Technical Report

Best Practices Guide for Microsoft SQL Server with ONTAP

Pat Sinthusan, NetApp

August 2020 | TR-4590

Abstract

This best practices guide enables storage administrators and database administrators to successfully deploy Microsoft SQL Server on NetApp® storage.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction

SQL Server is the foundation of Microsoft's data platform, delivering mission-critical performance with in-memory technologies and faster insights on any data, whether on the premises or in the cloud. Microsoft SQL Server builds on the mission-critical capabilities delivered in prior releases by providing breakthrough performance, availability, and manageability for mission-critical applications. The storage system is a key factor in the overall performance of a SQL Server database. NetApp provides several products to allow your SQL Server database to deliver enterprise-class performance while providing world-class tools to manage your environment.

1.1 Purpose and Scope

This document describes best practices and offers insight into design considerations for deploying SQL Server on NetApp storage systems running NetApp ONTAP® software, with the goal of achieving effective and efficient storage deployment and end-to-end data protection and retention planning. The scope of this guide is limited to technical design guidelines based on the design principles and preferred standards that NetApp recommends for storage infrastructure when deploying SQL Server. The end-to-end implementation is out of the scope of this report.

The best practices and recommendations described in this guide enable SQL Server architects and NetApp storage administrators to plan a highly available and easy-to-manage SQL Server environment and to meet stringent SLAs. NetApp assumes that the reader has working knowledge of the following:

- NetApp ONTAP software
  - NetApp SnapCenter® as backup software, which includes:
    - SnapCenter Plug-in for Microsoft Windows
    - SnapCenter Plug-in for SQL Server
- Microsoft SQL Server architecture and administration

For configuration compatibility across the NetApp stack, see the NetApp Interoperability Matrix Tool (IMT).

1.2 Intended Audience

This technical report is intended for NetApp customers, partners, employees, and field personnel who are responsible for deploying a SQL Server database solution in a customer environment. NetApp assumes that the reader is familiar with the various components of the listed solution previously.

2 SQL Server Workload Types

The SQL Server database platform can support several applications. Before deploying SQL Server, you must understand the database workload requirements of the applications that your SQL Server instances support. Each application has different requirements for capacity, performance, and availability, and therefore each database should be designed to optimally support those requirements. Many organizations classify databases into multiple management tiers, using application requirements to define SLAs. SQL Server workloads can be described as follows:

- OLTP databases are often also the most critical databases in an organization. These databases usually back customer-facing applications and are considered essential to the company’s core operations. Mission-critical OLTP databases and the applications they support often have SLAs that require high levels of performance and are sensitive to performance degradation and availability. They might also be candidates for Always On Failover Clusters or Always On Availability Groups. The I/O mix of these types of databases is usually characterized by 75% to 90% random read and 25% to 10% write.

- Decision support system (DSS) databases can be also referred to as data warehouses. These databases are mission critical in many organizations that rely on analytics for their business. These databases are sensitive to CPU utilization and read operations from disk when queries are being run. In many organizations, DSS databases are the most critical during the month, quarter, and year end. This workload typically has a 100% read I/O mix.

2.1 Testing Benchmark

The Transaction Process Council (TPC) is a nonprofit corporation founded to define transaction processing and database benchmarks and to disseminate objective, verifiable TPC performance data to the industry. TPC tests simulate complete compute environments in which a population of users executes transactions against databases. Table 1 summarizes the typical SQL Server testing workloads.

Table 1) Typical SQL Server workloads.

Workload Type	TPC Test	Read/Write Ratio (Percentages)
OLTP	TPC-C	~75/25
	TPC-E	~90/10
DSS	TPC-H	~100/0

Although various workload generation options are available, we generally focus our efforts on measuring the performance of SQL Server databases when handling transactional workloads, and we use the TPC-E tools from Microsoft or TPC-H using HammerDB (HammerDB.com). The detailed instructions on how to use these specific benchmarks are beyond the scope of this document.

3 SQL Server Configuration

This section provides general guidance about how to configure SQL Server settings that should be considered before and after installing SQL Server.

3.1 Shared Instance Versus Dedicated Instance

If an application has many schemas and stored procedures, it could potentially affect other apps that share a SQL Server instance. Instance resources could potentially become divided or locked, which in turn causes performance issues for other apps that have databases hosted on the shared SQL Server instance.

Troubleshooting performance issues can be complicated because you must figure out which instance is the root cause. This question is weighed against the costs of operating system licenses and SQL Server licenses. If application performance is paramount, then a dedicated instance is highly recommended.

Microsoft licenses SQL Server per core at the server level and not per instance. For this reason, database administrators are tempted to install as many SQL Server instances as the server can handle to save on licensing costs, which can lead to major performance issues later.

To obtain optimal performance, NetApp recommends choosing dedicated SQL Server instances whenever possible.

3.2 CPU Configuration

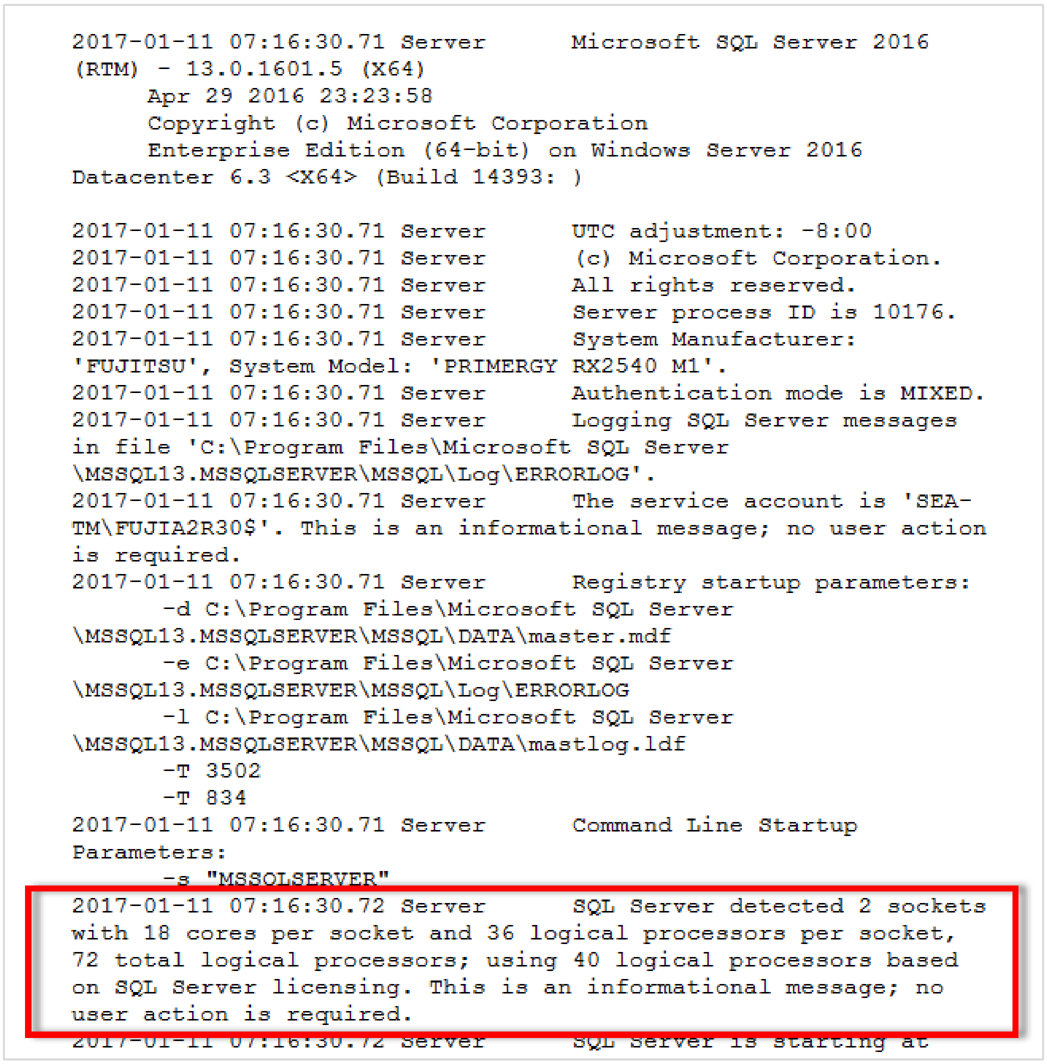
Hyperthreading

Hyperthreading is Intel’s proprietary simultaneous multithreading (SMT) implementation, which improves parallelization of computations (multitasking) performed on x86 microprocessors. Hardware that uses hyperthreading allows the logical hyperthread CPUs to appear as physical CPUs to the operating system. SQL Server then sees the physical CPUs, which the operating system presents, and so can use the hyperthreaded processors.

The caveat here is that each SQL Server version has its own limitations on the compute power it can use. For more information, see Compute Capacity Limits by Edition of SQL Server.

There are two main schools of thought when licensing SQL Server. The first is known as a server + client access license (CAL) model; the second is the per processor core model. Although you can access all the product features available in SQL Server with the server + CAL strategy, there is a hardware limit of 20 CPU cores per socket. Even if you have SQL Server Enterprise Edition + CAL for a server with more than 20 CPU cores per socket, the application cannot use all those cores at a time on that instance. Figure 1 shows the SQL Server log message after startup indicating the enforcement of the core limit.

Figure 1) Log entries indicate number of cores being used after SQL Server startup.



```
2017-01-11 07:16:30.71 Server      Microsoft SQL Server 2016
(RTM) - 13.0.1601.5 (X64)
     Apr 29 2016 23:23:58
     Copyright (c) Microsoft Corporation
     Enterprise Edition (64-bit) on Windows Server 2016
Datacenter 6.3 <X64> (Build 14393: )

2017-01-11 07:16:30.71 Server      UTC adjustment: -8:00
2017-01-11 07:16:30.71 Server      (c) Microsoft Corporation.
2017-01-11 07:16:30.71 Server      All rights reserved.
2017-01-11 07:16:30.71 Server      Server process ID is 10176.
2017-01-11 07:16:30.71 Server      System Manufacturer:
'FUJITSU', System Model: 'PRIMERGY RX2540 M1'.
2017-01-11 07:16:30.71 Server      Authentication mode is MIXED.
2017-01-11 07:16:30.71 Server      Logging SQL Server messages
in file 'C:\Program Files\Microsoft SQL Server
\MSSQL13.MSSQLSERVER\MSSQL\Log\ERRORLOG'.
2017-01-11 07:16:30.71 Server      The service account is 'SEA-
TM\FUJIA2R30$'. This is an informational message; no user action
is required.
2017-01-11 07:16:30.71 Server      Registry startup parameters:
      -d C:\Program Files\Microsoft SQL Server
\MSSQL13.MSSQLSERVER\MSSQL\DATA\master.mdf
      -e C:\Program Files\Microsoft SQL Server
\MSSQL13.MSSQLSERVER\MSSQL\Log\ERRORLOG
      -l C:\Program Files\Microsoft SQL Server
\MSSQL13.MSSQLSERVER\MSSQL\DATA\mastlog.ldf
      -T 3502
      -T 834
2017-01-11 07:16:30.71 Server      Command Line Startup
Parameters:
      -s "MSSQLSERVER"
2017-01-11 07:16:30.72 Server      SQL Server detected 2 sockets
with 18 cores per socket and 36 logical processors per socket,
72 total logical processors; using 40 logical processors based
on SQL Server licensing. This is an informational message; no
user action is required.
2017-01-11 07:16:30.72 Server      SQL Server is starting at
```

Therefore, to use all CPUs, you should use the per-processor core license. For detailed information about SQL Server licensing, see SQL Server 2019: Your modern data platform.

Nonuniform Memory Access

Nonuniform memory access (NUMA) is a memory-access optimization method that helps increase processor speed without increasing the load on the processor bus. If NUMA is configured on the server where SQL Server is installed, no additional configuration is required because SQL Server is NUMA aware and performs well on NUMA hardware.

Processor Affinity

You are unlikely ever to need to alter the processor affinity defaults unless you encounter performance problems, but it is still worth understanding what they are and how they work.

SQL Server supports processor affinity by two options:

- CPU affinity mask
- Affinity I/O mask

SQL Server uses all CPUs available from the operating system (if the per-processor core license is chosen). It creates schedulers on all the CPUs to make best use of the resources for any given workload. When multitasking, the operating system or other applications on the server can switch process threads from one processor to another. SQL Server is a resource-intensive application, and so performance can be affected when this occurs. To minimize the effect, you can configure the processors such that all the SQL Server load is directed to a preselected group of processors. This is achieved by using the CPU affinity mask.

The affinity I/O mask option binds SQL Server disk I/O to a subset of CPUs. In SQL Server OLTP environments, this extension can enhance the performance of SQL Server threads issuing I/O operations.

Max Degree of Parallelism (MAXDOP)

By default, SQL Server uses all available CPUs during query execution (if the per-processor core license chosen). Although this is great for large queries, it can cause performance problems and limit concurrency. A better approach is to limit parallelism to the number of physical cores in a single CPU socket. For example, on a server with two physical CPU sockets with 12 cores per socket, regardless of hyperthreading, MAXDOP should be set to 12. MAXDOP cannot restrict or dictate which CPU is to be used. Instead, it restricts the number of CPUs that can be used by a single batch query.

For DSSs such as data warehouses, NetApp recommends starting with this setting at 50 or so and tuning up or down as appropriate. Make sure you measure for the critical queries in your application and adjust if necessary.

3.3 Memory Configuration

Max Server Memory

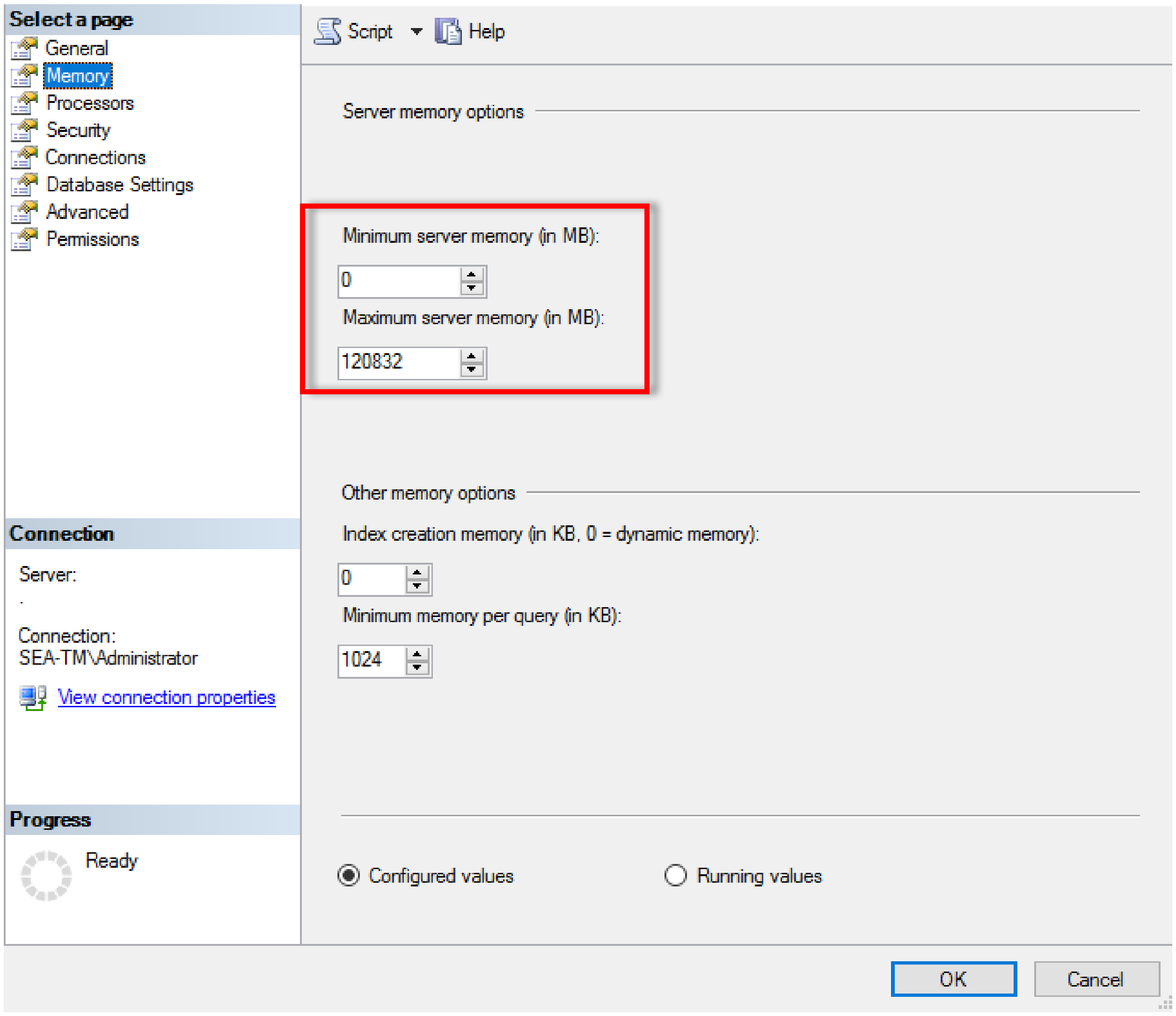
The max server memory option sets the maximum amount of memory that the SQL Server instance can use. It is generally used if multiple applications are running on the same server where SQL Server is running and you want to guarantee that these applications have sufficient memory to function properly.

Some applications only use whatever memory is available when they start and do not request more even if needed. That is where the max server memory setting comes into play.

On a SQL Server cluster with several SQL Server instances, each instance could be competing for resources. Setting a memory limit for each SQL Server instance can help guarantee best performance for each instance.

NetApp recommends leaving at least 4GB to 6GB of RAM for the operating system to avoid performance issues. Figure 2 displays how to set up minimum and maximum server memory.

Figure 2) Adjusting minimum and maximum server memory using SQL Server Management Studio.



Select a page: General, Memory, Processors, Security, Connections, Database Settings, Advanced, Permissions

Server memory options

Minimum server memory (in MB): 0

Maximum server memory (in MB): 120832

Other memory options

Index creation memory (in KB, 0 = dynamic memory): 0

Minimum memory per query (in KB): 1024

[✓ Configured values] Running values

Connection

Server:
.

Connection:
SEA-TM\Administrator

View connection properties

Progress

Ready

Using SQL Server Management Studio to adjust minimum or maximum server memory requires a restart of the SQL Server service. You can adjust server memory using transact SQL (T-SQL) using this code:

```
EXECUTE sp_configure 'show advanced options', 1
GO
EXECUTE sp_configure 'min server memory (MB)', 2048
GO
EXEC sp_configure 'max server memory (MB)', 120832
GO
RECONFIGURE WITH OVERRIDE
```

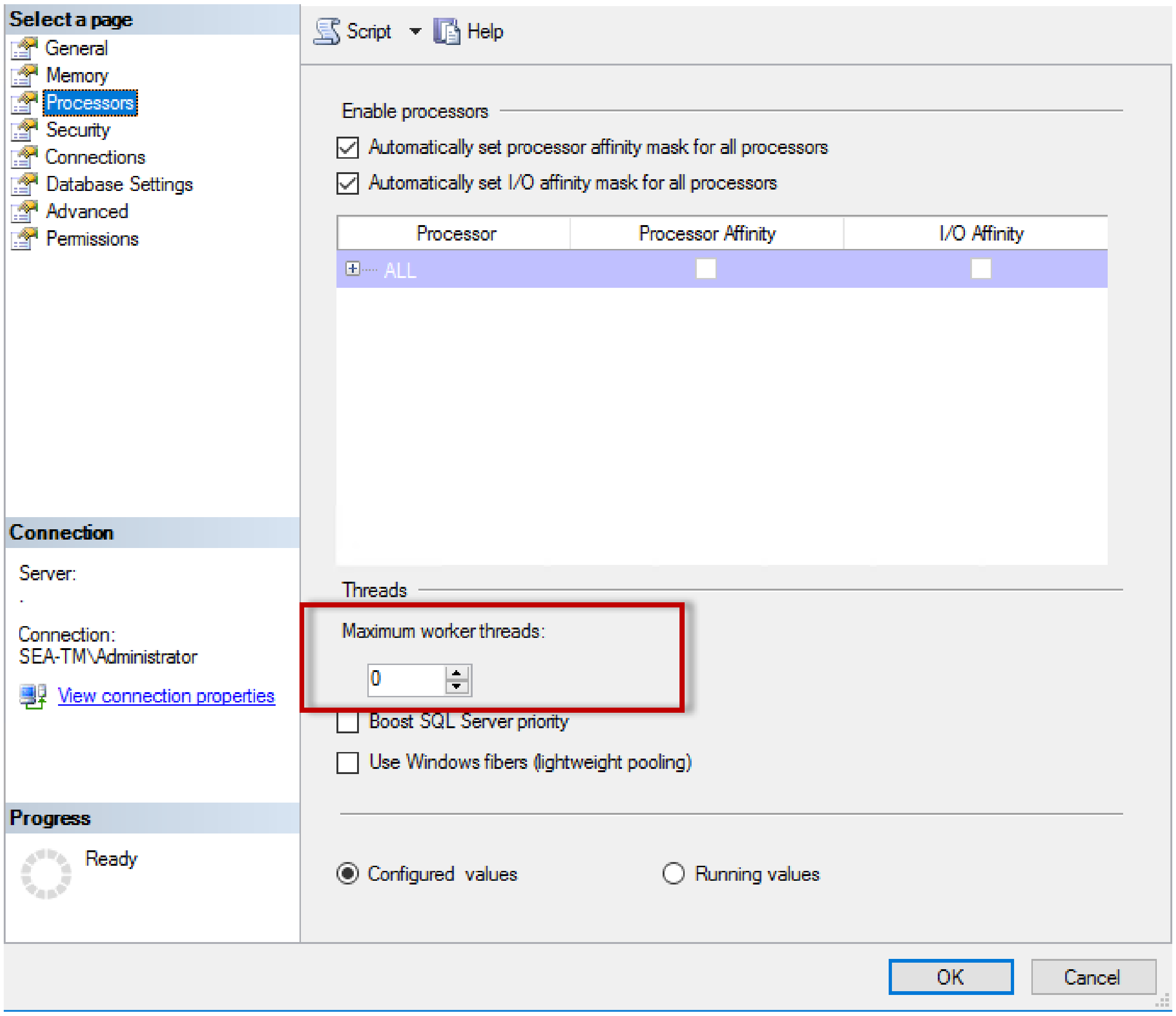
Max Worker Threads

The max worker threads option helps to optimize performance when large numbers of clients are connected to SQL Server. Normally, a separate operating system thread is created for each query request. If hundreds of simultaneous connections are made to SQL Server, then one thread per query request consumes large amounts of system resources. The max worker threads option helps improve performance by enabling SQL Server to create a pool of worker threads to service a larger number of query requests.

The default value is 0, which allows SQL Server to automatically configure the number of worker threads at startup. This works for most systems. Max worker threads is an advanced option and should not be altered without assistance from an experienced database administrator (DBA).

When should you configure SQL Server to use more worker threads? If the average work queue length for each scheduler is above 1, you might benefit from adding more threads to the system, but only if the load is not CPU-bound or experiencing any other heavy waits. If either of those is happening, adding more threads does not help because they end up waiting for other system bottlenecks. For more information about max worker threads, see [Configure the max worker threads Server Configuration Option](). Figure 3 indicates how to adjust maximum worker threads.

Figure 3) Configuring max worker threads using SQL Server Management Studio.



Server:
.

Connection:
SEA-TM\Administrator

Enable processors

☑ Automatically set processor affinity mask for all processors

☑ Automatically set I/O affinity mask for all processors

Processor	Processor Affinity	I/O Affinity
ALL	☐	☐

Threads

Maximum worker threads: 0

☐ Boost SQL Server priority

☐ Use Windows fibers (lightweight pooling)

[✓ Configured values] Running values

The following example shows how to configure the max work threads option using T-SQL.

```
EXEC sp_configure 'show advanced options', 1;
GO
RECONFIGURE ;
GO
EXEC sp_configure 'max worker threads', 900 ;
GO
RECONFIGURE;
```

```
GO
```

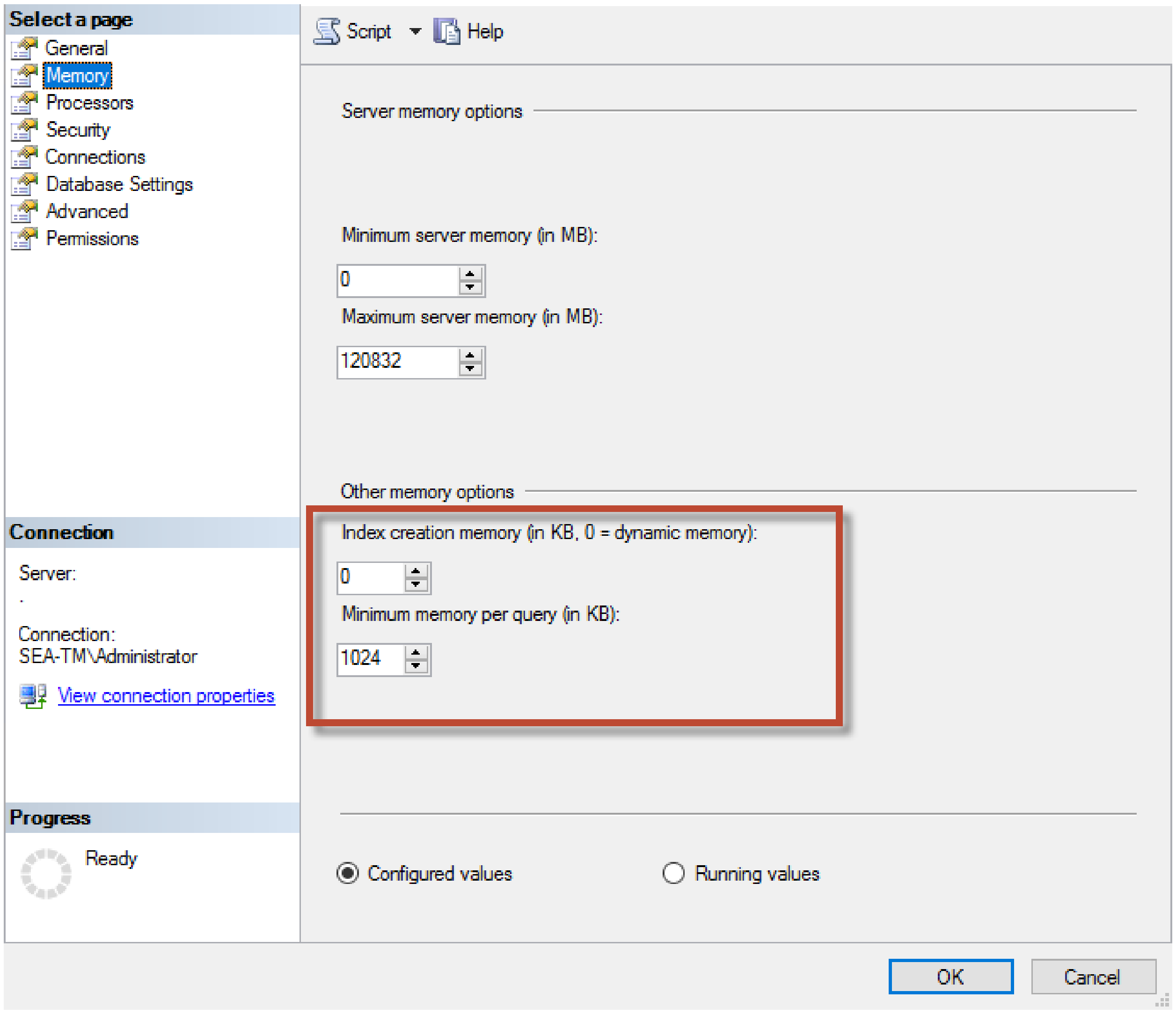
Index Create Memory

The index create memory option is another advanced option that you should not usually change. It controls the maximum amount of RAM initially allocated for creating indexes. The default value for this option is 0, which means that it is managed by SQL Server automatically. However, if you encounter difficulties creating indexes, consider increasing the value of this option.

Min Memory per Query

When a query is run, SQL Server tries to allocate the optimum amount of memory for it to run efficiently. By default, the `min memory per query` setting allocates > or = to 1024KB for each query to run. It is a best practice is to leave this setting at the default value of 0 to allow SQL Server to dynamically manage the amount of memory allocated for index creation operations. However, if SQL Server has more RAM than it needs to run efficiently, the performance of some queries can be boosted if you increase this setting. Therefore, as long as memory is available on the server that is not being used by SQL Server, any other applications, or the operating system, then boosting this setting can help overall SQL Server performance. If no free memory is available, increasing this setting might hurt overall performance.

Figure 4) Configuring index create memory and min memory per query using SQL Server Management Studio.



Server memory options

Minimum server memory (in MB): 0

Maximum server memory (in MB): 120832

Other memory options

Index creation memory (in KB, 0 = dynamic memory): 0

Minimum memory per query (in KB): 1024

Server: .

Connection: SEA-TM\Administrator

Ready

[✓ Configured values] Running values

Buffer Pool Extensions

The buffer pool extension provides seamless integration of an NVRAM extension with the database engine buffer pool to significantly improve I/O throughput. The buffer pool extension is not available in every SQL Server edition. It is available only with the 64-bit SQL Server Standard, Business Intelligence, and Enterprise editions.

The buffer pool extension feature extends the buffer pool cache with nonvolatile storage (usually SSDs). The extension allows the buffer pool to accommodate a larger database working set, forcing the paging of I/O between the RAM and the SSDs and effectively offloading small random I/Os from mechanical disks to SSDs. Because of the lower latency and better random I/O performance of SSDs, the buffer pool extension significantly improves I/O throughput.

The buffer pool extension feature offers the following benefits:

- Increased random I/O throughput
- Reduced I/O latency
- Increased transaction throughput

- Improved read performance with a larger hybrid buffer pool
- A caching architecture that can take advantage of existing and future low-cost memory

For buffer pool extensions, NetApp recommends the following:

- Make sure that an SSD-backed LUN (such as NetApp AFF) is presented to the SQL Server host so that it can be used as a buffer pool extension target disk.
- The extension file must be the same size as or larger than the buffer pool.

The following example shows a T-SQL command to set up a buffer pool extension of 32GB.

```
USE master
GO
ALTER SERVER CONFIGURATION
SET BUFFER POOL EXTENSION ON
  (FILENAME = ‘P:\BUFFER POOL EXTENSION\SQLServerCache.BUFFER POOL EXTENSION’, SIZE = 32 GB);
GO
```

4 Database Files and Filegroups

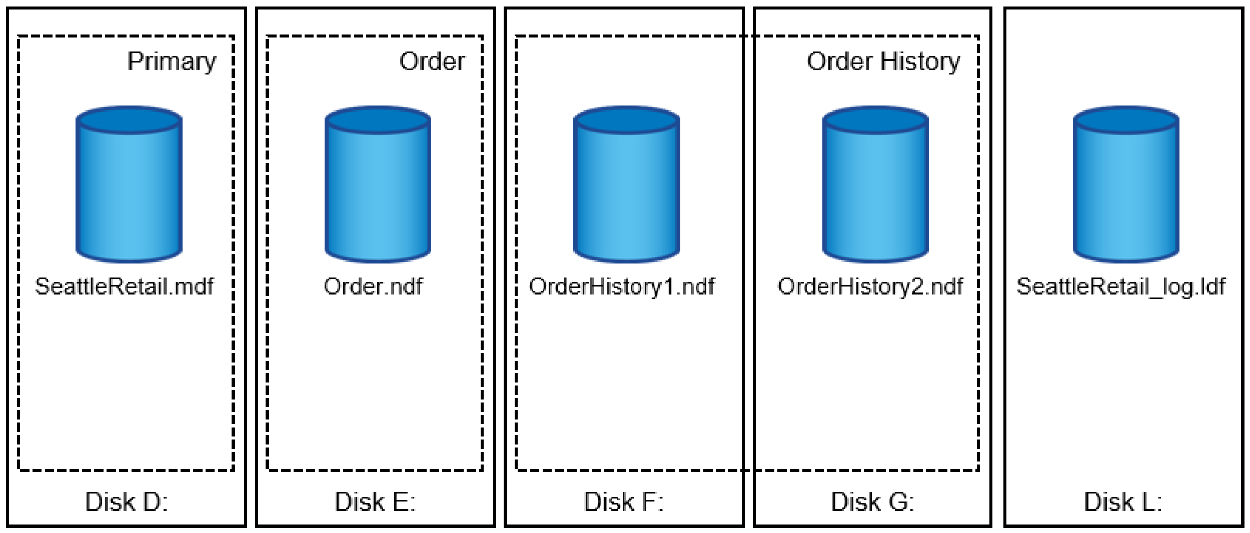
A SQL Server database is a collection of objects that allows you to store and manipulate data. In theory, SQL Server (64-bit) supports 32,767 databases per instance and 524,272TB of database size, although the typical installation usually has several databases. However, the number of the databases SQL Server can handle depends on the load and hardware. It is not unusual to see SQL Server instances hosting dozens, hundreds, or even thousands of small databases.

Each database consists of one or more data files and one or more transaction log files. The transaction log stores the information about database transactions and all data modifications made by each session. Every time the data is modified, SQL Server stores enough information in the transaction log to undo (roll back) or redo (replay) the action. A SQL Server transaction log is an integral part of SQL Server’s reputation for data integrity and robustness. The transaction log is vital to the atomicity, consistency, isolation, and durability (ACID) capabilities of SQL Server. SQL Server writes to the transaction log as soon as any change to the data page happens. Every Data Manipulation Language (DML) statement (for example, select, insert, update, or delete) is a complete transaction, and the transaction log makes sure that the entire set-based operation takes place, making sure of the atomicity of the transaction.

Each database has one primary data file, which, by default, has the .mdf extension. In addition, each database can have secondary database files. Those files, by default, have .ndf extensions.

All database files are grouped into filegroups. A filegroup is the logical unit, which simplifies database administration. They allow the separation between logical object placement and physical database files. When you create the database objects tables, you specify in what filegroup they should be placed without worrying about the underlying data file configuration.

Figure 5) Filegroups.



The ability to put multiple data files inside the filegroup allows you to spread the load across different storage devices, which helps to improve the I/O performance of the system. The transaction log in contrast does not benefit from the multiple files because SQL Server writes to the transaction log in a sequential manner.

The separation between logical object placement in the filegroups and physical database files allows you to fine-tune the database file layout, getting the most from the storage subsystem. For example, independent software vendors (ISVs) who are deploying their products to different customers can adjust the number of database files based on the underlying I/O configuration and the expected amount of data during the deployment stage. Those changes are transparent to the application developers, who are placing the database objects in the filegroups rather than database files.

NetApp recommends avoiding the use of the primary filegroup for anything but system objects. Creating a separate filegroup or set of filegroups for the user objects simplifies database administration and disaster recovery, especially in the case of large databases.

You can specify initial file size and autogrowth parameters at the time when you create the database or add new files to an existing database. SQL Server uses a proportional fill algorithm when choosing which data file it should write data into. It writes an amount of data proportionally to the free space available in the files. The more free space in the file, the more writes it handles.

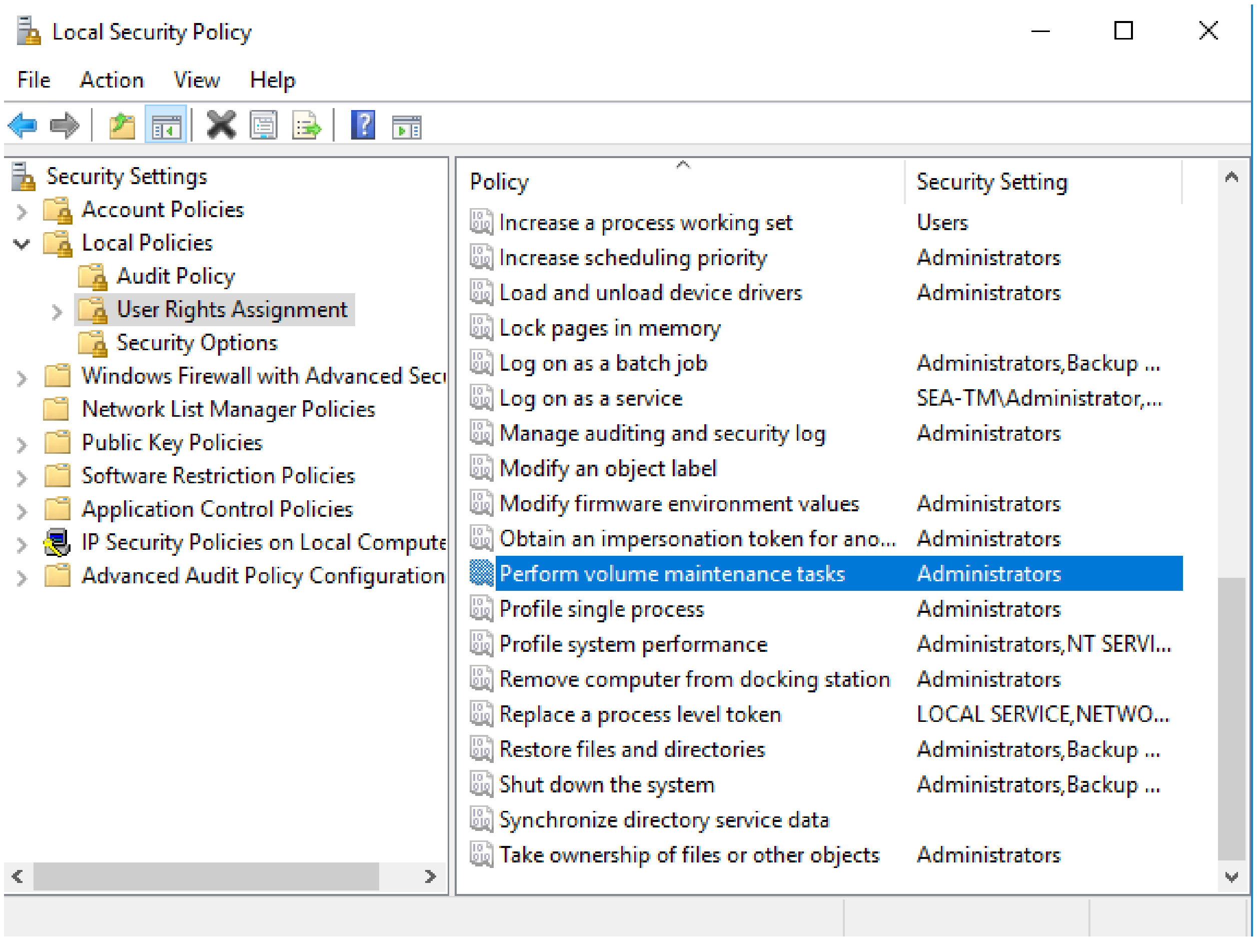
NetApp recommends that all files in the single filegroup have the same initial size and autogrowth parameters, with the grow size defined in megabytes rather than percentages. This helps the proportional fill algorithm evenly balance write activities across data files.

Every time SQL Server grows the files, it fills newly allocated space in the files with zeros. That process blocks all sessions that need to write to the corresponding file or, in case of transaction log growth, generate transaction log records.

SQL Server always zeroes out the transaction log, and that behavior cannot be changed. However, you can control whether data files are zeroing out by enabling or disabling instant file initialization. Enabling instant file initialization helps to speed up data file growth and reduces the time required to create or restore the database.

A small security risk is associated with instant file initialization. When this option is enabled, unallocated parts of the data file can contain information from previously deleted OS files. Database administrators can examine such data.

You can enable instant file initialization by adding the SA_MANAGE_VOLUME_NAME permission, also known as “perform volume maintenance task,” to the SQL Server startup account. You can do this under the local security policy management application (secpol.msc), as shown in the following figure. Open the properties for the “perform volume maintenance task” permission and add the SQL Server startup account to the list of users there.

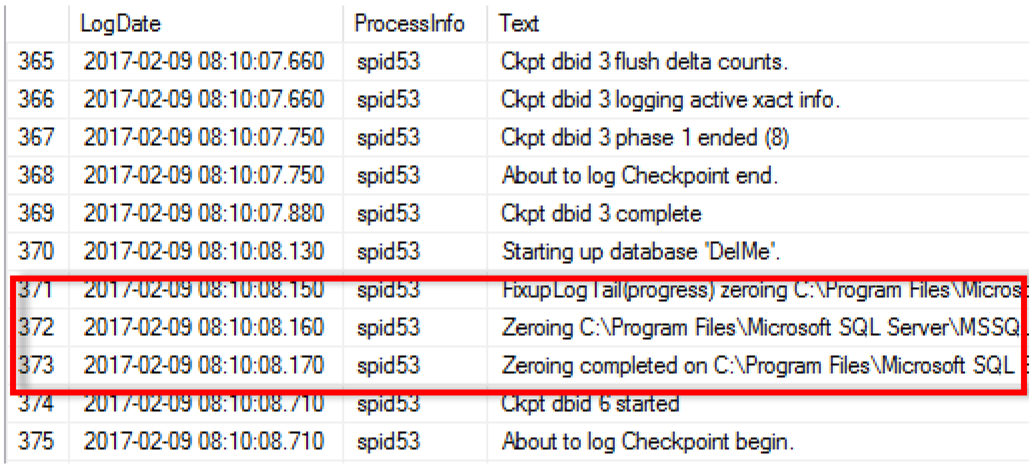

Local Security Policy
File
Action
View
Help
Security Settings
Account Policies
Local Policies
Audit Policy
User Rights Assignment
Security Options
Windows Firewall with Advanced Secu
Network List Manager Policies
Public Key Policies
Software Restriction Policies
Application Control Policies
IP Security Policies on Local Compute
Advanced Audit Policy Configuration

Policy	Security Setting
Increase a process working set	Users
Increase scheduling priority	Administrators
Load and unload device drivers	Administrators
Lock pages in memory	
Log on as a batch job	Administrators,Backup ...
Log on as a service	SEA-TM\Administrator,...
Manage auditing and security log	Administrators
Modify an object label	
Modify firmware environment values	Administrators
Obtain an impersonation token for ano...	Administrators
Perform volume maintenance tasks	Administrators
Profile single process	Administrators
Profile system performance	Administrators,NT SERVI...
Remove computer from docking station	Administrators
Replace a process level token	LOCAL SERVICE,NETWO...
Restore files and directories	Administrators,Backup ...
Shut down the system	Administrators,Backup ...
Synchronize directory service data	
Take ownership of files or other objects	Administrators

To check if the permission is enabled, you can use the code from the following example. This code sets two trace flags that force SQL Server to write additional information to the error log, create a small database, and read the content of the log.

```
DBCC TRACEON(3004,3605,-1)
GO
CREATE DATABASE DelMe
GO
EXECUTE sp_readerrorlog
GO
DROP DATABASE DelMe
GO
DBCC TRACEOFF(3004,3605,-1)
GO
```

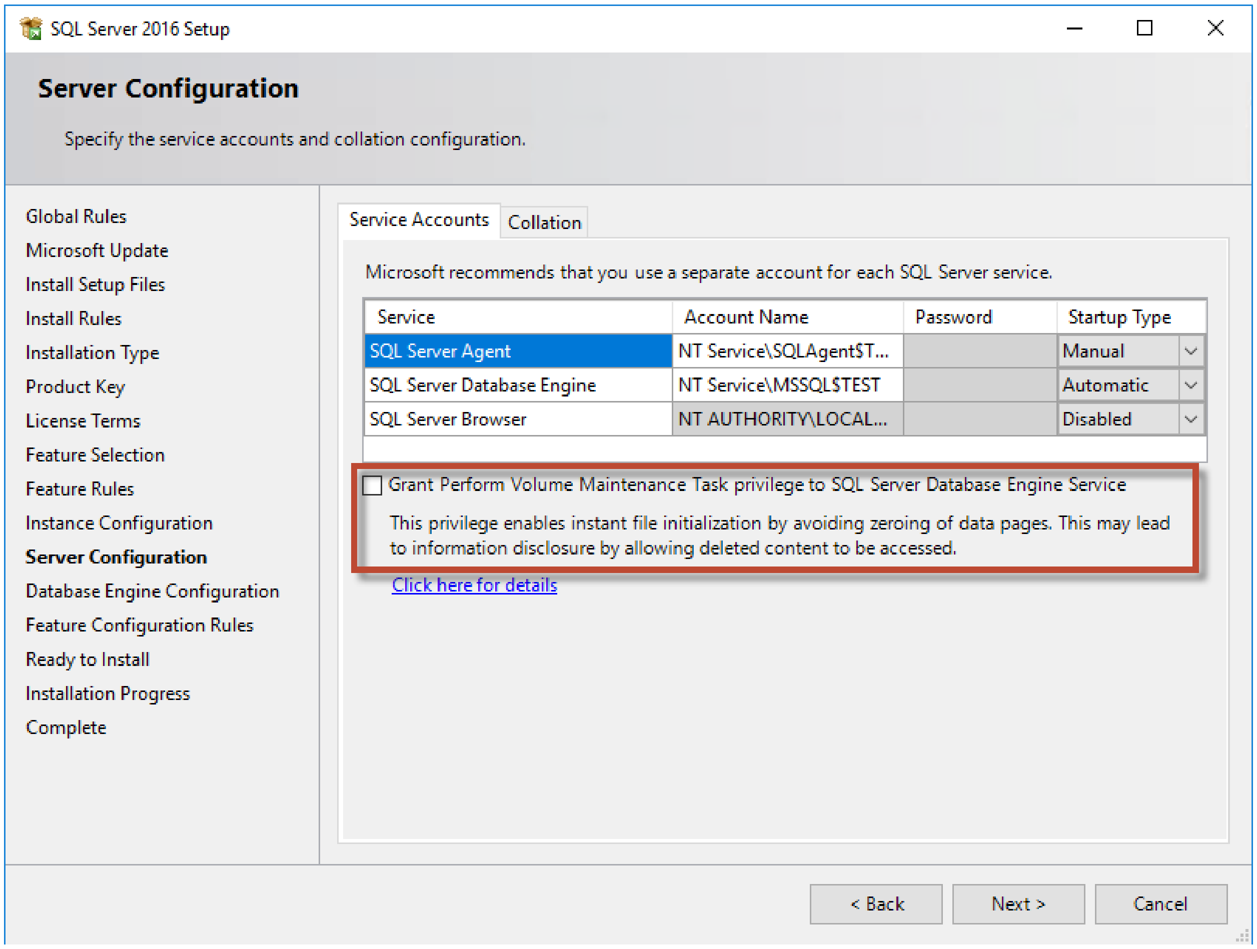
When instant file initialization is not enabled, the SQL Server error log shows that SQL Server is zeroing the mdf data file in addition to zeroing the ldf log file, as shown in the following example. When instant file initialization is enabled, it displays only zeroing of the log file.



	LogDate	ProcessInfo	Text
365	2017-02-09 08:10:07.660	spid53	Ckpt dbid 3 flush delta counts.
366	2017-02-09 08:10:07.660	spid53	Ckpt dbid 3 logging active xact info.
367	2017-02-09 08:10:07.750	spid53	Ckpt dbid 3 phase 1 ended (8)
368	2017-02-09 08:10:07.750	spid53	About to log Checkpoint end.
369	2017-02-09 08:10:07.880	spid53	Ckpt dbid 3 complete
370	2017-02-09 08:10:08.130	spid53	Starting up database 'DelMe'.
371	2017-02-09 08:10:08.150	spid53	FixupLogTail(progress) zeroing C:\Program Files\Micros
372	2017-02-09 08:10:08.160	spid53	Zeroing C:\Program Files\Microsoft SQL Server\MSSQ
373	2017-02-09 08:10:08.170	spid53	Zeroing completed on C:\Program Files\Microsoft SQL
374	2017-02-09 08:10:08.710	spid53	Ckpt dbid 6 started
375	2017-02-09 08:10:08.710	spid53	About to log Checkpoint begin.

The perform volume maintenance task is simplified in SQL Server 2016 and is later provided as an option during the installation process. Figure 6 displays the option to grant the SQL Server database engine service the privilege to perform the volume maintenance task.

Figure 6) Option for granting perform volume maintenance task privilege during SQL Server installation.



Server Configuration

Specify the service accounts and collation configuration.

Microsoft recommends that you use a separate account for each SQL Server service.

Service	Account Name	Password	Startup Type
SQL Server Agent	NT Service\SQLAgent$T...		Manual
SQL Server Database Engine	NT Service\MSSQL$TEST		Automatic
SQL Server Browser	NT AUTHORITY\LOCAL...		Disabled

☐ Grant Perform Volume Maintenance Task privilege to SQL Server Database Engine Service

This privilege enables instant file initialization by avoiding zeroing of data pages. This may lead to information disclosure by allowing deleted content to be accessed.

Another important database option that controls the database file sizes is autoshrink. When this option is enabled, SQL Server regularly shrinks the database files, reduces their size, and releases space to the operating system. This operation is resource intensive and is rarely useful because the database files grow again after some time when new data comes into the system. Autoshrink must never be enabled on the database.

5 Database and Storage Considerations

The combination of NetApp storage solutions and Microsoft SQL Server enables the creation of enterprise-level database storage designs that can meet today's most demanding application requirements. To optimize both technologies, it is vital to understand the SQL Server I/O pattern and characteristics. A well-designed storage layout for a SQL Server database supports the performance of SQL Server and the management of the SQL Server infrastructure. A good storage layout also allows the initial deployment to be successful and the environment to grow smoothly over time as your business grows.

5.1 Aggregates

Aggregates are the primary storage containers for NetApp storage configurations and contain one or more RAID groups consisting of both data disks and parity disks.

NetApp has performed various I/O workload characterization tests using shared and dedicated aggregates with data files and transaction log files separated. The tests show that one large aggregate with more RAID groups and spindles optimizes and improves storage performance and is easier for administrators to manage for two reasons:

- One large aggregate makes the I/O capabilities of all spindles available to all files.
- One large aggregate enables the most efficient use of disk space.

For high availability (HA), place the SQL Server Always On Availability Group secondary synchronous replica on a separate storage virtual machine (SVM) in the aggregate. For disaster recovery purposes, place the asynchronous replica on an aggregate that is part of a separate storage cluster in the DR site, with content replicated by using NetApp SnapMirror® technology.

NetApp recommends having at least 10% free space available in an aggregate for optimal storage performance.

5.2 Volumes

NetApp FlexVol volumes are created and reside inside aggregates. Many volumes can be created in a single aggregate, and each volume can be expanded, shrunk, or moved between aggregates with no user downtime.

Volume Design Considerations

Before you create a database volume design, it is important to understand how the SQL Server I/O pattern and characteristics vary depending on the workload and on the backup and recovery requirements. See the following NetApp recommendations for flexible volumes:

- Use flexible volumes to store SQL Server database files and avoid sharing volumes between hosts.
- Use NTFS mount points instead of drive letters to surpass the 26-drive-letter limitation in Windows. When using volume mount points, it is a general recommendation to give the volume label the same name as the mount point.
- When appropriate, configure a volume autosize policy to help prevent out-of-space conditions.

- Enable read reallocation on the volume when the SQL Server database I/O profile consists of mostly large sequential reads, such as with decision support system workloads. Read reallocation optimizes the blocks to provide better performance.
- If you install SQL Server on an SMB share, make sure that Unicode is enabled on the SMB/CIFS volumes for creating folders.
- Set the NetApp `snapshot copy reserve` value in the volume to zero for ease of monitoring from an operational perspective.
- Disable storage Snapshot™ copy schedules and retention policies. Instead, use SnapCenter to coordinate Snapshot copies of the SQL Server data volumes.
- For SnapCenter, place the SQL Server system databases on a dedicated volume or VMDK, because collocating system databases with user databases, including availability group databases, prevents Snapshot backups of the user databases. Backups of system databases stream into the HostLog Directory LUN. This LUN is typically the same volume or VMDK that hosts the Windows operating system files and SQL Server binaries, which are random read/write workloads.
- tempdb is a system database used by SQL Server as a temporary workspace, especially for I/O-intensive DBCC CHECKDB operations. Therefore, place this database on a dedicated volume with a separate set of spindles. In large environments in which volume count is a challenge, you can consolidate tempdb into fewer volumes and store it in the same volume as other system databases after careful planning. Data protection for tempdb is not a high priority because this database is recreated every time SQL Server is restarted.
- Place user data files (.mdf) on separate volumes because they are random read/write workloads. It is common to create transaction log backups more frequently than database backups. For this reason, place transaction log files (.ldf) on a separate volume or VMDK from the data files so that independent backup schedules can be created for each. This separation also isolates the sequential write I/O of the log files from the random read/write I/O of data files and significantly improves SQL Server performance.
- Create the host log directory on the dedicated FlexVol volume to which SnapCenter copies transaction logs.

5.3 LUNs

tempdb Files

NetApp recommends proactively inflating tempdb files to their full size to avoid disk fragmentation. Page contention can occur on lobal allocation map (GAM), shared global allocation map (SGAM), or page free space (PFS) pages when SQL Server has to write to special system pages to allocate new objects. Latches protect (lock) these pages in memory. On a busy SQL Server intance, it can take a long time to get a latch on a system page in tempdb. This results in slower query run times and is known as latch contention. See the following best practices for creating tempdb data files:

- For < or = to 8 cores: tempdb data files = number of cores
- For > 8 cores: 8 tempdb data files

The following example script modifies tempdb by creating eight tempdb files and moving tempdb to the mount point `C:\MSSQL\tempdb` for SQL Server 2012 and later.

```
use master
go
-- Change logical tempdb file name first since SQL Server shipped with logical file name called
tempdev
alter database tempdb modify file (name = 'tempdev', newname = 'tempdev01');

-- Change location of tempdev01 and log file
alter database tempdb modify file (name = 'tempdev01', filename =
'C:\MSSQL\tempdb\tempdev01.mdf');
alter database tempdb modify file (name = 'templog', filename = 'C:\MSSQL\tempdb\templog.ldf');
```

```
GO
-- Assign proper size for tempdev01
ALTER DATABASE [tempdb] MODIFY FILE ( NAME = N'tempdev01', SIZE = 10GB );
ALTER DATABASE [tempdb] MODIFY FILE ( NAME = N'templog', SIZE = 10GB );
GO
-- Add more tempdb files
ALTER DATABASE [tempdb] ADD FILE ( NAME = N'tempdev02', FILENAME =
N'C:\MSSQL\tempdb\tempdev02.ndf' , SIZE = 10GB , FILEGROWTH = 10%);
ALTER DATABASE [tempdb] ADD FILE ( NAME = N'tempdev03', FILENAME =
N'C:\MSSQL\tempdb\tempdev03.ndf' , SIZE = 10GB , FILEGROWTH = 10%);
ALTER DATABASE [tempdb] ADD FILE ( NAME = N'tempdev04', FILENAME =
N'C:\MSSQL\tempdb\tempdev04.ndf' , SIZE = 10GB , FILEGROWTH = 10%);
ALTER DATABASE [tempdb] ADD FILE ( NAME = N'tempdev05', FILENAME =
N'C:\MSSQL\tempdb\tempdev05.ndf' , SIZE = 10GB , FILEGROWTH = 10%);
ALTER DATABASE [tempdb] ADD FILE ( NAME = N'tempdev06', FILENAME =
N'C:\MSSQL\tempdb\tempdev06.ndf' , SIZE = 10GB , FILEGROWTH = 10%);
ALTER DATABASE [tempdb] ADD FILE ( NAME = N'tempdev07', FILENAME =
N'C:\MSSQL\tempdb\tempdev07.ndf' , SIZE = 10GB , FILEGROWTH = 10%);
ALTER DATABASE [tempdb] ADD FILE ( NAME = N'tempdev08', FILENAME =
N'C:\MSSQL\tempdb\tempdev08.ndf' , SIZE = 10GB , FILEGROWTH = 10%);
GO
```

Beginning with SQL Server 2016, the number of CPU cores visible to the operating system is automatically detected during installation, and, based on that number, SQL Server calculates and configures the number of tempdb files required for optimum performance.

Host Log Directory

SnapCenter uses a host log directory to store transaction log backup data. This is at the host and instance level. Each SQL Server host used by SnapCenter must have a host log directory configured to perform log backups. SnapCenter has a database repository, so metadata related to backup, restore, or cloning operations is stored in a central database repository.

The sizes of the host log directory is calculated as follows:

Size of host log directory = ((system database size + (maximum DB LDF size × daily log change rate %)) × (Snapshot copy retention) ÷ (1 – LUN overhead space %)

The host log directory sizing formula assumes the following:

- A system database backup that does not include the tempdb database
- A 10% LUN overhead space

Place the host log directory on a dedicated volume or LUN. The amount of data in the host log directory depends on the size of the backups and the number of days that backups are retained. The host log directory can be configured at any time by running the configuration wizard. SnapCenter allows only one host log directory per SQL Server host. The host log directories can be configured from SnapCenter > Host > Configure Plug-in.

You can back up SQL Server databases to a NetApp SnapVault® location and perform the following operations:

- Restore all LUNs in a volume.
- Restore a single LUN from the vault.
- Access the latest Snapshot copy of a LUN directly from the backup vault.

The following are NetApp recommendations for host log directories:

- Make sure that the host log directory is not shared by any other type of data that can potentially corrupt the backup Snapshot copies.
- Do not place user databases or system databases on a LUN that hosts mount points.

- Always use the SnapCenter configuration wizards to migrate databases to NetApp storage so that the databases are stored in valid locations, enabling successful SnapCenter backup and restore operations. Keep in mind that the migration process is disruptive and can cause the databases to go offline while the migration is in progress.
- The following conditions must be in place for failover cluster instances (FCIs) of SQL Server:
  - If you are using a failover cluster instance, the host log directory LUN must be a cluster disk resource in the same cluster group as the SQL Server instance being backed up SnapCenter.
  - If you are using a failover cluster instance, user databases must be placed on shared LUNs that are physical disk cluster resources assigned to the cluster group associated with the SQL Server instance.
- Make sure that the user database and the host log directory are on separate volumes to prevent the retention policy from overwriting Snapshot copies when these are used with SnapVault technology.
- Make sure that SQL Server databases reside on LUNs separate from LUNs that have nondatabase files, such as full-text search-related files.
- Placing database secondary files (as part of a filegroup) on separate volumes improves the performance of the SQL Server database. This separation is valid only if the database's .mdf file does not share its LUN with any other .mdf files.
- If any database filegroups share LUNs, the database layout must be identical for all databases. However, this restriction is not applicable when the unrestricted database layout (UDL) option is enabled.
- Create LUNs with SnapCenter Plug-in for Microsoft Windows whenever possible. If you want to create LUNs with DiskManager or other tools, make sure that the allocation unit size is set to 64K for partitions when formatting the LUNs.

5.4 Data Storage Design

For SQL Server databases that do not use SnapCenter to perform backups, Microsoft recommends placing the data and log files on separate drives. For applications that simultaneously update and request data, the log file is write intensive, and the data file (depending on your application) is read/write intensive. For data retrieval, the log file is not needed. Therefore, requests for data can be satisfied from the data file placed on its own drive.

When you create a new database, Microsoft recommends specifying separate drives for the data and logs. To move files after the database is created, the database must be taken offline. For more Microsoft recommendations, see Place Data and Log Files on Separate Drives.

For NetApp storage considerations in environments that use SnapCenter with the SQL Server plug-in, see TR-4714: Best Practice Guide for SQL Server Using NetApp SnapCenter.

6 Storage Efficiency and Manageability

Storage efficiency is the ability to store and manage SQL Server data in a way that consumes the least amount of storage space with little or no effect on the overall performance of the system. Storage efficiency goes beyond data deduplication; it is a combination of RAID, provisioning (overall layout and utilization), mirroring, and other data protection technologies.

The following NetApp technologies implement storage efficiency and create cost benefits by optimizing existing storage in the infrastructure and deferring or avoiding future storage expenditures. The more you use these technologies together, the larger the savings.

6.1 Snapshot Copies

NetApp Snapshot technology provides fast, low-cost backup by creating point-in-time copies of the file system (volume) or LUN by preserving ONTAP architecture WAFL® consistency points. NetApp

SnapCenter integrates with the SQL Server virtual device interface (VDI) for the creation of application-consistent Snapshot copies of production SQL Server databases with no downtime for the production database.

There is no performance penalty for creating Snapshot copies because data is never moved, as it is with other copy-out technologies. The cost for Snapshot copies is only at the rate of block-level changes; it is not 100% for each backup, as is the case with mirror copies. Snapshot technology can create significant savings in storage costs for backup and restore purposes and creates several efficient data-management possibilities.

If a database uses multiple LUNs on the same volume, then all Snapshot copies of these LUNs are made simultaneously because Snapshot copies are volume-based. In certain situations, a SnapCenter clone operation restores a LUN from a Snapshot copy for temporary read/write access to an alternative location by using a writable Snapshot copy during the SnapCenter verification process.

6.2 Thin Provisioning

Thin provisioning is a method for optimizing the use of available storage space. It relies on on-demand allocation of data blocks versus the traditional method of allocating all of the blocks up front. This methodology eliminates almost all white space and thus helps avoid poor utilization rates.

FlexVol volumes are the enabling technology behind NetApp thin provisioning and can be thought of as the virtualization layer of ONTAP. When a LUN is created, it does not dedicate specific data blocks out of the volume for itself or for its Snapshot copies. Instead, it allocates the blocks from the aggregate when the data is written. This allocation method allows the administrator to provision more storage space, as seen from the connected servers, than the space that is physically available in the storage system.

When storage consumption is unpredictable or highly volatile, it is best to reduce the level of storage overcommitment so that storage is available for any growth spikes. Consider limiting storage commitment to 100%—no overcommitment—and using the trending functionality to determine how much if any overcommitment is acceptable. Overcommitment of storage must be carefully considered and managed for mission-critical applications (such as SQL Server) for which even a minimal outage is intolerable. In such cases, it is best to monitor storage consumption trends to determine how much overcommitment is tolerable.

If the time required to procure new storage is long, storage overcommitment thresholds should be adjusted accordingly. The overcommitment threshold should alert administrators early enough to allow new storage to be procured and installed. The potential risk when configuring the SQL Server environment for thin provisioning is a LUN going offline when not enough space is available to write further data. Use the volume autogrow functionality as a mitigation mechanism to safely allow thin provisioning and higher storage utilization.

6.3 Space Guarantee

Space guarantees enable thin provisioning. The space guarantee option can be set at the volume or LUN level. If the space guarantee at the volume level is set to volume (default setting), the amount of space required by the FlexVol volume is always available from its aggregate during the creation of that volume.

If the space guarantee for the volume is set to none, the volume reserves no space from the aggregate during volume creation. Space is taken from the aggregate when data is written to the volume. Write operations to space-reserved LUNs in a volume with the setting guarantee=none fail if the containing aggregate does not have enough available space.

LUN reservation makes sure that the LUN has space in the volume, but setting guarantee=none does not guarantee that the volume has space in the aggregate. When the space guarantee for the volume is set to file, the aggregate keeps space available for completely rewriting LUNs that have space reservation enabled.

NetApp recommends using thin provisioning in SQL Server environments to improve space utilization and to reduce the overall storage requirements when the space guarantee functionality is used.

6.4 Space Reclamation

Space reclamation can be initiated periodically to recover unused space in a LUN. With SnapCenter, you can use the following PowerShell command to start space reclamation.

```
Invoke-SdHostVolumeSpaceReclaim -Path drive_path
```

If you need to run space reclamation, this process should be run during periods of low activity because it initially consumes cycles on the host.

6.5 Fractional Reserve

Fractional reserve is a volume option that determines how much space ONTAP reserves to be used for Snapshot copy overwrite for LUNs after all other space in the volume has been used.

NetApp storage appliances can be configured in many different ways for LUN thin provisioning, and each method has advantages and disadvantages. It is possible to have thin provisioned volumes and non-thin-provisioned volumes on the same storage system or even on the same aggregate. The following two options are considered best practice configurations for using thin provisioning with SQL Server.

Volume Guarantee Set to None

The advantage of the configuration in Table 2 is that the free space in the aggregate is used as a shared pool of free space. The disadvantages of this configuration are the high level of dependency between volumes and the fact that the level of thin provisioning cannot be easily tuned on an individual volume basis.

When using the configuration in Table 2, the total size of the volumes is greater than the actual storage available in the host aggregate. With this configuration, storage administrators can generally size the volume so that they only need to manage and monitor the used space in the aggregate. This option does not affect the space for hosting live data but rather allows the backup space to dynamically change.

Table 2) Volume guarantee set to none.

Setting	Value
volume guarantee	none
LUN reservation	enabled
fractional_reserve	0%
snap_reserve	0%
autodelete	volume/oldest_first
autosize	off
try_first	snap_delete

Using Autogrow and Autodelete

The configuration in Table 3 allows the administrator to finely tune the level of thin provisioning for SQL Server environments. With this configuration, the volume size defines or guarantees an amount of space that is available only to LUNs within that volume. The aggregate provides a shared storage pool of available space for all the volumes contained within it.

If the LUNs or Snapshot copies require more space than the space available in the volume, the volumes automatically grow, taking more space from the containing aggregate. Additionally, the advantage of having the LUN space reservation setting disabled is that Snapshot copies can use the space that is not needed by the LUNs. The LUNs themselves are not in danger of running out of space because the autodelete feature removes the Snapshot copies that are consuming space.

Note: Snapshot copies used for creating NetApp FlexClone® volumes are not deleted by the autodelete option.

Table 3) Setting up volume with autodelete and autogrow.

Setting	Value
volume guarantee	volume
LUN reservation	disabled
fractional_reserve	0%
snap_reserve	0%
autodelete	volume/oldest_first
autosize	on
try_first	autogrow

NetApp recommends using autogrow for most common deployment configurations because the storage admin only needs to monitor space usage in the aggregate.

6.6 NetApp FlexClone

NetApp FlexClone technology can be used to quickly create a writable copy of a FlexVol volume, eliminating the need for additional copies of the data. FlexClone volumes are great in the following situations:

- When testing or development occurs
- When progress is made by locking in incremental improvements
- When there is a desire to distribute data in changeable form without endangering the integrity of the original

A common scenario for using FlexClone is for test/dev purposes before a rollup patch or hotfix installation.

FlexClone technology can be leveraged both at the primary storage system and at the SnapMirror destination for effective utilization of resources. FlexClone can also be used for disaster recovery testing without affecting the operational continuity of the SQL Server environment.

SQL Server requires application-consistent Snapshot copies to create FlexClone volumes based on a Snapshot. NetApp recommends using SnapCenter to create Snapshot copies. For more details about the SQL Server Plug-in for SnapCenter, see TR-4714: Best Practice Guide for SQL Server Using NetApp SnapCenter.

6.7 Compression, Compaction, and Deduplication

Compression and deduplication are two storage efficiency options that increase the amount of logical data that fits on a given amount of physical storage. At a high level, compression is a mathematical process whereby patterns in data are detected and encoded in a way that reduces space requirements. In contrast, deduplication detects actual repeated blocks of data and removes the extraneous copies.

Although they deliver similar results, they work in significantly different ways and therefore must be managed differently.

Inline data compaction, a technology introduced in ONTAP 9, improves compression efficiency. Adaptive compression alone can provide at best 2:1 savings because it is limited to storing an 8K I/O in a 4K WAFL block. Compression methods such as secondary compression use a larger block size and deliver better efficiency but are not suitable for data that is subject to small block overwrites. Decompressing 32KB units of data, updating an 8K portion, recompressing, and writing back to disk create overhead.

Compression

There are multiple ways to compress a database. Until recently, compression was of limited value because most databases required many spindles to provide sufficient performance. One side effect of building a storage array with acceptable performance was that the array generally offered more capacity than was required. The situation has changed with the rise of solid-state storage. There is no longer a need to vastly overprovision the drive count to obtain good performance.

Even without compression, migrating a database to a partially or fully solid-state storage platform can yield significant cost savings because doing so avoids the need to purchase drives only needed to support I/O. For example, NetApp has examined some storage configurations from recent large database projects and compared the costs with and without the use of solid-state drives (SSDs) using NetApp Flash Cache™ and Flash Pool™ intelligent data caching or all-flash arrays.

As stated earlier, the increased IOPS capability of SSDs almost always yields cost savings, but compression can achieve further savings by increasing the effective capacity of solid-state media.

SQL Server currently supports two types of data compression: row compression and page compression. Row compression changes the data storage format. For example, it changes integers and decimals to the variable-length format instead of their native fixed-length format. It also changes fixed-length character strings to the variable-length format by eliminating blank spaces. Page compression implements row compression and two other compression strategies (prefix compression and dictionary compression). You can find more details about page compression in Page Compression Implementation.

Data compression is currently supported in the Enterprise, Developer, and Evaluation editions of SQL Server 2008 and later. Although compression can be performed by the database itself, this is rarely observed in a SQL Server environment.

NetApp Adaptive Compression

Adaptive compression has been thoroughly tested with SQL Server workloads, and the performance effect has been found to be negligible, even in an all-flash environment (where it is enabled by default) in which latency is measured in microseconds. In initial testing, some customers have reported a performance increase with the use of compression. This increase is the result of compression effectively increasing the amount of SSD available to the database.

ONTAP manages physical blocks in 4KB units. Therefore, the maximum possible compression ratio is 2:1 with a typical SQL Server database using an 8KB block. Early testing with real customer data has shown compression ratios approaching this level, but results vary based on the type of data stored.

NetApp Secondary Compression

Secondary compression uses a larger block size that is fixed at 32KB. This feature enables ONTAP to compress data with increased efficiency, but secondary compression is primarily designed for data at rest or data that is written sequentially and requires maximum compression.

NetApp recommends secondary compression for data such as transaction logs and backup files. These types of files are written sequentially and are not updated. This point does not mean that adaptive compression is discouraged. However, if the volume of data being stored is large, then secondary compression delivers better savings when compared to adaptive compression.

Consider secondary compression of data files when the amount of data is very large and the data files themselves are either read-only or rarely updated. Data files using a 32KB block size should see more compression under secondary compression that has a matching 32KB block size. However, care must be taken to verify that data using block sizes other than 32KB are not placed on these volumes. Only use this method in cases in which the data is not frequently updated.

Inline Data Compaction

Inline data compaction works by allowing logical WAFL blocks to be stored within physical WAFL blocks. For example, a database with highly compressible data such as text or partially full blocks might compress from 8KB to 1KB. Without compaction, that 1KB of data still occupies an entire 4KB block. Inline data compaction allows that 1KB of compressed data to be stored in just 1KB of physical space alongside other compressed data. It is not a compression technology. It is simply a more efficient way of allocating space on disk and therefore should not create any detectable performance effect.

The degree of savings obtained varies. Data that is already compressed or encrypted cannot generally be further compressed, and therefore such datasets do not benefit from compaction. Newly initialized SQL Server data files that contain little more than block metadata and zeros compress up to 80:1. This creates an extremely wide range of possibilities. The best way to evaluate potential savings is using the NetApp Space Savings Estimation Tool (SSET) available on NetApp Field Portal or through your NetApp representative.

Deduplication

NetApp does not recommend using deduplication with SQL Server database files primarily because this process is almost entirely ineffective. A SQL Server page contains a header that is globally unique to the database and a trailer that is nearly unique. One percent space savings are possible, but this is at the expense of significant overhead caused by data deduplication.

Many competing arrays claim the ability to deduplicate SQL Server databases based on the presumption that a database is copied multiple times. In this respect, NetApp deduplication could also be used, but ONTAP offers a better option: NetApp FlexClone technology. The result is the same; multiple copies of a SQL Server database that share most of the underlying physical blocks are created. Using FlexClone is much more efficient than taking the time to copy data files and then deduplicate them. It is, in effect, nonduplication rather than deduplication, because a duplicate is never created in the first place.

In the unusual case in which multiple copies of the same data files exist, deduplication can be used.

NetApp recommends that you not enable deduplication on any volumes containing SQL Server data files unless the volume is known to contain multiple copies of the same data, such as restoring database from backups to a single volume.

6.8 NetApp SnapMirror

NetApp SnapMirror technology offers a fast and flexible enterprise solution for mirroring or replicating data over LANs and WANs. SnapMirror technology transfers only modified 4KB data blocks to the destination after the initial base transfer, significantly reducing network bandwidth requirements. SnapMirror provides asynchronous volume-level replication that is based on a configured replication update interval. The following are recommendations for SnapMirror for SQL Server:

- The destination SVM must be a member of the same Active Directory domain of which the source SVM is a member so that the access control lists (ACLs) stored within NAS files are not broken during recovery from a disaster.

- Using destination volume names that are the same as the source volume names is not required but can make the process of mounting destination volumes into the destination simpler to manage. If CIFS is used, you must make the destination NAS namespace identical in paths and directory structure to the source namespace.
- For consistency purposes, do not schedule SnapMirror update from the controllers. However, enable SnapMirror update from either SMSQL or SnapCenter to update SnapMirror after either full or log backup is completed.
- Distribute volumes that contain SQL Server data across different nodes in the cluster to allow all cluster nodes to share SnapMirror replication activity. This distribution optimizes the use of node resources.
- Mirror the CIFS share used by the availability group to the secondary data center for disaster recovery purposes.

For more information about SnapMirror, see the following resources:

- TR-5015 SnapMirror Configuration and Best Practices Guide for ONTAP 9
- TR-4733 SnapMirror Synchronous for ONTAP 9.7

7 NetApp Cloud Data Services and SQL Server

Whether you are targeting an all-cloud, hybrid cloud, or multicloud strategy, NetApp Cloud Data Services reduces the time it takes for you to deploy or develop SQL Server databases by making the data requirements seamless to the application layer. You can create a highly available solution in the cloud supporting native network file system services based on NetApp ONTAP technology.

For more information on SQL Server Solution and NetApp Cloud Data Services, see Gain Data Intelligence for SQL Databases.

8 Conclusion

SQL Server users typically face a series of significant challenges in their effort to increase the return on their SQL Server investments and optimize their infrastructure to support business and IT requirements. They must:

- Accelerate new database implementations or migrations and lower the risk of these operations.
- Make sure that the underlying storage infrastructure is fully optimized to support SLAs, including performance, scalability, and availability.
- Consolidate existing databases and infrastructure to lower costs.
- Reduce complexity and simplify IT infrastructure.
- Increase the productivity of IT personnel.

To succeed in these challenges, architects, sysadmins, and DBAs look to deploy their databases and storage infrastructure based on proven best practices and technology.

This document covers NetApp's recommendations for designing, optimizing, and scaling Microsoft SQL Server deployments, which can vary greatly between implementations. Options such as cluster awareness and virtualization introduce further variables. The right solution depends on both the technical details of the implementation and the business requirements driving the project.

This document gives common recommendations in the following areas:

- SQL Server workload type
- SQL Server configuration
- Database storage layout

- Storage efficiency

SQL Server databases can be quickly and easily protected using NetApp SnapCenter software with the Plug-in for SQL Server and the Plug-in for Microsoft Windows. These products enable application-consistent backup, automated cloning, and restore and recovery of SQL Server databases, instances, or availability groups.

NetApp and partner professional services experts are available for assistance in complex projects. Even if assistance is not required during the project, NetApp strongly encourages new customers to use professional services for assistance in developing a high-level approach.

Where to Find Additional Information

To learn more about the information described in this document, refer to the following documents and/or websites:

- Gain Data Intelligence for SQL Databases
  https://cloud.netapp.com/solutions/sql-server-database
- NetApp Interoperability Matrix Tool
  http://mysupport.netapp.com/NOW/products/interoperability/
- NetApp Product Documentation:
  docs.netapp.com
- TPC
  http://www.tpc.org/
- HammerDB
  http://www.hammerdb.com/
- SQL Server 2019: Your modern data platform
  https://www.microsoft.com/en-us/sql-server/sql-server-2019-comparison
- Configure the max worker threads Server Configuration Option
  https://msdn.microsoft.com/en-us/library/ms190219.aspx
- NetApp AFF8080 EX Performance and Server Consolidation with Microsoft SQL Server 2014
  https://fieldportal.netapp.com/content/248568?assetComponentId=248696
- Page Compression Implementation
  https://msdn.microsoft.com/en-us/library/cc280464.aspx
- SnapMirror Configuration and Best Practices Guide for ONTAP 9
  https://www.netapp.com/us/media/tr-4015.pdf
- SnapMirror Synchronous for ONTAP 9.7
  https://www.netapp.com/us/media/tr-4733.pdf
- TR-4714: Best Practice Guide for SQL Server using SnapCenter
  https://www.netapp.com/us/media/tr-4714.pdf