



# **Storage-Konfiguration**

## Enterprise applications

NetApp

March 26, 2025

# Inhalt

- Storage-Konfiguration ..... 1
  - Überblick ..... 1
    - Datenspeicher Design ..... 1
    - Aggregate ..... 1
    - Volumes ..... 1
    - LUNs ..... 2
  - Datenbankdateien und Dateigruppen ..... 3
  - Protokollverzeichnis ..... 7
  - Tempdb-Dateien ..... 8
- Storage-Effizienz ..... 10
  - Komprimierung ..... 10
  - Data-Compaction ..... 12
  - Deduplizierung ..... 12
  - Effizienz und Thin Provisioning ..... 13
  - Best Practices für Effizienz ..... 13
  - Datenbankkomprimierung ..... 14
  - Speicherplatzrückgewinnung ..... 14

# Storage-Konfiguration

## Überblick

Die Kombination aus ONTAP Storage-Lösungen und Microsoft SQL Server ermöglicht Datenbank-Storage-Designs der Enterprise-Klasse, die auch die anspruchsvollsten Applikationsanforderungen von heute erfüllen.

Um eine SQL Server auf ONTAP-Lösung zu optimieren, müssen Sie das I/O-Muster und die Merkmale des SQL Servers kennen. Ein gut konzipiertes Storage Layout für eine SQL Server Datenbank muss die Performance-Anforderungen von SQL Server unterstützen und gleichzeitig maximale Management-Fähigkeit der Infrastruktur als Ganzes bieten. Ein gutes Storage-Layout ermöglicht außerdem eine erfolgreiche Erstimplementierung und ein reibungsloses Wachstum der Umgebung im Laufe der Zeit, während das Unternehmen wächst.

## Datenspeicher Design

Für SQL Server-Datenbanken, die keine Backups mit SnapCenter durchführen, empfiehlt Microsoft, die Daten und Log-Dateien auf separaten Laufwerken zu platzieren. Bei Anwendungen, die gleichzeitig Daten aktualisieren und anfordern, ist die Protokolldatei schreibintensiv und die Datendatei (je nach Anwendung) ist Lese-/schreibintensiv. Für den Datenabruf wird die Protokolldatei nicht benötigt. Daher können Datenanfragen aus der Datendatei auf dem eigenen Laufwerk bearbeitet werden.

Wenn Sie eine neue Datenbank erstellen, empfiehlt Microsoft, getrennte Laufwerke für die Daten und Protokolle anzugeben. Um Dateien nach der Datenbankerstellung zu verschieben, muss die Datenbank offline geschaltet werden. Weitere Empfehlungen von Microsoft finden Sie unter ["Platzieren Sie Daten- und Protokolldateien auf separaten Laufwerken"](#).

## Aggregate

Aggregate sind die Storage-Container der niedrigsten Ebene für NetApp Storage-Konfigurationen. Im Internet gibt es eine ältere Dokumentation, die die Trennung von E/A auf verschiedene Sätze zugrunde liegender Laufwerke empfiehlt. Dies wird bei ONTAP nicht empfohlen. NetApp hat verschiedene I/O-Workload-Merkmalstests mit gemeinsam genutzten und dedizierten Aggregaten mit getrennten Datendateien und Transaktions-Log-Dateien durchgeführt. Tests zeigen, dass ein großes Aggregat mit mehr RAID-Gruppen und -Laufwerken die Storage Performance optimiert und verbessert und Administratoren aus zwei Gründen einfacher zu managen sind:

- Ein großes Aggregat macht die I/O-Funktionen aller Laufwerke für alle Dateien verfügbar.
- Ein großes Aggregat ermöglicht die effizienteste Nutzung von Festplattenspeicher.

Platzieren Sie für Hochverfügbarkeit (HA) das sekundäre synchrone Replikat der SQL Server Always On Availability Group auf einer separaten Storage Virtual Machine (SVM) im Aggregat. Platzieren Sie zum Zweck der Disaster Recovery das asynchrone Replikat in einem Aggregat, das Teil eines separaten Storage-Clusters am DR-Standort ist, und Inhalte werden mithilfe der NetApp SnapMirror Technologie repliziert. NetApp empfiehlt eine Verfügbarkeit von mindestens 10 % freien Speicherplatz in einem Aggregat zugunsten der optimalen Storage-Performance.

## Volumes

Volumes werden erstellt und befinden sich in den Aggregaten. Dieser Begriff verursacht manchmal Verwirrung,

weil ein ONTAP Volume keine LUN ist. Ein ONTAP Volume ist ein Management-Container für Daten. Ein Volume kann Dateien, LUNs oder sogar S3 Objekte enthalten. Ein Volume benötigt keinen Speicherplatz, sondern wird nur für das Management der enthaltenen Daten verwendet.

## Überlegungen zum Volume-Design

Bevor Sie ein Datenbank-Volume-Design erstellen, ist es wichtig zu wissen, wie das I/O-Muster und die Merkmale von SQL Server je nach Workload und Backup- und Recovery-Anforderungen variieren. Beachten Sie die folgenden NetApp Empfehlungen für flexible Volumes:

- Vermeiden Sie die gemeinsame Nutzung von Volumes zwischen Hosts. Beispielsweise wäre es möglich, 2 LUNs in einem einzelnen Volume zu erstellen und jede LUN mit einem anderen Host zu teilen. Dies sollte jedoch vermieden werden, da das Management dadurch komplizierter wird. Vermeiden Sie bei der Ausführung mehrerer SQL Server-Instanzen auf demselben Host, sofern Sie sich nicht nahe an der Volume-Grenze auf einem Node befinden, die gemeinsame Nutzung von Volumes und stattdessen ein separates Volume pro Instanz pro Host zur Vereinfachung des Datenmanagements.
- Verwenden Sie NTFS-Bereitstellungspunkte anstelle von Laufwerksbuchstaben, um die Beschränkung auf 26 Laufwerksbuchstaben in Windows zu überschreiten. Bei der Verwendung von Volume-Mount-Punkten wird generell empfohlen, dem Volume-Label den gleichen Namen wie dem Mount-Punkt zu geben.
- Konfigurieren Sie bei Bedarf eine Richtlinie für die automatische Größenanpassung von Volumes, um Speicherplatzbelegung zu verhindern.
- Wenn Sie SQL Server auf einer SMB-Freigabe installieren, stellen Sie sicher, dass Unicode auf den SMB-Volumes zum Erstellen von Ordnern aktiviert ist.
- Setzen Sie den Wert der Snapshot-Reserve im Volume auf null, um die Überwachung aus betrieblicher Sicht zu vereinfachen.
- Snapshot Zeitpläne und Aufbewahrungsrichtlinien deaktivieren Stattdessen können Sie SnapCenter verwenden, um Snapshot Kopien der SQL Server-Daten-Volumes zu koordinieren.
- Platzieren Sie die SQL Server Systemdatenbanken auf einem dedizierten Volume.
- Tempdb ist eine Systemdatenbank, die von SQL Server als temporärer Arbeitsbereich verwendet wird, insbesondere für I/O-intensive DBCC-CHECKDB-Vorgänge. Platzieren Sie diese Datenbank daher auf einem dedizierten Volume mit einem separaten Satz von Spindeln. In großen Umgebungen, in denen die Volume-Anzahl eine Herausforderung ist, können Sie tempdb in weniger Volumes konsolidieren und im gleichen Volume wie andere Systemdatenbanken nach einer sorgfältigen Planung speichern. Datenschutz für tempdb hat keine hohe Priorität, da diese Datenbank bei jedem Neustart von SQL Server neu erstellt wird.
- Legen Sie Benutzerdatendateien (.mdf) auf separaten Volumes, da es sich um zufällige Lese-/Schreib-Workloads handelt. Es ist üblich, Transaktions-Log-Backups häufiger zu erstellen als Datenbank-Backups. Legen Sie daher Transaktions-Log-Dateien (.ldf) auf ein separates Volume oder VMDK aus den Datendateien, so dass für jedes Volume unabhängige Backup-Zeitpläne erstellt werden können. Durch diese Trennung werden auch die I/O-Vorgänge bei sequenziellen Schreibvorgängen aus den I/O-Vorgängen für zufällige Lese-/Schreibzugriffe von Datendateien isoliert und die SQL Server Performance deutlich verbessert.

## LUNs

- Stellen Sie sicher, dass sich die Benutzerdatenbankdateien und das Protokollverzeichnis für das Protokoll-Backup auf separaten Volumes befinden, damit die Aufbewahrungsrichtlinie Snapshots bei Verwendung der SnapVault-Technologie nicht überschreitet.
- Mischen Sie keine Datenbank- und nicht-Datenbankdateien, wie z. B. Dateien mit Volltextsuche, auf derselben LUN.

- Wenn sekundäre Datenbankdateien (als Teil einer Dateigruppe) auf separate Volumes platziert werden, wird die Performance der SQL Server Datenbank verbessert. Diese Trennung ist nur gültig, wenn die Datei der Datenbank `.mdf` ihre LUN nicht mit anderen Dateien teilt `.mdf`.
- Wenn Sie LUNs mit DiskManager oder anderen Werkzeugen erstellen, stellen Sie sicher, dass die Größe der Zuordnungseinheit beim Formatieren der LUNs auf 64K für Partitionen festgelegt ist.
- Siehe "[Microsoft Windows und natives MPIO unter den Best Practices von ONTAP für modernes SAN](#)" So wenden Sie Multipathing-Unterstützung unter Windows auf iSCSI-Geräte in den MPIO-Eigenschaften an.

## Datenbankdateien und Dateigruppen

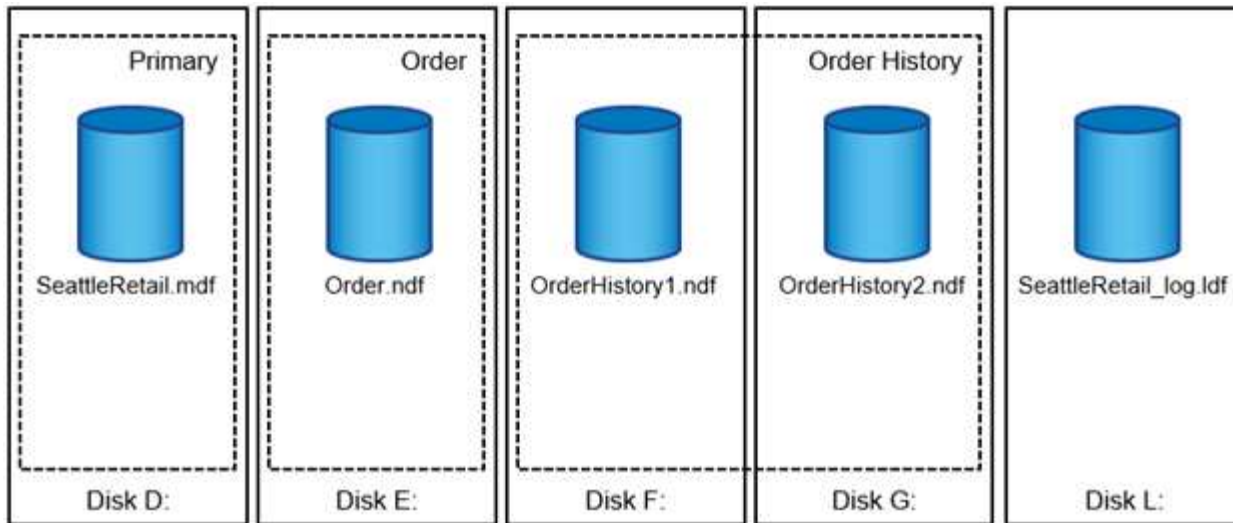
Die korrekte Platzierung von SQL Server-Datenbankdateien auf ONTAP ist in der ersten Implementierungsphase entscheidend. Dies sorgt für optimale Performance, Speicherplatz-Management, Backup- und Wiederherstellungszeiten, die Ihren geschäftlichen Anforderungen entsprechend konfiguriert werden können.

Theoretisch unterstützt SQL Server (64-Bit) 32,767 Datenbanken pro Instanz und 524.272 TB Datenbankgröße, obwohl die typische Installation normalerweise über mehrere Datenbanken verfügt. Die Anzahl der Datenbanken, die SQL Server verarbeiten kann, hängt jedoch von der Last und der Hardware ab. Es ist nicht ungewöhnlich, dass SQL Server Instanzen Dutzende, Hunderte oder sogar Tausende kleine Datenbanken hosten.

Jede Datenbank besteht aus einer oder mehreren Datendateien und einer oder mehreren Transaktions-Log-Dateien. Das Transaktionsprotokoll speichert die Informationen über Datenbanktransaktionen und alle von jeder Sitzung vorgenommenen Datenänderungen. Jedes Mal, wenn die Daten geändert werden, speichert SQL Server genügend Informationen im Transaktionsprotokoll, um die Aktion rückgängig zu machen (Rollback) oder zu wiederholen (Replay). Ein SQL Server-Transaktionsprotokoll ist ein integraler Bestandteil des Rufs von SQL Server für Datenintegrität und Robustheit. Das Transaktionsprotokoll ist für die Atomizität, Konsistenz, Isolation und Strapazierfähigkeit (ACID) von SQL Server von entscheidender Bedeutung. SQL Server schreibt in das Transaktionsprotokoll, sobald eine Änderung an der Datenseite erfolgt. Jede DML-Anweisung (Data Manipulation Language) (z. B. SELECT, Insert, Update oder delete) ist eine vollständige Transaktion, und das Transaktionsprotokoll stellt sicher, dass der gesamte Set-basierte Vorgang durchgeführt wird, um die Atomizität der Transaktion sicherzustellen.

Jede Datenbank verfügt über eine primäre Datendatei, die standardmäßig über die Erweiterung `.mdf` verfügt. Darüber hinaus kann jede Datenbank sekundäre Datenbankdateien enthalten. Diese Dateien haben standardmäßig `.ndf`-Erweiterungen.

Alle Datenbankdateien werden in Dateigruppen gruppiert. Eine Dateigruppe ist die logische Einheit, die die Datenbankverwaltung vereinfacht. Sie ermöglichen die Trennung zwischen einer logischen Objektplatzierung und physischen Datenbankdateien. Wenn Sie die Tabellen für Datenbankobjekte erstellen, geben Sie an, in welcher Dateigruppe sie platziert werden sollen, ohne sich um die zugrunde liegende Datendateikonfiguration zu sorgen.



Die Fähigkeit, mehrere Datendateien innerhalb der Dateigruppe zu speichern, ermöglicht es Ihnen, die Last auf verschiedene Speichergeräte zu verteilen, wodurch die I/O-Performance des Systems verbessert wird. Der Kontrast für die Transaktionsprotokollanmeldung profitiert nicht von den mehreren Dateien, da SQL Server in sequenzieller Weise in das Transaktionsprotokoll schreibt.

Die Trennung zwischen der Platzierung logischer Objekte in den Dateigruppen und physischen Datenbankdateien ermöglicht es Ihnen, das Layout von Datenbankdateien zu optimieren und so das Storage-Subsystem optimal zu nutzen. Die Anzahl der Datendateien, die einen mitgebenden Workload unterstützen, kann nach Bedarf variiert werden, um I/O-Anforderungen und erwartete Kapazität ohne Auswirkungen auf die Applikation zu erfüllen. Diese Variationen im Datenbank-Layout sind für Anwendungsentwickler transparent, die die Datenbankobjekte in Dateigruppen statt in Datenbankdateien platzieren.



**NetApp empfiehlt** die Verwendung der primären Dateigruppe für alles andere als Systemobjekte zu vermeiden. Das Erstellen einer separaten Dateigruppe oder einer Gruppe von Dateigruppen für die Benutzerobjekte vereinfacht die Datenbankverwaltung und Disaster Recovery, insbesondere bei großen Datenbanken.

Sie können die ursprüngliche Dateigröße und die automatischen Wachstumsparameter angeben, wenn Sie die Datenbank erstellen oder neue Dateien zu einer vorhandenen Datenbank hinzufügen. SQL Server verwendet einen proportionalen Füllalgorithmus bei der Auswahl der Datendatei, in die Daten geschrieben werden sollen. Es schreibt eine Datenmenge proportional zum verfügbaren freien Speicherplatz in den Dateien. Je mehr Speicherplatz in der Datei verfügbar ist, desto mehr Schreibvorgänge werden verarbeitet.



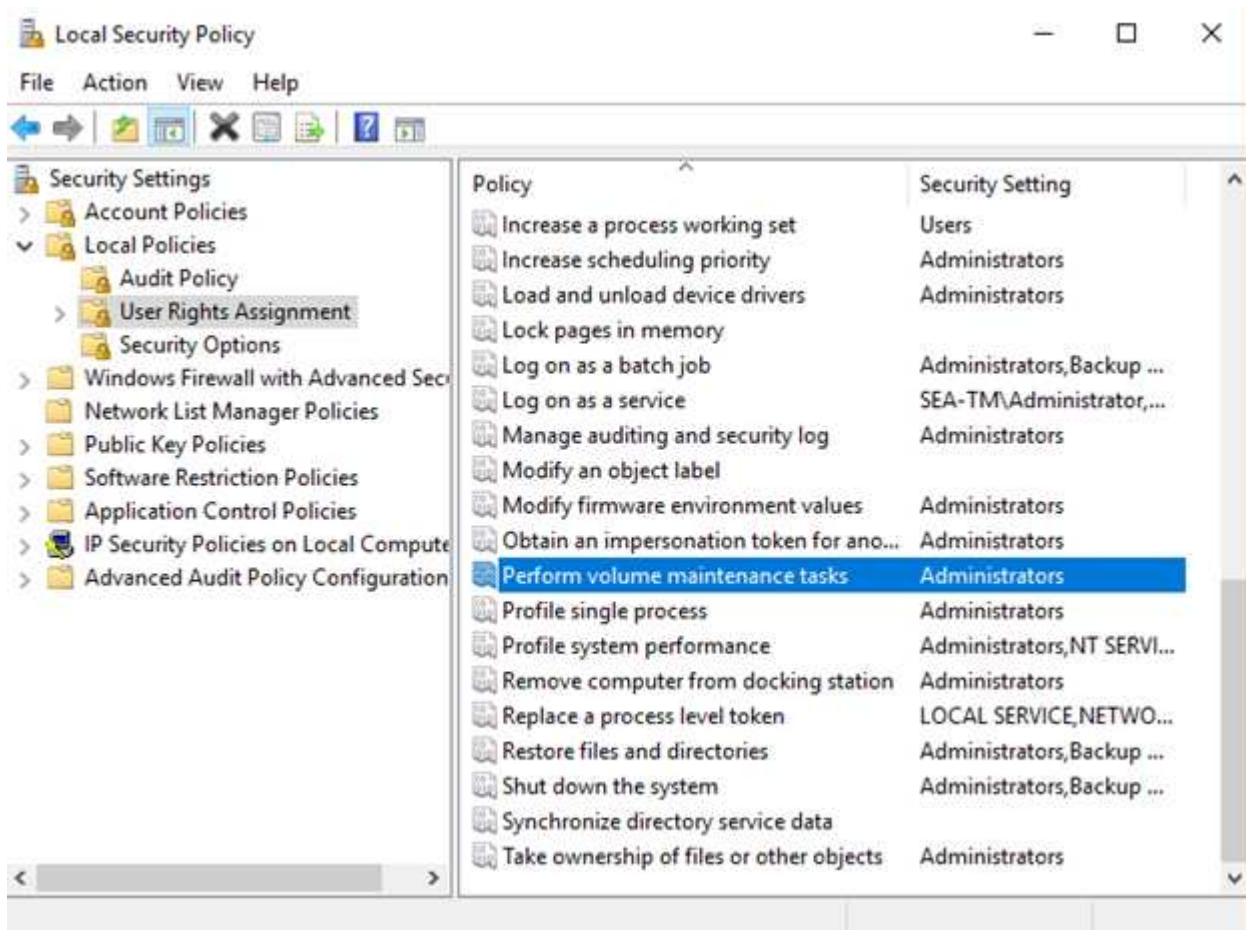
**NetApp empfiehlt**, dass alle Dateien in der einzelnen Dateigruppe die gleiche Anfangsgröße und die gleichen Autogrowth-Parameter haben, wobei die Grow-Größe in Megabyte und nicht in Prozentsätzen definiert ist. Dies hilft dem proportionalen Füllalgorithmus, Schreibaktivitäten gleichmäßig über Datendateien hinweg auszugleichen.

Jedes Mal, wenn SQL Server Dateien vergrößert, füllt es neu zugewiesenen Speicherplatz mit Nullen. Dieser Prozess blockiert alle Sitzungen, die in die entsprechende Datei geschrieben werden müssen, oder generiert im Falle eines Wachstums des Transaktionsprotokolls Transaktionsprotokolle.

SQL Server löscht das Transaktionsprotokoll immer auf Null, und dieses Verhalten kann nicht geändert werden. Sie können jedoch festlegen, ob Datendateien auf Null gesetzt werden, indem Sie die sofortige Dateiinitialisierung aktivieren oder deaktivieren. Durch die sofortige Dateiinitialisierung wird das Wachstum von Datendateien beschleunigt und der Zeitaufwand für die Erstellung oder Wiederherstellung der Datenbank verringert.

Mit der sofortigen Dateinitialisierung ist ein kleines Sicherheitsrisiko verbunden. Wenn diese Option aktiviert ist, können nicht zugewiesene Teile der Datendatei Informationen aus zuvor gelöschten Betriebssystemdateien enthalten. Datenbankadministratoren können solche Daten prüfen.

Sie können die sofortige Dateinitialisierung aktivieren, indem Sie dem SQL Server-Startkonto die Berechtigung SA\_MANAGE\_VOLUME\_NAME, auch bekannt als „Perform Volume Maintenance Task“, hinzufügen. Sie können dies unter der Anwendung zur Verwaltung lokaler Sicherheitsrichtlinien (secpol.msc) tun, wie in der folgenden Abbildung dargestellt. Öffnen Sie die Eigenschaften für die Berechtigung zum Ausführen von Volume-Wartungsaufgaben und fügen Sie das SQL Server-Startkonto zur Liste der Benutzer dort hinzu.



Um zu überprüfen, ob die Berechtigung aktiviert ist, können Sie den Code aus dem folgenden Beispiel verwenden. Dieser Code setzt zwei Trace-Flags, die SQL Server zwingen, zusätzliche Informationen in das Fehlerprotokoll zu schreiben, eine kleine Datenbank zu erstellen und den Inhalt des Protokolls zu lesen.

```

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

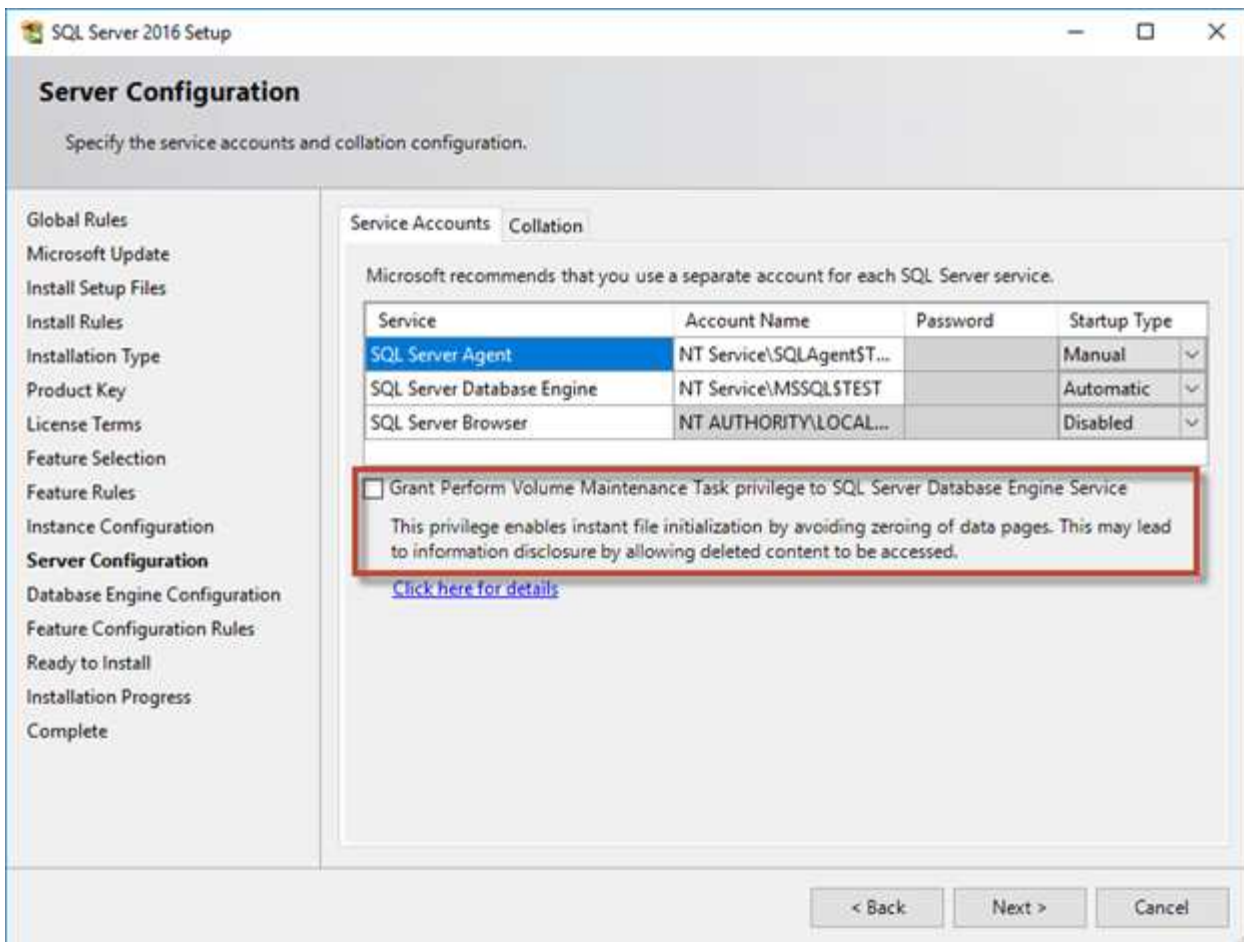
```

Wenn die sofortige Dateiinitalisierung nicht aktiviert ist, zeigt das SQL Server-Fehlerprotokoll an, dass SQL Server die mdf-Datendatei zusätzlich zum Nullsetzen der ldf-Protokolldatei auf Null setzt, wie im folgenden Beispiel gezeigt. Wenn die sofortige Dateiinitalisierung aktiviert ist, wird nur das Nullsetzen der Protokolldatei angezeigt.

|     | 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      | FixupLog Tail(progress) zeroing C:\Program Files\Microsoft SQL Server\MSSQL |
| 372 | 2017-02-09 08:10:08.160 | spid53      | Zeroing C:\Program Files\Microsoft SQL Server\MSSQL                         |
| 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.  |

Die Aufgabe „Volume Maintenance durchführen“ wird in SQL Server 2016 vereinfacht und später während des Installationsprozesses als Option bereitgestellt. In dieser Abbildung wird die Option angezeigt, dem SQL Server-Datenbank-Engine-Service die Berechtigung zum Ausführen der Volume-Wartungsaufgabe zu gewähren.





Eine weitere wichtige Datenbankoption, die die Größe der Datenbankdateien steuert, ist Autoshrink. Wenn diese Option aktiviert ist, verkleinert SQL Server die Datenbankdateien regelmäßig, reduziert deren Größe und gibt Speicherplatz für das Betriebssystem frei. Dieser Vorgang ist ressourcenintensiv und nur selten sinnvoll, da die Datenbankdateien nach einiger Zeit wieder wachsen, wenn neue Daten in das System gelangen. Autoshrink sollte in der Datenbank nicht aktiviert sein.

## Protokollverzeichnis

Das Protokollverzeichnis wird in SQL Server angegeben, um die Backup-Daten des Transaktionsprotokolls auf Hostebene zu speichern. Wenn Sie SnapCenter zum Sichern von Protokolldateien verwenden, muss für jeden von SnapCenter verwendeten SQL Server-Host ein Hostprotokollverzeichnis konfiguriert sein, um Protokollsicherungen durchzuführen. Bei SnapCenter gibt es ein Datenbank-Repository, sodass Metadaten, die mit Backup-, Restore- oder Klonvorgängen verbunden sind, in einem zentralen Datenbank-Repository gespeichert werden.

Die Größe des Host-Log-Verzeichnisses wird wie folgt berechnet:

Größe des Host-Log-Verzeichnisses = ( (maximale DB-LDF-Größe x tägliche Log-Änderungsrate %) x (Snapshot-Aufbewahrung) ÷ (1 - LUN Overhead-Speicherplatz %) )

Die Formel zur Größenbestimmung des Host-Protokollverzeichnisses nimmt einen LUN Overhead von 10 % an

Platzieren Sie das Protokollverzeichnis auf einem dedizierten Volume oder LUN. Die Datenmenge im Host-Log-Verzeichnis hängt von der Größe der Backups und der Anzahl der Tage ab, die Backups aufbewahrt

werden. SnapCenter erlaubt nur ein Host-Protokollverzeichnis pro SQL Server-Host. Sie können die Host-Protokollverzeichnisse unter SnapCenter → Host → Configure Plug-in konfigurieren.

**NetApp empfiehlt** für ein Host-Log-Verzeichnis:

- Stellen Sie sicher, dass das Host-Protokollverzeichnis nicht von anderen Datentypen gemeinsam genutzt wird, die möglicherweise die Backup-Snapshot-Daten beschädigen können.
- Platzieren Sie keine Benutzerdatenbanken oder Systemdatenbanken auf einer LUN, die Bereitstellungspunkte hostet.
- Erstellen Sie das Host-Protokollverzeichnis auf einem dedizierten Volume, auf das SnapCenter Transaktionsprotokolle kopiert.
- Migrieren Sie Datenbanken mithilfe von SnapCenter-Assistenten in NetApp Storage, damit die Datenbanken an gültigen Speicherorten gespeichert werden und so erfolgreiche SnapCenter-Backup- und -Restore-Vorgänge ermöglichen. Beachten Sie, dass der Migrationsprozess für den Fall von Unterbrechungen verantwortlich ist und dazu führen kann, dass die Datenbanken offline gehen, während die Migration durchgeführt wird.
- Die folgenden Bedingungen müssen für Failover-Cluster-Instanzen (FCIs) von SQL Server gelten:
  - Wenn Sie eine Failover-Cluster-Instanz verwenden, muss das Host-Log-Verzeichnis LUN eine Cluster-Festplattenressource in derselben Cluster-Gruppe sein wie die SQL Server-Instanz, die SnapCenter gesichert wird.
  - Wenn Sie eine Failover-Cluster-Instanz verwenden, müssen Benutzerdatenbanken auf gemeinsam genutzte LUNs platziert werden, bei denen es sich um physische Festplatten-Cluster-Ressourcen handelt, die der Cluster-Gruppe zugewiesen sind, die der SQL Server-Instanz zugeordnet ist.



## Tempdb-Dateien

Die Tempdb-Datenbank kann stark genutzt werden. Neben der optimalen Platzierung von Benutzerdatenbankdateien auf ONTAP ist die Platzierung von tempdb-Datendateien auch wichtig, um die Zuweisungskonflikte zu verringern. Tempdb sollte auf einer separaten Festplatte abgelegt und nicht für Benutzerdatendateien freigegeben werden.

Seitenkonflikte können auf den Seiten Global Allocation Map (GAM), Shared Global Allocation Map (SGAM) oder Page Free Space (PFS) auftreten, wenn SQL Server auf spezielle Systemseiten schreiben muss, um neue Objekte zuzuweisen. Verriegelungen sperren diese Seiten im Speicher. In einer stark ausgelasteten SQL Server-Instanz kann es lange dauern, bis ein Latch auf einer Systemseite in tempdb abgerufen wird. Dies führt zu längeren Abfragezeiten und wird als Latch Contention bezeichnet. Lesen Sie die folgenden Best Practices für das Erstellen von tempdb-Datendateien:

- Für < oder = bis 8 Kerne: Tempdb-Datendateien = Anzahl der Kerne
- Für > 8 Kerne: 8 tempdb-Datendateien
- Die tempdb-Datendatei sollte mit gleicher Größe erstellt werden

Mit dem folgenden Beispielskript wird tempdb geändert, indem acht tempdb-Dateien gleicher Größe erstellt und tempdb auf den Mount-Punkt für SQL Server 2012 und höher verschoben C:\MSSQL\tempdb wird.

```

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
```

Ab SQL Server 2016 wird die Anzahl der für das Betriebssystem sichtbaren CPU-Kerne während der Installation automatisch erkannt. Auf Basis dieser Anzahl berechnet und konfiguriert SQL Server die Anzahl der für eine optimale Performance erforderlichen tempdb-Dateien.

## Storage-Effizienz

Die ONTAP Storage-Effizienz wurde für das Speichern und Managen von SQL Server-Daten optimiert, sodass Sie den Speicherplatz auf ein Minimum beschränken und keine Auswirkungen auf die Performance haben.

Funktionen für Platzeffizienz wie Komprimierung, Data-Compaction und Deduplizierung sind darauf ausgelegt, die Menge der logischen Daten zu einer bestimmten Menge des physischen Storage zu erhöhen. Das Ergebnis sind niedrigere Kosten und geringerer Management-Overhead.

Auf hohem Niveau ist Komprimierung ein mathematischer Prozess, bei dem Muster in Daten erkannt und so kodiert werden, dass der Platzbedarf reduziert wird. Dagegen erkennt die Deduplizierung tatsächlich wiederholte Datenblöcke und entfernt die fremden Kopien. Durch Data-Compaction können mehrere logische Datenblöcke denselben physischen Block auf den Medien gemeinsam nutzen.



In den nachfolgenden Abschnitten zu Thin Provisioning finden Sie eine Erläuterung des Wechselspiels zwischen Storage-Effizienz und fraktionaler Reservierung.

## Komprimierung

Vor der Verfügbarkeit von All-Flash-Storage-Systemen war die Array-basierte Komprimierung nur eingeschränkt verfügbar, da die meisten I/O-intensiven Workloads eine sehr große Anzahl von Spindeln erforderten, um eine akzeptable Performance zu erreichen. Als Nebeneffekt der großen Anzahl von Laufwerken enthielten Storage-Systeme grundsätzlich viel mehr Kapazität als erforderlich. Mit dem Trend hin zu Solid-State-Storage hat sich die Situation verändert. Eine enorme Überprovisionierung von Laufwerken entfällt, nur weil eine gute Performance erzielt werden kann. Der Speicherplatz in einem Storage-System kann den tatsächlichen Kapazitätsanforderungen angepasst werden.

Die gesteigerte IOPS-Fähigkeit von Solid-State-Laufwerken (SSDs) bringt im Vergleich zu rotierenden Laufwerken fast immer Kosteneinsparungen mit sich. Allerdings kann die Komprimierung durch eine höhere effektive Kapazität von Solid-State-Medien weitere Einsparungen erzielen.

Es gibt verschiedene Möglichkeiten, Daten zu komprimieren. Viele Datenbanken verfügen über eigene Komprimierungsfunktionen, dies wird jedoch in Kundenumgebungen selten beobachtet. Der Grund dafür ist in der Regel die Performance-Einbußen bei einem **Wechsel** zu komprimierten Daten. Bei einigen Anwendungen fallen zudem hohe Lizenzierungskosten für die Komprimierung auf Datenbankebene an. Und schließlich gibt es noch die allgemeinen Performance-Auswirkungen auf die Datenbankvorgänge. Es macht wenig Sinn, für eine CPU, die Datenkomprimierung und -Dekomprimierung durchführt, hohe Lizenzkosten pro CPU zu zahlen, anstatt eine echte Datenbankarbeit zu erledigen. Eine bessere Option ist, die Komprimierungsarbeiten auf das

Storage-System zu verlagern.

## Anpassungsfähige Komprimierung

Die adaptive Komprimierung wurde vollständig mit Enterprise-Workloads getestet, ohne dabei die Performance zu beeinträchtigen – selbst in einer All-Flash-Umgebung, in der die Latenz im Mikrosekunden-Bereich gemessen wird. Einige Kunden haben bei Verwendung der Komprimierung sogar eine Performance-Steigerung festgestellt, da die Daten im Cache komprimiert bleiben. Dadurch konnte die Menge des verfügbaren Cache in einem Controller erhöht werden.

ONTAP managt physische Blöcke in 4-KB-Einheiten. Die anpassungsfähige Komprimierung verwendet eine Standardkomprimierung von 8 KB. Dies bedeutet, dass Daten in 8-KB-Einheiten komprimiert werden. Dies entspricht der 8-KB-Blockgröße, die von relationalen Datenbanken am häufigsten verwendet wird. Kompressionsalgorithmen werden effizienter, da mehr Daten als eine Einheit komprimiert werden. Eine Komprimierungs-Blockgröße von 32 KB wäre speichereffizienter als eine Komprimierungsblockeinheit mit 8 KB. Das bedeutet, dass die adaptive Komprimierung bei Verwendung der standardmäßigen 8-KB-Blockgröße zu etwas niedrigeren Effizienzzraten führt, jedoch bietet die Verwendung kleinerer Blockgrößen zur Komprimierung auch einen signifikanten Vorteil. Datenbank-Workloads umfassen einen großen Anteil an Überschreibungsaktivitäten. Beim Überschreiben eines komprimierten 32-KB-Datenblocks müssen die gesamten 32-KB-Daten zurückgelesen, dekomprimiert, der erforderliche 8-KB-Bereich aktualisiert, neu komprimiert und dann die gesamten 32-KB-Daten wieder auf die Laufwerke geschrieben werden. Dies ist für ein Storage-System ein sehr teurer Vorgang und der Grund dafür, dass bei einigen Storage Arrays anderer Anbieter, die auf größeren Komprimierungsblockgrößen basieren, auch die Performance bei Datenbank-Workloads erheblich beeinträchtigt wird.



Die von der anpassungsfähigen Komprimierung verwendete Blockgröße kann auf bis zu 32 KB gesteigert werden. Dies kann die Speichereffizienz verbessern und sollte bei stillgelegten Dateien wie Transaktionsprotokollen und Backup-Dateien in Betracht gezogen werden, wenn eine große Menge solcher Daten auf dem Array gespeichert wird. In manchen Situationen profitieren aktive Datenbanken mit 16-KB- oder 32-KB-Blockgröße möglicherweise auch von der Erhöhung der Blockgröße der anpassungsfähigen Komprimierung. Wenden Sie sich an einen Mitarbeiter von NetApp oder einen unserer Partner, um Rat zu erhalten, ob diese Lösung für Ihren Workload geeignet ist.



Blockgrößen der Komprimierung von mehr als 8 KB sollten nicht zusammen mit der Deduplizierung an Streaming-Backup-Zielen verwendet werden. Der Grund dafür ist, dass kleine Änderungen an den gesicherten Daten das 32-KB-Komprimierungsfenster beeinflussen. Wenn sich das Fenster verschiebt, unterscheiden sich die resultierenden komprimierten Daten in der gesamten Datei. Die Deduplizierung erfolgt nach der Komprimierung. Das heißt, die Deduplizierungs-Engine sieht jedes komprimierte Backup unterschiedlich. Wenn eine Deduplizierung von Streaming-Backups erforderlich ist, sollte nur eine blockadaptive Komprimierung von 8 KB verwendet werden. Die adaptive Komprimierung ist vorzuziehen, da sie bei kleineren Blöcken arbeitet und die Deduplizierungseffizienz nicht stört. Aus ähnlichen Gründen wirkt sich die Host-seitige Komprimierung auch in die Effizienz der Deduplizierung aus.

## Kompressionsausrichtung

Die anpassungsfähige Komprimierung in einer Datenbankumgebung erfordert bestimmte Überlegungen zur Blockausrichtung der Komprimierung. Dies ist nur für Daten relevant, die Random Überschreibungen sehr spezifischer Blöcke unterliegen. Dieser Ansatz ähnelt im Konzept der gesamten Filesystem-Ausrichtung, wobei der Beginn eines Dateisystems an einer Grenze von 4K-Geräten ausgerichtet werden muss und die Blockgröße eines Dateisystems ein Vielfaches von 4K sein muss.

Ein Schreibvorgang von 8 KB in eine Datei wird beispielsweise nur komprimiert, wenn er an einer 8-KB-Grenze

innerhalb des Dateisystems selbst ausgerichtet ist. Dieser Punkt bedeutet, dass er auf die ersten 8 KB der Datei, die zweiten 8 KB der Datei usw. fallen muss. Der einfachste Weg, um eine korrekte Ausrichtung zu gewährleisten, ist die Verwendung des korrekten LUN-Typs. Jede erstellte Partition sollte einen Offset vom Anfang des Geräts an haben, der ein Vielfaches von 8K ist, und eine Dateisystem-Blockgröße verwenden, die ein Vielfaches der Datenbank-Blockgröße ist.

Daten wie Backups oder Transaktions-Logs werden sequenziell geschrieben und umfassen mehrere Blöcke. Alle Blöcke werden komprimiert. Daher besteht keine Notwendigkeit, eine Ausrichtung zu erwägen. Das einzige I/O-Muster, das Bedenken hinsichtlich des zufälligen Überschreibens von Dateien hat, ist das zufällige Überschreiben von Dateien.

## **Data-Compaction**

Data-Compaction ist eine Technologie, die die Komprimierungseffizienz verbessert. Wie bereits erwähnt, erzielt die anpassungsfähige Komprimierung allein schon Einsparungen von 2:1, da sie auf das Speichern eines 8-KB-I/O-Blocks in einem 4-KB-WAFL-Block beschränkt ist. Komprimierungsmethoden mit größeren Blockgrößen verbessern die Effizienz. Sie sind jedoch nicht für Daten geeignet, die mit Überschreibungen kleiner Blöcke verbunden sind. Die Dekomprimierung von 32-KB-Dateneinheiten durch die Aktualisierung eines 8-KB-Abschnitts, die Datenkomprimierung und das Zurückschreiben auf die Laufwerke verursacht Overhead.

Data-Compaction sorgt dafür, dass mehrere logische Blöcke innerhalb physischer Blöcke gespeichert werden können. Beispielsweise kann eine Datenbank mit stark komprimierbaren Daten wie Text oder teilweise vollständigen Blöcken von 8 KB bis 1 KB komprimieren. Ohne Data-Compaction belegen diese 1 KB Daten immer noch einen gesamten 4-KB-Block. Durch die Inline-Data-Compaction können 1 KB komprimierte Daten zusammen mit anderen komprimierten Daten auf nur 1 KB physischen Speicherplatz gespeichert werden. Es handelt sich nicht um eine Komprimierungstechnologie. Es ist einfach eine effizientere Möglichkeit, Speicherplatz auf den Laufwerken zuzuweisen und sollte daher keine erkennbaren Performance-Auswirkungen verursachen.

Der Grad der erzielten Einsparungen variiert. Bereits komprimierte oder verschlüsselte Daten können in der Regel nicht weiter komprimiert werden. Daher profitieren diese Datensätze von der Data-Compaction nicht. Im Gegensatz dazu werden neu initialisierte Datendateien, die nur wenig mehr als Block-Metadaten und Nullen enthalten, mit bis zu 80 komprimiert.

## **Temperaturempfindliche Speichereffizienz**

Temperaturempfindliche Speichereffizienz (TSSE) ist ab ONTAP 9.8 verfügbar. Es basiert auf Block-Zugriffs-Heatmaps, um selten genutzte Blöcke zu identifizieren und sie effizienter zu komprimieren.

## **Deduplizierung**

Deduplizierung ist die Entfernung von Blockduplikaten aus einem Datensatz. Wenn beispielsweise derselbe 4-KB-Block in 10 verschiedenen Dateien vorhanden war, leitet die Deduplizierung diesen 4-KB-Block innerhalb aller 10 Dateien auf denselben physischen 4-KB-Block um. Im Ergebnis würde sich die Effizienz dieser Daten um 10:1 verbessern.

Daten wie Boot-LUNs von VMware lassen sich in der Regel sehr gut deduplizieren, da sie aus mehreren Kopien derselben Betriebssystemdateien bestehen. Es wurde eine Effizienz von 100:1 und höher festgestellt.

Einige Daten enthalten keine Datenduplikate. Ein Oracle-Block enthält beispielsweise einen Header, der global nur für die Datenbank gilt, und einen Trailer, der fast einzigartig ist. Aus diesem Grund führt die Deduplizierung einer Oracle Database selten zu Einsparungen von mehr als 1 %. Die Deduplizierung mit MS SQL Datenbanken ist etwas besser, aber eindeutige Metadaten auf Blockebene stellen immer noch eine

Einschränkung dar.

In einigen Fällen wurde eine Speicherersparnis von bis zu 15 % bei Datenbanken mit 16 KB und großen Blockgrößen beobachtet. Die ersten 4-KB-Blöcke enthalten die global eindeutige Kopfzeile, und der letzte 4-KB-Block enthält den nahezu einzigartigen Trailer. Die internen Blöcke eignen sich für eine Deduplizierung, obwohl dies in der Praxis fast vollständig der Deduplizierung von gelöschten Daten zugeordnet ist.

Viele Arrays anderer Anbieter behaupten, Datenbanken unter der Annahme zu deduplizieren, dass eine Datenbank mehrfach kopiert wird. In dieser Hinsicht kann auch NetApp Deduplizierung eingesetzt werden, allerdings bietet ONTAP die bessere Option: NetApp FlexClone Technologie. Das Endergebnis ist das gleiche. Es werden mehrere Kopien einer Datenbank erstellt, die die meisten zugrunde liegenden physischen Blöcke nutzen. Ein Einsatz von FlexClone ist wesentlich effizienter, als Datenbankdateien zu kopieren und anschließend zu deduplizieren. Der Effekt ist die Nichtdeduplizierung und nicht die Deduplizierung, da ein Duplikat von vornirgends erstellt wird.

## Effizienz und Thin Provisioning

Effizienzfunktionen sind Formen von Thin Provisioning. Beispielsweise kann eine 100-GB-LUN, die ein 100-GB-Volume belegt, bis zu 50 GB komprimiert werden. Es wurden noch keine tatsächlichen Einsparungen realisiert, da das Volume noch 100 GB beträgt. Das Volume muss zunächst verkleinert werden, damit der eingesparte Speicherplatz an anderer Stelle im System genutzt werden kann. Wenn spätere Änderungen an der 100GB-LUN dazu führen, dass die Daten weniger komprimierbar werden, dann vergrößert sich die LUN und das Volume könnte sich füllen.

Thin Provisioning wird nachdrücklich empfohlen, da es das Management vereinfachen und gleichzeitig eine deutliche Verbesserung der nutzbaren Kapazität mit den damit verbundenen Kosteneinsparungen ermöglichen kann. Der Grund hierfür ist einfach: Datenbankumgebungen enthalten oft viel leeren Speicherplatz, eine große Anzahl an Volumes und LUNs sowie komprimierbare Daten. Durch Thick Provisioning wird Speicherplatz auf Storage für Volumes und LUNs reserviert, für den Fall, dass sie eines Tages zu 100 % voll werden und 100 % nicht komprimierbare Daten enthalten. Das wird wohl nie passieren. Dank Thin Provisioning kann dieser Speicherplatz zurückgewonnen und an anderer Stelle verwendet werden. Das Kapazitätsmanagement kann auf dem Storage-System selbst basieren, anstatt auf vielen kleineren Volumes und LUNs.

Einige Kunden bevorzugen Thick Provisioning entweder für bestimmte Workloads oder generell basierend auf bestehenden Betriebs- und Beschaffungsmethoden.



Bei einem Volume mit Thick Provisioning müssen unbedingt alle Effizienzfunktionen des Volumes deaktiviert werden, einschließlich der Dekomprimierung und der Entfernung der Deduplizierung mit dem `sis undo` Befehl. Die Lautstärke sollte nicht in der Ausgabe angezeigt `volume efficiency show` werden. Ist dies der Fall, ist das Volume für Effizienzfunktionen noch teilweise konfiguriert. Daher funktionieren Überschreibungsgarantien anders. Dies erhöht die Wahrscheinlichkeit, dass Konfigurationsübersehungen dazu führen, dass das Volume unerwartet aus dem Speicherplatz kommt und zu Datenbank-I/O-Fehlern führt.

## Best Practices für Effizienz

NetApp empfiehlt Folgendes:

### AFF-Standards

Volumes, die auf ONTAP erstellt wurden und auf einem rein Flash-basierten AFF System ausgeführt werden, werden über Thin Provisioning mit allen Inline-Effizienzfunktionen bereitgestellt. Obwohl Datenbanken im Allgemeinen nicht von der Deduplizierung profitieren und nicht komprimierbare Daten enthalten können, sind die Standardeinstellungen dennoch für fast alle Workloads geeignet. ONTAP wurde mit dem Ziel entwickelt,



alle Arten von Daten und I/O-Muster effizient zu verarbeiten. Dabei spielt es keine Rolle, ob es zu Einsparungen kommt oder nicht. Standardwerte sollten nur dann geändert werden, wenn die Gründe vollständig verstanden sind und es einen Vorteil gibt, dass sie abweichen.

## Allgemeine Empfehlungen

- Wenn Volumes und/oder LUNs nicht über Thin Provisioning bereitgestellt werden, müssen Sie alle Effizienzeinstellungen deaktivieren, da die Verwendung dieser Funktionen keine Einsparungen bietet. Die Kombination von Thick Provisioning mit aktivierter Speicherplatzeffizienz kann zu unerwartetem Verhalten führen, einschließlich Fehlern aufgrund von fehlerhafterem Speicherplatz.
- Wenn Daten nicht überschrieben werden, wie etwa bei Backups oder Datenbanktransaktionsprotokollen, können Sie die Effizienz steigern, indem Sie TSSE mit einem niedrigen Kühlzeitraum aktivieren.
- Einige Dateien enthalten möglicherweise eine beträchtliche Menge an nicht komprimierbaren Daten. Ein Beispiel: Wenn die Komprimierung bereits auf Applikationsebene aktiviert ist, werden Dateien verschlüsselt. Wenn eines dieser Szenarien zutrifft, sollten Sie die Komprimierung deaktivieren, um einen effizienteren Betrieb auf anderen Volumes mit komprimierbaren Daten zu ermöglichen.
- Verwenden Sie für Datenbank-Backups nicht sowohl die 32-KB-Komprimierung als auch die Deduplizierung. Siehe Abschnitt [Anpassungsfähige Komprimierung](#) Entsprechende Details.

## Datenbankkomprimierung

SQL Server selbst verfügt auch über Funktionen zur Komprimierung und zum effizienten Management von Daten. SQL Server unterstützt derzeit zwei Arten der Datenkomprimierung: Row Compression und Page Compression.

Durch die Zeilenkomprimierung wird das Datenspeicherformat geändert. So werden beispielsweise ganze Zahlen und Dezimalzahlen anstelle des nativen Formats mit fester Länge in das Format mit variabler Länge geändert. Außerdem werden Zeichenketten mit fester Länge durch das Entfernen von Leerzeichen in das Format mit variabler Länge geändert. Die Seitenkomprimierung implementiert die Zeilenkomprimierung und zwei weitere Komprimierungsstrategien (Prefix-Komprimierung und Wörterbuchkomprimierung). Weitere Details zur Seitenkomprimierung finden Sie unter "[Implementierung Der Seitenkomprimierung](#)".

Die Datenkomprimierung wird derzeit in den Enterprise-, Developer- und Evaluation-Editionen von SQL Server 2008 und höher unterstützt. Obwohl die Komprimierung von der Datenbank selbst durchgeführt werden kann, ist dies in einer SQL Server Umgebung nur selten der Fall.

Hier sind die Empfehlungen für die Verwaltung von Speicherplatz für SQL Server-Datendateien

- Verwenden Sie Thin Provisioning in SQL Server-Umgebungen, um die Speicherplatzauslastung zu verbessern und bei Einsatz der Speicherplatzgarantiefunktion den gesamten Storage-Bedarf zu senken.
  - Verwenden Sie Autogrow für die meisten gängigen Implementierungskonfigurationen, da der Storage-Administrator nur die Speicherplatznutzung im Aggregat überwachen muss.
- Aktivieren Sie die Deduplizierung auf Volumes mit SQL Server-Datendateien nicht, es sei denn, das Volume ist bekannt, dass es mehrere Kopien derselben Daten enthält, wie beispielsweise die Wiederherstellung von Datenbanken aus Backups auf einem einzelnen Volume.

## Speicherplatzrückgewinnung

Die Rückgewinnung von ungenutztem Speicherplatz in einer LUN kann regelmäßig gestartet werden. Bei SnapCenter können Sie den folgenden PowerShell Befehl verwenden, um die Rückgewinnung von ungenutztem Speicherplatz zu starten.



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

Wenn Sie die Speicherplatzrückgewinnung durchführen müssen, sollte dieser Prozess in Zeiten geringer Aktivität ausgeführt werden, da er anfangs Hostzyklen beansprucht.

## Copyright-Informationen

Copyright © 2025 NetApp. Alle Rechte vorbehalten. Gedruckt in den USA. Dieses urheberrechtlich geschützte Dokument darf ohne die vorherige schriftliche Genehmigung des Urheberrechtsinhabers in keiner Form und durch keine Mittel – weder grafische noch elektronische oder mechanische, einschließlich Fotokopieren, Aufnehmen oder Speichern in einem elektronischen Abrufsystem – auch nicht in Teilen, vervielfältigt werden.

Software, die von urheberrechtlich geschütztem NetApp Material abgeleitet wird, unterliegt der folgenden Lizenz und dem folgenden Haftungsausschluss:

DIE VORLIEGENDE SOFTWARE WIRD IN DER VORLIEGENDEN FORM VON NETAPP ZUR VERFÜGUNG GESTELLT, D. H. OHNE JEGLICHE EXPLIZITE ODER IMPLIZITE GEWÄHRLEISTUNG, EINSCHLIESSLICH, JEDOCH NICHT BESCHRÄNKT AUF DIE STILLSCHWEIGENDE GEWÄHRLEISTUNG DER MARKTGÄNGIGKEIT UND EIGNUNG FÜR EINEN BESTIMMTEN ZWECK, DIE HIERMIT AUSGESCHLOSSEN WERDEN. NETAPP ÜBERNIMMT KEINERLEI HAFTUNG FÜR DIREKTE, INDIREKTE, ZUFÄLLIGE, BESONDERE, BEISPIELHAFTE SCHÄDEN ODER FOLGESCHÄDEN (EINSCHLIESSLICH, JEDOCH NICHT BESCHRÄNKT AUF DIE BESCHAFFUNG VON ERSATZWAREN ODER -DIENSTLEISTUNGEN, NUTZUNGS-, DATEN- ODER GEWINNVERLUSTE ODER UNTERBRECHUNG DES GESCHÄFTSBETRIEBS), UNABHÄNGIG DAVON, WIE SIE VERURSACHT WURDEN UND AUF WELCHER HAFTUNGSTHEORIE SIE BERUHEN, OB AUS VERTRAGLICH FESTGELEGTER HAFTUNG, VERSCHULDENSUNABHÄNGIGER HAFTUNG ODER DELIKTSHAFTUNG (EINSCHLIESSLICH FAHRLÄSSIGKEIT ODER AUF ANDEREM WEGE), DIE IN IRGEND EINER WEISE AUS DER NUTZUNG DIESER SOFTWARE RESULTIEREN, SELBST WENN AUF DIE MÖGLICHKEIT DERARTIGER SCHÄDEN HINGEWIESEN WURDE.

NetApp behält sich das Recht vor, die hierin beschriebenen Produkte jederzeit und ohne Vorankündigung zu ändern. NetApp übernimmt keine Verantwortung oder Haftung, die sich aus der Verwendung der hier beschriebenen Produkte ergibt, es sei denn, NetApp hat dem ausdrücklich in schriftlicher Form zugestimmt. Die Verwendung oder der Erwerb dieses Produkts stellt keine Lizenzierung im Rahmen eines Patentrechts, Markenrechts oder eines anderen Rechts an geistigem Eigentum von NetApp dar.

Das in diesem Dokument beschriebene Produkt kann durch ein oder mehrere US-amerikanische Patente, ausländische Patente oder anhängige Patentanmeldungen geschützt sein.

ERLÄUTERUNG ZU „RESTRICTED RIGHTS“: Nutzung, Vervielfältigung oder Offenlegung durch die US-Regierung unterliegt den Einschränkungen gemäß Unterabschnitt (b)(3) der Klausel „Rights in Technical Data – Noncommercial Items“ in DFARS 252.227-7013 (Februar 2014) und FAR 52.227-19 (Dezember 2007).

Die hierin enthaltenen Daten beziehen sich auf ein kommerzielles Produkt und/oder einen kommerziellen Service (wie in FAR 2.101 definiert) und sind Eigentum von NetApp, Inc. Alle technischen Daten und die Computersoftware von NetApp, die unter diesem Vertrag bereitgestellt werden, sind gewerblicher Natur und wurden ausschließlich unter Verwendung privater Mittel entwickelt. Die US-Regierung besitzt eine nicht ausschließliche, nicht übertragbare, nicht unterlizenzierbare, weltweite, limitierte unwiderrufliche Lizenz zur Nutzung der Daten nur in Verbindung mit und zur Unterstützung des Vertrags der US-Regierung, unter dem die Daten bereitgestellt wurden. Sofern in den vorliegenden Bedingungen nicht anders angegeben, dürfen die Daten ohne vorherige schriftliche Genehmigung von NetApp, Inc. nicht verwendet, offengelegt, vervielfältigt, geändert, aufgeführt oder angezeigt werden. Die Lizenzrechte der US-Regierung für das US-Verteidigungsministerium sind auf die in DFARS-Klausel 252.227-7015(b) (Februar 2014) genannten Rechte beschränkt.

## Markeninformationen

NETAPP, das NETAPP Logo und die unter <http://www.netapp.com/TM> aufgeführten Marken sind Marken von NetApp, Inc. Andere Firmen und Produktnamen können Marken der jeweiligen Eigentümer sein.