스토리지 구성

Enterprise applications

NetApp
May 09, 2024

목차

스토리지 구성

Microsoft SQL Server 스토리지 고려 사항

ONTAP 스토리지 솔루션과 Microsoft SQL Server를 결합하여 오늘날의 가장 까다로운 애플리케이션 요구사항을 충족할 수 있는 엔터프라이즈급 데이터베이스 스토리지 설계를 작성할 수 있습니다.

두 기술을 모두 최적화하려면 SQL Server I/O 패턴과 특성을 이해해야 합니다. SQL Server 데이터베이스를 위해 잘 설계된 스토리지 레이아웃은 SQL Server의 성능과 SQL Server 인프라의 관리를 지원합니다. 또한 우수한 스토리지 레이아웃을 통해 초기 구축을 성공적으로 수행할 수 있으며 비즈니스 성장에 따라 환경이 원활하게 확장될 수 있습니다.

데이터 스토리지 설계

SnapCenter를 사용하여 백업을 수행하지 않는 SQL Server 데이터베이스의 경우 데이터와 로그 파일을 별도의 드라이브에 배치하는 것이 좋습니다. 데이터를 동시에 업데이트하고 요청하는 응용 프로그램의 경우 로그 파일은 쓰기 작업이 많고 데이터 파일(응용 프로그램에 따라 다름)은 읽기/쓰기 작업이 많이 사용됩니다. 데이터 검색을 위해 로그 파일이 필요하지 않습니다. 따라서 자체 드라이브에 있는 데이터 파일에서 데이터 요청을 처리할 수 있습니다.

새 데이터베이스를 만들 때는 데이터와 로그에 대해 별도의 드라이브를 지정하는 것이 좋습니다. 데이터베이스를 만든 후 파일을 이동하려면 데이터베이스를 오프라인으로 전환해야 합니다. Microsoft 권장 사항에 대한 자세한 내용은 을 참조하십시오 "데이터 및 로그 파일을 별도의 드라이브에 저장합니다".

애그리게이트

애그리게이트는 NetApp 스토리지 구성에서 사용할 수 있는 최저 수준의 스토리지 컨테이너입니다. IO를 다른 기본 드라이브 세트로 분리할 것을 권장하는 일부 레거시 문서가 인터넷에 있습니다. ONTAP에서는 이 기능을 사용하지 않는 것이 좋습니다. NetApp은 데이터 파일 및 트랜잭션 로그 파일이 분리된 공유 및 전용 애그리게이트를 사용하여 다양한 I/O 워크로드 특성 테스트를 수행했습니다. 테스트 결과, 더 많은 RAID 그룹 및 드라이브를 포함하는 하나의 대형 Aggregate는 스토리지 성능을 최적화 및 개선했으며 다음과 같은 두 가지 이유로 관리자가 보다 쉽게 관리할 수 있는 것으로 나타났습니다.

- 하나의 대형 Aggregate를 통해 모든 드라이브의 I/O 기능을 모든 파일에서 사용할 수 있습니다.
- 하나의 대형 Aggregate는 디스크 공간을 가장 효율적으로 사용합니다.

고가용성(HA)을 위해 SQL Server Always On Availability Group 보조 동기식 복제본을 애그리게이트의 별도의 SVM(스토리지 가상 머신)에 배치합니다. 재해 복구를 위해 NetApp SnapMirror 기술을 사용하여 복제된 콘텐츠와 함께 DR 사이트에서 별도의 스토리지 클러스터의 일부인 애그리게이트에 비동기식 복제를 배치하십시오. NetApp은 최적의 스토리지 성능을 위해 애그리게이트에서 최소 10% 이상의 여유 공간을 사용할 것을 권장합니다.

볼륨

NetApp FlexVol 볼륨이 생성되어 애그리게이트 내에 상주합니다. 이 용어는 ONTAP 볼륨이 LUN이 아니기 때문에 혼동을 일으킬 수 있습니다. ONTAP 볼륨은 데이터를 위한 관리 컨테이너입니다. 볼륨에는 파일, LUN 또는 S3 오브젝트가 포함될 수 있습니다. 볼륨은 공간을 차지하지 않으며 포함된 데이터를 관리하는 데만 사용됩니다.

볼륨 설계 고려 사항

데이터베이스 볼륨 설계를 생성하기 전에 SQL Server 입출력 패턴과 특성이 워크로드와 백업 및 복구 요구 사항에 따라

어떻게 다른지 이해해야 합니다. 확장 가능한 볼륨에 대한 다음 NetApp 권장 사항을 참조하십시오.

- 호스트 간에 볼륨을 공유하지 마십시오. 예를 들어, 단일 볼륨에 2개의 LUN을 생성하고 각 LUN을 다른 호스트와 공유할 수 있지만 관리가 복잡해질 수 있으므로 이러한 작업을 피해야 합니다.
- Windows의 26개 드라이브 문자 제한을 능가하려면 드라이브 문자 대신 NTFS 마운트 지점을 사용하십시오. 볼륨 마운트 지점을 사용할 때는 볼륨 레이블에 마운트 지점과 동일한 이름을 지정하는 것이 좋습니다.
- 필요한 경우 볼륨 자동 크기 조정 정책을 구성하여 공간 부족 상태를 방지하십시오. ONTAP c 2022 NetApp, Inc.의 Microsoft SQL Server에 대한 17 모범 사례 가이드 저작권 본사 소유.
- SMB 공유에 SQL Server를 설치하는 경우 폴더를 생성할 수 있도록 SMB/CIFS 볼륨에 유니코드가 설정되어 있는지 확인합니다.
- 운영 관점에서 쉽게 모니터링할 수 있도록 볼륨의 스냅숏 예비 공간 값을 0으로 설정합니다.
- 스냅샷 스케줄 및 보존 정책을 해제합니다. 대신 SnapCenter를 사용하여 SQL Server 데이터 볼륨의 스냅샷 복사본을 조정합니다.
- SQL Server 시스템 데이터베이스를 전용 볼륨에 배치합니다.
- tempdb는 SQL Server가 임시 작업 공간으로 사용하는 시스템 데이터베이스로, 특히 I/O를 많이 사용하는 DBCC CHECKDB 작업에 사용됩니다. 따라서 이 데이터베이스를 별도의 스핀들 세트가 있는 전용 볼륨에 배치하십시오. 볼륨 수가 문제가 되는 대규모 환경에서는 신중하게 계획을 수립한 후 tempdb를 더 적은 볼륨으로 통합하고 동일한 볼륨에 저장할 수 있습니다. SQL Server를 다시 시작할 때마다 이 데이터베이스가 다시 생성되므로 tempdb에 대한 데이터 보호는 높은 우선 순위가 아닙니다.
- 사용자 데이터 파일(.mdf)은 랜덤 읽기/쓰기 워크로드이므로 별도의 볼륨에 배치하십시오. 일반적으로 트랜잭션 로그 백업은 데이터베이스 백업보다 더 자주 생성됩니다. 이러한 이유로 트랜잭션 로그 파일(.ldf)을 데이터 파일과 별도의 볼륨 또는 VMDK에 배치하여 각각에 대해 독립적인 백업 일정을 생성할 수 있도록 합니다. 또한 이 분리 방식은 로그 파일의 순차적 쓰기 I/O를 데이터 파일의 랜덤 읽기/쓰기 I/O에서 격리하고 SQL Server 성능을 크게 향상시킵니다.

LUN을 클릭합니다

- 사용자 데이터베이스 파일과 로그 백업을 저장할 로그 디렉토리가 별도의 볼륨에 있어야 보존 정책이 SnapVault 기술과 함께 사용될 때 스냅샷을 덮어쓰지 않도록 할 수 있습니다.
- SQL Server 데이터베이스가 전체 텍스트 검색 관련 파일과 같이 데이터베이스 파일이 아닌 LUN과 분리된 LUN에 상주해야 합니다.
- 데이터베이스 보조 파일(파일 그룹의 일부로)을 별도의 볼륨에 배치하면 SQL Server 데이터베이스의 성능이 향상됩니다. 이 분리는 데이터베이스의 .mdf 파일이 LUN을 다른 .mdf 파일과 공유하지 않는 경우에만 유효합니다.
- DiskManager 또는 다른 툴을 사용하여 LUN을 생성하는 경우 LUN을 포맷할 때 파티션의 할당 단위 크기가 64K로 설정되어 있는지 확인하십시오.
- 를 참조하십시오 "최신 SAN에 대한 ONTAP 모범 사례 하의 Microsoft Windows 및 네이티브 MPIO" Windows에서 MPIO 속성의 iSCSI 장치에 다중 경로 지원을 적용하려면 다음을 수행합니다.

Microsoft SQL Server 데이터베이스 파일 및 파일 그룹

초기 구축 단계에서는 ONTAP에 SQL Server 데이터베이스 파일을 적절하게 배치하는 것이 중요합니다. 따라서 비즈니스 요구 사항에 맞게 구성할 수 있는 최적의 성능, 공간 관리, 백업 및 복원 시간이 보장됩니다.

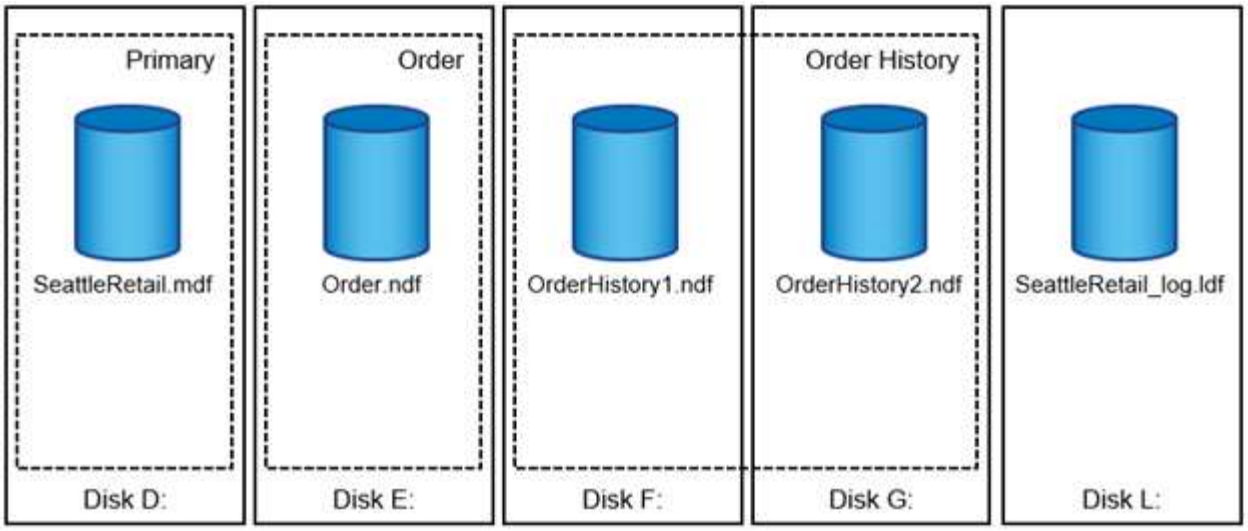
이론적으로 SQL Server(64비트)는 인스턴스당 32,767개의 데이터베이스와 524,272TB의 데이터베이스 크기를

지원하지만, 일반적인 설치에는 일반적으로 여러 개의 데이터베이스가 있습니다. 그러나 SQL Server에서 처리할 수 있는 데이터베이스 수는 로드 및 하드웨어에 따라 다릅니다. SQL Server 인스턴스가 수십, 수백 또는 수천 개의 소규모 데이터베이스를 호스팅하는 것은 드문 일이 아닙니다.

각 데이터베이스는 하나 이상의 데이터 파일과 하나 이상의 트랜잭션 로그 파일로 구성됩니다. 트랜잭션 로그에는 데이터베이스 트랜잭션에 대한 정보와 각 세션에서 수행한 모든 데이터 수정에 대한 정보가 저장됩니다. 데이터가 수정될 때마다 SQL Server는 작업을 실행 취소(롤백)하거나 다시 실행(재생)할 수 있는 충분한 정보를 트랜잭션 로그에 저장합니다. SQL Server 트랜잭션 로그는 데이터 무결성과 견고성에 대한 SQL Server의 평판에 필수적인 부분입니다. 트랜잭션 로그는 SQL Server의 원자성, 일관성, 격리 및 내구성(ACID) 기능에 매우 중요합니다. SQL Server는 데이터 페이지가 변경되는 즉시 트랜잭션 로그에 기록합니다. 모든 DML(Data Manipulation Language) 문(예: SELECT, INSERT, UPDATE 또는 DELETE)은 완전한 트랜잭션이며, 트랜잭션 로그에서는 전체 집합 기반 작업이 수행되도록 하여 트랜잭션의 원자성을 확인합니다.

각 데이터베이스에는 기본 데이터 파일이 하나 있으며 기본적으로 확장명은 .mdf입니다. 또한 각 데이터베이스에는 보조 데이터베이스 파일이 있을 수 있습니다. 이러한 파일의 확장명은 기본적으로 .ndf입니다.

모든 데이터베이스 파일은 파일 그룹으로 그룹화됩니다. 파일 그룹은 논리적 단위로, 데이터베이스 관리를 간소화합니다. 논리 객체 배치와 물리적 데이터베이스 파일 간의 구분이 가능합니다. 데이터베이스 개체 테이블을 만들 때 기본 데이터 파일 구성에 대해 걱정하지 않고 파일 그룹을 배치할 파일 그룹을 지정합니다.



파일 그룹 내에 여러 데이터 파일을 배치할 수 있으므로 여러 스토리지 디바이스에 로드를 분산시킬 수 있으므로 시스템의 입출력 성능을 향상시킬 수 있습니다. 반면 SQL Server는 트랜잭션 로그에 순차적으로 기록하므로 트랜잭션 로그는 여러 파일의 이점을 얻지 못합니다.

파일 그룹에서 논리적 객체 배치와 물리적 데이터베이스 파일 간의 구분을 통해 데이터베이스 파일 레이아웃을 세밀하게 조정하여 스토리지 서브시스템에서 최대한 활용할 수 있습니다. 예를 들어, 서로 다른 고객에게 제품을 배포하는 ISV(Independent Software Vendor)는 기본 I/O 구성과 구축 단계에서 예상되는 데이터 양에 따라 데이터베이스 파일 수를 조정할 수 있습니다. 이러한 변경 사항은 데이터베이스 파일이 아닌 파일 그룹에 데이터베이스 개체를 배치하는 응용 프로그램 개발자에게 영향을 주지 않습니다.

* NetApp는 * 시스템 객체를 제외한 모든 항목에 대해 기본 파일 그룹을 사용하지 않을 것을 권장합니다. 사용자 객체에 대해 별도의 파일 그룹 또는 파일 그룹 집합을 만들면 특히 대규모 데이터베이스의 경우 데이터베이스 관리 및 재해 복구가 간소화됩니다.

데이터베이스를 만들거나 기존 데이터베이스에 새 파일을 추가할 때 초기 파일 크기 및 자동 증가 매개 변수를 지정할 수 있습니다. SQL Server는 데이터를 기록할 데이터 파일을 선택할 때 비례 채우기 알고리즘을 사용합니다. 파일에서 사용할 수 있는 여유 공간에 비례하여 데이터의 양을 기록합니다. 파일의 여유 공간이 많을수록 처리하는 쓰기 횟수가

많아집니다.

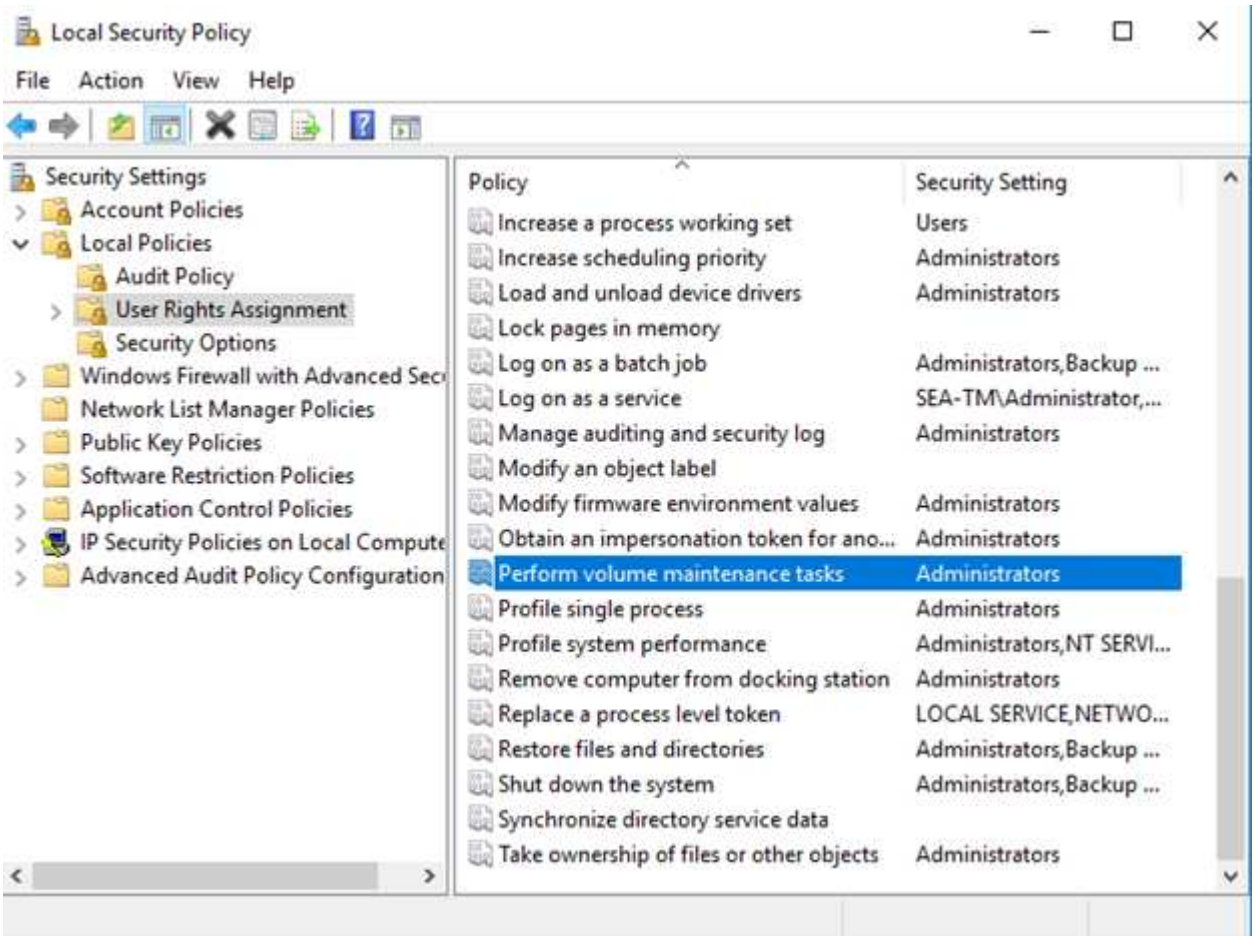
* NetApp는 단일 파일 그룹에 있는 모든 파일의 초기 크기 및 자동 증가 매개 변수가 같고 증가 크기가 백분율이 아닌 메가바이트로 정의됨을 * 권장합니다. 이렇게 하면 비례 채우기 알고리즘이 데이터 파일 간에 쓰기 작업의 균형을 고르게 유지할 수 있습니다.

SQL Server는 파일을 늘릴 때마다 새로 할당된 공간을 0으로 채웁니다. 이 프로세스는 해당 파일에 기록해야 하는 모든 세션을 차단하거나 트랜잭션 로그가 증가하는 경우 트랜잭션 로그 레코드를 생성합니다.

SQL Server는 항상 트랜잭션 로그를 0으로 설정하며 이 동작은 변경할 수 없습니다. 그러나 인스턴트 파일 초기화를 사용하거나 사용하지 않도록 설정하여 데이터 파일의 제로화 여부를 제어할 수 있습니다. 즉각적인 파일 초기화를 사용하면 데이터 파일 증가 속도를 높이고 데이터베이스를 만들거나 복원하는 데 필요한 시간을 줄일 수 있습니다.

즉각적인 파일 초기화와 관련된 보안 위험이 작습니다. 이 옵션을 활성화하면 데이터 파일의 할당되지 않은 부분에 이전에 삭제된 OS 파일의 정보가 포함될 수 있습니다. 데이터베이스 관리자가 이러한 데이터를 검토할 수 있습니다.

SQL Server 시작 계정에 "볼륨 유지 관리 작업 수행"이라고도 하는 SA_MANAGE_VOLUME_NAME 권한을 추가하여 즉각적인 파일 초기화를 활성화할 수 있습니다. 이 작업은 다음 그림과 같이 로컬 보안 정책 관리 응용 프로그램(secpol.msc)에서 수행할 수 있습니다. "볼륨 유지 관리 작업 수행" 권한에 대한 속성을 열고 SQL Server 시작 계정을 사용자 목록에 추가합니다.



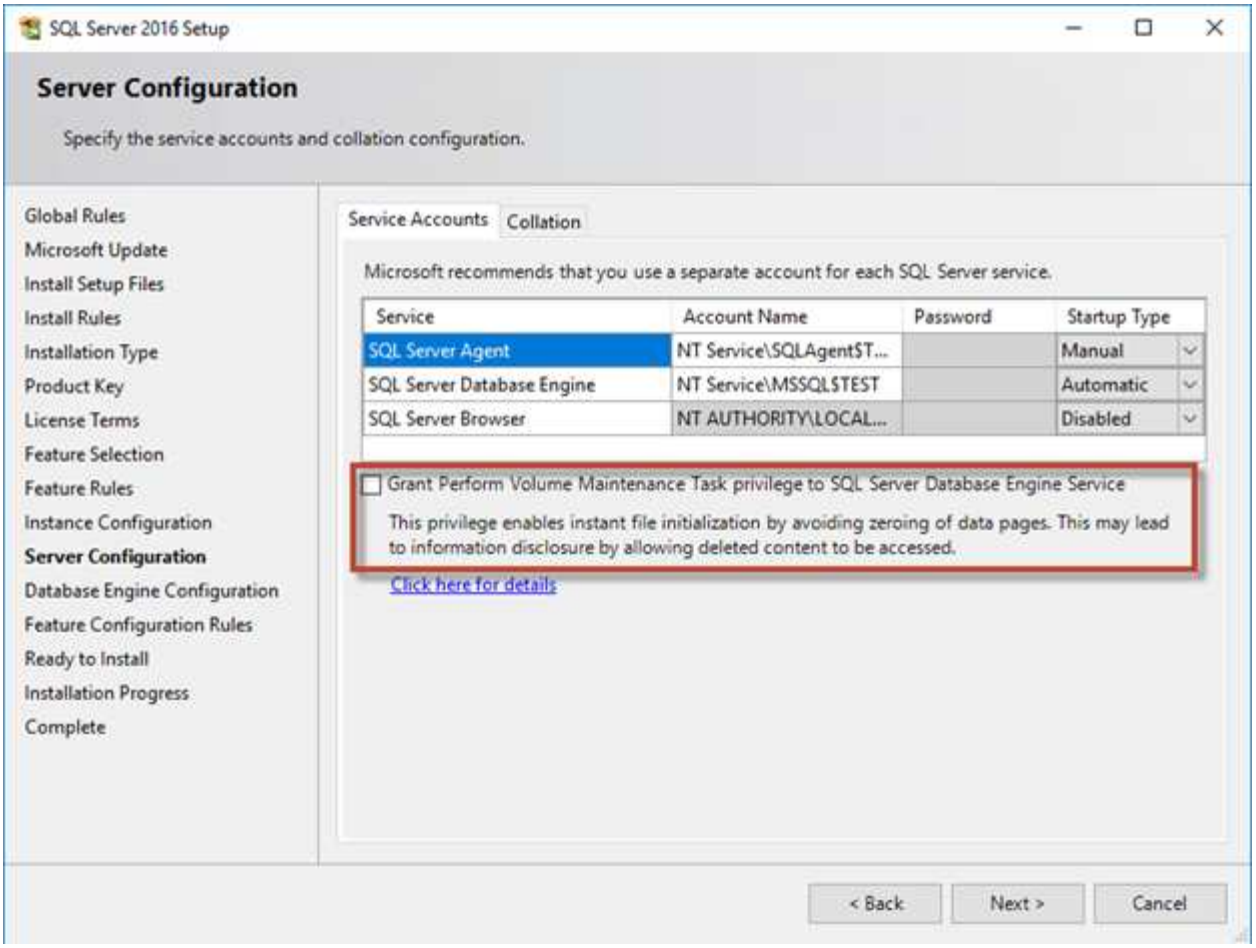
사용 권한이 설정되어 있는지 확인하려면 다음 예제의 코드를 사용합니다. 이 코드는 SQL Server가 오류 로그에 추가 정보를 쓰고, 작은 데이터베이스를 만들고, 로그 내용을 읽도록 하는 두 개의 추적 플래그를 설정합니다.

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

인스턴트 파일 초기화가 사용되지 않는 경우 SQL Server 오류 로그는 다음 예와 같이 SQL Server가 MDF 데이터 파일을 제로화하는 것 외에 LDF 로그 파일을 제로화하는 것을 보여 줍니다. 인스턴트 파일 초기화가 설정된 경우 로그 파일의 제로화만 표시됩니다.

	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\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.

볼륨 유지 관리 수행 작업은 SQL Server 2016에서 간소화되며 나중에 설치 프로세스 중에 옵션으로 제공됩니다. 다음 그림에서는 SQL Server 데이터베이스 엔진 서비스에 볼륨 유지 관리 작업을 수행할 수 있는 권한을 부여하는 옵션을 보여 줍니다.



데이터베이스 파일 크기를 제어하는 또 다른 중요한 데이터베이스 옵션은 자동 축소입니다. 이 옵션을 사용하면 SQL Server에서 정기적으로 데이터베이스 파일을 축소하고 크기를 줄이며 운영 체제에 공간을 해제합니다. 이 작업은 리소스를 많이 사용하며 새 데이터가 시스템에 유입될 때 일정 시간이 지난 후에 데이터베이스 파일이 다시 증가하기 때문에 거의 유용하지 않습니다. 데이터베이스에서 자동 축소를 사용하지 않아야 합니다.

Microsoft SQL Server 로그 디렉터리

로그 디렉토리는 트랜잭션 로그 백업 데이터를 호스트 레벨에서 저장하기 위해 SQL Server에 지정됩니다. SnapCenter를 사용하여 로그 파일을 백업하는 경우 SnapCenter에서 사용하는 각 SQL Server 호스트에 로그 백업을 수행하도록 구성된 호스트 로그 디렉토리가 있어야 합니다. SnapCenter에는 데이터베이스 저장소가 있으므로 백업, 복원 또는 클론 복제 작업과 관련된 메타데이터가 중앙 데이터베이스 저장소에 저장됩니다.

호스트 로그 디렉토리의 크기는 다음과 같이 계산됩니다. 호스트 로그 디렉토리의 크기 = ((최대 DB LDF 크기 x 일일 로그 변경률 %) x (스냅샷 보존) ÷ (1 - LUN 오버헤드 공간 %) 호스트 로그 디렉토리 크기 조정 공식에서는 10%의 LUN 오버헤드 공간을 가정합니다

로그 디렉토리를 전용 볼륨 또는 LUN에 배치합니다. 호스트 로그 디렉토리의 데이터 양은 백업 크기 및 백업 보존 일수에 따라 달라집니다. SnapCenter는 SQL Server 호스트당 하나의 호스트 로그 디렉토리만 허용합니다. 호스트 로그 디렉토리는 SnapCenter -→ 호스트 → 플러그인 구성에서 구성할 수 있습니다.

- NetApp는 호스트 로그 디렉토리에 대해 다음을 권장합니다 *.
- 호스트 로그 디렉토리가 백업 스냅샷 데이터를 손상시킬 수 있는 다른 유형의 데이터와 공유되지 않도록 하십시오.
- 마운트 지점을 호스팅하는 LUN에 사용자 데이터베이스나 시스템 데이터베이스를 배치하지 마십시오.
- SnapCenter에서 트랜잭션 로그를 복제할 전용 FlexVol 볼륨에 호스트 로그 디렉토리를 생성합니다.
- SnapCenter 마법사를 사용하여 데이터베이스를 NetApp 스토리지로 마이그레이션하여 데이터베이스가 유효한 위치에 저장되도록 함으로써 SnapCenter 백업 및 복구 작업을 성공적으로 수행할 수 있습니다. 마이그레이션 프로세스는 중단되며 마이그레이션이 진행 중인 동안 데이터베이스가 오프라인 상태가 될 수 있습니다.
- SQL Server의 FCI(Failover Cluster Instance)에 대해 다음 조건이 충족되어야 합니다.
  - 장애 조치 클러스터 인스턴스를 사용하는 경우 호스트 로그 디렉토리 LUN은 백업 중인 SQL Server 인스턴스와 동일한 클러스터 그룹에 있는 클러스터 디스크 리소스여야 SnapCenter 합니다.
  - 장애 조치 클러스터 인스턴스를 사용하는 경우 SQL Server 인스턴스와 연결된 클러스터 그룹에 할당된 물리적 디스크 클러스터 리소스인 공유 LUN에 사용자 데이터베이스를 배치해야 합니다.

Microsoft SQL Server tempdb 파일

tempdb 데이터베이스를 많이 활용할 수 있습니다. ONTAP에 사용자 데이터베이스 파일을 최적으로 배치할 수 있을 뿐만 아니라 tempdb 데이터 파일을 변경하여 할당 경합을 줄입니다

페이지 경합은 SQL Server가 새 개체를 할당하기 위해 특수 시스템 페이지에 써야 하는 경우 전역 할당 맵(GAM), 공유 전역 할당 맵(SGAM) 또는 페이지 사용 가능 공간(PFS) 페이지에서 발생할 수 있습니다. 래치는 이러한 페이지를 메모리에서 보호(잠금)합니다. 사용 중인 SQL Server 인스턴스의 경우 tempdb의 시스템 페이지에 래치가 표시되는 데 시간이 오래 걸릴 수 있습니다. 이로 인해 쿼리 실행 시간이 느려지며 래치 경합이라고 합니다. tempdb 데이터 파일을 생성하는 방법은 다음 Best Practice를 참조하십시오.

- 또는 = 8코어: tempdb 데이터 파일 = 코어 수입니다
- 8코어 이상의 경우: 8tempdb 데이터 파일

다음 예제 스크립트는 tempdb 파일 8개를 생성하고 tempdb를 마운트 지점으로 이동하여 tempdb를 수정합니다 `C:\MSSQL\tempdb` SQL Server 2012 이상

```
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
```

SQL Server 2016부터 운영 체제에 표시되는 CPU 코어 수가 설치 중에 자동으로 감지되며, 이 수에 따라 SQL Server는 최적의 성능을 위해 필요한 tempdb 파일 수를 계산 및 구성합니다.

Microsoft SQL Server 및 스토리지 효율성

ONTAP 스토리지 효율성은 전체 시스템 성능에 거의 또는 전혀 영향을 미치지 않고 최소한의 스토리지 공간을 사용하는 방식으로 SQL Server 데이터를 저장하고 관리하는 데 최적화되어 있습니다.

스토리지 효율성은 RAID, 프로비저닝(전체 레이아웃 및 활용률), 미러링, 기타 데이터 보호 기술의 조합입니다. 스냅샷, 씬 프로비저닝, 클론 복제와 같은 NetApp 기술은 인프라의 기존 스토리지를 최적화하고 향후 스토리지 지출을 연기하거나 방지합니다. 이러한 기술을 함께 사용할수록 더 많은 비용을 절감할 수 있습니다.

압축, 컴팩션, 중복제거와 같은 공간 효율성 기능은 지정된 양의 물리적 스토리지에 적합한 논리적 데이터의 양을 늘리기 위해 설계되었습니다. 결과적으로 비용과 관리 부담이 줄어듭니다.

상위 수준에서 압축은 수학적 프로세스이며, 그 패턴은 공간 요구사항을 감소시키는 방식으로 데이터를 감지하고 인코딩합니다. 이와 반대로, 중복제거는 실제로 반복되는 데이터 블록을 감지하여 불필요한 복사본을 제거합니다. 컴팩션을 사용하면 여러 개의 논리적 데이터 블록이 미디어에서 동일한 물리적 블록을 공유할 수 있습니다.

스토리지 효율성과 부분 예약 간의 상호 작용에 대한 설명은 아래의 씬 프로비저닝에 대한 섹션을 참조하십시오.

압축

All-Flash 스토리지 시스템을 사용할 수 이전에는 어레이 기반 압축의 값이 제한되었습니다. 대부분의 I/O 집약적인 워크로드에는 허용되는 성능을 제공하기 위해 매우 많은 수의 스핀들이 필요했기 때문입니다. 스토리지 시스템에는 항상 많은 수의 드라이브가 부작용으로 필요한 것보다 훨씬 많은 용량이 포함되어 있습니다. 그러나 솔리드 스테이트 스토리지가 부상하면서 상황이 달라졌습니다. 이제 우수한 성능을 얻기 위해 드라이브를 엄청나게 오버 프로비저닝하지 않아도 됩니다. 스토리지 시스템의 드라이브 공간은 실제 용량 요구 사항과 일치할 수 있습니다.

솔리드 스테이트 드라이브(SSD)의 IOPS 용량이 증가하면 대개 회전식 드라이브에 비해 비용이 절감되며 압축 덕분에 솔리드 스테이트 미디어의 실제 용량이 늘어나 추가 절감을 달성할 수 있습니다.

데이터를 압축하는 방법에는 여러 가지가 있습니다. 대부분의 데이터베이스에는 자체 압축 기능이 포함되어 있지만 고객 환경에서는 이런 일이 거의 발생하지 않습니다. 그 이유는 일반적으로 압축된 데이터로 * 변경 * 시 성능 저하가 발생하며 일부 애플리케이션의 경우 데이터베이스 수준 압축에 대한 라이센스 비용이 많이 듭니다. 마지막으로, 데이터베이스 작업의 전반적인 성능에 영향을 미칩니다. 실제 데이터베이스 작업이 아닌 데이터 압축과 압축 해제를 수행하는 CPU를 위해 CPU당 라이센스 비용으로 높은 금액을 지불하는 것은 합리적이지 않습니다. 더 좋은 옵션은 압축 작업을 스토리지 시스템으로 오프로드하는 것입니다.

적응형 압축

적응형 압축은 지연 시간이 마이크로초 단위로 측정되는 All-Flash 환경에서조차 성능에 미치는 영향 없이 엔터프라이즈 워크로드로 철저히 테스트되었습니다. 심지어 일부 고객은 데이터가 캐시에 압축된 상태로 남아 있으므로 압축을 사용하여 성능이 향상되었다고 보고했습니다. 따라서 컨트롤러에서 가용 캐시의 양이 실질적으로 증가하기 때문입니다.

ONTAP는 4KB 유닛의 물리적 블록을 관리하며 적응형 압축은 기본 압축 블록 크기 8KB를 사용하며, 이는 데이터가 8KB 단위로 압축된다는 것을 의미합니다. 이 크기는 관계형 데이터베이스에서 가장 많이 사용되는 8KB 블록 크기와 일치합니다. 압축 알고리즘은 단일 유닛으로 더 많은 데이터가 압축되므로 효율성이 더욱 향상됩니다. 32KB의 압축 블록 크기는 8KB 압축 블록 유닛보다 더 공간 효율적입니다. 이는 기본 8KB 블록 크기를 사용하는 적응형 압축을

사용하면 효율성이 약간 낮지만 압축 블록 크기를 더 작게 만들면 큰 이점이 있습니다. 데이터베이스 워크로드에는 많은 양의 덮어쓰기 활동이 포함됩니다. 압축된 32KB 데이터 블록의 8KB를 덮어쓰려면 전체 32KB의 논리적 데이터를 다시 읽고, 압축을 풀고, 필요한 8KB 영역을 업데이트하고, 재압축을 수행한 다음 전체 32KB를 드라이브에 다시 써야 합니다. 이는 스토리지 시스템의 경우 매우 많은 비용이 드는 작업이며, 이로 인해 압축 블록 크기가 더 큰 경쟁 스토리지 어레이에서도 데이터베이스 워크로드의 성능이 크게 저하될 수 있습니다.

적응형 압축에서 사용되는 블록 크기는 32KB까지 늘릴 수 있습니다. 이렇게 하면 스토리지 효율성이 향상될 수 있으며, 스토리지에 상당한 양의 데이터가 저장될 경우 트랜잭션 로그 및 백업 파일과 같은 대기 상태의 파일에 대해 고려해야 합니다. 경우에 따라 16KB 또는 32KB 블록 크기를 사용하는 액티브 데이터베이스가 이에 맞춰 적응형 압축의 블록 크기를 늘렸을 수도 있습니다. NetApp 또는 파트너 담당자에게 문의하여 이 솔루션이 현재 워크로드에 적합한지 여부를 확인하십시오.

8KB보다 큰 압축 블록 크기는 스트리밍 백업 대상에서 중복제거와 함께 사용해서는 안 됩니다. 백업된 데이터의 작은 변화가 32KB 압축 기간에 영향을 미치기 때문입니다. 기간이 바뀌면 그에 따라 파일 전체에서 압축된 데이터가 달라집니다. 압축 후 중복제거가 발생하며, 이는 중복제거 엔진이 압축된 각 백업을 다르게 간주한다는 의미입니다. 스트리밍 백업의 중복제거가 필요한 경우 8KB 블록 적응형 압축만 사용해야 합니다. 더 작은 블록 크기를 사용할 수 있고 중복제거 효율성을 방해하지 않기 때문에 적응형 압축이 더 낫습니다. 유사한 이유로 호스트 측 압축도 중복제거 효율성에 지장을 줍니다.

압축 정렬

데이터베이스 환경에서 적응형 압축을 수행할 때는 압축 블록 정렬과 관련된 몇 가지 사항을 고려해야 합니다. 이는 특정 블록의 랜덤 덮어쓰기가 데이터에 적용되는 경우만 해당합니다. 이 접근 방식은 파일 시스템의 시작이 4K 디바이스 경계에 맞춰 정렬되어야 하고 파일 시스템의 블록 크기가 4K의 배수여야 하는 전체 파일 시스템 정렬과 개념이 비슷합니다.

예를 들어, 파일에 대한 8KB 쓰기는 파일 시스템 자체 내에서 8KB 경계와 일치하는 경우에만 압축됩니다. 즉 파일의 첫 번째 8KB에, 두 번째 8KB 에 그리고 그 이후로도 동일하게 포함되어야 합니다. 올바른 정렬을 보장하는 가장 간단한 방법은 올바른 LUN 유형을 사용하는 것입니다. 생성된 모든 파티션은 8K의 배수인 디바이스의 시작 부분에서 오프셋을 가지며 데이터베이스 블록 크기의 배수인 파일 시스템 블록 크기를 사용해야 합니다.

백업이나 트랜잭션 로그 같은 데이터는 압축된 여러 블록을 확장하는 순차적 쓰기 작업이며 따라서 정렬을 고려할 필요가 없습니다. I/O 패턴에서 고려해야 할 한 가지는 파일의 랜덤 덮어쓰기입니다.

데이터 컴팩션

데이터 컴팩션은 압축 효율성을 향상하는 기술입니다. 앞서 설명한 것처럼, 적응형 압축만 사용했을 때는 절감 비율이 최대 2:1입니다. 4KB WAFL 블록에 8KB I/O를 저장하도록 제한되어 있기 때문입니다. 블록 크기가 더 큰 압축 방법을 통해 효율성이 향상됩니다. 그러나 이러한 복사본은 작은 블록 덮어쓰기가 적용되는 데이터에는 적합하지 않습니다. 32KB 단위 데이터의 압축 해제, 8KB 부분 업데이트, 재압축, 드라이브에 다시 쓰기 작업은 오버헤드를 발생시킵니다.

데이터 컴팩션은 여러 논리적 블록이 물리적 블록 내에 저장될 수 있게 합니다. 예를 들어, 텍스트 또는 부분 전체 블록과 같이 고도로 압축 가능한 데이터가 포함된 데이터베이스는 8KB에서 1KB로 압축될 수 있습니다. 컴팩션을 적용하지 않으면 이 1KB 데이터는 여전히 4KB 블록 전체를 점유할 것입니다. 인라인 데이터 컴팩션에서는 압축된 데이터 1KB를 다른 압축된 데이터와 함께 단 1KB의 물리적 공간에 저장할 수 있습니다. 이 방식은 압축 기술이 아니라 그저 드라이브의 공간을 더 효율적으로 할당하는 것이며 감지할 수 있는 성능 영향을 발생시키지 않습니다.

이로써 얻어지는 절감의 수준은 다양합니다. 이미 압축되었거나 암호화된 데이터는 일반적으로 더 압축할 수 없기 때문에 이 데이터 세트는 컴팩션의 이점을 얻지 못합니다. 반면 제로와 블록 메타데이터보다 조금 더 많이 포함하고 있으며 새롭게 초기화된 데이터 파일의 경우 80:1까지 압축합니다.

온도에 민감한 스토리지 효율성

TSSE(Temperature Sensitive Storage Efficiency)는 블록 액세스 히트 맵을 사용하여 자주 액세스하지 않는 블록을 식별하고 보다 효율적으로 압축하는 ONTAP 9.8 이상에서 사용할 수 있습니다.

중복 제거

중복 제거는 데이터 세트에서 중복된 블록 크기가 제거됩니다. 예를 들어, 동일한 4KB 블록이 10개 파일에 존재하면 중복제거는 파일 10개 전체에서 해당 4KB 블록을 동일한 4KB 물리적 블록으로 리디렉션합니다. 그 결과 데이터의 효율성이 10:1로 향상됩니다.

VMware 게스트 부팅 LUN과 같은 데이터는 동일한 운영 체제 파일의 여러 복사본으로 구성되어 있기 때문에 중복 제거가 매우 용이합니다. 100:1 이상의 효율성이 관찰되었습니다.

일부 데이터에 중복 데이터가 없습니다. 예를 들어, Oracle 블록에는 데이터베이스에 관해 전역적으로 고유한 헤더와 거의 고유한 트레일러가 포함되어 있습니다. 따라서 Oracle 데이터베이스의 중복 제거 기능을 사용하면 1%를 넘는 비용을 절감하는 경우는 거의 없습니다. MS SQL 데이터베이스의 중복 제거는 약간 더 낫지만 블록 수준에서 고유한 메타데이터는 여전히 제한 사항입니다.

일부 경우 16KB 및 대형 블록 크기의 데이터베이스에서 공간이 최대 15% 절약되었습니다. 각 블록의 처음 4KB에는 전역적으로 고유한 헤더가 포함되어 있고 마지막 4KB 블록에는 거의 고유한 트레일러가 포함되어 있습니다. 실제로는 거의 전적으로 제로화 데이터의 중복제거에 기인하지만 내부 블록은 중복제거 후보입니다.

많은 경쟁업체의 어레이는 데이터베이스가 여러 차례 복사된다는 추정을 기반으로 데이터베이스의 중복제거 기능을 내세웁니다. 이런 측면에서 NetApp 중복제거도 사용할 수 있지만 ONTAP은 더 나은 옵션인 NetApp FlexClone 기술을 제공합니다. 최종 결과는 같으며 기본 물리적 블록의 대부분을 공유하는 데이터베이스의 복사본이 여러 개 생성됩니다. FlexClone은 시간을 들여 데이터베이스 파일을 복사한 다음 중복제거하는 것보다 훨씬 더 효율적입니다. 실제로 이는 중복제거가 아니라 비중복이라 할 수 있습니다. 애초에 중복을 생성하지 않기 때문입니다.

효율성 및 씬 프로비저닝

효율성 기능은 씬 프로비저닝의 한 형태입니다. 예를 들어, 100GB 볼륨을 점유하는 100GB LUN은 50GB까지 압축할 수 있을 것이고 볼륨은 여전히 100GB이기 때문에 실제로 절감이 실현되지는 않았습니다. 먼저 볼륨의 크기를 줄여 절감된 공간을 시스템의 어느 곳에서든 사용할 수 있게 해야 합니다. 나중에 100GB LUN으로 변경하면 데이터 압축률이 줄어들어 LUN 크기가 커지고 볼륨을 가득 채울 수 있습니다.

씬 프로비저닝은 관리를 단순화하는 동시에 가용 용량을 크게 개선하면서 비용을 절감할 수 있기 때문에 적극 권장합니다. 단순한 데이터베이스 환경에서 많은 빈 공간, 많은 수의 볼륨 및 LUN, 압축 가능한 데이터가 포함되는 경우가 많습니다. 일반 프로비저닝은 언젠가 100% 채워지고 100% 압축할 수 없는 데이터가 포함될 경우에 대비해 볼륨 및 LUN에 대한 스토리지 공간을 예약합니다. 그런 일은 일어나지 않을 것입니다. 씬 프로비저닝을 사용하면 공간을 재확보하고 다른 위치에서 사용할 수 있으며 더 작은 볼륨 및 LUN이 아닌 스토리지 시스템 자체를 기반으로 용량을 관리할 수 있습니다.

일부 고객은 특정 워크로드에 대해 또는 일반적으로 확립된 운영 및 조달 사례를 기반으로 일반 프로비저닝을 사용하는 것을 선호합니다.

- 주의: * 볼륨이 일반 프로비저닝되면 압축 해제 및 를 사용한 중복 제거 제거를 포함하여 해당 볼륨에 대한 모든 효율성 기능을 완전히 비활성화하도록 주의해야 합니다 `sis undo` 명령. 볼륨은 에 나타나지 않아야 합니다 `volume efficiency show` 출력. 그렇지 않을 경우, 효율성 기능을 위해 볼륨이 부분적으로 구성됩니다. 결과적으로 덮어쓰기 보장은 서로 다르게 동작하므로 구성 과다 사용으로 인해 볼륨의 공간이 예기치 않게 부족해져서 데이터베이스 I/O 오류가 발생할 가능성이 높아집니다.

효율성 모범 사례

NetApp에서 권장하는 사항은 다음과 같습니다.

AFF 기본값

All-Flash AFF 시스템에서 실행되는 ONTAP에서 생성된 볼륨은 모든 인라인 효율성 기능을 사용하는 씬 프로비저닝됩니다. 일반적으로 데이터베이스에는 중복제거를 통해 이점을 얻을 수 없고 압축할 수 없는 데이터가 포함될 수 있지만 그럼에도 불구하고 기본 설정은 거의 모든 워크로드에 적합합니다. ONTAP는 절감 여부와 관계없이 모든 유형의 데이터와 I/O 패턴을 효율적으로 처리하도록 설계되었습니다. 원인을 완전히 이해하고 편차가 있는 경우에만 기본값을 변경해야 합니다.

일반 권장 사항

- 볼륨 및/또는 LUN이 씬 프로비저닝되지 않는 경우 모든 효율성 설정을 비활성화해야 합니다. 이러한 기능을 사용하면 절약 효과가 없고 일반 프로비저닝과 공간 효율성이 활성화된 조합을 통해 공간 부족 오류를 포함하여 예기치 않은 동작이 발생할 수 있기 때문입니다.
- 백업 또는 데이터베이스 트랜잭션 로그와 같이 데이터를 덮어쓰지 않는 경우 냉각 기간이 짧은 TSSE를 활성화하여 효율성을 높일 수 있습니다.
- 일부 파일에는 압축할 수 없는 많은 양의 데이터가 포함되어 있을 수 있습니다. 예를 들어 파일의 응용 프로그램 수준에서 압축이 이미 활성화되어 있는 경우 암호화됩니다. 이러한 시나리오가 적용되는 경우 압축 데이터를 포함하는 다른 볼륨에서 더 효율적으로 작업할 수 있도록 압축을 해제하는 것이 좋습니다.
- 데이터베이스 백업에 32KB 압축 및 중복제거를 모두 사용하지 마십시오. 섹션을 참조하십시오 적응형 압축 를 참조하십시오.

데이터베이스 압축

SQL Server 자체에는 데이터를 압축하고 효율적으로 관리하는 기능도 있습니다. SQL Server는 현재 행 압축과 페이지 압축이라는 두 가지 유형의 데이터 압축을 지원합니다.

행 압축은 데이터 저장소 형식을 변경합니다. 예를 들어, 정수와 소수를 네이티브 고정 길이 형식 대신 가변 길이 형식으로 변경합니다. 또한 빈 공백을 제거하여 고정 길이 문자 문자열을 가변 길이 형식으로 변경합니다. 페이지 압축은 행 압축과 두 가지 다른 압축 전략(접두사 압축 및 사전 압축)을 구현합니다. 페이지 압축에 대한 자세한 내용은 에서 확인할 수 있습니다 "페이지 압축 구현".

데이터 압축은 현재 SQL Server 2008 이상의 Enterprise, Developer 및 Evaluation 에디션에서 지원됩니다. 데이터베이스가 자체적으로 압축을 수행할 수 있긴 하지만 SQL Server 환경에서는 이런 일이 거의 발생하지 않습니다.

다음은 SQL Server 데이터 파일의 공간을 관리하기 위한 권장 사항입니다

- SQL Server 환경에서 씬 프로비저닝을 사용하여 공간 사용률을 개선하고 공간 보장 기능을 사용할 때 전체 스토리지 요구 사항을 줄입니다.
- 스토리지 관리자는 애그리게이트의 공간 사용량만 모니터링하면 되기 때문에 가장 일반적인 구축 구성에 대해 자동 확장 기능을 사용합니다.
- SQL Server 데이터 파일이 포함된 볼륨에 백업을 단일 볼륨으로 복원하는 것과 같이 동일한 데이터의 여러 복사본이 포함되어 있는 것으로 알려져 있지 않은 경우 SQL Server 데이터 파일이 포함된 볼륨에서 중복 제거를 사용하지 않는 것이 좋습니다.

공간 재확보

LUN에서 사용되지 않는 공간을 복구하기 위해 공간 재확보를 주기적으로 시작할 수 있습니다. SnapCenter에서는 다음 PowerShell 명령을 사용하여 공간 재확보를 시작할 수 있습니다.

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

공간 재확보를 실행해야 하는 경우 이 프로세스는 처음에 호스트의 주기를 소비하기 때문에 작업이 적은 기간 동안 실행해야 합니다.