Microsoft SQL Server

Enterprise applications

NetApp
May 03, 2024

Table of Contents

Microsoft SQL Server

Microsoft SQL Server on ONTAP

ONTAP delivers an enterprise-class security and performance solution for your Microsoft SQL Server databases while also providing world-class tools to manage your environment.

This documentation replaces the previously published technical report *TR-4590: Best practice guide for Microsoft SQL Server with ONTAP*

NetApp assumes that the reader has working knowledge of the following:

- 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

The scope of this best practices section is limited to technical design based on the design principles and preferred standards that NetApp recommends for storage infrastructure. The end-to-end implementation is out of the scope.

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

Microsoft SQL Server workloads

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.

Database configuration

Microsoft SQL Server CPU configuration

To improve system performance, you need modify SQL Server settings and server configuration to use appropriate number of processors for execution.

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 can use the hyperthreaded processors. This improves performance by increasing parallelization.

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 options for 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.

The figure below shows the SQL Server log message after startup indicating the enforcement of the core limit.

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 2022: Your modern data platform.

CPU affinity

You are unlikely 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 performance can be affected when this occurs. To minimize the impact, you can configure the processors so that all of 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 helpful 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.

NetApp recommends for DSS such as data warehouses, start with MAXDOP at 50 and explore tuning up or down if required. Make sure you measure the critical queries in your application when making changes.

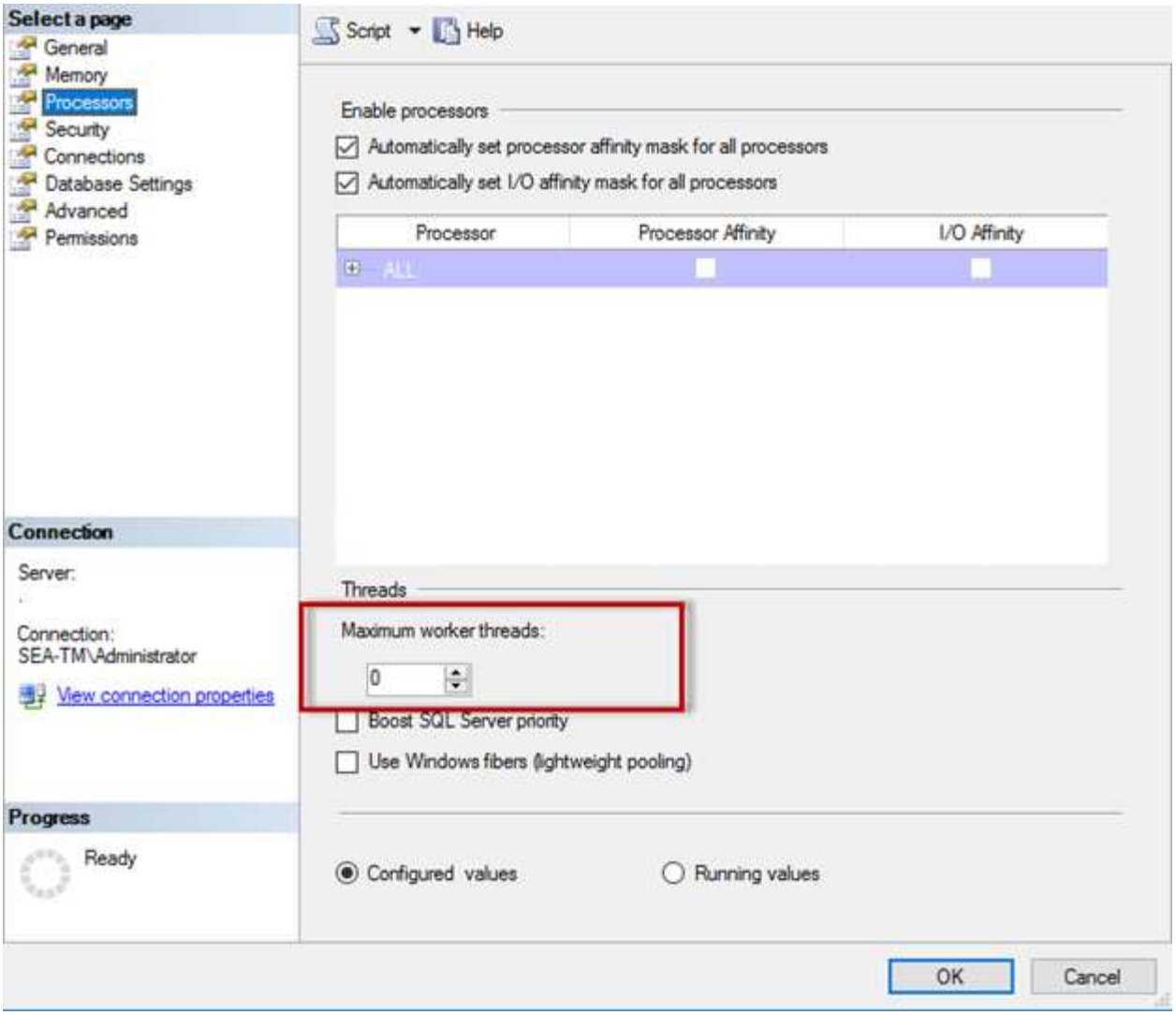
Max worker threads

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.



Configuring max worker threads using SQL Server Management Studio.

```
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
```

Microsoft SQL Server memory configuration

The following section explain configuring SQL server memory settings to optimize database performance.

Max server memory

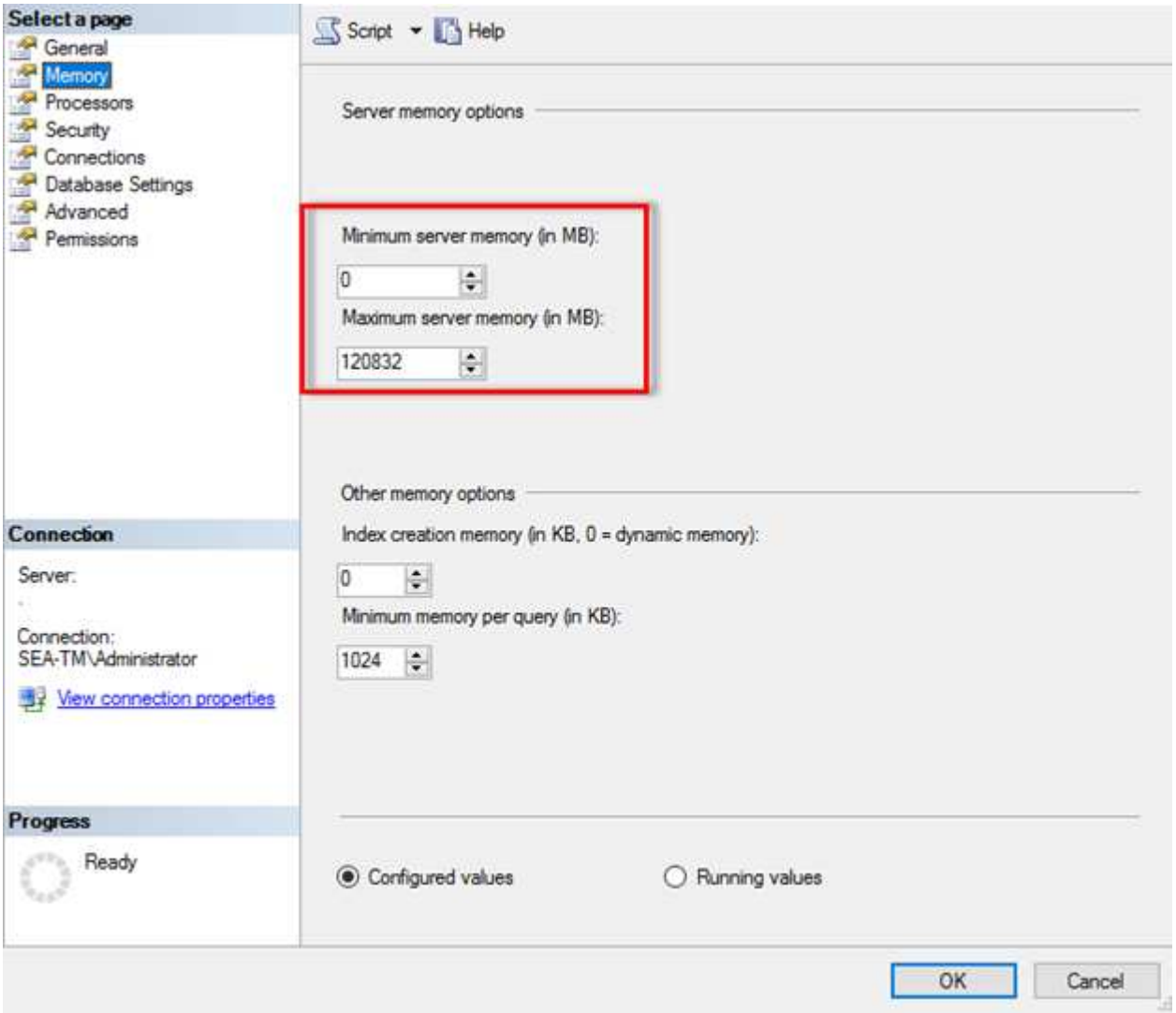
The max server memory option sets the maximum amount of memory that the SQL Server instance can use.

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.



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

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
```

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.

Index create memory

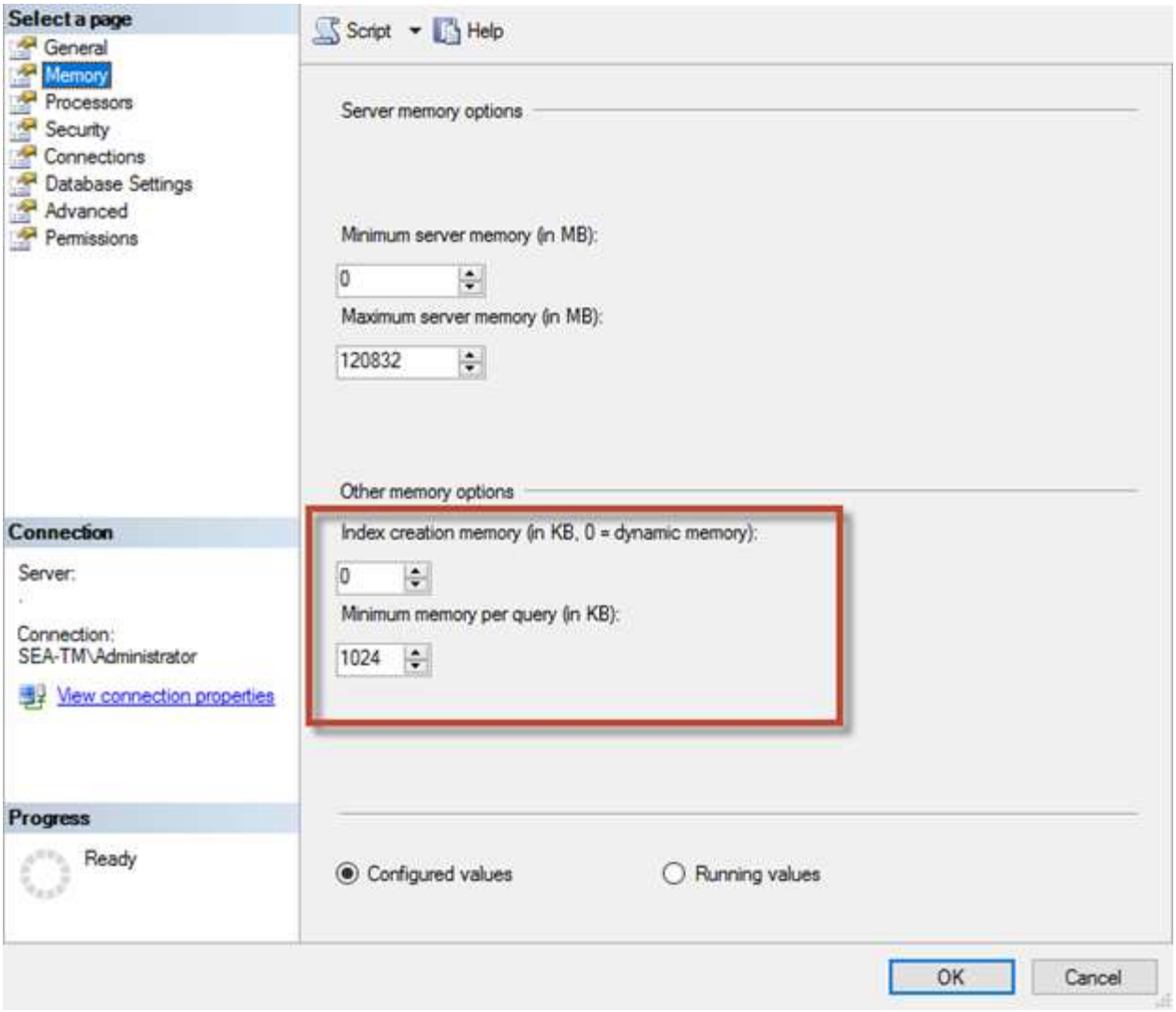
The index create memory option is another advanced option that you should not usually change.

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 to run efficiently.

By default, the min memory per query setting allocates >= 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.



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

NetApp recommends configuring the buffer pool extensions to:

- 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
```

Microsoft SQL Server shared instance versus dedicated instance

Multiple SQL Server can be configured as a single instance per server or as multiple instances. The right decision usually depends on factors such as whether the server is to be used for production or development, whether the instance is considered critical to business operations and performance goals.

Shared instance configurations may be initially easier to configure, but it can lead to problems where resources 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.

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

Storage configuration

Microsoft SQL Server storage considerations

The combination of ONTAP 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 the business grows.

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.

Aggregates

Aggregates are the lowest level storage containers for NetApp storage configurations. Some legacy documentation exists on the internet that recommends separating IO onto different sets of underlying drives. This is not recommended with ONTAP. 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 drives 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 drives 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.

Volumes

NetApp FlexVol volumes are created and reside inside aggregates. This term sometimes causes confusion because an ONTAP volume is not a LUN. An ONTAP volume is a management container for data. A volume could contain files, LUNs or even S3 objects. A volume does not take up space, it is only used for management of the contained data.

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:

- Avoid sharing volumes between hosts. For example, while it would be possible to create 2 LUNs in a single volume and share each LUN to a different host, this should be avoided because it can complicate management.
- 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. 17 Best practice guide for Microsoft SQL Server with ONTAP © 2022 NetApp, Inc. All rights reserved.
- 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 snapshot reserve value in the volume to zero for ease of monitoring from an operational perspective.
- Disable snapshot schedules and retention policies. Instead, use SnapCenter to coordinate Snapshot copies of the SQL Server data volumes.
- Place the SQL Server system databases on a dedicated volume.
- 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.

LUNs

- Make sure that the user database files and the log directory to store log backup are on separate volumes to prevent the retention policy from overwriting snapshots when these are used with SnapVault technology.
- Make sure that SQL Server databases reside on LUNs separate from LUNs that have non-database 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 you create LUNs with DiskManager or other tools, make sure that the allocation unit size is set to 64K for partitions when formatting the LUNs.
- See the Microsoft Windows and native MPIO under ONTAP best practices for modern SAN to apply multipathing support on Windows to iSCSI devices in the MPIO properties.

Microsoft SQL Server database files and filegroups

Proper SQL Server database file placement on ONTAP is critical during initial deployment stage. This ensures optimal performance, space management, backup and restore times that can be configured to match your business requirements.

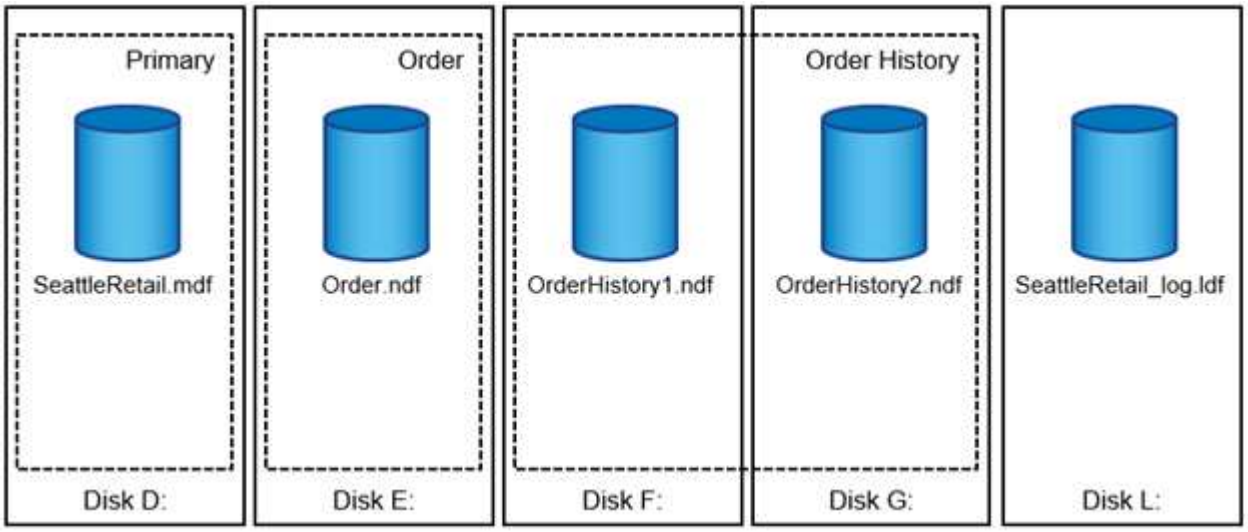
In theory, SQL Server (64-bit) supports 32,767 databases per instance and 524,272TB of database size, although the typical installation usually has several databases. However, the number of the databases SQL Server can handle depends on the load and hardware. It is not unusual to see SQL Server instances 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.



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.

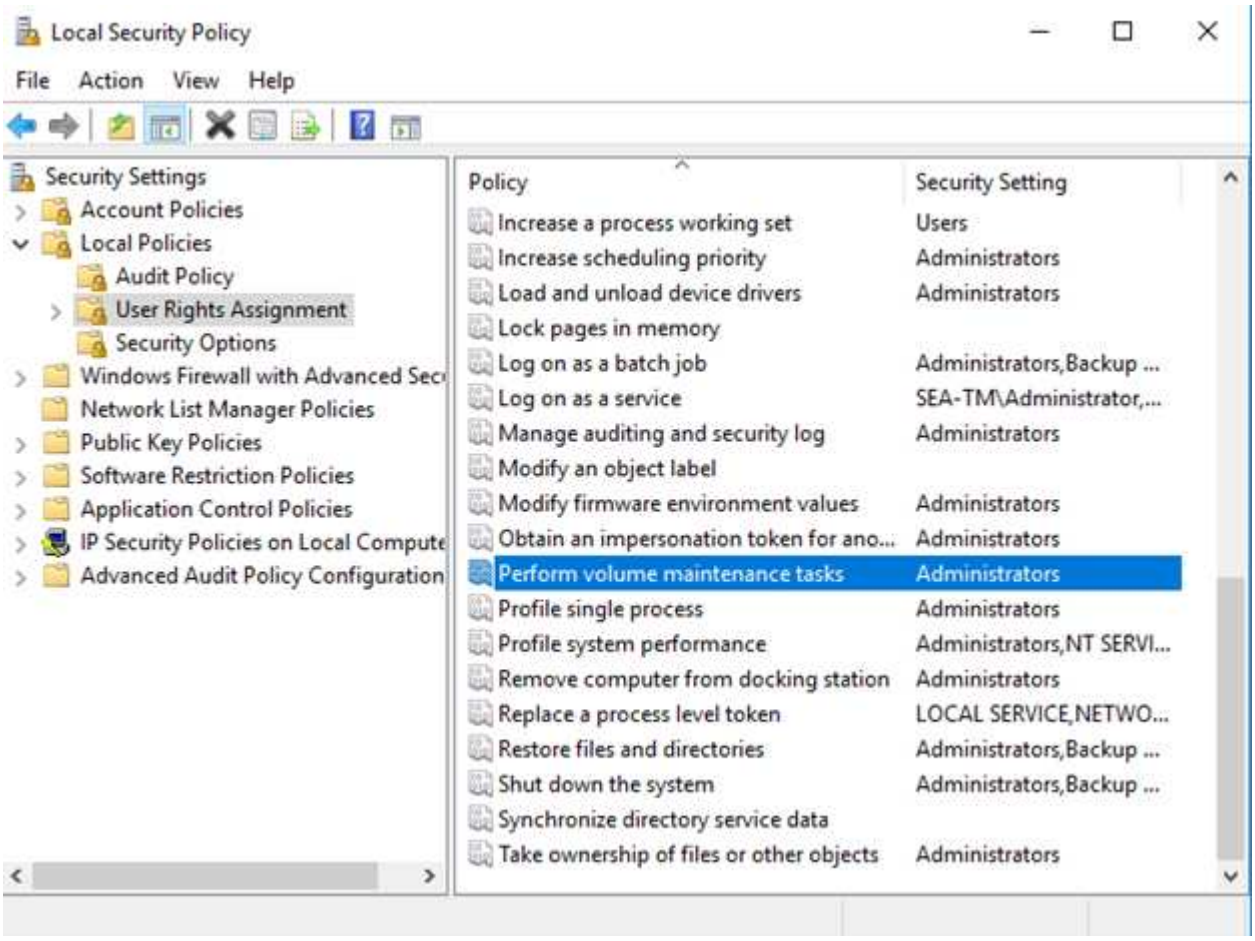
NetApp recommends that all files in the single filegroup have the same initial size and autogrowth parameters, with the grow size defined in megabytes rather than percentages. This helps the proportional fill algorithm evenly balance write activities across data files.

Every time SQL Server grows files, it fills newly allocated space 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.

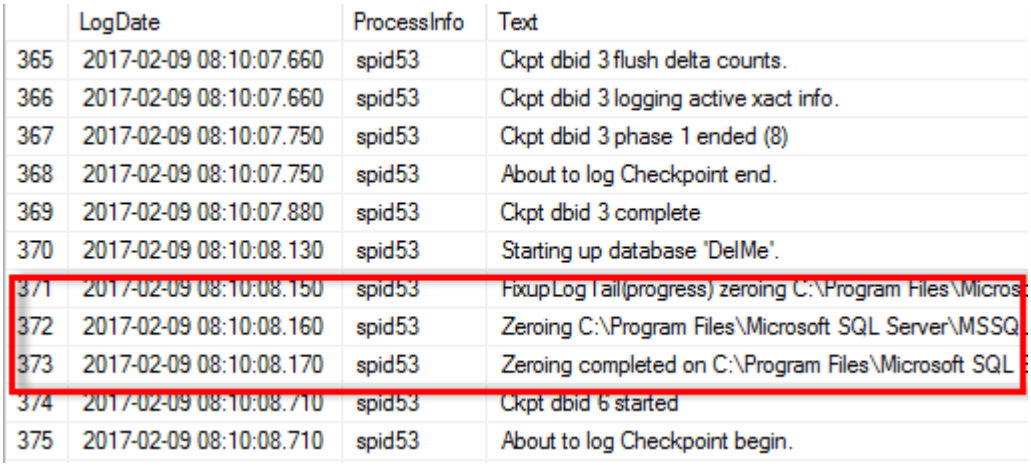
You can enable instant file initialization by adding the SA_MANAGE_VOLUME_NAME permission, also known as “perform volume maintenance task,” to the SQL Server startup account. You can do this under the local security policy management application (secpol.msc), as shown in the following figure. Open the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



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.

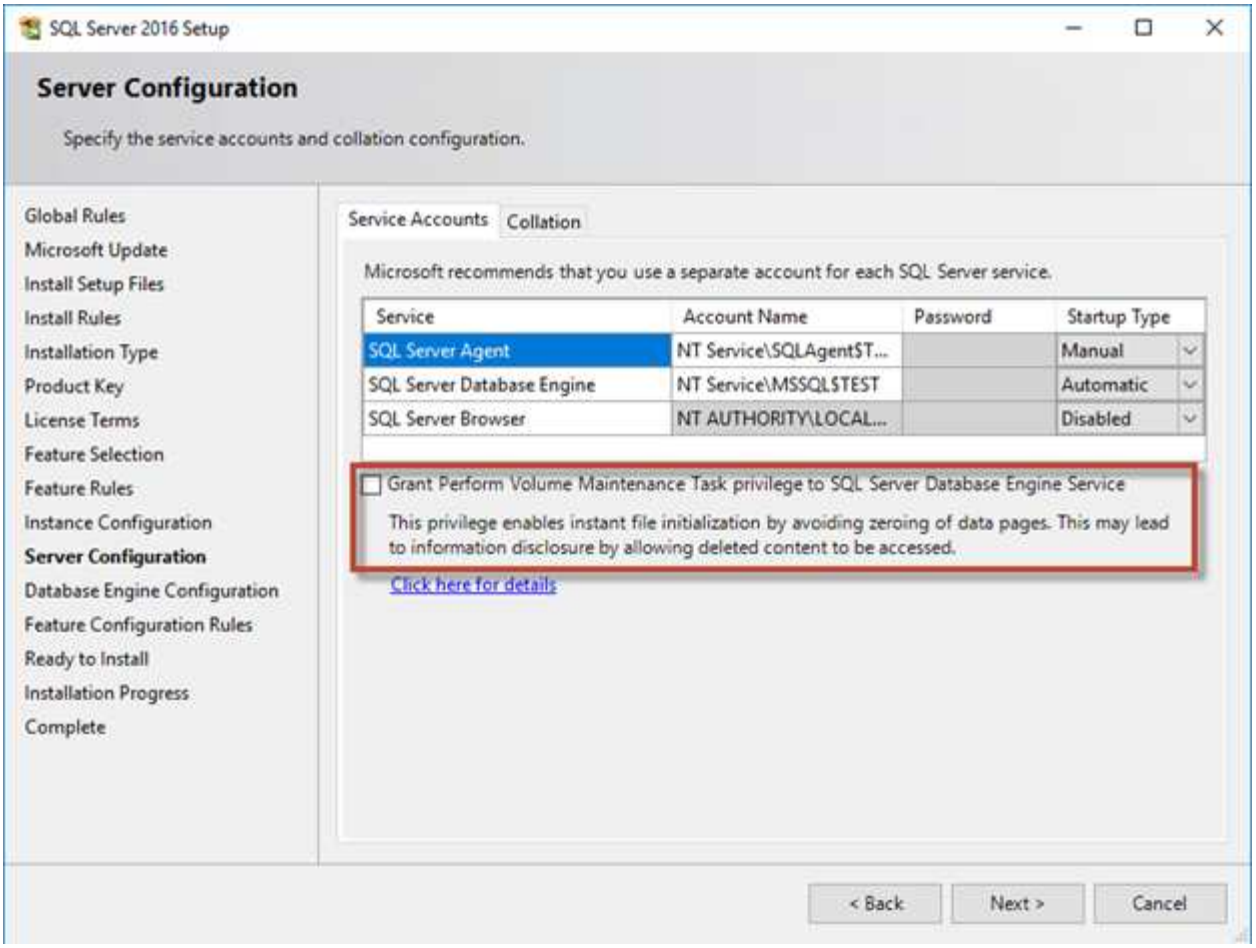
```
DBCC TRACEON(3004,3605,-1)
GO
CREATE DATABASE DelMe
GO
EXECUTE sp_readerrorlog
GO
DROP DATABASE DelMe
GO
DBCC TRACEOFF(3004,3605,-1)
GO
```

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. This figure displays the option to grant the SQL Server database engine service the privilege to perform the volume maintenance task.



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.

Microsoft SQL Server log directory

The log directory is specified in SQL Server to store transaction log backup data at the host level. If you are using SnapCenter to backup log files then 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 = ((maximum DB LDF size x daily log change rate %) x (snapshot retention) ÷ (1 - LUN overhead space %) The host log directory sizing formula assumes a 10% LUN overhead space

Place the 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. SnapCenter allows only one host log directory per SQL Server host. You can configure the host log directories at SnapCenter -→ Host -→ Configure Plug-in.

NetApp recommends the following for a host log directory:

- Make sure that the host log directory is not shared by any other type of data that can potentially corrupt the backup snapshot data.
- Do not place user databases or system databases on a LUN that hosts mount points.
- Create the host log directory on the dedicated FlexVol volume to which SnapCenter copies transaction logs.
- Use SnapCenter 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.

Microsoft SQL Server tempdb files

Tempdb database can be heavily utilized. In addition to optimal placement of user database files on ONTAP, alter tempdb datafiles to reduce allocation contention

Page contention can occur on lobal allocation map (GAM), shared global allocation map (SGAM), or page free space (PFS) pages when SQL Server must write to special system pages to allocate new objects. Latches protect (lock) these pages in memory. On a busy SQL Server instance, 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.

Microsoft SQL Server and storage efficiency

ONTAP storage efficiency is optimized 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 is a combination of RAID, provisioning (overall layout and utilization), mirroring, and other data protection technologies. NetApp technologies including snapshots, thin provisioning, and cloning optimizes existing storage in the infrastructure and deferring or avoiding future storage expenditures. The more you use these technologies together, the larger the savings.

Space efficiency features, such as compression, compaction, and deduplication are designed to increase the amount of logical data that fits on a given amount of physical storage. The result is lower costs and management overhead.

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. Compaction allows multiple logical blocks of data to share the same physical block on media.

See the sections below on thin provisioning for an explanation of the interaction between storage efficiency and fractional reservation.

Compression

Prior to the availability of all-flash storage systems, array-based compression was of limited value because most I/O-intensive workloads required a very large number of spindles to provide acceptable performance. Storage systems invariably contained much more capacity than required as a side effect of the large number of drives. The situation has changed with the rise of solid-state storage. There is no longer a need to vastly overprovision drives purely to obtain good performance. The drive space in a storage system can be matched to actual capacity needs.

The increased IOPS capability of solid-state drives (SSDs) almost always yields cost savings compared to spinning drives, but compression can achieve further savings by increasing the effective capacity of solid-state media.

There are several ways to compress data. Many databases include their own compression capabilities, but this is rarely observed in customer environments. The reason is usually the performance penalty for a **change** to compressed data, plus with some applications there are high licensing costs for database-level compression. Finally, there is the overall performance consequences to database operations. It makes little sense to pay a high per-CPU license cost for a CPU that performs data compression and decompression rather than real database work. A better option is to offload the compression work on to the storage system.

Adaptive compression

Adaptive compression has been thoroughly tested with enterprise workloads with no observed effect on performance, even in an all-flash environment in which latency is measured in microseconds. Some customers have even reported a performance increase with the use of compression because the data remains compressed in cache, effectively increasing the amount of available cache in a controller.

ONTAP manages physical blocks in 4KB units. Adaptive compression uses a default compression block size of 8KB, which means data is compressed in 8KB units. This matches the 8KB block size most often used by relational databases. Compression algorithms become more efficient as more data is compressed as a single unit. A 32KB compression block size would be more space-efficient than an 8KB compression block unit. This

does mean that adaptive compression using the default 8KB block size does lead to slightly lower efficiency rates, but there is also a significant benefit to using a smaller compression block size. Database workloads include a large amount of overwrite activity. Overwriting a 8KB of a compressed 32KB block of data requires reading back the entire 32KB of logical data, decompressing it, updating the required 8KB region, recompressing, and then writing the entire 32KB back to the drives. This is a very expensive operation for a storage system and is the reason some competing storage arrays based on larger compression block sizes also incur a significant performance penalty with database workloads.

The block size used by adaptive compression can be increased up to 32KB. This may improve storage efficiency and should be considered for quiescent files such as transaction logs and backup files when a substantial amount of such data is stored on the array. In some situations, active databases that use a 16KB or 32KB block size may also benefit from increasing the block size of adaptive compression to match. Consult a NetApp or partner representative for guidance on whether this is appropriate for your workload.

Compression block sizes larger than 8KB should not be used alongside deduplication on streaming backup destinations. The reason is small changes to the backed-up data affect the 32KB compression window. If the window shifts, the resulting compressed data differs across the entire file. Deduplication occurs after compression, which means the deduplication engine sees each compressed backup differently. If deduplication of streaming backups is required, only 8KB block adaptive compression should be used. Adaptive compression is preferable, because it works at a smaller block size and does not disrupt deduplication efficiency. For similar reasons, host-side compression also interferes with deduplication efficiency.

Compression alignment

Adaptive compression in a database environment requires some consideration of compression block alignment. Doing so is only a concern for data that is subject to random overwrites of very specific blocks. This approach is similar in concept to overall file system alignment, where the start of a filesystem must be aligned to a 4K device boundary and the blocksize of a filesystem must be a multiple of 4K.

For example, an 8KB write to a file is compressed only if it aligns with an 8KB boundary within the file system itself. This point means that it must fall on the first 8KB of the file, the second 8KB of the file, and so forth. The simplest way to ensure correct alignment is to use the correct LUN type, any partition created should have an offset from the start of the device that is a multiple of 8K, and use a filesystem block size that is a multiple of the database block size.

Data such as backups or transaction logs are sequentially written operations that span multiple blocks, all of which are compressed. Therefore, there is no need to consider alignment. The only I/O pattern of concern is the random overwrites of files.

Data compaction

Data compaction is a technology that improves compression efficiency. As stated previously, adaptive compression alone can provide at best 2:1 savings because it is limited to storing an 8KB I/O in a 4KB WAFL block. Compression methods with larger block sizes deliver better efficiency. However, they are not suitable for data that is subject to small block overwrites. Decompressing 32KB units of data, updating an 8KB portion, recompressing, and writing back to the drives creates overhead.

Data compaction works by allowing multiple logical blocks to be stored within physical blocks. For example, a database with highly compressible data such as text or partially full blocks may compress from 8KB to 1KB. Without compaction, that 1KB of data would still occupy 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 the drives and therefore

should not create any detectable performance effect.

The degree of savings obtained vary. Data that is already compressed or encrypted cannot generally be further compressed, and therefore such datasets do not benefit from compaction. In contrast, newly initialized datafiles that contain little more than block metadata and zeros compress up to 80:1.

Temperature sensitive storage efficiency

Temperature sensitive storage efficiency (TSSE) is a available in ONTAP 9.8 and later that relies on block access heat maps to identify infrequently accessed blocks and compress them with greater efficiency.

Deduplication

Deduplication is the removal of duplicate block sizes from a dataset. For example, if the same 4KB block existed in 10 different files, deduplication would redirect that 4KB block within all 10 files to the same 4KB physical block. The result would be a 10:1 improvement in efficiency for that data.

Data such as VMware guest boot LUNs usually deduplicate extremely well because they consist of multiple copies of the same operating system files. Efficiency of 100:1 and greater have been observed.

Some data does not contain duplicate data. For example, an Oracle block contains a header that is globally unique to the database and a trailer that is nearly unique. As a result, deduplication of an Oracle database rarely delivers more than 1% savings. Deduplication with MS SQL databases is slightly better, but unique metadata at the block level is still a limitation.

Space savings of up to 15% in databases with 16KB and large block sizes have been observed in a few cases. The initial 4KB of each block contains the globally unique header, and the final 4KB block contains the nearly unique trailer. The internal blocks are candidates for deduplication, although in practice this is almost entirely attributed to the deduplication of zeroed data.

Many competing arrays claim the ability to deduplicate 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 end result is the same; multiple copies of a database that share most of the underlying physical blocks are created. Using FlexClone is much more efficient than taking the time to copy database files and then deduplicating them. It is, in effect, nonduplication rather than deduplication, because a duplicate is never created in the first place.

Efficiency and thin provisioning

Efficiency features are forms of thin provisioning. For example, a 100GB LUN occupying a 100GB volume might compress down to 50GB. There are no actual savings realized yet because the volume is still 100GB. The volume must first be reduced in size so that the space saved can be used elsewhere on the system. If later changes to the 100GB LUN result in the data becoming less compressible, then the LUN grows in size and the volume could fill up.

Thin provisioning is strongly recommended because it can simplify management while delivering a substantial improvement in usable capacity with associated cost savings. The reason is simple - database environments frequently include a lot of empty space, a large number of volumes and LUNs, and compressible data. Thick provisioning results in the reservation of space on storage for volumes and LUNs just in case they someday become 100% full and contain 100% uncompressible data. That is unlikely to ever occur. Thin provisioning allows that space to be reclaimed and used elsewhere and allows capacity management to be based on the storage system itself rather than many smaller volumes and LUNs.

Some customers prefer to use thick provisioning, either for specific workloads or generally based on established operational and procurement practices.

Caution: If a volume is thick provisioned, care must be taken to completely disable all efficiency features for that volume, including decompression and the removal of deduplication using the `sis undo` command. The volume should not appear in `volume efficiency show` output. If it does, the volume is still partially configured for efficiency features. As a result, overwrite guarantees work differently, which increases the chance that configuration oversights cause the volume to unexpectedly run out of space, resulting in database I/O errors.

Efficiency best practices

NetApp recommends the following:

AFF defaults

Volumes created on ONTAP running on an all-flash AFF system are thin provisioned with all inline efficiency features enabled. Although databases generally do not benefit from deduplication and may include uncompressible data, the default settings are nevertheless appropriate for almost all workloads. ONTAP is designed to efficiently process all types of data and I/O patterns, whether or not they result in savings. Defaults should only be changed if the reasons are fully understood and there is a benefit to deviating.

General recommendations

- If volumes and/or LUNs are not thin provisioned, you should must disable all efficiency settings because using these features provides no savings and the combination of thick provisioning with space efficiency enabled can cause unexpected behavior, including out-of-space errors.
- If data is not subject to overwrites, such as with backups or database transaction logs, you can achieve greater efficiency by enabling TSSE with a low cooling period.
- Some files might contain a significant amount of uncompressible data, for example when compression is already enabled at the application level of files are encrypted. If any of these scenarios are true, consider disabling compression to allow more efficient operation on other volumes containing compressible data.
- Do not use both 32KB compression and deduplication with database backups. See the section Adaptive compression for details.

Database compression

SQL Server itself also has features to compress and efficiently manage data. 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.

Here are the recommendation for managing space for SQL Server data files

- Use thin provisioning in SQL Server environments to improve space utilization and to reduce the overall storage requirements when the space guarantee functionality is used.
- Use autogrow for most common deployment configurations because the storage admin only needs to monitor space usage in the aggregate.

- Advice not to 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.

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.

Microsoft SQL Server data protection with NetApp management software

Planning database backup is based on business requirements. By combining ONTAP's NetApp Snapshot technology and leveraging Microsoft SQL Server API's, you can quickly take application consistent backup irrespective of size of user databases. For more advanced or scale-out data management requirements, NetApp offers SnapCenter.

SnapCenter

SnapCenter is the NetApp data protection software for enterprise applications. SQL Server databases can be quickly and easily protected with the SnapCenter Plug-in for SQL Server and with OS operations managed by the SnapCenter Plug-in for Microsoft Windows.

SQL Server instance can be a standalone setup, failover cluster instance or it can be always on availability group. The result is that from single-pane-of-glass, databases can be protected, cloned and restored from primary or secondary copy. SnapCenter can manage SQL Server databases both on-premises, in the cloud, and hybrid configurations.Database copies can also be created in few minutes on the orignal or alternate host for development or for reporting purpose.

> **NetApp recommends** using SnapCenter to create Snapshot copies. The T-SQL method described below also works, but SnapCenter offers complete automation over the backup, restore, and cloning process. It also performs discovery to ensure the correct snapshots are being created. No pre-configuration is required. … SQL Server also requires coordination between the OS and the storage to ensure the correct data is present in snapshots at the time of creation. In most cases, the only safe method to do this is with SnapCenter or T-SQL. Snapshots created without this additional coordination may not be reliably recoverable.

For more details about the SQL Server Plug-in for SnapCenter, see TR-4714: Best practice guide for SQL Server using NetApp SnapCenter.

Protecting database using T-SQL snapshots

In SQL Server 2022, Microsoft introduced T-SQL snapshots that offers a path to scripting and automation of backups operations. Rather than performing full-sized copies, you can prepare the database for snapshots. Once the database is ready for backup, you can leveraging ONTAP REST API's to create snapshots..

The following is a sample backup workflow:

1. Freeze a database with ALTER command. This prepares the database for a consistent snapshot on the underlying storage. After the freeze you can thaw the database and record the snapshot with BACKUP command.
2. Perform snapshots of multiple databases on the storage volumes simultaneously with the new BACKUP GROUP and BACKUP SERVER commands.
3. Perform FULL backups or COPY_ONLY FULL backups. These backups are recorded in msdb as well.
4. Perform point-in-time recovery using log backups taken with the normal streaming approach after the snapshot FULL backup. Streaming differential backups are also supported if desired.

To learn more, see Microsoft documentation to know about the T-SQL snapshots.

Microsoft SQL Server disaster recovery with ONTAP

Enterprise databases and application infrastructures often require replication to protect from natural disaster or unexpected business disruption with minimal downtime.

The SQL Server Always-On availability group replication feature can be an excellent option, and NetApp offers optiosn to integrate data protection with Always-On. In some cases, however, you might want to consider ONTAP replication technology. ONTAP replication options, including MetroCluster and SnapMirror, can scale better with minimal performance impact, protect non-SQL data, and generally provide a full-infrastructure replication and DR solution.

SnapMirror asynchronous

SnapMirror technology offers a fast and flexible asynchronous enterprise solution for replicating data over LANs and WANs. SnapMirror technology transfers only changed data blocks to the destination after the initial mirror is created, significantly reducing network bandwidth requirements.

The following are recommendations for SnapMirror for SQL Server:

- If CIFS is used, 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 updates from the controllers. Instead, enable SnapMirror updates from 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.

For more information about SnapMirror, see TR-4015: SnapMirror Configuration and Best Practices Guide for ONTAP 9.

Securing Microsoft SQL Server on ONTAP

Securing a SQL Server database environment is a multdimensional effort that goes

beyond managing the database itself. ONTAP offers several unique features designed to secure the storage aspect of your database infrastructure.

Snapshot copies

Storage snapshots are point-in-time replicas of the target data. ONTAP's implementation includes the abilities to set various policies and store up to 1024 snapshots per volume. Snapshots in ONTAP are space-efficient. Space is only consumed as the original dataset changes. They are also read-only. A snapshot can be deleted, but it cannot be changed.

In some cases, snapshots can be scheduled directly on ONTAP. In other cases, software such as SnapCenter may be required to orchestrate application or OS operations before creating snapshots. Whichever approach is best for your workload, an aggressive snapshot strategy can provide data security through frequent, easily-accessible access to backups of everything from boot LUNs to mission-critical databases.

Note: An ONTAP Flexible Volume, or more simply, a volume is not synonymous with a LUN. Volumes are management containers for data such as files or LUNs. For example, a database might be placed on an 8-LUN stripe set, with all LUNs contained in a single volume.

For more information on snapshots, click here.

Tamperproof snapshots

Beginning with ONTAP 9.12.1, snapshots are not just read-only, they can also be protected from accidental or intentional deletion. The feature is called Tamperproof Snapshots. A retention period can be set and enforced via snapshot policy. The resulting snapshots cannot be deleted until they have reached their expiration date. There are no administrative or support center overrides.

This ensures that an intruder, a malicious insider, or even a ransomware attack is unable to compromise the backups even if it resulted in access to the ONTAP system itself. When combined with an frequent snapshot schedule, the result is extremely powerful data protection with a very low RPO.

For more information on Tamperproof Snapshots, click here.

SnapMirror replication

Snapshots can also be replicated to a remote system. This includes Tamperproof Snapshots, where the retention period is applied and enforced on the remote system. The result is the same data protection benefits as local snapshots, but the data is located on a second storage array. This ensures that destruction of the original array does not compromise the backups.

A second system also opens new options for administrative security. For example, some NetApp customers segregate authentication credentials for the primary and secondary storage systems. No single administrative user has access to both systems, which means a malicious administrator cannot delete all copies of data.

For more information on SnapMirror, click here.

Storage Virtual Machines

A newly configured ONTAP storage system is similar to a newly provisioned VMware ESX server because neither of them can support any users until a virtual machine is created. With ONTAP, you create a Storage Virtual Machine (SVM) which becomes the most basic unit of storage management. Each SVM has its own storage resources, protocol configurations, IP addresses, and FCP WWNs. This is the foundation of ONTAP mult-tenancy.

For example, you might configure one SVM for critical production workloads, and a second SVM on a different network segment for development activities. You could then restrict access to the production SVM to certain administrators, while granting developers more expansive control over the storage resources in the development SVM. You might also need to provide a third SVM to your financial and HR teams to store especially critical eyes-only data.

For more information about SVMs, click here.

Administrative RBAC

ONTAP offers powerful role-based access control (RBAC) for administrative logins. Some admins might need full cluster access, while others might only need access to certain SVMs. Advanced helpdesk personnel might need the ability to increase volumes sizes. The result is you can grant administrative users the access required to perform their job responsibilities, and nothing more. Furthermore, you can secure these logins using PKI from various vendors, restrict access to ssh keys only, and enforce failed login attempt lockouts.

For more information on administrative access control, click here.

Multfactor authentication

ONTAP and certain other NetApp products now support multifactor authentication (MFA) using a variety of methods. The result is a compromised username/password alone is not a security thread without the data from the second factor, such as a FOB or a smartphone app.

For more information, click here.

API RBAC

Automation requires API calls, but not all tools require full administrative access. To help secure automation systems, RBAC is also available at the API level. You can limit the automation user accounts to the API calls required. For example, monitoring software does not need change access, it only requires read access. Workflows that provision storage do not need the ability to delete storage.

To learn more, start chttps://docs.netapp.com/us-en/ontap-automation/rest/rbac_overview.html[here.]

Multi-admin verification (MAV)

Multi "factor" authentication can be taken even further by requiring two different administrators, each with their own credentials, to approve certain activities. This includes changing login permissions, running diagnostic commands, and deleting data.

For more information on multi-admin verification (MAV), click here

Copyright information

Trademark information