Best Practices

Dell EMC SC Series: Microsoft SQL Server Best Practices

Abstract

This document describes best practices when using Dell EMC™ SC Series arrays with Microsoft® SQL Server®.

June 2019

CML1057

Revisions

Date	Description
October 2007	Initial release
December 2007	Technical review
June 2008	Technical review
July 2010	Technical review
June 2012	Technical review
May 2014	Technical review; included some of the new features of SQL Server 2014 related to I/O
July 2014	Technical review; updated Storage Validation and guidance on SQLIO
March 2016	Added content for in-memory, new storage profiles, and disk types
July 2016	Technical review for SQL Server 2016
June 2019	Technical review; template update

Acknowledgements

Author: Doug Bernhardt

Table of contents

Executive summary

This document describes the best practices for running Microsoft® SQL Server® (versions 2008 and later) with Dell EMC™ SC Series arrays. SQL Server performance tuning and profiling are beyond the scope of this paper. Due to dependencies on application design and requirements, environments and their recommendations can differ greatly. Visit Microsoft SQL Server Documentation for in-depth information on tuning SQL Server databases.

The audience for this document is database administrators, system administrators, storage administrators, and architects who analyze, design, and maintain robust database and storage systems. Readers should be familiar with Microsoft SQL Server, Microsoft Windows Server®, and SC Series arrays.

1 SQL Server on SC Series arrays

An SC Series array provides a flexible, easy-to-manage storage platform that helps meet the demanding uptime and performance requirements common in SQL Server environments.

1.1 Manageability

SQL Server environments are often very dynamic. An SC Series array makes it easy to adapt the storage configuration to meet changing database requirements. Volumes can be created or expanded with very little effort. Data can be easily moved between tiers or RAID levels. Disks can be added to increase performance or capacity on the fly. New storage technology can be integrated when it becomes available. All of this can be done while SQL Server remains online and available for applications.

1.2 High availability

The uptime of storage systems is more important than ever, as organizations increasingly grow dependent on SQL Server databases. The SC Series array was designed with high availability and fault tolerance in mind. Redundant hardware and RAID technology protect against an unplanned outage due to the failure of a component. Maintenance tasks such as controller firmware upgrades, hardware expansions, or hardware replacements can all be done without taking databases offline. SC Series features such as Live Migrate and Live Volume allow workloads to be shifted between SC Series arrays with zero downtime. The result is a storage system that rarely needs to be taken down and provides a solid foundation for SQL Server databases.

1.3 Performance

SQL Server is an I/O-intensive application that requires high-performance storage. SC Series arrays make it easy to leverage a large number of drives to support SQL Server workloads. Solid state drives (SSDs) can also be integrated into an SC Series array for those environments that require extremely low latency. A variety of connectivity options exist, including 32Gb Fibre Channel and 100Gb iSCSI, to ensure that data can be moved quickly and efficiently between the database server and the storage. Increasing performance by adding drives or new storage technology can be done dynamically to ensure good performance in the future.

2 Sizing for SQL Server

The I/O subsystem is a critical component of any SQL Server environment. Sizing and configuring a storage system without understanding the I/O requirements can have disastrous consequences. Monitoring performance in an existing environment using a tool like Live Optics can help define the I/O requirements. For best results, capture performance statistics for a time period of at least 24 hours that includes the system peak workload.

2.1 Key I/O performance metrics

A good understanding of the key metrics used to describe I/O performance is required to successfully define I/O requirements and size storage systems.

2.1.1 IOPS

I/O operations per second (IOPS) describe the number of reads and writes occurring each second. This is the primary metric used when designing OLTP systems. For arrays using spinning disks, this metric is key for determining the number of disks required in the array. Arrays using SSDs typically provide enough IOPS once throughput and capacity requirements are met. The number of IOPS is reported by Windows Performance Monitor using the disk reads/sec, disk writes/sec and disk transfers/sec physical disk counters.

2.1.2 Throughput

Typically expressed in megabytes per second (MB/s), throughput describes the amount of data transferred between the server and the storage. Throughput can be calculated using the number of IOPS and the average I/O size (throughput = IOPS * average I/O size). This is the primary metric used to design the path between the server(s) and the storage as well as the number of drives required. A small number of SSDs can often meet IOPS requirements but may not meet throughput requirements. Throughput is reported by Windows Performance Monitor using the disk read bytes/sec, disk write bytes/sec and disk bytes/sec physical disk counters.

2.1.3 Latency

Typically expressed in milliseconds (ms), latency describes the amount of time an I/O operation takes to complete. When a server experiences high latency, this generally indicates an I/O bottleneck in the system. Latency is reported by Windows Performance Monitor using the avg. disk sec/read, avg. disk sec/write and avg. disk sec/transfer physical disk counters.

2.2 The write penalty

RAID technology is used to protect data from a disk failure on SC Series arrays. For each write sent to an SC Series volume, additional I/O is performed on the physical disks to provide the redundancy needed to protect the incoming data. The amount of additional I/O is determined by the RAID configuration for the volume. The following table shows the number of I/Os required when performing a write on the volume. The number of I/Os required to complete a write is referred to as the write penalty.

Table 1 Write penalty by RAID type

RAID type	Write penalty	I/O description
RAID 10	2	2 writes
RAID 10 DM	3	3 writes
RAID 5	4	2 reads, 2 writes
RAID 6	6	3 reads, 3 writes

By default, writes to SC Series volumes use RAID 10. Since RAID 10 has the lowest write penalty, it is strongly recommended to always use RAID 10 for writes. All standard SC Series storage profiles write using RAID 10.

When sizing an array, it is important to factor in the write penalty when determining the number of disks required. The number of IOPS reported by the server when gathering I/O statistics does not include the write penalty. If the IOPS target for the array design does not include the write penalty, it will be undersized. To adjust the number of IOPS to include the write penalty, use the following formula:

Number of IOPS + (*Number of IOPS* * *Write Percentage* * (*Write Penalty* – 1))

Definitions:

Number of IOPS	=	Number of IOPS generated by the server
Write Percentage	=	Percent of the I/O attributed to writes
Write Penalty	=	Number of I/Os required to complete a write

As an example, consider a server that is performing a workload of 10,000 IOPS with a mix of 70 percent reads and 30 percent writes. Since the default storage profile is used, writes use RAID 10, which has a write penalty of two. For this workload, the SC Series array needs enough disks to support 13,000 IOPS (10,000 + (10,000 * .30 * (2-1)) = 13,000).

2.3 Random I/O compared to sequential I/O

Disks can perform more sequential I/O than random I/O. This is because sequential I/O requires less overhead on the disk. While I/O statistics show how much I/O is occurring, they do not show how much is sequential I/O compared to random I/O. The virtualized storage architecture of SC Series arrays may cause sequential I/O requests from the server to become large-block random I/O on the physical disks. Performance will still be good because large-block random I/O provides comparable performance to sequential I/O. For best results when sizing an SC Series array, assume all of the I/O is random. This will ensure good performance for both random and sequential operations.

2.4 OLTP workloads

While every environment is unique, an online transaction processing (OLTP) workload typically consists of small, random reads and writes. A storage system servicing this type of workload is primarily sized based on the number of IOPS required. Different drive types have different performance capabilities. Faster spinning drives are able to provide more IOPS with lower latency than slower spinning drives. For best performance, SSDs are recommended for OLTP workloads (see section 2.7).

2.5 OLAP/DSS workloads

An online analytic processing (OLAP) or decision support system (DSS) workload is typically dominated by large, sequential reads. A storage system servicing this type of workload is primarily sized based on throughput. When designing for throughput, the performance of the entire path between the server and the disks in the SC Series array need to be considered. For best throughput, consider using at least 16Gbps Fibre Channel (FC) or 10Gbps iSCSI connectivity to the array, and 12Gbps SAS connectivity from the controllers to the disk enclosures. To meet high throughput requirements, multiple HBAs may be required in the server, the SC Series array, or both.

2.6 Sizing tiers

When sizing an SC Series array with two tiers, start by sizing tier 1 for all of the IOPS with at least 30 percent of the capacity and tier 3 for 70 percent of the capacity with at least 30 percent of the IOPS. In a two-tier system, the fastest disks are tier 1 and the slower disks are tier 3. When sizing each tier for SQL Server, consider the following:

- Accessible pages on a volume are not eligible to move to a lower tier until they are at least 12 days old.

  In environments where indexes are rebuilt frequently, accessible pages on the database volume may never be older than 12 days. All accessible pages will always be in tier 1. In those environments, tier 1 needs to be sized for 100 percent of the capacity required by SQL Server.

- The automated tiering feature of SC Series arrays automatically moves infrequently accessed data to slower, cheaper storage (tier 3), reducing storage costs without impacting performance.

  For environments where the access pattern is even across the entire data set, all data may need to reside on tier 1 to ensure good performance. In those environments, size tier 1 for 100 percent of the capacity required by SQL Server.

2.7 Using SSDs

SSDs offer significantly better performance than traditional hard drives for most I/O patterns. From a performance perspective, SSDs are suitable for storing any type of SQL Server file. While SSDs can support a large number of IOPS, it is easy to exceed throughput limits of the drive or other components in the path with the large I/Os that can be generated by SQL Server. It is important to understand the performance characteristics of SSDs as well as the I/O requirements of the SQL Server environment before implementing SSDs.

Both write-intensive and read-intensive SSDs can be used in an SC Series array. Although they provide lower capacity than read-intensive drives, write-intensive drives can perform a larger number of write cycles, making them better suited for heavy write workloads. Read-intensive drives offer a much higher capacity than write-intensive drives, but cannot endure as many write cycles, making them better suited for read-intensive workloads. Traditional SSD configurations contain only write-intensive drives used as a tier in the same way as spinning drives.

2.7.1 Traditional SSD configurations

A single SSD type can be used as the high-performance tier in an SC Series array either alone (all-flash) or combined with spinning drives (hybrid). When using SSDs as a tier that participates in automated tiering, there needs to be enough drives in the tier to handle 100 percent of the IOPS and at least 30 percent of the capacity of the volumes that use the SSD tier. Since SSDs can perform a large number of IOPS, a small number of drives may meet the IOPS requirement. However, the capacity or throughput requirements may need many drives. It is very important to have enough SSDs to meet the IOPS as well as the throughput and capacity requirements. If the SSD tier becomes full, performance will degrade significantly.

In cases where SSD capacity is limited, it may be more cost effective to place a subset of the SQL Server files on SSDs, instead of placing all files on SSDs with automated tiering. Storage profiles allow an administrator to easily control which volumes reside on SSDs. While transaction log files and tempdb files require high performance and tend to be small enough to fit on a small number of SSDs, they are not always the best choice for SSDs. In some cases, it may be better to place data files on SSDs. Analyze the access patterns of the database files to determine the best fit. If a data file is too large to fit on the available SSD drives, the portion of the data requiring high performance can be moved to a separate file group and placed on a volume that resides on SSDs.

2.7.2 Flash-optimized configurations

By leveraging the characteristics of both write-intensive and read-intensive SSDs, a flash-optimized configuration can provide both high capacity and high performance at an affordable price point, making it practical to store a large SQL Server data set solely on SSDs. When sizing a flash-optimized configuration, it is important to use enough write-intensive drives to handle the volume of writes and enough read-intensive drives to provide sufficient capacity. When working with traditional SSD configurations, the number of drives was determined primarily by capacity requirements that were often reduced due to cost considerations. Using read-intensive drives makes it cost effective to provide a large amount of capacity. However, additional write or read-intensive drives may be needed for the configuration to keep from exceeding the throughput limit of the drives.

3 Validating the storage design

Once the I/O requirements have been defined, it is easy to determine whether the hardware can provide the desired performance by running some simple tests. Diskspd is a free Microsoft utility that can simulate I/O patterns generated by SQL Server. There are several other utilities available as well. When selecting a utility to simulate I/O, it should meet the following requirements:

- Ability to configure block size
- Ability to specify number of outstanding requests
- Ability to configure test file size
- Ability to configure number of threads
- Support for multiple test files
- Does not write blocks of zeros during tests

3.1 Validating the I/O path

The first thing to test on a new configuration is the path between the server and the array. Running a large block sequential read test using small files should saturate the path between the server and array. This test verifies that all paths are fully functional and can be used for I/O traffic. This test should be run on a server and array that is dedicated to this test. Running it on a live system could cause significant performance issues.

To validate the I/O path, run a large block sequential read test using the following guidelines:

- If testing a single controller system, create one volume. If using a dual controller system, create two volumes with a volume owned by each controller.
- Format the volumes using a 64 KB allocation unit.
- Use a block size of 512 KB for the test.
- Configure the test for 32 outstanding I/Os.
- Use multiple threads. Eight is the recommended starting point.

If the displayed throughput matches the expected throughput for the number of HBA ports in the server, the paths between the server and SC Series array are set up correctly.

3.2 Validating the disks

Once the I/O path has been validated, the next step is to test the disks. For best results when testing disks on an SC Series array, use the following guidelines when configuring the test.

- In a dual-controller system, use at least one volume per controller. This ensures that I/O will be distributed across both controllers. Using both controllers more closely simulates real world activity. For best results, use the same number of volumes on each controller.
- When performing I/O tests on any storage platform, it is important to use files that are larger than the controller cache. For more accurate results, use a file size that matches the amount of data being stored. In an environment where that is not practical due to a large data set, use a file size of at least 100 GB.

- Some I/O test tools, including Diskspd, SQLIO, and IOMeter, generate files full of zeros. By default, SC Series arrays track pages full of zeroes in the metadata, but do not store them on a disk. This behavior, known as a thin write, causes inaccurate results when testing with files containing only zeros. Avoid using test utilities that write zeros for disk validation. The contents of the test file can be verified by viewing the test file with a hex editor after different stages of a test. For example, create a small test file and view it after the initial creation, as well as after the test has run for a few seconds. If the file is filled with zeros, select another utility. Diskspd and IOMeter initially create test files filled with zeros, and then writes random characters when performing write tests. To properly initialize a Diskspd or IOMeter test file, run a sequential write test until the entire file has been overwritten with non-zero data. Unfortunately, SQLIO writes zeros during write tests and therefore is not recommended for disk validation.

The purpose of this type of testing is to validate that the storage design will provide the required throughput and IOPS with acceptable latency. It is important that the test does not exceed the designed capacity of the array. For example, an SC Series array designed for a workload of 25,000 IOPS is likely to perform poorly with a workload of 50,000 IOPS. If a test is generating a workload higher than the designed capacity, adjust the workload being generated by reducing the number of threads and/or outstanding I/Os.

The results of the DPACK analysis provide an I/O target to simulate using these tests. To get an idea of the performance capabilities of the array, run I/O tests with a range of I/O sizes commonly seen with SQL Server. When testing random I/O, test with an I/O size of 8 KB and 64 KB. When testing sequential I/O, start with I/O sizes of 8 KB and 64 KB. Since processes like read ahead scans and backups can issue much larger sequential I/O, it is a good idea to also test block sizes up to 1024 KB.

4 Storage setup and configuration

4.1 Single or multiple disk pools

Dell EMC recommends using a single virtual disk pool when implementing SQL Server. This provides better performance by leveraging the aggregate I/O bandwidth of all disks to service I/O requests from SQL Server. A single disk pool is also easier to manage, allowing an administrator to quickly and easily adapt the storage system to satisfy the ever-changing workloads that are common in SQL Server environments.

4.2 Creating volumes

SC Series storage is virtualized to take advantage of all the drives in the disk pool. Characteristics such as RAID level and storage tier can have a big impact on performance and are configured at the volume level. SC Series snapshots are also configured at the volume level. There are many different types of files that are part of a SQL Server instance. Those different types of data often have different performance and snapshot requirements. For performance sensitive applications, Dell EMC recommends creating at least five volumes for an instance of SQL Server as shown in the following table.

Table 2 Volume provisioning recommendations

User DB data	**At least 1 per instance**	**Lower performance may be acceptable**	**Frequent snapshots, same schedule as log volume**
User DB transaction log	At least 1 per instance	High performance required	Frequent snapshots, same schedule as data volume(s)
Data root directory (includes system DBs)	1 per instance	Lower performance may be acceptable	Infrequent snapshots, independent schedule
Tempdb data and transaction log	1 per instance	High performance may be required	No snapshots
Native SQL Server backup	1 per instance	Lower performance may be acceptable	Snapshots optional, independent schedule
Memory-Optimized Filegroup (if used)	At least 1 per instance	High performance required	Frequent snapshots, same schedule as log volume

4.2.1 Performance considerations

If there is a group of databases that require high performance and another group that does not, consider creating a set of volumes for each group of databases. Even if there is only one tier of storage, this strategy will make it easier to adjust the storage configuration in the future. For example, having the high-performance databases on separate volumes will make it very easy to move them to a higher performance tier in the future.

Having multiple volumes has an additional benefit in dual-controller systems. All I/O requests for a given volume are processed by the controller that owns the volume. While a volume can be owned by either controller, a volume is only owned by one controller at a time. Having many volumes makes it easier to distribute the I/O load evenly across both controllers. Databases that have very high-performance requirements can be spread across two or more data files on separate volumes to leverage resources on both controllers.

4.2.2 Flexibility and manageability

For ultimate flexibility, create a volume for each user database file. This provides the ability to independently optimize the storage and snapshot configuration for each individual database. With thin provisioning, there is no space penalty for creating a lot of volumes. However, a large number of volumes can be difficult to manage, especially in virtualized environments. It is up to the DBA or storage administrator to find the right balance between flexibility and maintainability when determining the number of volumes to create. Virtualized SQL Server environments are a good example of where it may make sense to place multiple file types on a single volume. Understanding the database I/O patterns is critical to making the best decisions.

4.3 Storage profiles

Storage profiles define the RAID level used to protect data on a volume and the tiers where that data is stored. The information in the profile is used by Data Progression when moving existing pages, as well as by new SC Series pages. In most environments, using the default storage profile **Recommended (All Tiers)** provides good I/O performance for all types of database volumes. It is strongly recommended to use this storage profile first and evaluate its suitability before attempting to change the storage profiles.

4.3.1 User database data file considerations

On volumes storing the data files, the **Recommended (All Tiers)** storage profile keeps the highly active parts of the database on tier 1 and the less active parts on lower tiers. In most cases, this provides the best performance. However, there are some environments where the entire dataset is highly active and needs the performance of tier 1. Before changing the storage profile to force the data volume to live only on tier 1, consider the impact that the frequency of both index maintenance and snapshots has on Data Progression. Using the **Recommended (All Tiers)** storage profile allows inaccessible frozen pages to be moved to tier 3, freeing up tier 1 resources for active pages.

4.3.2 User database transaction log file considerations

The transaction log requires good performance and should reside on tier 1. This storage component has the greatest impact on transaction latency. Writes to the transaction log start at the beginning of the file, go to the end of the file and then start again at the beginning of the file. The **Recommended (All Tiers)** storage profile allows inaccessible frozen pages to be moved to tier 3, freeing up tier 1 resources for active pages.

4.3.3 Tempdb database file considerations

Tempdb files can require high performance and generally should reside on tier 1. Some applications use tempdb heavily and others hardly use it at all. If your applications require high performance for tempdb and you want it to remain on tier 1 this is accomplished by selecting the High Priority Storage Profile for Standard storage or the Write Intensive Storage Profile for Flash Optimized storage.

4.3.4 System database file considerations

The system databases (master, model and msdb) generally do not have special performance requirements. Use the **Recommended (All Tiers)** storage profile for the system database volumes.

4.3.5 SSD considerations

The type of files that benefit the most from SSDs varies from environment to environment. Monitor the performance of the database volumes to determine the best fit for SSDs. If there are not enough SSDs to function as tier 1 storage for all SC Series volumes, the Storage Profiles for Flash Optimized Storage can be used to control tiering.

4.4 SC Series snapshots

SC Series snapshots, previously called Replays, are very powerful when combined with SQL Server. By leveraging snapshots, protecting and recovering databases is very fast and space efficient, even for large databases.

Dell EMC strongly recommends using Replay Manager to take snapshots of database volumes while SQL Server is online. Using Replay Manager is the only way to ensure transactional consistency, since writes are frozen inside of SQL Server while the snapshots are taken. Snapshots of active database volumes taken without Replay Manager may not be usable for database recovery. Replay Manager also allows snapshots to be used as the starting point of a restore chain that includes transaction log backups, as databases can be restored without recovery. In addition, restore points created by Replay Manager are fully logged in SQL Server providing complete backup integration.

Even though Replay Manager allows specific databases to be selected, snapshots are taken on the entire volume. If all databases on a given set of volumes are not backed up together, disk space could be wasted on the array. Consider placing the databases for each backup set on a separate set of volumes.

Placing a large database on its own set of volumes allows additional recovery flexibility when using Replay Manager. If a database is on its own volumes and in its own backup set, the database can be recovered very quickly using the Resync method in the Replay Manager Explorer or the **Resync-RMRestorePoint** PowerShell cmdlet without affecting other databases. This can be a big time saver when recovering a large database using a Replay Manager restore point and transaction log backups.

When taking snapshots of database volumes without Replay Manager, it is recommended to use a consistency group. This will increase the odds of a successful database recovery using those snapshots. Using a consistency group on the SC Series array allows snapshots to be taken on multiple volumes at the same point in time. For information on how to setup a consistency group, see the "Managing Snapshot Profiles" section in the *Dell Storage Manager Administrator's Guide*.

Dell EMC recommends taking a snapshot of each user database volume at least once per day, with a retention period of 25 hours or longer. Data Progression moves frozen pages in the snapshot according to settings in the storage profile. If the default storage profile **Recommended (All Tiers)** is used, frozen pages in the snapshot are moved to RAID 5, while writes will still occur at RAID 10. This allows tier 1 space to be used more efficiently.

Tasks ideally suited for using snapshots with SQL Server include the following:

- Restoring a copy of a database to quickly recover data or objects after errant updates or deletes
- Restoring a copy of a database on the original server or another server for development, testing or reporting
- Replicating databases to a DR location (during a DR test, databases can be recovered without interrupting replication)
- Big data environments where the size of the databases makes frequent backups impractical.

In environments where other Dell EMC array models exist, Dell EMC AppSync™ can also be used for taking SQL Server database snapshots on SC Series arrays, as well as several other Dell EMC arrays. For more information about using snapshots with SQL Server using Replay Manager, see the *Replay Manager Administrator's Guide*. A wealth of resources on Dell EMC AppSync can be found in the Dell EMC AppSync site.

5 Windows setup and configuration

5.1 Allocation unit size

Use a 64 KB allocation unit size when formatting volumes that will contain database files (transaction log and data) or database backups.

5.2 MPIO

Set the MPIO policy to **Round Robin** for all database volumes. This is the default for Windows Server 2008 and newer. It allows all paths to be used, enabling higher throughput between the server and the array. This setting works best for most environments as it is easy to manage and performs very well.

For database servers with a large number of I/O ports, the overhead of the **Round Robin** MPIO policy can reduce the maximum throughput to the storage array. To maximize throughput, create a volume for each port on the server and use the **Fail Over Only** MPIO policy to define a single, unique, active path for each volume. All other paths should be defined as standby paths. By using a data file of the same size on each volume, the data will be evenly distributed across the volumes. This strategy is more complex to setup and maintain. It should only be used in environments that require maximum throughput.

5.3 Partition alignment

An offset of 64 KB, or a multiple of 64 KB, is recommended for volumes created on Windows Server 2003 and earlier. New volumes created on Windows Server 2008 or newer should already be aligned since 1024 KB is the default offset.

6 Server setup and configuration

6.1 HBA considerations

Be sure that the HBA firmware and drivers are up to date. As with any update, it is important to verify functionality in a test or QA environment before implementing in production.

SQL Server is a very I/O intensive application. It is important to configure the queue depth of the HBA properly to optimize performance. See the "HBA Server Settings" section in the Dell Storage Manager *Administrator's Guide*.

6.2 iSCSI considerations

Use at least 1Gb Ethernet to connect to the SC Series storage array. For best performance, use 10Gb Ethernet.

To provide consistent performance, create a separate network for iSCSI traffic.

If using the Microsoft iSCSI Initiator, Dell EMC recommends the following:

- Make the SQL Server service dependent on the Microsoft iSCSI Initiator Service.
- Use dedicated network interface cards (NICs) for iSCSI connections to the array.
- On each of the dedicated NICs, disable all features, protocols, and clients except for TCP/IPv4.
- Explicitly create each of the iSCSI connections, rather than using the **Quick Connect** button.

7 Optimizing SQL Server I/O

7.1 Memory

Unnecessary I/O can be avoided and performance can be increased by allocating the proper amount of memory to SQL Server. SQL Server performs all I/O through the buffer pool (cache) and therefore uses a large portion of its memory allocation for the buffer pool. Ideally, when SQL Server performs I/O, the data is already in the buffer pool and it does not need to go to disk. This type of I/O is referred to as logical I/O and is the most desirable because it results in the best performance. If the data SQL Server needs does not reside in the buffer pool, it will need to access disk resulting in physical I/O.

Proper memory allocation is critical to SQL Server performance and can improve storage performance as well. In many cases, SQL Server and storage performance can be further improved by adding additional memory. Generally speaking, the more memory the better but there is a point of diminishing returns that is unique to each environment.

7.2 Buffer pool extension

Starting with SQL Server 2014, the buffer pool can be extended to a file on the file system to provide additional space to cache data or index pages. Only pages that have not been modified can be stored in the buffer pool extension. While the extension file can reside on any type of storage, it is best to store it on local solid-state devices. Using this feature can provide significant performance benefits without adding memory to the database server. By caching more pages on the server, the I/O load on the array is reduced.

While the SSD tier on the SC Series array can be used for the buffer pool extension, the SSD tier is typically more effective when used as primary storage. In cases where there is not enough SSD capacity to be effective for primary storage of the database volumes, the SSD tier can be considered for the buffer pool extension. When placing the buffer pool extension on the array, use the following guidelines:

- Create a separate volume for the buffer pool extension.
- Assign a storage profile that is configured to store all data on the SSD tier (tier 1).
- Do not take snapshots of the buffer pool extension volume.

7.3 Persistent memory

SQL Server 2016 introduced support for persistent memory (PMEM) and the capabilities are being expanded with SQL Server 2019 to cover more scenarios as well as the Linux® operating system. In many cases, PMEM can be used to accelerate challenging I/O workloads and make I/O patterns more efficient with SQL Server PMEM features such as tail-of-log-cache and Hybrid Buffer Pool. Virtualized environments running Hyper-V® or VMware® can also take advantage of PMEM, making it a wise investment. Dell EMC supports PMEM starting with the Dell EMC PowerEdge™ 14th generation (14G) servers. For more information, see the Microsoft article Storage Class Memory in SQL Server 2016 SP1 and the Dell EMC document Dell EMC NVDIMM-N Persistent Memory User Guide.

7.4 Database compression

The overall I/O workload can be reduced by enabling database compression in SQL Server. While there is a tradeoff in terms of CPU utilization on the database server, it is still a viable option to consider and test in any environment. Database compression reduces I/O by reducing the amount of data that needs to be stored. The

SQL Server data pages are compressed in memory before being written to disk resulting in fewer pages needed to store the same number of rows and therefore less I/O.

7.5 Instant file initialization

By default, SQL Server writes zeros to the data file during the allocation process. The process of zeroing out the data files consumes I/O and acquires locks as the SQL Server data pages are written. This activity can occur for minutes or even hours depending on the file size. The thin write feature saves space by not physically storing all the zero data. It will simply mark a bit in the metadata indicating these pages are filled with zeros. Although thin writing mitigates the space concern, there is still processing that occurs on the controller to receive and inspect the data. While this may seem minor, writing zeros to these files can occur at critical periods when time and performance are critical such as database auto growth, expanding a full data file, replication, or restoring a database as part of a disaster recovery event.

When Instant File Initialization is enabled, SQL Server will skip the process of zeroing out its data files when allocating space. Dell EMC recommends enabling Instant File Initialization.

7.6 Resource Governor

The Resource Governor was added in SQL Server 2008 to allow database administrators to limit the CPU and memory resources a query is able to consume. This feature was enhanced in SQL Server 2014 to allow I/O resources to be limited as well. For example, the Resource Governor can be used to reduce the impact of a user running an I/O intensive report by limiting the maximum number of IOPS that user can perform. While a query throttled by the Resource Governor will take more time to complete, overall database performance will be better.

7.7 Database design considerations

Reducing SQL Server I/O requires a holistic approach. Many of the items in this section will require involvement from the whole team responsible for the SQL Server applications including the business owner, architect, developer, database administrator, and system administrator. Decisions at the design level have a multiplied downstream impact as data is written and read multiple times and duplicated in various types of database copies including databases copied for other uses such as testing and reporting, replicated databases, replicated storage, and backups. There are more copies of databases that people retain in various forms than they realize.

One of the most challenging aspects of SQL Server is that the I/O pattern and the amount of I/O that is generated can vary greatly depending on the application, even if those applications have databases of the same size. This is because the design of both the database and the data access code control SQL Server I/O.

Database tuning can be one of the most cost-effective ways to reduce I/O and improve scalability. Database tuning is a very large topic and is beyond the scope of this document. However, there are numerous resources available in both online and print form. At a high level, the following areas should be considered when tuning a database to reduce I/O.

7.7.1 Database design

The foundation of the entire database and the schema for how data will be stored and ultimately accessed is determined by the database design. The database design should support both usability and efficient data access. This includes efficient table design and data types as well as indexes, partitioning, and other features that can improve efficiency. It is common for database design to only be focused on usability.

7.7.2 Query design

How a query is written can greatly affect the amount of I/O SQL Server needs to perform when executing the query. Queries should return only the required amount of data in the most efficient manner possible. There are many great tools and resources available to help design efficient queries.

7.7.3 Application design

Consider how applications are using the data and the manner in which it is requested. Sometimes code and component reuse can result in the same data being unnecessarily retrieved over and over again. All data access should be purposeful.

7.7.4 Maintenance

SQL Server uses a cost-based optimizer to generate query plans for data access. These plans are based on the statistics regarding how data is distributed in the tables. If the statistics are inaccurate, bad query plans may result and unnecessary I/O will be performed. Proper database maintenance includes ensuring that statistics are up to date.

Frequent data modifications can also lead to fragmentation within SQL Server data files, producing unnecessary I/O. Fragmentation can be addressed through index reorganization or rebuilds as part of regular database maintenance.

The database maintenance process itself can also have a large I/O impact. Typically, every table and index does not need to be rebuilt or reorganized every time maintenance is run. In addition, table partitioning strategies can also be leveraged to make the maintenance process more selective. Consider implementing maintenance scripts to perform maintenance on an as-needed basis.

For mission critical databases these maintenance activities need to be considered as part of the overall design. If maintenance is not considered as part of the overall process, issues can arise, such as unmanageable sizes and feature incompatibilities that limit available options and strategies.

A Additional resources

A.1 Technical support and resources

Dell.com/support is focused on meeting customer needs with proven services and support.

Storage technical documents and videos provide expertise that helps to ensure customer success on Dell EMC storage platforms.

A.1 SQL Server resources

There is a tremendous amount of SQL Server information available online. While not a complete list, the table below contains several useful links.

Table 3 Useful links

Description	**Link**
Dell EMC SQL Server Solutions	http://www.dell.com/sql
Microsoft SQL Server homepage	http://www.microsoft.com/sql
SQL Server Customer Advisory Team	https://techcommunity.microsoft.com/t5/DataCAT/bg-p/DataCAT
Microsoft online SQL Server forum	http://social.msdn.microsoft.com/Forums/en-US/category/sqlserver
Professional Association for SQL Server	http://sqlpass.org
Online SQL Server resources	http://www.sqlservercentral.com http://www.sqlteam.com http://www.sql-server-performance.com