=g

*Paul* [October 16, 2014 | 3:48 am](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/#comment-1291744)

This is good to know and quite interesting. But I feel like you’ve shown me a problem without suggesting an answer.

[Reply](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/?replytocom=1291744#respond)

* + http://2.gravatar.com/avatar/2d6e491fa773660aff4414c84be25579?s=50&d=mm&r=g

[*Adam Machanic*](http://sqlblog.com) [October 16, 2014 | 12:57 pm](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/#comment-1294647)

Here’s one possible solution:

<http://sqlblog.com/blogs/adam_machanic/archive/2006/08/04/scalar-functions-inlining-and-performance-an-entertaining-title-for-a-boring-post.aspx>

[Reply](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/?replytocom=1294647#respond)

1. http://0.gravatar.com/avatar/6a3fd27f1cf0f8e8c994be560e7db26a?s=50&d=mm&r=g

*Darek* [October 16, 2014 | 5:30 am](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/#comment-1292231)

Hi there. As Daniil S noticed it’s not at all surprising that a scalar function is called as many times as there are rows flowing. In what other way would SQL ensure that a value is being calculated for the rows? THERE IS NONE OTHER WAY. It’s true for any function that has to CALCULATE (not retrieve) a value based on the values in any particular row.

[Reply](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/?replytocom=1292231#respond)

1. Pingback: [(SFTW) SQL Server Links 17/10/14 - John Sansom](http://www.johnsansom.com/sftw-sql-server-links-171014/)
2. http://2.gravatar.com/avatar/b54c5a839222bd0b5e38ae5aac566fdc?s=50&d=mm&r=g

*Joe Celko* [October 18, 2014 | 8:10 pm](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/#comment-1314105)

Being a SQL purist, I do not want user functions at all. This is more than the awful performance, but the basic model for a declarative language. In SQL/PSM, I can declare a module to be DETERMINISTIC, modifies SQL data, etc, so the SQL engine can “do stuff” with it. I do not have that in SQL Server so I am screwed and have to assume the worst.

[Reply](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/?replytocom=1314105#respond)

* + http://1.gravatar.com/avatar/77f776c2eaf0cc691e8a0880bb8a191f?s=50&d=mm&r=g

[*Brent Ozar*](http://www.brentozar.com) [October 19, 2014 | 9:35 am](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/#comment-1319126)

Joe – that’s an interesting take. Out of curiosity, what other things would you remove from the SQL Server engine if you could?

[Reply](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/?replytocom=1319126#respond)

1. Pingback: [The Week That Was – Week Ending 18th Oct 2014 | Practical Performance Analyst](http://www1.practicalperformanceanalyst.com/2014/10/21/the-week-that-was-week-ending-18th-oct-2014/)
2. http://1.gravatar.com/avatar/75de267745397bb9a896c301df80c9d3?s=50&d=mm&r=g

*Doug* [May 29, 2015 | 3:49 pm](http://www.brentozar.com/archive/2014/10/sql-server-functions-dragging-query/#comment-2189200)

Just came across this. I know it was posted some time back, but I was wondering what happens if you have a scenario like:

CREATE FUNCTION dbo.test (@date AS DATETIME, @days INT) RETURNS DATETIME  
AS  
BEGIN  
RETURN  
(  
SELECT DATEADD(DD, @days, @date)  
)  
END

and use that in a WHERE clause

SELECT …  
FROM …  
WHERE ship\_date > dbo.test(getdate(), 10)