<http://msdn.microsoft.com/en-us/library/ms191275.aspx>

# Specify Query Parameterization Behavior by Using Plan Guides

**SQL Server 2012**

[Other Versions](javascript:;)

http://i3.msdn.microsoft.com/Areas/Epx/Content/Images/ImageSprite.png

* [SQL Server 2008 R2](http://msdn.microsoft.com/en-us/library/ms191275(v=sql.105).aspx)
* [SQL Server 2008](http://msdn.microsoft.com/en-us/library/ms191275(v=sql.100).aspx)
* [SQL Server 2005](http://msdn.microsoft.com/en-us/library/ms191275(v=sql.90).aspx)

0 out of 1 rated this helpful - [Rate this topic](http://msdn.microsoft.com/en-us/library/ms191275.aspx#feedback)

When the PARAMETERIZATION database option is set to SIMPLE, the SQL Server query optimizer may choose to parameterize the queries. This means that any literal values that are contained in a query are substituted with parameters. This process is referred to as simple parameterization. When SIMPLE parameterization is in effect, you cannot control which queries are parameterized and which queries are not. However, you can specify that all queries in a database be parameterized by setting the PARAMETERIZATION database option to FORCED. This process is referred to as forced parameterization.

You can override the parameterization behavior of a database by using plan guides in the following ways:

* When the PARAMETERIZATION database option is set to SIMPLE, you can specify that forced parameterization is attempted on a certain class of queries. You do this by creating a TEMPLATE plan guide on the parameterized form of the query, and specifying the PARAMETERIZATION FORCED query hint in the [sp\_create\_plan\_guide](http://msdn.microsoft.com/en-us/library/ms179880.aspx) stored procedure. You can consider this kind of plan guide as a way to enable forced parameterization only on a certain class of queries, instead of all queries.
* When the PARAMETERIZATION database option is set to FORCED, you can specify that for a certain class of queries, only simple parameterization is attempted, not forced parameterization. You do this by creating a TEMPLATE plan guide on the force-parameterized form of the query, and specifying the PARAMETERIZATION SIMPLE query hint in **sp\_create\_plan\_guide**.

Consider the following query on the AdventureWorks2012 database:

[Copy](javascript:if%20(window.epx.codeSnippet)window.epx.codeSnippet.copyCode('CodeSnippetContainerCode_941a95b6-62cb-4c98-b79b-b0d7b7439d28');" \o "Copy to clipboard.)

SELECT pi.ProductID, SUM(pi.Quantity) AS Total

FROM Production.ProductModel AS pm

INNER JOIN Production.ProductInventory AS pi

ON pm.ProductModelID = pi.ProductID

WHERE pi.ProductID = 101

GROUP BY pi.ProductID, pi.Quantity HAVING SUM(pi.Quantity) > 50;

As a database administrator, you have determined that you do not want to enable forced parameterization on all queries in the database. However, you do want to avoid compilation costs on all queries that are syntactically equivalent to the previous query, but differ only in their constant literal values. In other words, you want the query to be parameterized so that a query plan for this kind of query is reused. In this case, complete the following steps:

1. Retrieve the parameterized form of the query. The only safe way to obtain this value for use in **sp\_create\_plan\_guide** is by using the [sp\_get\_query\_template](http://msdn.microsoft.com/en-us/library/ms186908.aspx) system stored procedure.
2. Create the plan guide on the parameterized form of the query, specifying the PARAMETERIZATION FORCED query hint.

|  |
| --- |
| **Important noteImportant** |
| As part of parameterizing a query, SQL Server assigns a data type to the parameters that replace the literal values, depending on the value and size of the literal. The same process occurs to the value of the constant literals passed to the **@stmt** output parameter of **sp\_get\_query\_template**. Because the data type specified in the **@params** argument of **sp\_create\_plan\_guide** must match that of the query as it is parameterized by SQL Server, you may have to create more than one plan guide to cover the complete range of possible parameter values for the query. |

The following script can be used both to obtain the parameterized query and then create a plan guide on it:

[Copy](javascript:if%20(window.epx.codeSnippet)window.epx.codeSnippet.copyCode('CodeSnippetContainerCode_9229ad2b-211e-4ca4-9315-9fd520dca8d1');)

DECLARE @stmt nvarchar(max);

DECLARE @params nvarchar(max);

EXEC sp\_get\_query\_template

N'SELECT pi.ProductID, SUM(pi.Quantity) AS Total

FROM Production.ProductModel AS pm

INNER JOIN Production.ProductInventory AS pi ON pm.ProductModelID = pi.ProductID

WHERE pi.ProductID = 101

GROUP BY pi.ProductID, pi.Quantity

HAVING sum(pi.Quantity) > 50',

@stmt OUTPUT,

@params OUTPUT;

EXEC sp\_create\_plan\_guide

N'TemplateGuide1',

@stmt,

N'TEMPLATE',

NULL,

@params,

N'OPTION(PARAMETERIZATION FORCED)';

Similarly, in a database in which forced parameterization is already enabled, you can make sure that the sample query, and others that are syntactically equivalent, except for their constant literal values, are parameterized according to the rules of simple parameterization. To do this, specify PARAMETERIZATION SIMPLE instead of PARAMETERIZATION FORCED in the OPTION clause.

|  |
| --- |
| **NoteNote** |
| TEMPLATE plan guides match statements to queries submitted in batches that consist of a single statement only. Statements inside multistatement batches are not eligible to be matched by TEMPLATE plan guides. |