

Workflow for IO troubleshooting

Last updated by | Vitor Tomaz | Feb 24, 2023 at 3:31 AM PST

Contents

- [1 Generate an ASC report](#)
- [2 looking at the ASC report](#)
- [3 Log Write](#)
 - [31 WRITELOG](#)
 - [32 LOGBUFFER](#)
- [4 Data IO](#)
- [5 Data Background IO](#)
- [6 PAGELATCH and PAGEIOLATCH](#)

The following TSG is intended to help troubleshoot IO issues.

Start at point [1](#)

1 Generate an ASC report

First identify where the issue started. Generate the ASC report according with the timelines given. Move to step [2](#)

2 looking at the ASC report

Open the ASC report. Go to **Performance->Overview**. Look at the graph **Database Resource Consumption Statistics**. You can also open **Performance->IO**. What is the highest metric?

Before jumping into any further step, first look into the instance (or Azure SQL) SLO. On ASC, go to the **Properties** tab. This might be useful down the road

If its Log_Write proceed to [3](#) For Data IO, proceed to [4](#) For Data Background IO, step into [5](#)

3 Log Write

In short, we will concentrate on write performance.

On ASC go to **Performance->Overview**

Do you see high WRITELOG wait types?

If WRITELOG, go to [3.1](#). If LOGBUFFER, go to [3.2](#)

31 WRITELOG

After getting this info, get an idea of what [queries have high transaction log usage](#).

With the query information the solutions should be around actions that will reduce the amount of transaction log generated, for example:

- reduce the number of indexes (if possible). Check [this TSG](#) for details on why the amount on indexes matter and also guidelines on how an index can be designed.
- reduce the amount of [page splits](#) (page split is a logged operation).
- reduce the transaction size. In other words, break big transactions in smaller ones.
- If on point 2 you saw that the MI SLO is General Purpose, you might want to check for the [increase of the log file size](#) ☐. Note also that Business Critical offers better log io. - **Note:** only applies to Managed Instance.

If during your troubleshooting you can also see PAGELATCH or PAGEIOLATCH (assuming that the customer is doing heavy writes), go to [6](#)

32 LOGBUFFER

LOGBUFFER is different from WRITELOG, in sense that it is waiting for space to be allocated in the log buffer. This wait type is very frequent when we have huge transactions.

The resolution is similar to WRITELOG.

4 Data IO

If Data IO is high, most likely you will see PAGEIOLATCH waits and queries with IO consumption. This can be for a variety of reasons:

- large scans
- indexing opportunities
- memory pressure
- lack of maintenance

In this order of ideas, check:

- [queries with high IO](#). From here you can check for tuning opportunities. Look for large scans and other physical operators like Key or RID lookups. This all can be seen on [Execution plans](#)
- [implicit conversions](#). It is true that implicit conversions are associated with high CPU, but since they can cause some queries to scan an index, can also be a point to look at. Take also a look at the ASC report on **Performance -> Queries -> Top CPU Consuming Queries with Anti-Patterns**
- [Fragmented indexes](#) will mean that more reads will be required to perform an IO operation. [Check fragmentation and solve if required](#).
- with [memory pressure](#), it means that more IO requests must be done. Memory pressure can be a reflection of multiple scenarios. Check [outdated statistics](#), query design, [indexing opportunities](#) and [memory limits](#) ☐. Here we can be in front of different scenarios like an undersized SLO, queries with high memory grants and/or queries that need optimization. Check also [Page life expectancy](#).

Also consider IO limits for Managed Instance, especially when we are dealing with the [General Purpose SLO](#) ☐.

5 Data Background IO

Data_background_IO is associated with CHECKPOINT and Lazy Writer events. Since its associated with CHECKPOINT, it's normal to see it spiking when BACKUPS occur. Lazy Writer flushes clean and dirty pages from the buffer pool to disk.


If Data_background_IO is a problem, look for memory pressure. This is described on point [4](#)

6 PAGELATCH and PAGEIOLATCH

PAGELATCH and PAGEIOLATCH can occur for [several reasons](#).

To get the queries waiting on a specific wait you can use the Kusto query referenced [here](#). Just change the value for the relevant wait type.

From here the solution will depend on what is seen.

- in [tempdb contention](#), the solution might me around increasing the number of files on tempdb (note that requires a restart / failover, so it might not be the best solution) or reviewing the number of accesses on tempdb.
- if it is related with [last page contention](#) .
- check the value for [Page life expectancy](#).

How good have you found this content?

