

# Compare two managed instances

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

---

## Contents

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

## Issue

Customer complains about performance differences between two managed instances.

## Investigation/Analysis


The investigation should naturally go with comparing both environments.

For SQL Server vs MI check [this TSG](#)

This checklist assumes that the workload (number of requests) is exactly the same - you can double check on ASC: **Performance** -> **Overview** and scroll down to **Query Execution Count Statistics**.

Below a checklist that can be used to compare differences on both environments (focusing on T-SQL commands to run on customer side):

1 - check the Managed Instance SLO of both instances - on **ASC** go to **Properties**.

2 - if the Managed instances are General Purpose, check data / log file size - [File IO characteristics in General Purpose tier](#) 

3 - compare the configurations of both instances like max degree of parallelism (Maxdop) and optimize for ad hoc workloads:

```
SELECT name, value, value_in_use, minimum, maximum, [description], is_dynamic, is_advanced
FROM sys.configurations WITH (NOLOCK)
ORDER BY name OPTION (RECOMPILE);
```

4 - compare database properties between both instances. For example, compatibility level since it impacts execution plan compilation.

```
select * from sys.databases
```

5 - Check if on both managed instances, the number of tempdb files is the same (type\_desc ROWS)

```

SELECT DB_NAME([database_id]) AS [Database Name],
       [file_id], [name], physical_name, [type_desc], state_desc,
       is_percent_growth, growth,
       CONVERT(bigint, growth/128.0) AS [Growth in MB],
       CONVERT(bigint, size/128.0) AS [Total Size in MB], max_size
FROM sys.master_files WITH (NOLOCK)
where DB_NAME([database_id]) = 'tempdb'
ORDER BY DB_NAME([database_id]), [file_id] OPTION (RECOMPILE);

```

6 - Focus on one database where the customer has activity. Check the table sizes. Check if there is a big skew between the two environments. If yes, the workload is not comparable (might need )

```

SELECT DB_NAME(DB_ID()) AS [Database Name], SCHEMA_NAME(o.Schema_ID) AS [Schema Name],
OBJECT_NAME(p.object_id) AS [Table Name],
CAST(SUM(ps.reserved_page_count) * 8.0 / 1024 AS DECIMAL(19,2)) AS [Object Size (MB)],
SUM(p.Rows) AS [Row Count],
p.data_compression_desc AS [Compression Type]
FROM sys.objects AS o WITH (NOLOCK)
INNER JOIN sys.partitions AS p WITH (NOLOCK)
ON p.object_id = o.object_id
INNER JOIN sys.dm_db_partition_stats AS ps WITH (NOLOCK)
ON p.object_id = ps.object_id
WHERE ps.index_id < 2
AND p.index_id < 2
AND o.type_desc = N'USER_TABLE'
GROUP BY SCHEMA_NAME(o.Schema_ID), p.object_id, ps.reserved_page_count, p.data_compression_desc
ORDER BY SUM(ps.reserved_page_count) DESC, SUM(p.Rows) DESC OPTION (RECOMPILE);

```

One way to find the most busy databases is to check the CPU consumption per database:

```

USE master
go
WITH DB_CPU_Stats
AS
(SELECT pa.DatabaseID, DB_Name(pa.DatabaseID) AS [Database Name],
       SUM(qs.total_worker_time/1000) AS [CPU_Time_Ms]
FROM sys.dm_exec_query_stats AS qs WITH (NOLOCK)
CROSS APPLY (SELECT CONVERT(int, value) AS [DatabaseID]
FROM sys.dm_exec_plan_attributes(qs.plan_handle)
WHERE attribute = N'dbid') AS pa
GROUP BY DatabaseID)
SELECT ROW_NUMBER() OVER(ORDER BY [CPU_Time_Ms] DESC) AS [CPU Rank],
       [Database Name], DatabaseID, [CPU_Time_Ms] AS [CPU Time (ms)],
       CAST([CPU_Time_Ms] * 1.0 / SUM([CPU_Time_Ms]) OVER() * 100.0 AS DECIMAL(5, 2)) AS [CPU Percent]
FROM DB_CPU_Stats
WHERE DatabaseID <> 32767
ORDER BY [CPU Rank] OPTION (RECOMPILE);

```

Or, IO per database

```

USE master
go
WITH Aggregate_IO_Statistics
AS (SELECT DB_NAME(database_id) AS [Database Name],
  CAST(SUM(num_of_bytes_read + num_of_bytes_written) / 1048576 AS DECIMAL(12, 2)) AS [ioTotalMB],
  CAST(SUM(num_of_bytes_read ) / 1048576 AS DECIMAL(12, 2)) AS [ioReadMB],
  CAST(SUM(num_of_bytes_written) / 1048576 AS DECIMAL(12, 2)) AS [ioWriteMB], database_id
  FROM sys.dm_io_virtual_file_stats(NULL, NULL) AS [DM_IO_STATS]
  GROUP BY database_id)
SELECT ROW_NUMBER() OVER (ORDER BY ioTotalMB DESC) AS [I/O Rank],
  [Database Name], database_id, ioTotalMB AS [Total I/O (MB)],
  CAST(ioTotalMB / SUM(ioTotalMB) OVER () * 100.0 AS DECIMAL(5, 2)) AS [Total I/O %],
  ioReadMB AS [Read I/O (MB)],
  CAST(ioReadMB / SUM(ioReadMB) OVER () * 100.0 AS DECIMAL(5, 2)) AS [Read I/O %],
  ioWriteMB AS [Write I/O (MB)],
  CAST(ioWriteMB / SUM(ioWriteMB) OVER () * 100.0 AS DECIMAL(5, 2)) AS [Write I/O %]
FROM Aggregate_IO_Statistics
ORDER BY [I/O Rank] OPTION (RECOMPILE);

```

Queries with higher Average Execution time (the first column will tell you the database where that query is executed):

```

SELECT TOP(50) DB_NAME(t.[dbid]) AS [Database Name],
  REPLACE(REPLACE(LEFT(t.[text], 255), CHAR(10), ''), CHAR(13), '') AS [Short Query Text],
  qs.total_elapsed_time/qs.execution_count AS [Avg Elapsed Time],
  qs.min_elapsed_time, qs.max_elapsed_time, qs.last_elapsed_time,
  qs.execution_count AS [Execution Count],
  qs.total_logical_reads/qs.execution_count AS [Avg Logical Reads],
  qs.total_physical_reads/qs.execution_count AS [Avg Physical Reads],
  qs.total_worker_time/qs.execution_count AS [Avg Worker Time],
  CASE WHEN CONVERT(nvarchar(max), qp.query_plan) LIKE N'%<MissingIndexes>%' THEN 1 ELSE 0 END AS [Has Missing I
  qs.creation_time AS [Creation Time]
FROM sys.dm_exec_query_stats AS qs WITH (NOLOCK)
CROSS APPLY sys.dm_exec_sql_text(plan_handle) AS t
CROSS APPLY sys.dm_exec_query_plan(plan_handle) AS qp
ORDER BY qs.total_elapsed_time/qs.execution_count DESC OPTION (RECOMPILE);

```

7 - Pick one of the queries that the customer is complaining about (our use the last query shared on point 6). Capture the execution plan from both sides. If there are differences on the execution plan, there are obvious differences (indexes, statistics, column structure, etc).

You can use the TSG [Execution Plans](#) to help you on the execution plans analysis.

8 - Picking up on of the tables from the query used on point 7, check on both managed instances if the statistics have the same sample.

```
SELECT
    ObjectSchema = OBJECT_SCHEMA_NAME(s.object_id),
    ObjectName = object_name(s.object_id),
    s.object_id,
    s.stats_id,
    s.name AS 'stats_name',
    CASE WHEN (s.stats_id > 2 AND s.auto_created = 1) THEN 'AUTOSTATS' WHEN (s.stats_id > 2 AND s.auto_created
    i.type_desc,
    sp.last_updated,
    sp.rows,
    sp.rows_sampled,
    sp.modification_counter
FROM sys.stats s
OUTER APPLY sys.dm_db_stats_properties(s.object_id,s.stats_id) sp
LEFT JOIN sys.indexes i ON sp.object_id = i.object_id AND sp.stats_id = i.index_id
WHERE OBJECT_SCHEMA_NAME(s.object_id) != 'sys'
and object_name(s.object_id) = '<table_name>' --table name
ORDER BY sp.last_updated DESC;
```

If not, [update statistics](#).

Might be worth to also check the [index fragmentation](#).

## Mitigation

The mitigation will depend on the differences found.

## How good have you found this content?



-