IDERA® | SQL Server Whitepaper

PREDICTABLE BI PERFORMANCE WITH UNPREDICTABLE QUERIES

BY JEREMIAH PESCHKA

INTRODUCTION

SQL Server database administration is difficult; queries and indexes require tuning in response to application changes. For many database administrators, change is regular; change happens after a development cycle with testing phases and a careful rollout process.

Business intelligence applications are different; they can be unpredictable. Schemas are designed to respond to changing user demands. Users may change query habits in response to changing market forces, and complex relationships may be redefined at the drop of a hat. Our best online transaction processing tuning efforts cannot keep up with the changing demands of business intelligence power users.

The purpose of this white paper is to teach database administrators on how to determine if SQL Server is configured correctly for BI workloads. This whitepaper introduces foundational ideas for sizing SQL Server storage and hardware. It then addresses more complex SQL Server features like SQL Server Resource Governor and Buffer Pool Extensions. Besides, we will examine tempdb configuration, parallelism, and licensing. That should provide a database administrator with the core knowledge needed to design a powerful business intelligence SQL Server.

CHOOSING TO VIRTUALIZE

There are several ways that database administrators are working virtual machines into their environments. Noncritical workloads, small footprint servers, and even critical production workloads are running on virtual platforms. Some decisions to virtualize should be easy, while others should require careful deliberation and testing.

HOW TO TELL YOU TO NEED TO BE HERE

Designing SQL Server to respond to business intelligence workloads does not have to be complicated or confusing. With a well-defined set of criteria, you can remove the guesswork from the design process. Before you start designing SQL Server for a BI workload, be aware of your end-users, your bottlenecks, and your hardware.

END USER PROFILE

In many other disciplines, practitioners create profiles for the target user of a product, process, or system. For a gaming system, the profile might be a young person between 18 and 25 with disposable income. For a luxury automobile, the target is likely to be someone over the age of 50 with disposable income.

The end-user profile for a business intelligence system is a power user. They can structure their questions and transform business questions into system questions. Whether they use a tool like PowerPivot, Tableau, or QlikView is irrelevant. These users know what they want, and they know how to ask for it. They are also likely to have deep domain knowledge and at least a passing familiarity with data analytics techniques or statistics.

COMMON WAITS

Before attempting any tuning effort, it is critical to understand the workload you are trying to tune. If you find yourself with any of the following SQL Server waits, you should start eliminating those waits before moving on:

- **LATCH_EX -** indicates that there are missing indexes or that there may be tempdb contention.
- **SOS_SCHEDULER_YIELD -** indicates that the SQL Server could be underprovisioned on CPU or that queries make heavy use of scalar functions.
- **PAGELATCH_EX, PAGELATCH_SH -** indicates bottlenecks during insert workload; a common solution is to randomize writes into the affected table.
- **WRITELOG -** many small database log writes are being performed; this is indicative of a transactional SQL Server.
- **RESOURCE_SEMAPHORE -** queries are waiting on a desired memory grant before executing. In short - you need more memory.
- **RESOURCE_SEMAPHORE_QUERY_COMPILE -** queries are waiting to acquire memory before query compilation can even begin.

We want to see a SQL Server that is free of these waits. If these waits exist in an existing reporting system, do not give up. Keep reading and see if there is something that you can do to address them. You may even see some of these waits during extract, transform, and load jobs or other non-production activities. Use monitoring tools such as IDERA SQL Diagnostic Manager to determine when the waits are occurring and figure out the best way to solve the problem.

There is one wait that you should not worry about CXPACKET. This wait is a sign that all is well in the world of a data warehouse. It is addressed in more detail below in the section "Preparing for Parallelism". For now, just know that CXPACKET is not a big deal. However, if you are working with the OLTP workload, you may also want to consider it in the list above.

VIRTUALIZATION

Virtualizing SQL Server has become more common. It is possible to virtualize many SQL Server applications. However, it is essential to note that virtualization can add unpredictability to an application through resource contention or resource oversubscription. In this case, the goal is to bring predictability to an unpredictable workload; virtualization makes that process more difficult. This paper assumes that you are using physical hardware.

SIZING SQL SERVER HARDWARE

A full guide to building a high-performance business intelligence SQL Server is outside the scope of this paper. However, there are several core ideas that DBAs can keep in mind to improve database performance.

Memory Requirements

It is tempting to purchase a large amount of memory for SQL Server. The semi-serious response to SQL Server memory planning is to ask for all of it. This approach works well for transactional applications. Business intelligence applications are different; large amounts of memory are not necessarily the best choice. A large execution plan cache, or even a large buffer pool, can lead to poor SQL Server performance.

Aa good recommendation is 128GB to 256GB of memory per CPU socket to manage unpredictable business intelligence workloads. A large amount of memory can be less efficient for a business intelligence focused SQL Server. When tables are especially large (10% or more of the buffer pool), reads may effectively bypass the buffer pool through a process called buffer pool disfavoring. Technically, the data is still read into the buffer pool. However, the data is immediately marked as the least recently used item in the buffer pool. That prevents a big table from clearing the rest of the buffer pool. Caching large tables in memory can increase online transaction processing performance. However, for business intelligence applications, the intent is to keep memory available for large sorts, joins, and other resource-intensive activities.

Disk performance

If the SQL Server will not be filled with memory, how fast should disks be? The entire I/O subsystem should be fast enough to saturate the CPUs. For our purposes, we can assume that CPUs can consume between 200 MB/s and 350 MB/s (depending on the CPU model). A two-socket hex core system from several years ago is capable of consuming 3.2GB/s of data. To take advantage of the CPU resources in this system, the entire input and output subsystem should be able to produce 3.2 GB/s of input and output. Ensure that disk performance lines up with SQL Server CPU performance. Hardware is used to mitigate performance issues; since user queries cannot be predicted or tuned, the hardware is designed to provide a predictable level of throughput.

Balanced performance

Designing a BI system this way ensures that there is a balance of performance throughout all system components. 128GB to 256GB of memory per CPU socket provides an effective balance for SQL Server. By configuring memory in these ranges, a database administrator can balance memory between caching frequently used query plans, sorts, joins, and prevent large tables from dominating the buffer pool. Designing a disk subsystem to produce data as quickly as CPUs can consume it alleviates the need to have a large buffer pool. When data can be consumed as fast as it can be produced, there is minimal need to worry about storage latency for a business intelligence system.

WHAT KIND OF HARDWARE DO YOU NEED?

Although this may sound like the domain of specialized enterprise hardware, it is possible to attain this type of storage performance using commodity hardware.

Consumer-grade solid-state storage

Although the cost per GB of solid state drives is still higher than the cost of rotational storage, consumer-grade solid state drives offer tremendous throughput (up to 450 MB/s in some cases). Solid state drives also offer minimal storage latency. The combination of high throughput, low latency, and relatively high storage density makes it possible to provide fast performance even under heavy load and at a relatively low price point. With some servers, it is possible to house 24TB of raw solid state drive storage in a single server chassis with a double unit height.

Rotational storage options and considerations

It is also possible to get adequate performance out of rotational storage. Rotational storage offers effective throughput at a lower price per GB than solid state drive. The one thing that rotational storage cannot offer over solid state drive is improved latency. While solid state drive latency is measured in microseconds, rotational drive latency is measured in milliseconds which is several orders of magnitude longer. To compensate for the increased latency, it is typical to overprovision rotational storage. Where ten solid state drives) would have sufficed, many more rotational drives are used. Four to five times as many rotational drives are needed to match the storage throughput of one consumer-grade solid state drive. Whatever storage option is chosen for the BI system, it is important to test and verify the throughput of the storage before going live. Tools like Crystal Disk Mark and SQLIO make it possible to test the raw throughput of the storage. Before installing SQL Server, verify storage performance with one or both of these tools. Resources on using Crystal Disk Mark and SQLIO are provided at the end of this whitepaper.

PREPARING FOR PARALLELISM

Decision support systems rely heavily on parallel query execution. Unfortunately, the default parallelism settings of SQL Server are less than perfect for guaranteeing predictable query performance. There are two steps recommended by Microsoft for the initial SQL Server configuration.

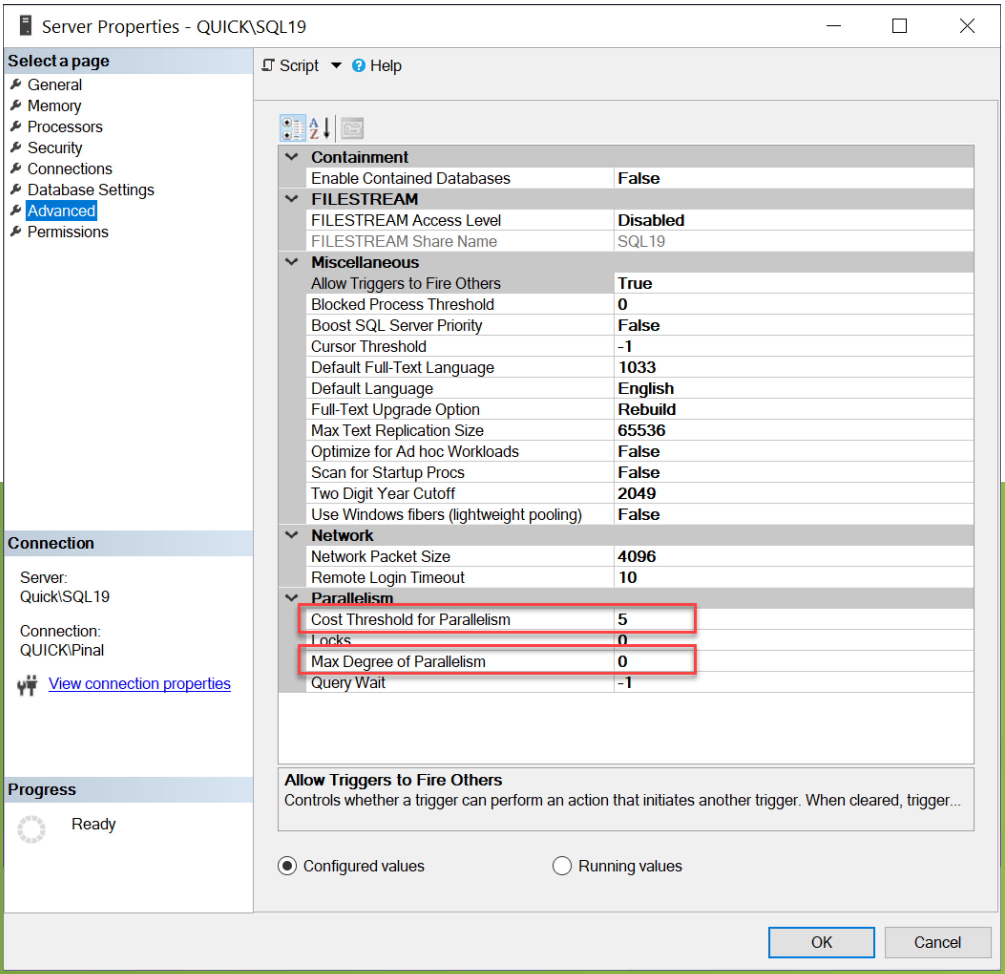
The first step is to correctly configure the “max degree of parallelism” setting (MAXDOP). In the Microsoft Knowledge Base article “Recommendations and Guidelines for the “max degree of parallelism” Configuration Option in SQL Server”, Microsoft recommends initially setting the maximum degree of parallelism to the number of physical CPU cores in a single CPU socket. For example, with a six-core CPU with hyper-threading enabled, Microsoft recommends setting MAXDOP to six.

When you read the article related to guidelines, you must focus on the version of your SQL Server as well. Microsoft has two different guidelines; 1) for SQL Server 2016 and later version and 2) SQL Server 2008 to 2014.

Additionally, it is worth remembering that you can set this value in three different ways for your workload.

- **Query level:** Use the MAXDOP query hint
- **Database level:** Use the MAXDOP database scoped configuration
- **Workload level:** Use the MAX_DOP Resource Governor workload group configuration option

It is also worth mentioning that the default cost threshold for parallelism setting is very low. The cost threshold for parallelism setting determines how expensive a query needs to be before SQL Server considers using multiple threads for executing the query. The default setting has not been changed since it was first introduced. It makes sense to increase this setting from the default; a reasonable cost threshold for parallelism setting is 50.



Server: Quick\SQL19

Connection: QUICK\Pinal

Ready

Containment	
Enable Contained Databases	False
FILESTREAM	
FILESTREAM Access Level	Disabled
FILESTREAM Share Name	SQL19
Miscellaneous	
Allow Triggers to Fire Others	True
Blocked Process Threshold	0
Boost SQL Server Priority	False
Cursor Threshold	-1
Default Full-Text Language	1033
Default Language	English
Full-Text Upgrade Option	Rebuild
Max Text Replication Size	65536
Optimize for Ad hoc Workloads	False
Scan for Startup Procs	False
Two Digit Year Cutoff	2049
Use Windows fibers (lightweight pooling)	False
Network	
Network Packet Size	4096
Remote Login Timeout	10
Parallelism	
Cost Threshold for Parallelism	5
Locks	0
Max Degree of Parallelism	0
Query Wait	-1

Allow Triggers to Fire Others
Controls whether a trigger can perform an action that initiates another trigger. When cleared, trigger...

[✓ Configured values] Running values

Keep in mind that both of these settings will need further testing; the goal of these defaults is to ensure that you have a reasonably sound system from the moment you start. Although it is tempting to eliminate CXPACKET waits, do not. A moderate amount of CXPACKET waits is a sign of a well-behaved BI SQL Server. Through parallel query processing, SQL Server can fully use the hardware we are providing and send data back to the users as fast as it can be read off of the disk.

LICENSING AND CORE COUNT

We now take a brief digression into the world of SQL Server licensing. Business intelligence systems are typically input and output bound. However, CPU can play a significant factor in system performance, including the number of concurrent queries and the amount of input and output a system can consume.

Does core count matter?

Yes. Core count is important for business intelligence database systems. A large core count allows for increased concurrency. Core count enables an increase in intra-query concurrency. The more cores that are available within a single socket, the more potential parallelism there is within any one query. Having a large number of cores available across many CPUs increases the number of concurrent queries that can be running within the business intelligence system.

If you are using the latest version of SQL Server 2019, you can refer here for additional help related to pricing and licensing.

Does CPU speed matter?

If core count is important, should you assume that core speed is essential also? Absolutely! Core speed influences how much data a single core can consume per second. A slower CPU will consume fewer data per second than a faster CPU. That plays directly into the performance and concurrency of the system.

The best approach is to strive for a mix of CPU speed and core count. A moderate number of relatively fast cores may provide better performance than a small number of fast cores or a large number of slow cores. Make sure to benchmark the system before deploying a production database. If you are not able to get acceptable performance, review the bottlenecks of the system.

SQL Server edition

For our discussion, SQL Server comes in two editions: Standard Edition and Enterprise Edition.

SQL Server Standard Edition limits both the number of CPUs and the amount of memory available to SQL Server for operations. Although Standard Edition is substantially cheaper than Enterprise Edition, the resource limitations of Standard Edition makes it unsuitable for production deployments.

SQL Server Enterprise Edition allows for unlimited CPUs and memory. Besides, it includes several features that help the management of a business intelligence system. Two critical features of SQL Server Enterprise Edition are read ahead and table partitioning. Read ahead, allow SQL Server to read additional data into the buffer pool during particularly read-heavy queries. Both editions of SQL Server have this feature. However, the read-ahead feature of SQL Server Enterprise Edition reads a much more considerable amount of data into memory. That can dramatically reduce the amount of time SQL Server spends waiting on disks during query processing. Table partitioning helps enhance the performance of data lifecycle management. New data can be loaded into a fresh partition, and old data can be easily swapped out. Table partitioning makes it possible to keep the data warehouse available for much larger periods.

If you remain unconvinced about the utility of SQL Server Enterprise Edition for BI systems, think about the queries that are commonly run against the system. Are the queries small and targeted, or are they large joins that analyze a significant portion of the system? Larger queries will require a larger workspace. Whether that workspace is in memory or on disk depends on the amount of memory available to SQL Server at runtime. While it is possible to tune SQL Server Standard Edition to use a small amount of memory per query, this requires a skilled performance tuner and a significant amount of time. It is typically more cost-effective to use the more substantial amount of memory available to SQL Server Enterprise Edition than to tune a workload to fit into the requirements of SQL Server Standard Edition.

SQL SERVER RESOURCE GOVERNOR

SQL Server Resource Governor is an Enterprise Edition feature that makes it possible to control the resources used by a particular query or workload. Depending on workload, it may be necessary to configure Resource Governor to limit resources available to anyone query. For business intelligence systems, a well-configured Resource Governor can add predictability to a workload and limit the workload of anyone from taking over the SQL Server.

Why you want Resource Governor

Resource Governor can be a mixed blessing. When incorrectly configured, it can add to the unpredictability to a workload. However, a well-configured Resource Governor allows a database administrator to control all resources available to the workload carefully.

After analyzing a workload, a DBA can create many resource pools and assign various tasks to pools with different resource allocations. Resource allocations can limit queries by CPU, memory consumption, or even input and output. Effective utilization of Resource Governor means that well-behaved queries will be able to access system resources even though resource-intensive queries are still running.

Resource Governor makes it possible to add additional users to a system with resource-consuming queries. Additional workloads can be added by limiting the amount that any single query can do. There is no magic to this feature. More queries will be able to execute. However, queries may take longer to execute under load.

Getting started with Resource Governor

The easiest way to get started with the SQL Server Resource Governor is to start slowly. Rather than attempting a full implementation of Resource Governor, start by focusing on one of the side effects: tracking SQL Server resource consumption for chargebacks. Chargebacks are a way to bill other departments for their consumption of valuable SQL Server resources. References to additional information about configuring resource governor for measuring chargebacks can be found at the end of this paper.

By implementing Resource Governor with a chargeback strategy, database administrators responsible for business intelligence systems will quickly become aware of more than just the most resource-intensive queries. When configured for chargeback, Resource Governor simply tracks resource utilization across all queries. The database administrator will be able to review significant aspects of the workload like CPU consumption, requests per pool, memory usage for queries in each pool, input and output metrics, and even query compilation metrics (for example, many bad plans generated). That allows the database administrator to make better decisions about resource allocation among different workloads. Having this information can also make it easier for the database administrator to assess the server load accurately. The technical details of implementing Resource Governor are beyond the scope of this paper. References are provided in the appendix for readers who are ready for more advanced material.

CONFIGURING TEMPDB

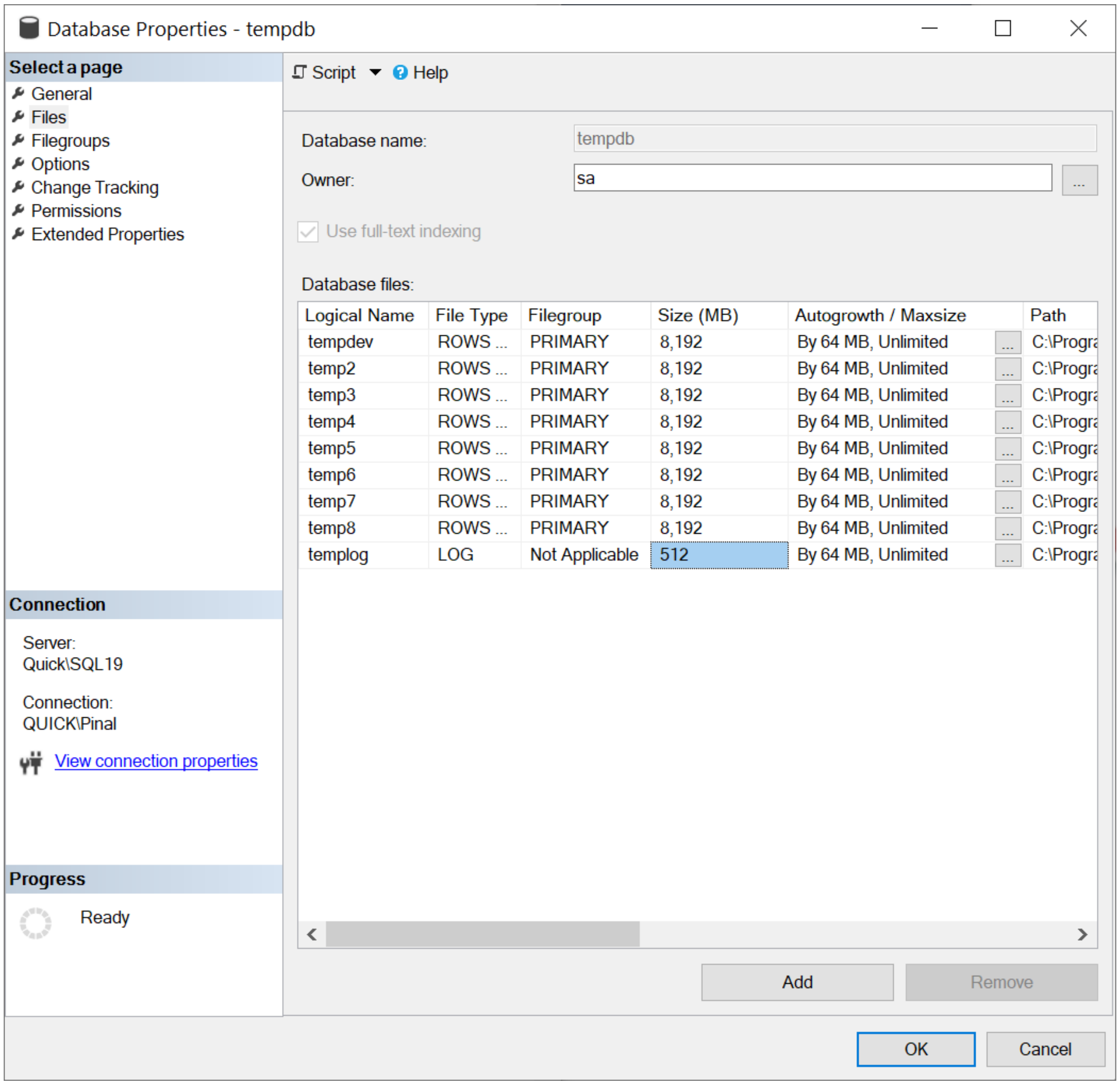
SQL Server uses tempdb for a variety of purposes. In effect, tempdb is used by almost every query and process within SQL Server as a temporary workspace. A poorly-configured tempdb can cause tremendous performance problems for SQL Server, much less a high-performance business intelligence SQL Server.

Physical files

The recommendation of Microsoft about this (see Recommendations to Reduce Allocation Contention in SQL Server tempdb Database) is to configure tempdb with the same number of data files as there are physical cores in a single processor: up to a maximum of eight files. These tempdb data files should be evenly sized and have the same file growth settings.

It is always a good idea to use database instant file initialization to improve the performance of data file grows operations with your user database as well as with the tempdb. Another best practice suggests pre-allocating space for all tempdb files by setting the file size to a value large enough to accommodate the typical workload in the environment.

For example, in a system with eight cores per socket, there should be eight tempdb data files, each equally sized, and with the same file growth settings. You can see a sample configuration below:


Database Properties - tempdb

Select a page
- General
- Files
- Filegroups
- Options
- Change Tracking
- Permissions
- Extended Properties

Connection

Server:
Quick\SQL19

Connection:
QUICK\Pinal

View connection properties

Progress

Ready

Database name: tempdb

Owner: sa

Use full-text indexing

Database files:

Logical Name	File Type	Filegroup	Size (MB)	Autogrowth / Maxsize		Path
tempdev	ROWS ...	PRIMARY	8,192	By 64 MB, Unlimited	...	C:\Progra
temp2	ROWS ...	PRIMARY	8,192	By 64 MB, Unlimited	...	C:\Progra
temp3	ROWS ...	PRIMARY	8,192	By 64 MB, Unlimited	...	C:\Progra
temp4	ROWS ...	PRIMARY	8,192	By 64 MB, Unlimited	...	C:\Progra
temp5	ROWS ...	PRIMARY	8,192	By 64 MB, Unlimited	...	C:\Progra
temp6	ROWS ...	PRIMARY	8,192	By 64 MB, Unlimited	...	C:\Progra
temp7	ROWS ...	PRIMARY	8,192	By 64 MB, Unlimited	...	C:\Progra
temp8	ROWS ...	PRIMARY	8,192	By 64 MB, Unlimited	...	C:\Progra
templog	LOG	Not Applicable	512	By 64 MB, Unlimited	...	C:\Progra

The linked Microsoft knowledge base article provides additional settings changes that should be considered if tempdb can be traced to a parallelism bottleneck. To verify if tempdb is a performance problem, refer to Demystify Tempdb Performance & Management, TempDB Performance Configuration, and How to Tell When TempDB is a Problem.

Monitoring tempdb is important. SQL Server ships with several performance counters and dynamic management objects to make monitoring tempdb easier. It is possible to work these counters and distributed management objects into an existing monitoring solution; both SQL Inventory Manager and SQL Diagnostic Manager ship with tempdb monitoring already configured. Whatever option you choose, make sure you monitor tempdb health. It is critical for SQL Server performance.

AUGMENTING PERFORMANCE WITH TRACE FLAGS

Multiple sources, including Microsoft knowledgebase articles, recommend enabling specific trace flags and startup options to improve SQL Server performance for business intelligence workloads. It is up to each database administrator to test these different trace flags and determine whether or not the modifications will work well for a given SQL Server workload. Before modifying the behavior of SQL Server, make sure the changes provide a measurable benefit.

FINAL THOUGHTS

BI SQL Server topology

Although it is tempting to combine SQL Server services on a single piece of hardware, the best practice is to separate these resources across multiple computers. SQL Server Reporting Services, SQL Server Integration Services, and SQL Server Analysis Services have a different performance profile. Mixing these services with SQL Server makes tuning and monitoring more difficult.

For the best business intelligence application performance, isolate SQL Server from other supporting services. Place tools like SQL Server Analysis Services, SQL Server Reporting Services, and SQL Server Integrations Services on a separate SQL Server. These tools, like SQL Server, need monitoring and fine-tuning. Although these are not the primary SQL Server engine, database administrators are frequently called upon to monitor these products since they are part of the SQL Server family.

Wrapping Up

When configuring SQL Server for high performance, unpredictable business intelligence workloads is not a mystery or secret art. It does require hard work and careful configuration from the database administrator. However, a mix of well-designed hardware and adequately configured software can ensure SQL Server success.

ABOUT THE AUTHOR

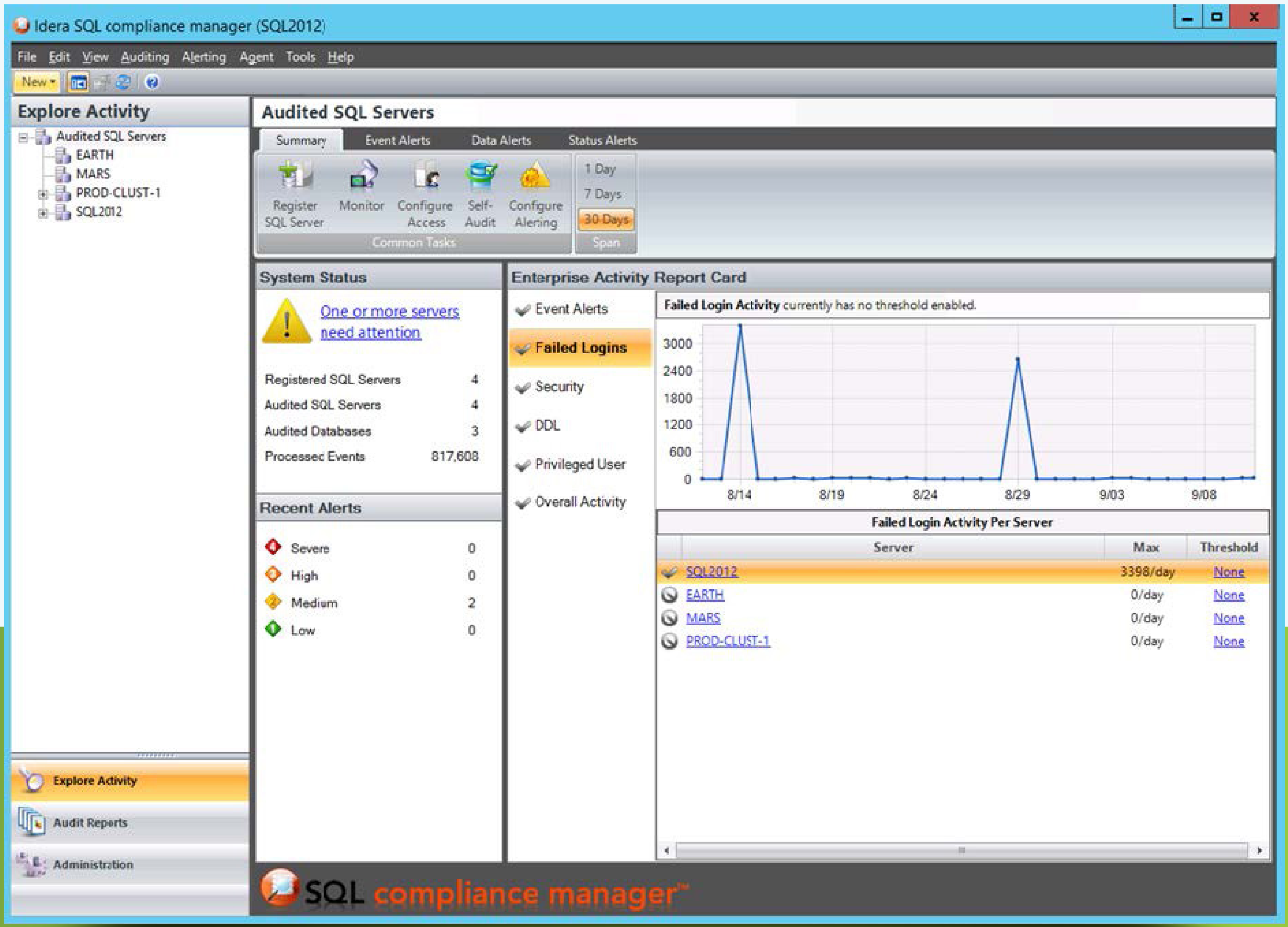
Jeremiah Peschka is a database administrator & developer with a history of delivering results and mitigating risks. He loves building software, helping developers, database administrators, and engineers build fast, robust, scalable solutions. Jeremiah is a Microsoft Most Valued Professional, Microsoft Certified IT Professional for SQL Server development and administration, and a Cloudera Certified Developer for Apache Hadoop. He is also a big fan of coffee.

SQL BI MANAGER

MONITOR SQL SERVER BUSINESS INTELLIGENCE

- Comprehensive monitoring of SSAS, SSRS, & SSIS health
- See detailed SSAS performance metrics, query activity, and sessions
- Identify issues with object utilization in both Cube and Tabular models
- Monitor both user and report activity in SSRS
- View SSIS package execution details and history

Start for FREE



System Status

One or more servers need attention

Registered SQL Servers	4
Audited SQL Servers	4
Audited Databases	3
Processed Events	817,608

Recent Alerts

Severe	0
High	0
Medium	2
Low	0

Enterprise Activity Report Card

Failed Login Activity currently has no threshold enabled.

Failed Login Activity Per Server

Server	Max	Threshold
SQL2012	3398/day	None
EARTH	0/day	None
MARS	0/day	None
PROD-CLUST-1	0/day	None

IDERA

IDERA.com

REFERENCES

Demystify Tempdb Performance & Management
http://www.idera.com/resourcecentral/whitepapers/demystify-tempdb-performance-and-management

Measuring Disk Performance with Crystal Disk Mark
http://www.brentozar.com/archive/2012/03/how-fast-your-san-or-how-slow/

Measuring Disk Performance with SQLIO
http://www.brentozar.com/archive/2008/09/finding-your-san-bottlenecks-with-sqlio/

Recommendations and Guidelines for the “max degree of parallelism” Configuration Option in SQL Server
http://support.microsoft.com/kb/2806535/en-us

Recommendations to Reduce Resource Allocation Contention in SQL Server tempdb Database
http://support.microsoft.com/kb/2154845

Resource Governor
http://msdn.microsoft.com/en-us/library/bb933866.aspx

SQL Server Chargeback Strategies, Part 1
http://www.informit.com/guides/content.aspx?g=sqlserver&seqNum=311

SQL Server Chargeback Strategies, Part 2
http://www.informit.com/guides/content.aspx?g=sqlserver&seqNum=312

TempDB Performance Configuration
http://www.brentozar.com/sql/tempdb-performance-and-configuration/

How to Tell When TempDB is a Problem
http://www.brentozar.com/archive/2011/11/how-tell-when-tempdb-problem-webcast-video/