6 Best Practices to Improve SQL Query Performance

August 2020

Bruce Boyers

Condusiv®
THE DISKEEPER COMPANY

6 Best Practices to Improve SQL Query Performance

MS SQL Server is the world's leading RDMS and has plentiful benefits and features that empower its efficient operation. As with any such robust platform, however—especially one which has matured as SQL Server has—there have been best practices evolved that allow for its best performance.

For any company utilizing it, Microsoft SQL Server is central to a company's management and storage of information. In business, time is money, so any company that relies on information to function (and in this digital age that would be pretty much all of them) needs access to that information as rapidly as possible. In that information is often obtained from databases through queries, optimizing SQL query performance is vital.

As 7 out of 10 Condusiv customers come to us because they were experiencing SQL performance issues and user complaints, we have amassed a great deal of experience on this topic and would like to share. We have done extensive work in eliminating I/O inefficiencies and streamlining I/O for optimum performance. It is especially important in SQL to reduce the number of random and "noisy" I/Os as they can be quite problematic. We will cover this as well as some additional best practices. Some of these practices may be more time-consuming, and may even require a SQL consultant, and some are easy to solve.

> ***"SQL Server, and frankly all relational databases, are all high I/O utilization systems. They're going to do a lot of workload against the storage array. Understanding their I/O patterns are important, and more important in virtual environments." — Joey D'Antoni, Senior Architect and SQL Server MVP***

Here are 6 best practices for the improvement of SQL query performance.

1. Tune queries

A great feature of the SQL language is that it is fairly easy to learn and to use in creating commands. Not all database functions are efficient, however. Two queries, while they might appear similar, could vary when it comes to execution time. The difference could be the way they are structured; this is a very involved subject and

open to some debate. It's best to engage a SQL consultant or expert and allow them to assist you with structuring your queries.

Aside from the query structure, there are some great guidelines to follow in defining business requirements before beginning.

- Identify relevant stakeholders
- Focus on business outcomes
- Ask the right questions to develop good query requirements
- Create very specific requirements and confirm them with stakeholders

2. Add memory

Adding memory will nearly always assist in SQL Server query performance, as SQL Server uses memory in several ways. These include:

- The buffer cache
- Plan cache, where query plans are stored for re-use
- Buffer pool, in which are stored recently written-to pages
- Sorting and matching data, which all takes place in memory

Some queries require lots of memory for joins, sorts, and other operations. All of these operations require memory, and the more data you aggregate and query, the more memory each query may require.

*Tips for current Condusiv DymaxIO™ users:

1. Provision an additional 4-16GB of memory to the SQL Server if you have additional memory to give.
2. Cap MS-SQL memory usage, leaving the additional memory for the OS and our software. Note – Condusiv software will leverage whatever is unused by the OS.
3. If no additional memory to add, cap SQL memory usage leaving 8GB for the OS and our software. Note – This may not achieve 2X gains but will likely boost performance 30-50% as SQL is not always efficient with its memory usage.

3. Perform index maintenance

Indexes are a key resource to SQL Server database performance gain. The downside, however, is that database indexes degrade over time.

Part of this performance degradation comes about through something that many system administrators will be familiar with: fragmentation. Fragmentation on a storage drive means data stored non-contiguously, so that the system has to search through thousands of fragments, meaning extra I/Os, to retrieve data. It is a similar situation with a database index.

There are two types of database index fragmentation:

- Internal fragmentation, which occurs when more than one data page is created, neither of which is full. Performance is affected because SQL Server must cache two full pages including empty yet allocated space.
- External fragmentation, which means pages that are out of order.

When an index is created, all pages are sequential, and rows are sequential across the pages. But as data is manipulated and added, pages are split, new pages are added, and tables become fragmented. This ultimately results in index fragmentation.

There are numerous measures to take in restoring an index so that all data is sequential again. One is to rebuild the index, which will result in a brand-new SQL index. Another is to reorganize the index, which will fix the physical order and compact pages.

There are other measures you can take as well, such as finding and removing unused indexes, detecting and creating missing indexes, and rebuilding or reorganizing indexes weekly.

It is recommended you do not perform such measures unless you are a DBA and/or have a thorough understanding of SQL Server.

4. Add extra spindles or flash drives

Like the increase of memory, increasing storage capacity can be beneficial.

Adding an SSD, the most expensive option, can provide the most benefit as there are no moving parts. The less expensive option is to add spindles. Both of these options can help with decreasing latency times, but it does not get rid of the extra I/Os occurring due to fragmentation. It is not really solving the root cause of I/O inefficiencies.

5. Optimize the I/O subsystem

Optimizing the I/O subsystem is highly important in optimizing SQL Server performance. When configuring a new server, or when adding or modifying the disk configuration of an existing system, determining the capacity of the I/O subsystem before deploying SQL Server is good practice.

There are three primary metrics that are most important when it comes to measuring I/O subsystem performance:

- Latency, which is the time it takes an I/O to complete.
- I/O operations per second, which is directly related to latency.
- Sequential throughput, which is the rate at which you can transfer data.

You can utilize an I/O stress tool to validate performance and ensure that the system is tuned optimally for SQL Server before deployment. This will help identify hardware or I/O configuration-related issues. One such tool is Microsoft DiskSpd, which provides the functionality needed to generate a wide variety of disk request patterns. These can be very helpful in the diagnosis and analysis of I/O performance issues.

You can download DiskSpd.exe here

Another tool is Condusiv's I/O Assessment Tool for identifying which systems suffer I/O issues and which systems do not. It identifies and ranks systems with the most I/O issues and displays what those issues are across 11 different key performance metrics by identifying performance deviations when workload is the heaviest.

You can download Condusiv's I/O Assessment Tool here

6. Use DymaxIO fast data software

"Reduce the number of I/Os that you are doing. Because remember, the fastest read from disk you can do is one you don't do at all. So, if you don't have to do a read, that's all the better.**"** —Joey D'Antoni, Senior Architect and SQL Server MVP

Reducing and streamlining small, random, fractured I/O will speed up slow SQL queries, reports and missed SLAs. DymaxIO makes this easy to solve.

Many companies have utilized virtualization to greatly increase server efficiency for SQL Server. While increasing efficiency, at the same time virtualization on Windows

systems has a downside. Virtualization itself adds complexity to the data path by mixing and randomizing I/O streams—something known as the "I/O blender effect." On top of that, when Windows is abstracted from the physical layer, it additionally utilizes very small random read and writes which are less efficient that larger contiguous reads and writes. SQL Server performance is penalized not once, but twice. The net effect is I/O characteristics more fractured and random than they need to be.

The result is that typically systems process workloads about 50 percent slower than they should be, simply because a great deal more I/O is required.

While hardware can help performance problems as covered above, it is only a temporary fix as the *cause* of Windows I/O inefficiencies is not being addressed. Many sites have discovered that DymaxIO fast data software, employed on any Windows server (virtual or physical) is a quicker and far more cost-effective solution. DymaxIO replaces tiny writes with large, clean, contiguous writes so that more payload is delivered with every I/O operation. I/O to storage is further reduced by establishing a tier 0 caching strategy which automatically serves hot reads from idle, otherwise unused memory. The software adjusts itself, moment to moment, to only use unused memory.

The use of DymaxIO can improve performance by 50 percent or more. Many sites see twice as much improvement or more, depending on the amount of DRAM available. Condusiv Technologies, developer of DymaxIO, actually provides a money-back guarantee that DymaxIO will solve the toughest application performance challenges on I/O intensive systems such as SQL Server.

Click here to get started with a free 30-day trial of DymaxIO