How to use uspQSFind to review procedure performance

Explanation

If enabled, SQL Server retains statement-level performance metrics (duration, plan used, page reads) on stored procedures in the Query Store. This is similar to the plan cache but it is supplemental and optional and unlike the plan cache, the data is retained on restart. While you can retrieve metrics from the Query Store by reviewing tables such as sys.query_store_query, it is easier to use this custom stored procedure, uspQSFind, to do that for you. It's a wrapper around the existing dynamic management views.

The query store is enabled for most databases in all environments.

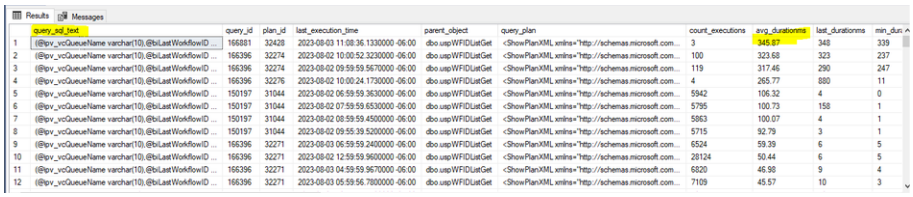
Usage Examples

What are the slowest statements in specified procedure

Retrieve a list of statements in uspWFIDListGet ordered by average duration (in milliseconds) from the past 2 days. You can see the slowest statements in the procedure below and get a query_id for each of them as well as the text of the statement and query plan in XML format as well as when that execution was recorded ("last_execution_time").

```
exec DBAdmin.dbo.uspQSFind @vcDBName = 'MLTCSData', @vcUSPName = 'uspWFIDListGet'
  , @vcOrderBy ='avgduration'
  , @iDaysBack = 2
GO


```



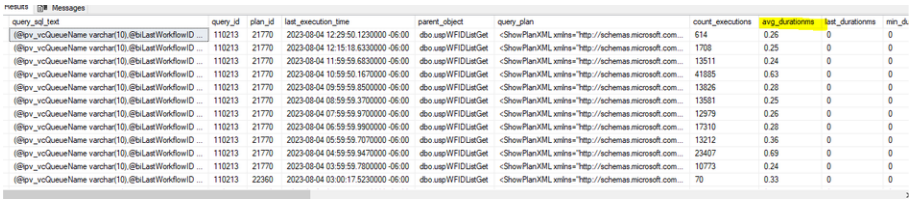
	query_sql_text	query_id	plan_id	last_execution_time	parent_object	query_plan	count_executions	avg_durationms	last_durationms	min_dur
1	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	166881	32428	2023-08-03 11:08:36.1330000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	3	345.67	348	339
2	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	166396	32274	2023-08-02 10:00:52.3230000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	100	323.68	323	237
3	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	166396	32274	2023-08-02 09:59:59.5670000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	119	317.46	290	247
4	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	166396	32276	2023-08-02 10:00:24.1730000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	4	265.77	880	11
5	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	150197	31044	2023-08-02 06:59:59.3630000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	5942	106.32	4	0
6	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	150197	31044	2023-08-02 07:59:59.6530000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	5795	100.73	158	1
7	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	150197	31044	2023-08-02 08:59:59.4500000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	5863	100.07	4	1
8	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	150197	31044	2023-08-02 09:55:39.5200000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	5715	92.79	3	1
9	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	166396	32271	2023-08-03 06:59:59.2400000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	6524	59.39	6	5
10	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	166396	32271	2023-08-02 12:59:59.9600000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	28124	50.44	6	5
11	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	166396	32271	2023-08-03 04:59:59.9670000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	6820	46.98	9	4
12	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	166396	32271	2023-08-03 05:59:56.7800000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	7109	45.57	10	3

When a stored procedure is altered or dropped and recreated, it is a brand new stored procedure as far as SQL Server is concerned so a the next time it runs, a new query_id will be generated. The historical records for the prior version will remain in the Query Store until they age out.

How has a specific statement been performing historically

When you have a specific statement in mind you can filter the results to just that statement. Ideally, you'd want the query_id which you could pass in but if not you can rely on the procedure to try to locate it by giving it a snippet of text from the statement. With the result below you can see that it's been performing typically under <1ms as recently as a few minutes ago (as of this writing).

```
exec DBAdmin.dbo.uspQSFind @vcDBName = 'MLTCSData', @vcUSPName = 'uspWFIDListGet'
  , @vcOrderBy ='exectime'
  , @vcUSPStatement = 'SELECT TOP 1 @biWorkflowID'
  , @iDaysBack = 10
GO
```



	query_sql_text	query_id	plan_id	last_execution_time	parent_object	query_plan	count_executions	avg_durationms	last_durationms	min_d
1	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	110213	21770	2023-08-04 12:29:50.1230000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	614	0.26	0	0
2	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	110213	21770	2023-08-04 12:15:18.6330000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	1708	0.25	0	0
3	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	110213	21770	2023-08-04 11:59:59.6830000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	13511	0.24	0	0
4	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	110213	21770	2023-08-04 10:59:59.1670000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	41885	0.63	0	0
5	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	110213	21770	2023-08-04 09:59:59.8500000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	13826	0.28	0	0
6	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	110213	21770	2023-08-04 08:59:59.3700000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	13581	0.25	0	0
7	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	110213	21770	2023-08-04 07:59:59.9700000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	12979	0.26	0	0
8	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	110213	21770	2023-08-04 06:59:59.9900000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	17310	0.28	0	0
9	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	110213	21770	2023-08-04 05:59:59.7070000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	13212	0.36	0	0
10	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	110213	21770	2023-08-04 04:59:59.9470000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	23407	0.69	0	0
11	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	110213	21770	2023-08-04 03:59:59.7800000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	10773	0.24	0	0
12	(@pv_vcQueueName varchar(10),@biLastWorkflowID ...	110213	22360	2023-08-04 03:00:17.5230000 -06:00	dbo.uspWFIDListGet	<ShowPlanXML xmlns="http://schemas.microsoft.com...	70	0.33	0	0

This would accomplish similar but would be more precise because you know the query id and can pass it in.

```
exec DBAdmin.dbo.uspQSFind @vcDBName = 'MLTCSData', @vcUSPName = 'uspWFIDListGet'
  , @vcOrderBy ='exectime'
  , @iQueryID = 110213
```

```
4   , @iDaysBack = 10
5 GO
```

Parameters

CREATE OR ALTER PROCEDURE uspQSFind
@vcDBName VARCHAR(100),
@vcUSPName VARCHAR(100),
@vcUSPSchema VARCHAR(100) = 'dbo',
@vcUSPStatement VARCHAR(100) = '',
@iQueryID INT = NULL,
@iDaysBack INT = NULL,
@vcOrderBy VARCHAR(100) = 'exectime', --@vcOrderBy must be "exectime" or "queryid" or "avgduration"
@vcTimeZone VARCHAR(100) = 'Mountain Standard Time',
@bOutputQry BIT = 0

You have to give it a database and procedure name. It defaults to the dbo schema but you can specify a different one. By default it goes back 7 days so you don't have to specify it. You can order (desc) by execution time, query id or average duration. The time is in UTC so the procedure converts it to mountain time by default. If you want the output of the query you can specify that it outputs the search query to the Messages column in SSMS.