

# Forced parameterization

Last updated by | Vitor Tomaz | Feb 24, 2023 at 3:32 AM PST

---

## Contents

- [Issue](#)
- [Investigation/Analysis](#)
- [Mitigation](#)

## Issue


When looking at an ASC report on the **Performance** -> **Queries** tab, you might notice a high number of compilations. High number of compilations can lead to high CPU - On the ASC report, on the **Performance** -> **CPU** tab you can check the amount of CPU used by query executed and query compilation.

Just as background, when a simple AdHoc query is executed, SQL will always try to parametrize the query. For example, if the query below is executed:

```
SELECT * FROM Table1 where ID = 1
```

Sql will parametrize the query, or in other words, will substitute the value 1 with a variable. For the example above, on the predicate search you will see something like:

```
<ColumnReference Database="[DatabaseName]" Schema="[dbo]" Table="[Table1]" Column="ID" /><ScalarOperator  
ScalarString="CONVERT_IMPLICIT(int,[@1],0)">
```

This is what is called [simple parametrization](#)  and will allow plan reuse for AdHoc queries that are executed.

When a query becomes complex, SQL might not be able to parametrize the query under the simple parametrization.


## Investigation/Analysis

Besides of the ASC report, we can also check query parametrization using [this Kusto queries](#).

Take a special attention to the second Kusto query. Will tell you per database and query hash the number of compiles and type of parametrization used. This information will be very useful for the decision making for mitigation.

## Mitigation

If the problem is located on a single query, you might want to suggest to fix the query, making it parametrized (using variables, with consistency on the size. In other words, for variables of data types, varchar, nvarchar, etc, the size should be always the same, i.e. doesn't change the size of the input variable depending on the string size that is being passed).

If the problem is more general, [FORCED PARAMETRIZATION](#)  might be a solution. But be careful with this setting for the following reasons (just to name a few):

- large integer inputs are parametrized to numeric(38,0). Smaller ones will be parametrized to int. Same happens with non-unicode strings: it will be parametrized to varchar(8000) or varchar(max) - in other words, more than one plan for the same query can exist.
- Can choose suboptimal plans on partitioned tables.
- Changing this setting will flush the plan cache, in other words, after changing the setting execution plans need to be compiled again.
- Since plans will now start to be reused, performance issues related with [parameter sniffing](#) can now occur.

In conclusion, setting FORCED PARAMETRIZATION will require testing.

Note also that appropriate query design might be preferable (functions, stored procedures, etc).

To enable forced parametrization:

```
ALTER DATABASE [database_name] SET PARAMETERIZATION FORCED WITH NO_WAIT
```

**How good have you found this content?**

