ILM 및 오브젝트 라이프사이클

StorageGRID

NetApp
March 12, 2025

목차

ILM 및 오브젝트 라이프사이클

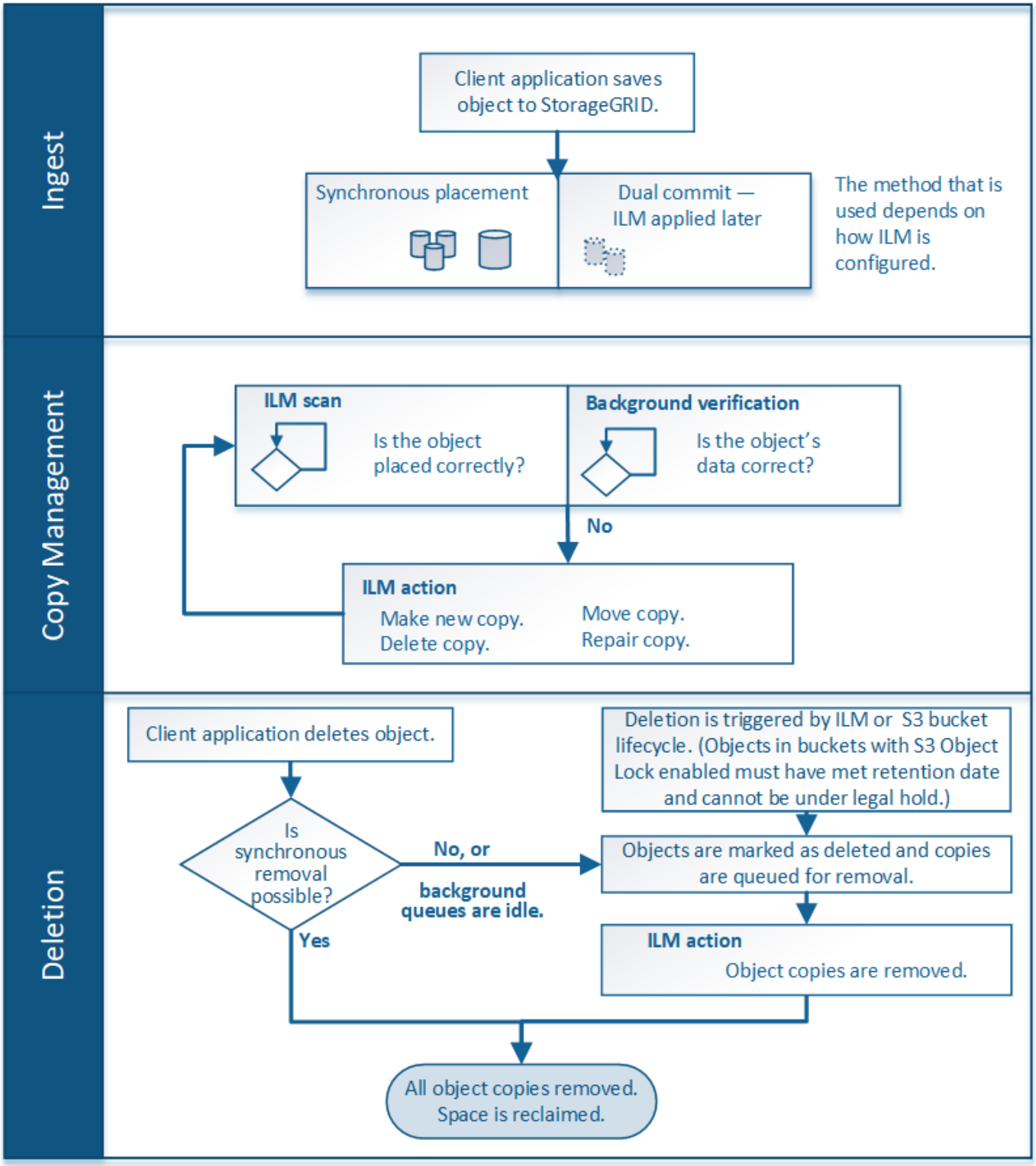
ILM이 개체 수명 전반에 걸쳐 작동하는 방식

StorageGRID에서 ILM을 사용하여 삶의 모든 단계에서 개체를 관리하는 방법을 이해하면 더 효과적인 정책을 설계하는 데 도움이 됩니다.

- * Ingest *: Ingest는 S3 클라이언트 응용 프로그램이 StorageGRID 시스템에 개체를 저장하기 위한 연결을 설정할 때 시작되며 StorageGRID가 클라이언트에 "수집 성공" 메시지를 반환할 때 완료됩니다. ILM 요구 사항이 지정된 방식에 따라 즉시(동기식 배치) ILM 명령을 적용하거나 나중에 ILM(이중 커밋)을 적용하여 수집 중에 오브젝트 데이터를 보호합니다.
- * 복사본 관리 *: ILM의 배치 명령에 지정된 오브젝트 복사본의 수와 유형을 생성한 후 StorageGRID는 오브젝트 위치를 관리하고 개체로부터 손실을 보호합니다.
  - * ILM 스캔 및 평가 *: StorageGRID는 그리드에 저장된 객체 목록을 지속적으로 검사하고 현재 복사본이 ILM 요구 사항을 충족하는지 확인합니다. 오브젝트 복사본의 유형, 숫자 또는 위치가 서로 다른 경우 StorageGRID은 필요에 따라 복사본을 생성, 삭제 또는 이동합니다.
  - * 배경 검증 * : StorageGRID는 객체 데이터의 무결성을 확인하기 위해 지속적으로 백그라운드 검증을 수행합니다. 문제가 발견되면 StorageGRID는 현재 ILM 요구사항을 충족하는 위치에 새 오브젝트 복사본 또는 삭제 코딩 된 대체 오브젝트 조각을 자동으로 생성합니다. 을 "개체 무결성을 확인합니다"참조하십시오.
- * 개체 삭제 *: StorageGRID 시스템에서 모든 복사본이 제거될 때 개체 관리가 종료됩니다. 클라이언트의 삭제 요청 결과로 또는 ILM에 의한 삭제 또는 S3 버킷 라이프사이클의 만료로 인한 삭제로 인해 오브젝트를 제거할 수 있습니다.

> ⓘ S3 오브젝트 잠금이 활성화된 버킷의 오브젝트는 법적 증거 자료 보관 중이거나 보존 기한이 지정되었지만 아직 충족되지 않은 경우 삭제할 수 없습니다.

이 다이어그램은 ILM이 개체 수명 주기 동안 어떻게 작동하는지를 요약합니다.



오브젝트를 섭취하는 방법

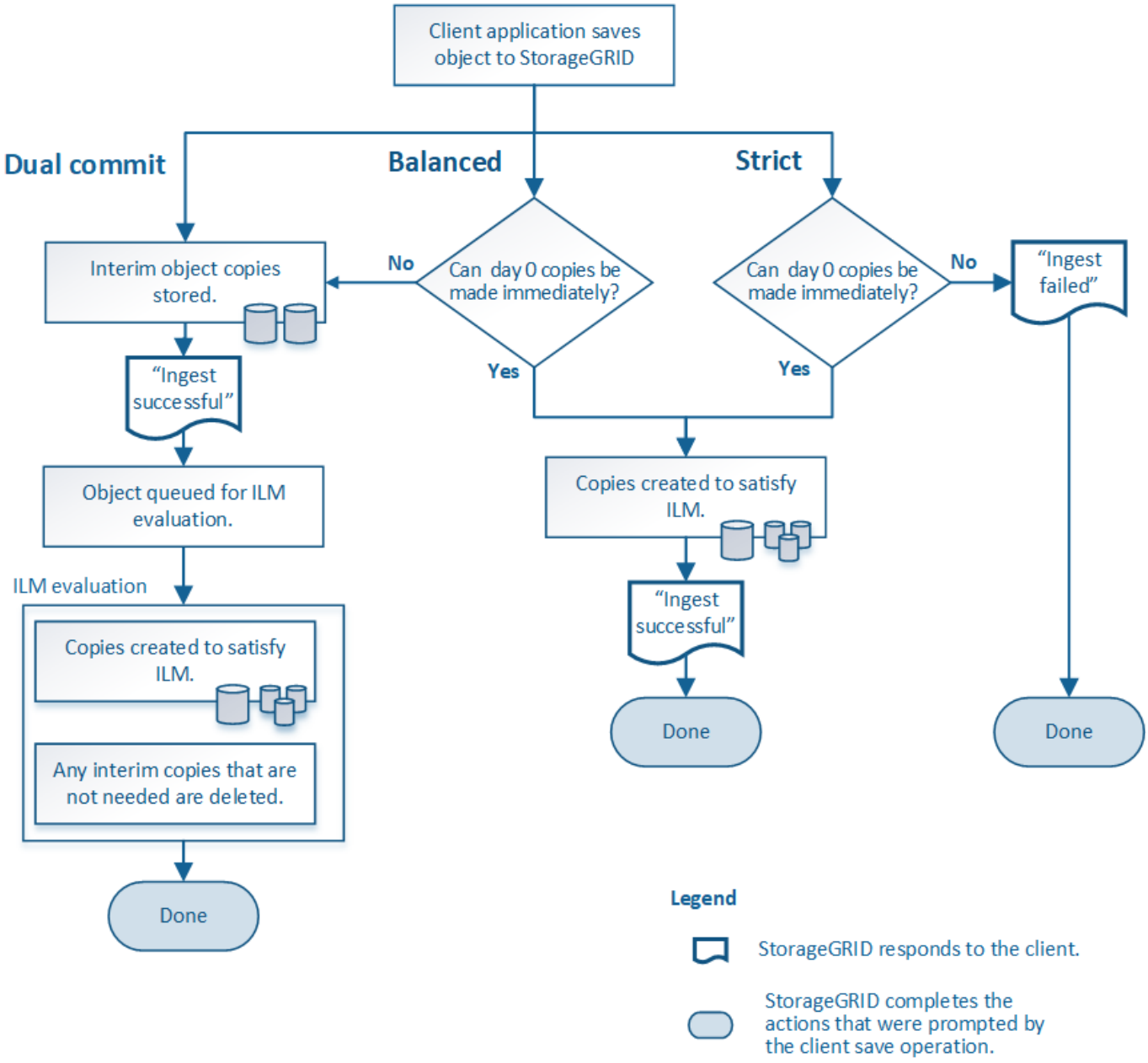
수집 옵션

ILM 규칙을 생성할 때 수집 시 개체를 보호하기 위한 세 가지 옵션 중 하나인 이중 커밋, Strict 또는 균형 을 지정합니다.

선택한 사항에 따라 StorageGRID은 임시 복사본을 만들고 나중에 ILM 평가를 위해 오브젝트를 큐에 대기시키거나 동기식 배치를 사용하여 ILM 요구 사항을 충족하도록 즉시 복사본을 만듭니다.

수집 옵션의 흐름도

이 순서도는 세 가지 수집 옵션 각각을 사용하는 ILM 규칙에 따라 오브젝트가 일치할 때 발생하는 상황을 보여 줍니다.



이중 커밋

Dual Commit 옵션을 선택하면 StorageGRID는 즉시 서로 다른 두 스토리지 노드에 임시 객체 복사본을 만들고 "ingest successful" 메시지를 클라이언트에 반환합니다. 개체는 ILM 평가를 위해 대기하며 규칙의 배치 지침을 충족하는 복사본은 나중에 만들어집니다. 이중 커밋 후 ILM 정책을 즉시 처리할 수 없는 경우 사이트 손실 보호를 달성하는 데 시간이 걸릴 수 있습니다.

다음 두 경우 중 하나에서 이중 커밋 옵션을 사용합니다.

- 멀티 사이트 ILM 규칙을 사용 중이며 클라이언트 수집 지연 시간이 중요하게 고려해야 합니다. 이중 커밋을 사용할 때 ILM을 충족하지 못하는 경우 그리드에서 이중 커밋 복사본을 만들고 제거하는 추가 작업을 수행할 수 있는지 확인해야 합니다. 주요 내용은 다음과 같습니다.
  - ILM 백로그를 방지할 수 있을 정도로 그리드의 부하가 낮아야 합니다.
  - 그리드에는 초과 하드웨어 리소스(IOPS, CPU, 메모리, 네트워크 대역폭 등)가 있어야 합니다.
- 다중 사이트 ILM 규칙을 사용 중이며 사이트 간 WAN 연결에 일반적으로 지연 시간이 길거나 대역폭이 제한되어 있습니다. 이 시나리오에서 이중 커밋 옵션을 사용하면 클라이언트 시간 초과를 방지할 수 있습니다. 이중 커밋 옵션을 선택하기 전에 실제 워크로드로 클라이언트 애플리케이션을 테스트해야 합니다.

균형(기본값)

균형 옵션을 선택하면 StorageGRID는 수집 시 동기식 배치를 사용하고 규칙의 배치 지침에 지정된 모든 복사본을 즉시 생성합니다. Strict 옵션과 달리 StorageGRID 에서 즉시 모든 복사본을 만들 수 없는 경우에는 Dual commit 을 대신 사용합니다. ILM 정책이 여러 사이트의 배치를 사용하고 즉각적인 사이트 손실 방지를 달성할 수 없는 경우 * ILM 배치 불가능 * 경고가 트리거됩니다.

균형 옵션을 사용하면 데이터 보호, 그리드 성능 및 수집 성공을 최적으로 조합하여 달성할 수 있습니다. ILM 규칙 만들기 마법사의 기본 옵션은 균형 조정입니다.

엄격한

Strict 옵션을 선택하면 StorageGRID에서는 수집 시 동기식 배치를 사용하고 규칙의 배치 지침에 지정된 모든 오브젝트 복사본을 즉시 생성합니다. 필요한 스토리지 위치를 일시적으로 사용할 수 없기 때문에 StorageGRID에서 모든 복사본을 생성할 수 없는 경우 수집에 실패합니다. 클라이언트가 작업을 재시도해야 합니다.

ILM 규칙에 요약된 위치에만 개체를 즉시 저장해야 하는 운영 또는 규정 요구사항이 있는 경우 Strict 옵션을 사용합니다. 예를 들어, 규정 요구 사항을 충족하려면 Strict 옵션 및 Location Constraint 고급 필터를 사용하여 개체가 특정 데이터 센터에 저장되지 않도록 해야 할 수 있습니다.

을 "예 5: 엄격한 수집 동작을 위한 ILM 규칙 및 정책"참조하십시오.

수집 옵션의 장점, 단점 및 제한 사항

수집 시 데이터를 보호하기 위한 세 가지 옵션(균형, 엄격 또는 이중 커밋)의 각 장단점을 이해하면 ILM 규칙에 대해 선택할 항목을 결정하는 데 도움이 됩니다.

수집 옵션에 대한 개요는 를 참조하십시오"수집 옵션".

균형 및 엄격 옵션의 장점

수집하는 동안 임시 사본을 생성하는 이중 커밋과 비교할 때 두 개의 동기식 배치 옵션은 다음과 같은 이점을 제공합니다.

- * 더 나은 데이터 보안 *: ILM 규칙의 배치 지침에 지정된 대로 개체 데이터가 즉시 보호됩니다. ILM은 둘 이상의 스토리지 위치 장애를 포함하여 다양한 장애 조건을 보호하도록 구성할 수 있습니다. 이중 커밋은 단일 로컬 복사본의 손실로부터 보호할 수 있습니다.
- * 더 효율적인 그리드 작업 *: 각 오브젝트는 수집될 때 한 번만 처리됩니다. StorageGRID 시스템은 중간 복사본을 추적하거나 삭제할 필요가 없으므로 처리 부하가 줄어들고 데이터베이스 공간이 더 적게 사용됩니다.
- * (Balanced) 권장 *: 최적의 ILM 효율성을 제공하는 균형 잡힌 옵션입니다. 엄격한 수집 동작이 필요하거나

그리드가 이중 커밋 사용에 대한 모든 기준을 충족하지 않는 한 균형 옵션을 사용하는 것이 좋습니다.

- * (Strict) 개체 위치에 대한 확실성 * : Strict 옵션은 ILM 규칙의 배치 지침에 따라 개체를 즉시 저장합니다.

균형 및 엄격 옵션의 단점

이중 커밋과 비교할 때 균형 및 엄격 옵션에는 다음과 같은 몇 가지 단점이 있습니다.

- * 더 긴 클라이언트 인제스트 *: 클라이언트 인제스트 지연 시간이 더 길어질 수 있습니다. Balanced 또는 Strict 옵션을 사용하는 경우 삭제 코딩 단편이나 복제된 복제본이 모두 생성 및 저장될 때까지 "수집 성공" 메시지가 클라이언트에 반환되지 않습니다. 하지만 오브젝트 데이터는 최종 위치에 훨씬 더 빠르게 도달할 수 있습니다.
- * (Strict) 수집 실패 비율 증가 *: Strict 옵션을 사용하면 StorageGRID에서 ILM 규칙에 지정된 모든 복사본을 즉시 만들 수 없을 때마다 수집이 실패합니다. 필요한 스토리지 위치가 일시적으로 오프라인이거나 네트워크 문제로 인해 사이트 간에 오브젝트 복제가 지연될 경우 수집 장애가 발생할 가능성이 높습니다.
- * (Strict) S3 멀티파트 업로드 배치가 일부 상황에서 예상과 다를 수 있습니다. *: Strict 를 사용하면 ILM 규칙에 설명된 대로 개체를 배치하거나 수집하지 못할 수 있습니다. 하지만 S3 멀티파트 업로드를 사용하면 ILM이 수집되는 개체의 각 부분에 대해 계산되고, 멀티파트 업로드가 완료되면 개체 전체에 대해 평가됩니다. 다음과 같은 상황에서는 예상과 다른 배치를 초래할 수 있습니다.
  - * S3 멀티파트 업로드가 진행 중일 때 ILM이 변경되는 경우 *: 각 파트는 파트를 인제스트할 때 활성 규칙에 따라 배치되므로 멀티파트 업로드가 완료될 때 개체의 일부 부분이 현재 ILM 요구 사항을 충족하지 못할 수 있습니다. 이 경우 오브젝트 수집은 실패하지 않습니다. 대신 올바르게 배치되지 않은 모든 부품은 ILM 재평가를 위해 대기하다가 나중에 올바른 위치로 이동됩니다.
  - * ILM 규칙이 크기 * 를 기준으로 필터링할 때: 파트에 대한 ILM을 평가할 때 StorageGRID은 개체의 크기가 아닌 파트 크기를 필터링합니다. 즉, 개체의 일부를 개체에 대한 ILM 요구 사항을 전체가 충족하지 않는 위치에 저장할 수 있습니다. 예를 들어, 규칙이 모든 오브젝트 10GB 이상이 DC1에 저장되는 반면 모든 작은 오브젝트는 DC2에 저장되는 것으로 지정하는 경우 10개 부분 멀티파트 업로드의 각 1GB 부분은 DC2에 저장됩니다. 개체에 대한 ILM을 평가할 때 개체의 모든 부분이 DC1로 이동합니다.
- * (Strict) Ingest는 오브젝트 태그 또는 메타데이터를 업데이트하고 새로 필요한 배치를 만들 수 없을 때 실패합니다. * : Strict 를 사용하면 ILM 규칙에 설명된 대로 개체를 배치하거나 수집 실패가 발생할 수 있습니다. 하지만 이미 그리드에 저장된 개체의 메타데이터 또는 태그를 업데이트하는 경우 객체가 다시 수집되지 않습니다. 즉, 업데이트로 인해 트리거되는 오브젝트 위치는 즉시 변경되지 않습니다. ILM을 정상적인 배경 ILM 프로세스에 의해 재평가할 때 배치 변경이 이루어집니다. 필요한 위치를 변경할 수 없는 경우(예: 새로 필요한 위치를 사용할 수 없는 경우), 업데이트된 개체는 배치를 변경할 수 있을 때까지 현재 위치를 유지합니다.

균형 및 엄격 옵션을 사용한 개체 배치 제한

다음 배치 지침이 있는 ILM 규칙에는 균형 또는 엄격 옵션을 사용할 수 없습니다.

- 0일에 클라우드 스토리지 풀에 배치.
- 규칙에 사용자 정의 생성 시간이 레퍼런스 시간으로 설정된 경우의 클라우드 스토리지 풀 배치

이러한 제한 사항은 StorageGRID가 클라우드 스토리지 풀에 대한 복제본을 동기식으로 만들 수 없고 사용자 정의 생성 시간이 현재로 해결될 수 있기 때문입니다.

ILM 규칙 및 일관성이 상호 작용하여 데이터 보호에 영향을 미치는 방법

ILM 규칙과 정합성 보장 선택은 모두 오브젝트를 보호하는 방법에 영향을 미칩니다. 이러한 설정은 상호 작용할 수 있습니다.

예를 들어, ILM 규칙을 위해 선택된 수집 동작은 오브젝트 복사본의 초기 배치에 영향을 미치며, 오브젝트가 저장될 때

사용되는 일관성은 오브젝트 메타데이터의 초기 배치에 영향을 미칩니다. StorageGRID에서는 클라이언트 요청을 이행하기 위해 오브젝트의 데이터와 메타데이터에 모두 액세스해야 하기 때문에 정합성 보장 및 수집 동작에 대해 일치하는 보호 수준을 선택하면 초기 데이터 보호 수준을 높이고 시스템 응답을 예측할 수 있습니다.

다음은 StorageGRID에서 사용할 수 있는 정합성 보장 값에 대한 간략한 요약입니다.

- * ALL *: 모든 노드가 즉시 객체 메타데이터를 수신하거나 요청이 실패합니다.
- **Strong-global**: 개체 메타데이터가 모든 사이트에 즉시 배포됩니다. 모든 사이트에서 모든 클라이언트 요청에 대해 쓰기 후 읽기 정합성을 보장합니다.
- **Strong-site**: 개체 메타데이터가 사이트의 다른 노드에 즉시 배포됩니다. 사이트 내의 모든 클라이언트 요청에 대해 쓰기 후 읽기 일관성을 보장합니다.
- **Read-after-new-write**: 새 개체에 대해 읽기-쓰기 후 일관성을 제공하고 개체 업데이트에 대한 최종 일관성을 제공합니다. 고가용성 및 데이터 보호 보장 제공 대부분의 경우에 권장됩니다.
- * 사용 가능 *: 새 객체 및 객체 업데이트 모두에 대한 최종 일관성을 제공합니다. S3 버킷의 경우 필요한 경우에만 사용하십시오(예: 거의 읽지 않는 로그 값이 포함된 버킷의 경우 또는 존재하지 않는 키의 헤드 또는 GET 작업의 경우). S3 FabricPool 버킷은 지원되지 않습니다.

> 정합성 보장 값을 선택하기 전에 를 "일관성에 대한 전체 설명을 읽어 보십시오"참조하십시오. 기본값을 변경하기 전에 이점과 제한 사항을 이해해야 합니다.

일관성과 **ILM** 규칙이 상호 작용하는 방법의 예

다음과 같은 ILM 규칙과 다음과 같은 일관성이 있는 2개 사이트 그리드가 있다고 가정합니다.

- * ILM 규칙 *: 로컬 사이트와 원격 사이트에 각각 하나씩, 두 개의 오브젝트 복사본을 만듭니다. 엄격한 수집 동작을 사용합니다.
- * Consistency *: 강력한 글로벌(오브젝트 메타데이터는 모든 사이트에 즉시 배포됨).

클라이언트가 오브젝트를 그리드에 저장할 때 StorageGRID는 오브젝트 복사본을 둘 다 만들고 메타데이터를 두 사이트에 분산한 다음 클라이언트에 성공을 반환합니다.

수집 성공 메시지가 표시된 시점에 객체가 손실로부터 완벽하게 보호됩니다. 예를 들어, 수집 직후 로컬 사이트가 손실되면 오브젝트 데이터와 오브젝트 메타데이터의 복사본이 원격 사이트에 계속 존재합니다. 개체를 완전히 검색할 수 있습니다.

대신 동일한 ILM 규칙과 강력한 사이트 일관성을 사용한 경우 개체 데이터가 원격 사이트에 복제된 후 개체 메타데이터가 이 사이트에 배포되기 전에 클라이언트에서 성공 메시지를 받을 수 있습니다. 이 경우 오브젝트 메타데이터의 보호 수준이 오브젝트 데이터의 보호 수준과 일치하지 않습니다. 수집 후 곧바로 로컬 사이트가 손실되면 오브젝트 메타데이터가 손실됩니다. 개체를 검색할 수 없습니다.

일관성과 ILM 규칙 간의 상호 관계는 복잡할 수 있습니다. 도움이 필요하면 NetApp에 문의하십시오.

관련 정보

"예 5: 엄격한 수집 동작을 위한 ILM 규칙 및 정책"

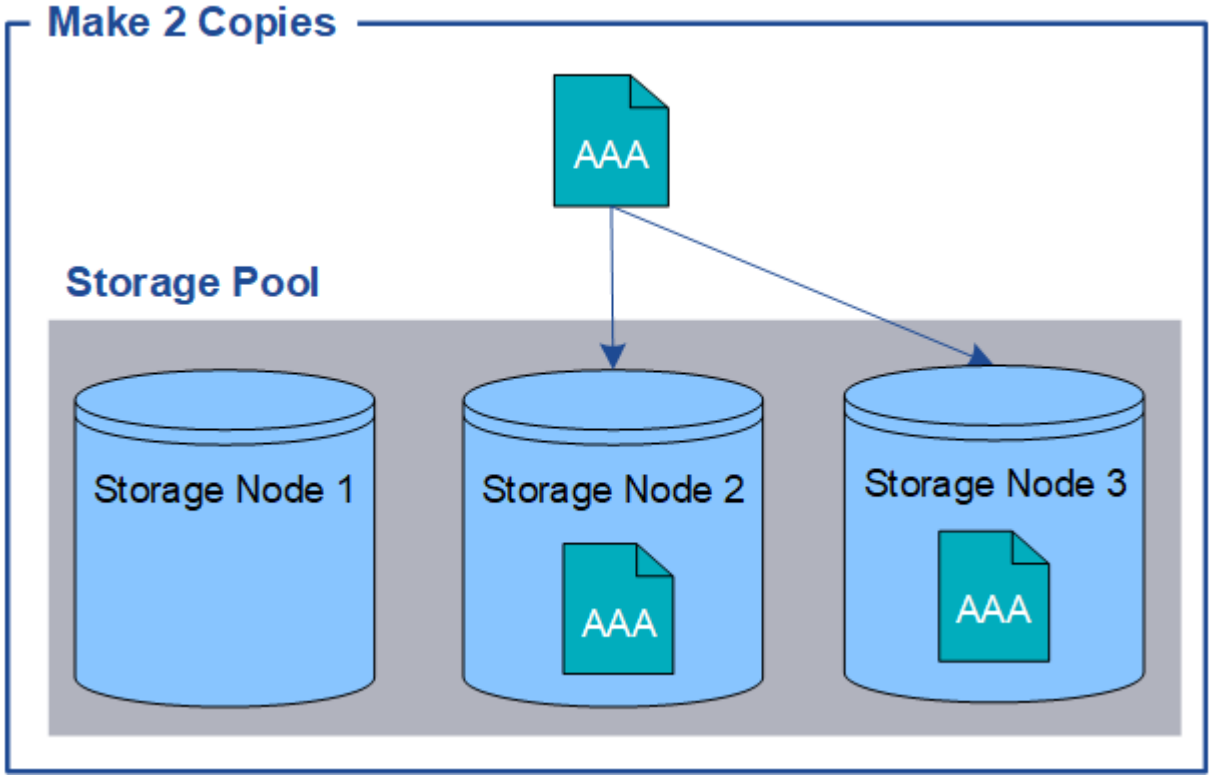
오브젝트 저장 방법(복제 또는 삭제 코딩)

복제란 무엇입니까?

복제는 StorageGRID에서 오브젝트 데이터를 저장하는 데 사용하는 두 가지 방법 중 하나입니다(삭제 코딩은 다른 방법). 오브젝트가 복제를 사용하는 ILM 규칙과 일치하면 시스템은 오브젝트 데이터의 정확한 복사본을 생성하고 복사본을 스토리지 노드에 저장합니다.

ILM 규칙을 구성하여 복제된 복사본을 생성할 때는 생성할 복사본 수, 복사본 배치 위치 및 각 위치에 복사본을 저장할 기간을 지정합니다.

다음 예제에서 ILM 규칙은 세 개의 스토리지 노드가 포함된 스토리지 풀에 각 개체의 복제된 복사본 2개를 배치하도록 지정합니다.



StorageGRID가 오브젝트를 이 규칙과 일치시키면 스토리지 풀의 다른 스토리지 노드에 각 복사본을 배치하여 객체의 복제본이 두 개 생성됩니다. 두 복제본은 세 개의 사용 가능한 스토리지 노드 중 어느 두 개에 배치될 수 있습니다. 이 경우 규칙은 스토리지 노드 2와 3에 오브젝트 복사본을 배치합니다. 두 개의 복제본이 있기 때문에 스토리지 풀의 노드 중 하나에 장애가 발생할 경우 객체를 검색할 수 있습니다.

> StorageGRID는 지정된 스토리지 노드에 복제된 객체 복제본을 하나만 저장할 수 있습니다. 그리드에 스토리지 노드 3개가 포함된 경우 4개 복사본 ILM 규칙을 생성하면 각 스토리지 노드에 대해 복사본 1개가 생성됩니다. ILM 규칙을 완전히 적용할 수 없음을 나타내기 위해 * ILM 배치 달성 안 됨 * 경고가 트리거됩니다.

관련 정보

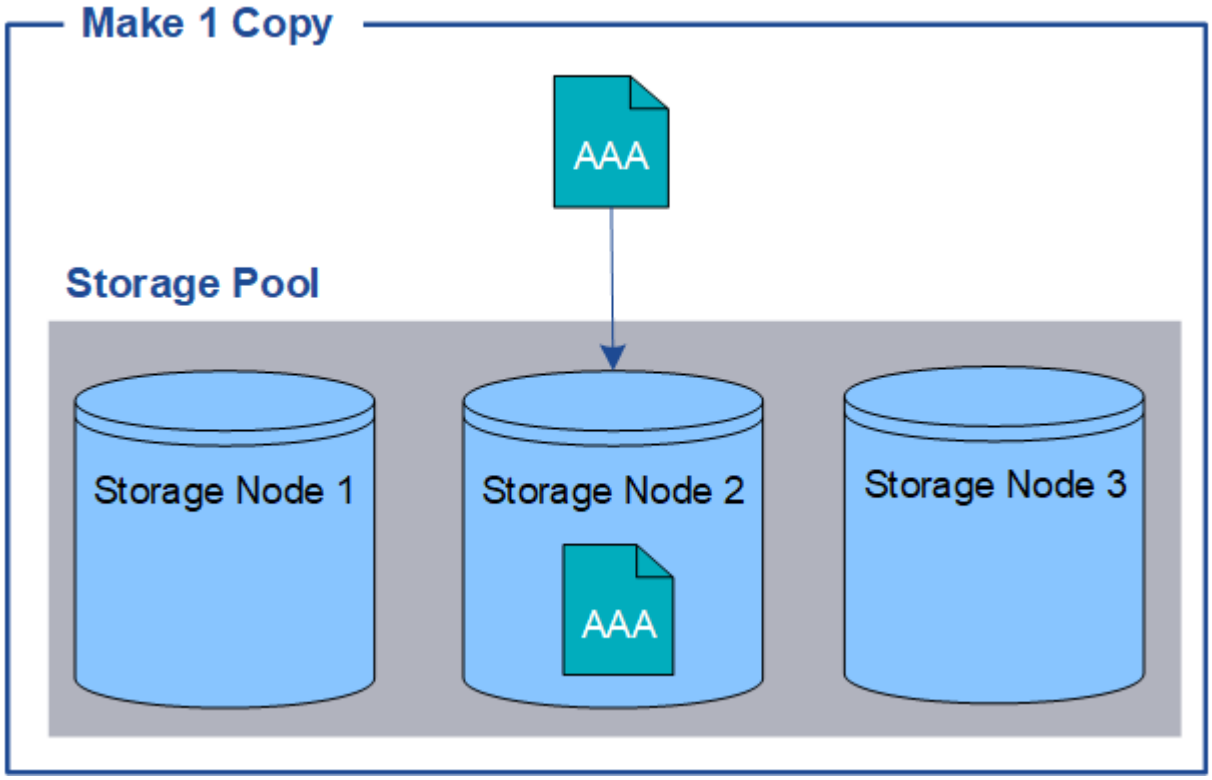
- "삭제 코딩이란 무엇입니까"
- "스토리지 풀이란 무엇입니까"
- "복제 및 삭제 코딩을 사용하여 사이트 손실을 보호합니다"

단일 복사본 복제를 사용하지 않아야 하는 이유

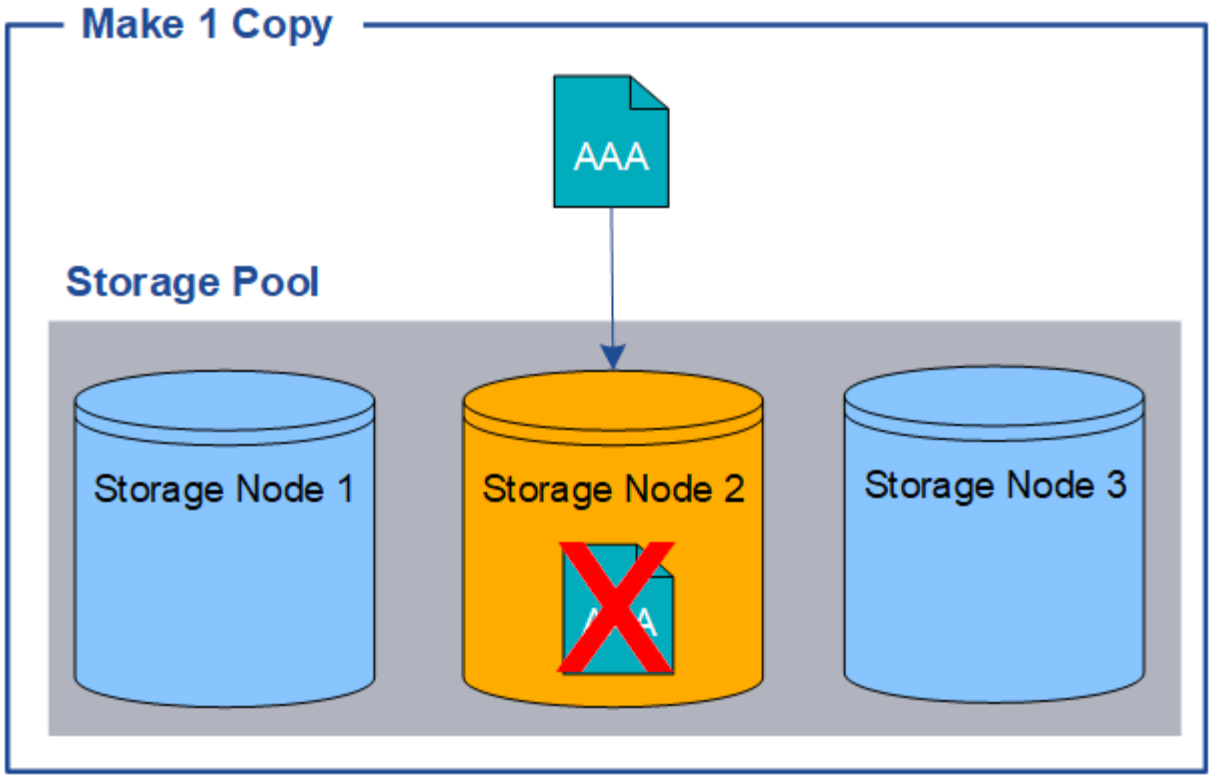
ILM 규칙을 생성하여 복제된 복사본을 만들 때는 항상 배치 지침에 두 개 이상의 복사본을 지정해야 합니다.

> 특정 기간 동안 복제된 복사본을 하나만 생성하는 ILM 규칙을 사용하지 마십시오. 복제된 객체 복제본이 하나만 있는 경우 스토리지 노드에 장애가 발생하거나 심각한 오류가 발생한 경우 해당 객체가 손실됩니다. 또한 업그레이드와 같은 유지보수 절차 중에는 개체에 대한 액세스가 일시적으로 중단됩니다.

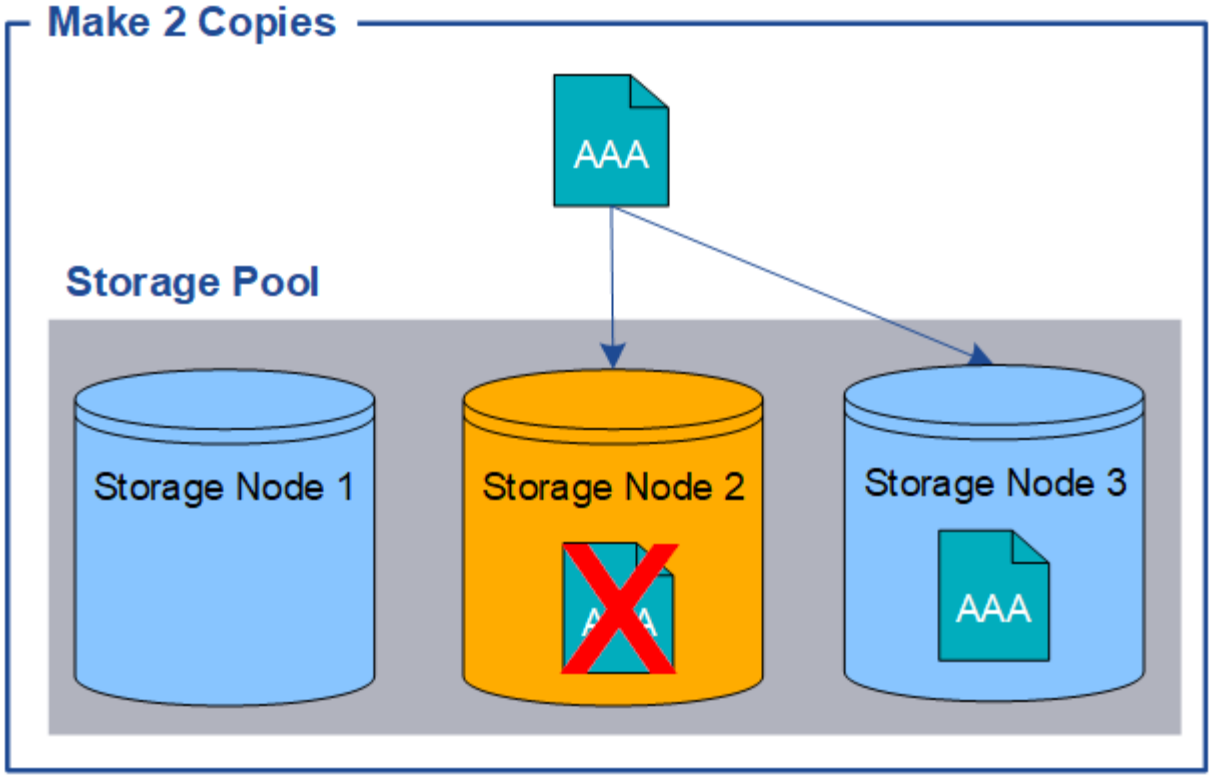
다음 예제에서 Make 1 Copy ILM 규칙은 세 개의 스토리지 노드가 포함된 스토리지 풀에 개체의 복제된 복사본 하나를 배치하도록 지정합니다. 이 규칙과 일치하는 객체가 수집되면 StorageGRID는 하나의 스토리지 노드에만 단일 복제본을 배치합니다.



ILM 규칙이 개체의 복제된 복사본을 하나만 만들면 스토리지 노드를 사용할 수 없을 때 개체에 액세스할 수 없게 됩니다. 이 예제에서는 업그레이드 또는 기타 유지 관리 절차 중에 스토리지 노드 2가 오프라인일 때마다 객체 AAA에 대한 액세스가 일시적으로 끊어집니다. 스토리지 노드 2에 장애가 발생하면 객체 AAA가 완전히 손실됩니다.



오브젝트 데이터의 손실을 방지하려면 항상 복제로 보호할 모든 오브젝트의 복사본을 두 개 이상 만들어야 합니다. 두 개 이상의 복제본이 있는 경우에도 하나의 스토리지 노드에 장애가 발생하거나 오프라인 상태가 되더라도 개체에 액세스할 수 있습니다.



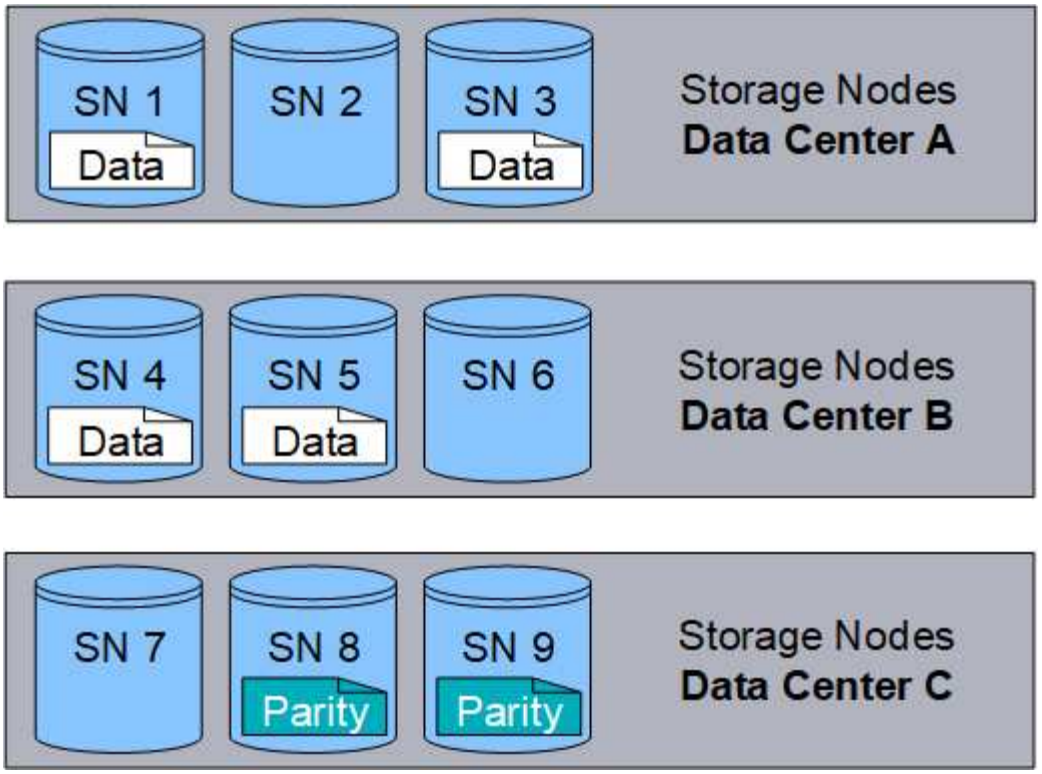
삭제 코딩이란 무엇입니까?

삭제 코딩은 StorageGRID에서 오브젝트 데이터를 저장하는 데 사용하는 두 가지 방법 중 하나입니다(복제는 다른 방법). 오브젝트가 삭제 코딩을 사용하는 ILM 규칙과 일치하면 해당 오브젝트는 데이터 조각으로 분할, 추가 패리티 조각들이 계산되고 각 조각은 다른 스토리지 노드에 저장됩니다.

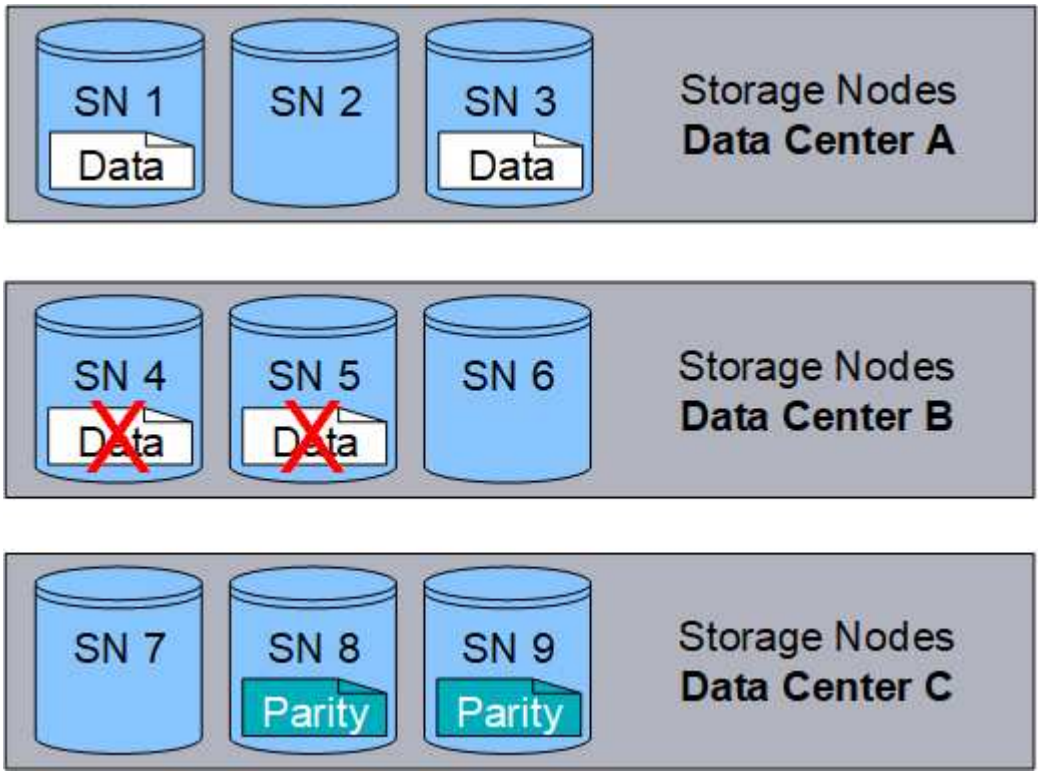
개체에 액세스하면 저장된 조각을 사용하여 다시 조립됩니다. 데이터 또는 패리티 조각이 손상되거나 손실될 경우, 삭제 코딩 알고리즘이 나머지 데이터 및 패리티 조각의 일부를 사용하여 해당 조각을 다시 생성할 수 있습니다.

ILM 규칙을 생성할 때 StorageGRID은 해당 규칙을 지원하는 삭제 코딩 프로필을 생성합니다. 삭제 코딩 프로필, "삭제 코딩 프로필의 이름을 바꿉니다"또는 의 목록을 볼 수 있습니다."삭제 코딩 프로필이 현재 ILM 규칙에 사용되지 않는 경우 비활성화합니다"

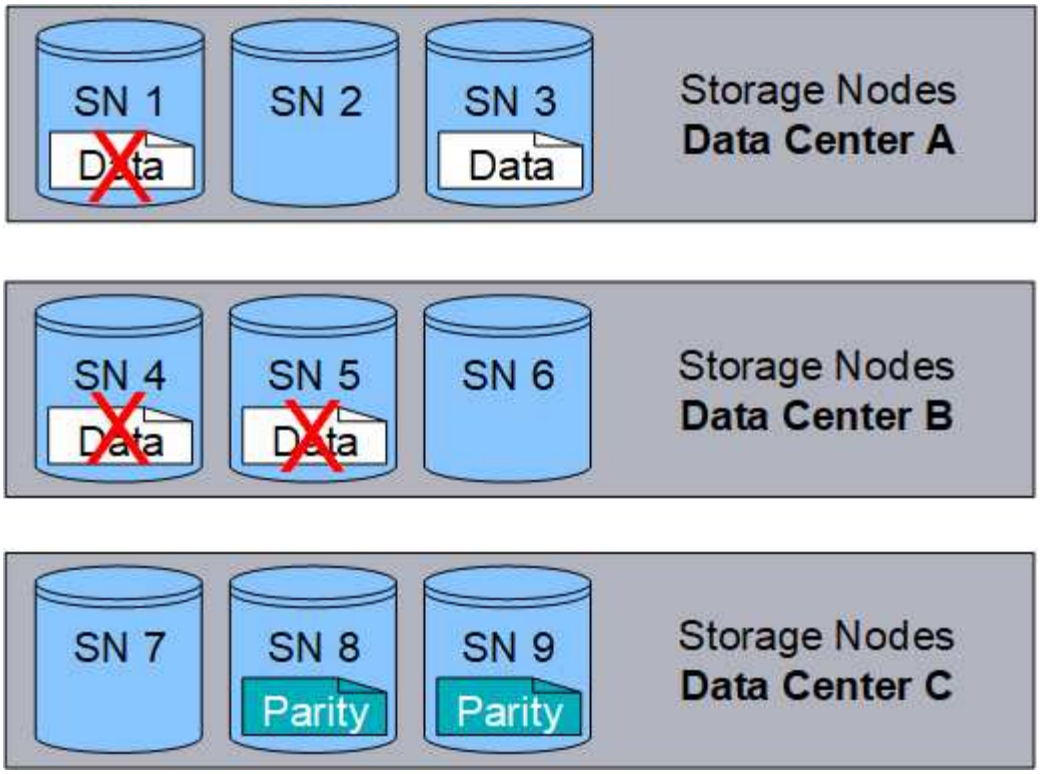
다음 예제에서는 오브젝트의 데이터에서 삭제 코딩 알고리즘을 사용하는 방법을 보여 줍니다. 이 예제에서 ILM 규칙은 4+2 삭제 코딩 체계를 사용합니다. 각 개체는 4개의 동일한 데이터 조각으로 분할되며 두 개의 패리티 조각은 개체 데이터에서 계산됩니다. 6개의 각 단편은 3개의 데이터 센터 사이트에서 서로 다른 노드에 저장되어 노드 장애 또는 사이트 손실에 대한 데이터 보호를 제공합니다.



4+2 삭제 코딩 방식은 다양한 방법으로 구성할 수 있습니다. 예를 들어 6개의 스토리지 노드가 포함된 단일 사이트 스토리지 풀을 구성할 수 있습니다. 의 경우 "사이트 손실 방지"각 사이트에 스토리지 노드 3개가 있는 사이트 3개가 포함된 스토리지 풀을 사용할 수 있습니다. 6개의 조각(데이터 또는 패리티) 중 4개를 사용할 수 있는 한 오브젝트를 검색할 수 있습니다. 개체 데이터를 손실하지 않고 최대 2개의 조각을 잃을 수 있습니다. 전체 사이트가 손실된 경우에도 다른 모든 조각에 액세스할 수 있는 한 개체를 검색하거나 복구할 수 있습니다.



두 개 이상의 스토리지 노드가 손실되면 객체를 검색할 수 없습니다.



관련 정보

- "복제란 무엇입니까"
- "스토리지 풀이란 무엇입니까"
- "삭제 코딩 체계란 무엇입니까"
- "삭제 코딩 프로필의 이름을 바꿉니다"
- "삭제 코딩 프로필을 비활성화합니다"

삭제 코딩 체계란 무엇입니까?

삭제 코딩 스키마를 통해 각 오브젝트에 대해 생성되는 데이터 조각과 패리티 조각의 수를 제어합니다.

ILM 규칙을 생성하거나 편집할 때 사용 가능한 삭제 코딩 체계를 선택합니다. 사용할 스토리지 풀을 구성하는 스토리지 노드 및 사이트의 수에 따라 StorageGRID에서 삭제 코딩 체계를 자동으로 생성합니다.

데이터 보호

StorageGRID 시스템은 Reed-Solomon 삭제 코딩 알고리즘을 사용합니다. 알고리즘은 오브젝트를 데이터 `m` 조각으로 분할하고 `k` 패리티 조각을 계산합니다.

`k + m = n`조각은 다음과 같이 데이터 보호를 제공하기 위해 스토리지 노드 전체에 분산됩니다. `n`

- 오브젝트를 검색하거나 복구하려면 `k` 조각이 필요합니다.

- 오브젝트는 손실되거나 손상된 조각까지 유지할 수 `m` 있습니다. 값이 클수록 `m` 오류 허용 오차가 커집니다.

최고의 데이터 보호는 스토리지 풀 내에서 가장 높은 노드 또는 볼륨 장애를 허용하는 삭제 코딩 체계를 통해 제공됩니다.

스토리지 오버헤드

삭제 코딩 체계의 스토리지 오버헤드는 패리티 조각의 (`m`수를 데이터 조각의 수로 나누어 (`k`계산합니다. 스토리지 오버헤드를 사용하여 각 삭제 코딩 오브젝트에 필요한 디스크 공간을 계산할 수 있습니다.

```
disk space = object size + (object size * storage overhead)
```

예를 들어, 스토리지 오버헤드가 50%인 4+2 체계를 사용하여 10MB 오브젝트를 저장할 경우 오브젝트는 15MB의 그리드 스토리지를 사용합니다. 스토리지 오버헤드가 33%인 6+2 체계를 사용하여 동일한 10MB 개체를 저장하는 경우, 개체는 약 13.3MB를 사용합니다.

귀사의 요구사항을 충족하는 의 총액이 가장 낮은 삭제 코딩 체계를 선택합니다 `k+m`. 조각 수가 적은 삭제 코딩 체계를 사용하는 것이 보다 효율적인 이유는 다음과 같습니다.

- 오브젝트당 생성 및 분산(또는 검색)되는 조각의 수가 더 적습니다
- 조각 크기가 크기 때문에 성능이 더 좋습니다
- 따라서 에 추가할 수 있는 노드 수를 줄일 수 있습니다 "추가 스토리지가 필요할 때 확장"

스토리지 풀에 대한 지침

삭제 코딩 복사본을 생성할 규칙에 사용할 스토리지 풀을 선택할 때는 스토리지 풀에 대해 다음 지침을 따르십시오.

- 스토리지 풀에는 3개 이상의 사이트 또는 정확히 하나의 사이트가 포함되어야 합니다.

  > ⓘ 스토리지 풀에 두 개의 사이트가 포함된 경우 삭제 코딩을 사용할 수 없습니다.

  - 3개 이상의 사이트가 포함된 스토리지 풀의 삭제 코딩 체계
  - 단일 사이트 스토리지 풀에 대한 삭제 코딩 구성표
- 모든 사이트 사이트를 포함하는 스토리지 풀을 사용하지 마십시오.
- 스토리지 풀에는 오브젝트 데이터를 저장할 수 있는 스토리지 노드 이상이 `k+m +1` 포함되어야 합니다.

  > ⓘ 스토리지 노드는 설치 중에 오브젝트 데이터가 아닌 오브젝트 메타데이터만 포함하도록 구성할 수 있습니다. 자세한 내용은 을 "스토리지 노드 유형"참조하십시오.

  필요한 최소 스토리지 노드 수는 입니다 `k+m`. 그러나 필요한 스토리지 노드를 일시적으로 사용할 수 없는 경우 하나 이상의 추가 스토리지 노드를 사용하면 수집 실패 또는 ILM 백로그를 방지할 수 있습니다.

3개 이상의 사이트가 포함된 스토리지 풀의 삭제 코딩 체계

다음 표에서는 3개 이상의 사이트가 포함된 스토리지 풀에 대해 StorageGRID에서 현재 지원하는 삭제 코딩 스키마를 설명합니다. 이러한 모든 스키마를 통해 사이트 손실을 보호할 수 있습니다. 한 사이트는 손실될 수 있으며 개체는 계속 액세스할 수 있습니다.

사이트 손실 보호를 제공하는 삭제 코딩 구성의 경우 각 사이트에는 최소 3개의 스토리지 노드가 필요하므로 스토리지 풀에서 권장 스토리지 노드 수가 을 초과합니다 `k+m +1`.

삭제 코딩 체계 (*k+m*)	배포된 사이트의 최소 수입니다	각 사이트에 권장되는 스토리지 노드 수입니다	총 권장 스토리지 노드 수입니다	사이트 손실 방지	스토리지 오버헤드
4 + +2	3	3	9	예	50%
6 + +2	4	3	12	예	33%
8 + +2	5	3	15	예	25%
6 + 3	3	4	12	예	50%
9 + 3	4	4	16	예	33%
2 + +1	3	3	9	예	50%
4 + +1	5	3	15	예	25%
6 + +1	7	3	21	예	17%
7 + +5	3	5	15	예	71%

StorageGRID에는 사이트당 최소 3개의 스토리지 노드가 필요합니다. 7+5 스키마를 사용하려면 각 사이트에 최소 4개의 스토리지 노드가 필요합니다. 사이트당 5개의 스토리지 노드를 사용하는 것이 좋습니다.

사이트 보호를 제공하는 삭제 코딩 스키마를 선택할 때는 다음 요소의 상대적 중요도를 균형 있게 조정합니다.

- * 조각 수 *: 전체 조각 수가 적으면 성능과 확장 유연성이 일반적으로 더 좋습니다.
- * 내결함성 *: 패리티 세그먼트가 많을수록 내결함성(즉, 값이 더 높은 경우 `m`)이 증가합니다.
- * 네트워크 트래픽 * : 실패에서 복구 할 때 더 많은 조각이 있는 체계(즉, 더 높은 총계 `k+m`)를 사용하면 더 많은 네트워크 트래픽이 생성됩니다.
- * 스토리지 오버헤드 *: 오버헤드가 높은 구성일수록 오브젝트당 스토리지 공간이 더 필요합니다.

예를 들어, 4+2 체계와 6+3 체계(둘 다 50%의 스토리지 오버헤드를 가짐) 중에서 결정할 때 추가 내결함성을 필요로 하는 경우 6+3 체계를 선택합니다. 네트워크 리소스가 제한된 경우 4+2 구성표를 선택합니다. 다른 모든 요소가 같으면 총 단편 수가 더 낮기 때문에 4+2를 선택합니다.

사용할 체계가 확실하지 않으면 4+2 또는 6+3을 선택하거나 기술 지원 부서에 문의하십시오.

단일 사이트 스토리지 풀에 대한 삭제 코딩 구성표

사이트에 충분한 스토리지 노드가 있는 경우 한 사이트 스토리지 풀은 세 개 이상의 사이트에 대해 정의된 모든 삭제

코딩 스키마를 지원합니다.

필요한 최소 스토리지 노드 수는 `k+m` 이지만 스토리지 노드가 있는 스토리지 풀을 사용하는 `k+m +1` 것이 좋습니다. 예를 들어, 2+1 삭제 코딩 구성표에 최소 3개의 스토리지 노드가 있는 스토리지 풀이 필요하지만 4개의 스토리지 노드를 사용하는 것이 좋습니다.

삭제 코딩 체계(*k+m*)	최소 스토리지 노드 수입니다	권장되는 스토리지 노드 수입니다	스토리지 오버헤드
4 + +2	6	7	50%
6 + +2	8	9	33%
8 + +2	10	11	25%
6 + 3	9	10	50%
9 + 3	12	13	33%
2 + +1	3	4	50%
4 + +1	5	6	25%
6 + +1	7	8	17%
7 + +5	12	13	71%

삭제 코딩의 장점, 단점 및 요구 사항

오브젝트 데이터의 손실로부터 보호하기 위해 복제 또는 삭제 코딩을 사용할지 결정하기 전에 삭제 코딩의 장점, 단점 및 요구 사항을 이해해야 합니다.

삭제 코딩의 장점

삭제 코딩은 복제와 비교할 때 안정성, 가용성 및 스토리지 효율성을 향상시킵니다.

- * 안정성 *: 신뢰성은 내결함성의 관점에서 측정되며, 즉 데이터 손실 없이 동시에 장애가 발생할 수 있는 횟수를 나타냅니다. 복제를 사용하면 동일한 여러 복사본이 여러 노드와 사이트 전체에 저장됩니다. 삭제 코딩을 사용하면 오브젝트는 데이터 및 패리티 조각으로 인코딩되어 여러 노드와 사이트에 분산됩니다. 이 분산은 사이트 및 노드 장애 보호를 모두 제공합니다. 복제와 비교할 때 삭제 코딩은 비슷한 스토리지 비용으로 향상된 안정성을 제공합니다.
- * 가용성 *: 스토리지 노드에 장애가 발생하거나 액세스할 수 없는 경우 객체를 검색하는 기능으로 가용성을 정의할 수 있습니다. 복제와 비교할 때 삭제 코딩은 비슷한 스토리지 비용으로 향상된 가용성을 제공합니다.
- * 스토리지 효율성 *: 유사한 수준의 가용성과 안정성을 위해 삭제 코딩을 통해 보호되는 오브젝트는 복제를 통해 보호될 경우 동일한 오브젝트보다 더 적은 디스크 공간을 사용합니다. 예를 들어, 두 사이트에 복제된 10MB 개체는 20MB의 디스크 공간(복사본 2개)을 소비하고, 6+3 삭제 코딩 체계를 사용하여 세 사이트에서 삭제 코딩된 개체는 15MB의 디스크 공간만 소비합니다.

삭제 코딩 오브젝트를 위한 디스크 공간은 오브젝트 크기와 스토리지 오버헤드로 계산됩니다. 스토리지 오버헤드 비율은 패리티 조각 수를 데이터 조각 수로 나눈 값입니다.

삭제 코딩의 단점

복제와 비교할 때 삭제 코딩에는 다음과 같은 단점이 있습니다.

- 삭제 코딩 체계에 따라 스토리지 노드 및 사이트의 수를 늘리는 것이 좋습니다. 반면, 오브젝트 데이터를 복제할 경우 각 복제본마다 스토리지 노드가 하나만 필요합니다. "3개 이상의 사이트가 포함된 스토리지 풀의 삭제 코딩 체계"및 을 "단일 사이트 스토리지 풀에 대한 삭제 코딩 구성표"참조하십시오.
- 스토리지 확장의 비용 및 복잡성 증가 복제를 사용하는 배포를 확장하려면 개체 복사본이 만들어지는 모든 위치에 스토리지 용량을 추가해야 합니다. 삭제 코딩을 사용하는 배포를 확장하려면 사용 중인 삭제 코딩 체계와 기존 스토리지 노드의 전체 용량을 고려해야 합니다. 예를 들어, 기존 노드가 100%로 꽉 찰 때까지 기다린 경우 스토리지 노드를 하나 이상 추가해야 `k+m` 하지만, 기존 노드가 70% 차 있을 때 확장하는 경우 사이트당 2개의 노드를 추가하여 사용 가능한 스토리지 용량을 최대화할 수 있습니다. 자세한 내용은 을 "삭제 코딩 오브젝트를 위한 스토리지 용량을 추가합니다"참조하십시오.
- 지리적으로 분산된 사이트에서 삭제 코딩을 사용하면 검색 지연 시간이 늘어납니다. 삭제 코딩되어 원격 사이트에 배포된 오브젝트의 오브젝트 조각은 복제되고 로컬에서 사용 가능한 오브젝트(클라이언트가 연결하는 동일한 사이트)에 비해 WAN 연결을 통해 검색하는 데 시간이 더 오래 걸립니다.
- 지리적으로 분산된 사이트에서 삭제 코딩을 사용하는 경우 검색 및 복구를 위해 WAN 네트워크 트래픽 사용량이 증가하고, 특히 자주 검색하는 오브젝트 또는 WAN 네트워크 연결을 통한 오브젝트 복구에서 더욱 그렇습니다.
- 여러 사이트에서 삭제 코딩을 사용하면 사이트 간의 네트워크 지연 시간이 증가함에 따라 최대 오브젝트 처리량이 급격히 줄어듭니다. 이러한 감소는 StorageGRID 시스템이 개체 조각을 저장하고 검색하는 데 영향을 미치는 TCP 네트워크 처리량이 감소하기 때문입니다.
- 컴퓨팅 리소스 사용량 증가.

삭제 코딩 사용 시기

삭제 코딩은 다음 요구사항에 가장 적합합니다.

- 크기가 1MB를 초과하는 객체

삭제 코딩은 1MB 이상의 오브젝트에 가장 적합합니다. 매우 작은 삭제 코딩 조각을 관리해야 하는 오버헤드를 방지하기 위해 200KB 미만의 오브젝트에 삭제 코딩을 사용하지 마십시오.

- 자주 검색되지 않는 콘텐츠의 장기 또는 콜드 스토리지
- 높은 데이터 가용성 및 안정성
- 전체 사이트 및 노드 장애로부터 보호
- 스토리지 효율성:
- 여러 개의 복제된 복사본이 아닌 하나의 삭제 코딩 복사본만으로 효율적인 데이터 보호가 필요한 단일 사이트 배포
- 사이트 간 지연 시간이 100ms 미만인 다중 사이트 구축

개체 보존이 결정되는 방식

StorageGRID는 그리드 관리자와 개별 테넌트 사용자 모두에게 개체 저장 기간을 지정할 수

있는 옵션을 제공합니다. 일반적으로 테넌트 사용자가 제공한 보존 지침은 그리드 관리자가 제공한 보존 지침보다 우선합니다.

테넌트 사용자가 객체 보존을 제어하는 방식

테넌트 사용자는 다음 방법을 사용하여 개체가 StorageGRID에 저장되는 기간을 제어할 수 있습니다.

- 그리드에 대해 글로벌 S3 오브젝트 잠금 설정이 활성화된 경우 S3 테넌트 사용자는 S3 오브젝트 잠금이 활성화된 상태로 버킷을 생성한 다음 각 버킷에 대해 * 기본 보존 기간 * 을 선택할 수 있습니다.
- 그리드에 글로벌 S3 오브젝트 잠금 설정이 활성화된 경우 S3 테넌트 사용자는 S3 오브젝트 잠금이 활성화된 버킷을 생성한 다음 S3 REST API를 사용하여 해당 버킷에 추가된 각 오브젝트 버전에 대한 보관 기한 및 법적 보류 설정을 지정할 수 있습니다.
  - 법적 증거 자료 보관 중인 개체 버전은 어떤 방법으로도 삭제할 수 없습니다.
  - 개체 버전의 보존 기한 에 도달하기 전에 어떤 방법으로도 해당 버전을 삭제할 수 없습니다.
  - S3 오브젝트 잠금이 설정된 버킷의 오브젝트는 ILM이 "영구"로 유지합니다. 그러나 보존 기한에 도달한 후에는 클라이언트 요청 또는 버킷 라이프사이클의 만료에 의해 오브젝트 버전을 삭제할 수 있습니다. 을 "S3 오브젝트 잠금으로 오브젝트 관리"참조하십시오.
- S3 테넌트 사용자는 만료 작업을 지정하는 버킷에 라이프사이클 구성을 추가할 수 있습니다. 버킷 라이프사이클이 있는 경우 StorageGRID는 클라이언트가 먼저 오브젝트를 삭제하지 않는 한 만료 작업에 지정된 날짜 또는 일 수가 충족될 때까지 오브젝트를 저장합니다. 을 "S3 라이프사이클 구성을 생성합니다"참조하십시오.
- S3 클라이언트에서 객체 삭제 요청을 실행할 수 있습니다. StorageGRID는 항상 S3 버킷 라이프사이클 또는 ILM을 통해 클라이언트 삭제 요청의 우선순위를 지정하고 오브젝트를 삭제 또는 보존 할 것인지 결정합니다.

그리드 관리자가 개체 보존을 제어하는 방법

그리드 관리자는 다음 방법을 사용하여 객체 보존을 제어할 수 있습니다.

- 각 테넌트의 S3 오브젝트 잠금 최대 보존 기간을 설정합니다. 그런 다음 테넌트 사용자는 각 버킷에 대해 기본 보존 기간을 설정할 수 있습니다. 최대 보존 기간은 해당 버킷에 대해 새로 수집된 객체(객체의 유지 종료 날짜)에도 적용됩니다.
- ILM 배치 지침을 생성하여 개체 저장 기간을 제어합니다. ILM 규칙에 따라 오브젝트가 일치하는 경우 StorageGRID는 ILM 규칙의 마지막 기간이 경과할 때까지 해당 오브젝트를 저장합니다. 배치 명령에 "영구"가 지정된 경우 객체는 무기한 유지됩니다.
- 누가 오브젝트 보존 기간을 제어하든 ILM 설정은 저장되는 오브젝트 복사본(복제 또는 삭제 코딩)과 복사본의 위치(스토리지 노드 또는 클라우드 스토리지 풀)를 제어합니다.

S3 버킷 수명 주기와 **ILM**이 상호 작용하는 방식

S3 버킷 라이프사이클이 구성된 경우 라이프사이클 만료 작업이 라이프사이클 필터와 일치하는 오브젝트에 대한 ILM 정책을 재정의합니다. 따라서 개체를 배치하기 위한 ILM 명령이 만료된 후에도 개체가 그리드에 유지될 수 있습니다.

오브젝트 보존의 예

S3 오브젝트 잠금, 버킷 수명 주기 설정, 클라이언트 삭제 요청 및 ILM 간의 상호 작용을 더 잘 이해하려면 다음 예제를 고려해 보십시오.

예 **1: S3** 버킷 수명 주기는 **ILM**보다 개체를 더 오래 유지합니다

ILM을 참조하십시오

1년(365일) 동안 2부 보관

버킷 수명 주기

2년 후 개체 만료(730일)

결과

StorageGRID 는 개체를 730일 동안 저장합니다. StorageGRID는 버킷 수명 주기 설정을 사용하여 오브젝트를 삭제 또는 유지할지 여부를 결정합니다.

> ⓘ 버킷 라이프사이클에서 ILM에서 지정한 것보다 더 오래 개체를 유지해야 한다고 지정하는 경우 StorageGRID는 저장할 복사본의 수와 유형을 결정할 때 ILM 배치 지침을 계속 사용합니다. 이 예제에서는 두 개의 개체 복사본이 StorageGRID에 계속 저장됩니다. 이 기간은 366일에서 730일입니다.

예 **2: S3** 버킷 라이프사이클이 **ILM** 전에 오브젝트를 만기합니다

ILM을 참조하십시오

2년(730일) 동안 2부 보관

버킷 수명 주기

1년 후 개체 만료(365일)

결과

StorageGRID에서는 365일 이후에 개체의 복사본을 모두 삭제합니다.

예 **3:** 클라이언트 삭제는 버킷 수명 주기와 **ILM**을 재정의합니다

ILM을 참조하십시오

스토리지 노드에 "영구" 복사본 2개 저장

버킷 수명 주기

2년 후 개체 만료(730일)

클라이언트 삭제 요청

400일째 발행

결과

StorageGRID는 클라이언트 삭제 요청에 대한 응답으로 400일째에 두 객체 복제본을 모두 삭제합니다.

예 **4: S3** 오브젝트 잠금이 클라이언트 삭제 요청을 재정의합니다

S3 오브젝트 잠금

개체 버전에 대한 보존 기한은 2026-03-31입니다. 법적 증거 자료 보관 은 적용되지 않습니다.

ILM 규칙 준수

스토리지 노드에 "영구" 복사본 2개 저장

클라이언트 삭제 요청

2024-03-31일에 발행되었습니다

결과

보존 기한이 2년 남지 않았으므로 StorageGRID는 개체 버전을 삭제하지 않습니다.

오브젝트 삭제 방법

StorageGRID는 클라이언트 요청에 직접 응답하거나 S3 버킷 라이프사이클의 만료 또는 ILM 정책 요구사항으로 인해 자동으로 오브젝트를 삭제할 수 있습니다. 개체를 삭제할 수 있는 다양한 방법과 StorageGRID에서 삭제 요청을 처리하는 방법을 이해하면 개체를 보다 효율적으로 관리할 수 있습니다.

StorageGRID는 다음 두 가지 방법 중 하나를 사용하여 오브젝트를 삭제할 수 있습니다.

- 동기 삭제: StorageGRID가 클라이언트 삭제 요청을 받으면 모든 개체 복사본이 즉시 제거됩니다. 복제본이 제거된 후 성공적으로 삭제되었다는 메시지가 클라이언트에 표시됩니다.
- 객체가 삭제 대기열에 저장됨: StorageGRID에서 삭제 요청을 수신하면 객체가 삭제 대기열에 추가되고 삭제 성공 사실을 즉시 클라이언트에 알립니다. 개체 복사본은 나중에 백그라운드 ILM 처리에 의해 제거됩니다.

오브젝트를 삭제할 때 StorageGRID는 삭제 성능을 최적화하고 잠재적인 삭제 백로그를 최소화하며 공간을 가장 빠르게 확보하는 방법을 사용합니다.

이 표에는 StorageGRID에서 각 방법을 사용하는 경우가 요약되어 있습니다.

삭제 수행 방법	사용 시
오브젝트는 삭제 대기열에 추가됩니다	다음 조건 중 * 어느 * 가 참일 경우: • 자동 개체 삭제는 다음 이벤트 중 하나에 의해 트리거되었습니다. ◦ S3 버킷에 대한 라이프사이클 구성의 만료 날짜 또는 일 수에 도달했습니다. ◦ ILM 규칙에 지정된 마지막 기간이 경과됩니다. • 참고: * S3 오브젝트 잠금이 활성화된 버킷의 오브젝트는 법적 보류 중이거나 보존 기한이 지정되었지만 아직 충족되지 않은 경우 삭제할 수 없습니다. • S3 클라이언트가 삭제를 요청하는데 다음 조건 중 하나 이상이 참입니다. ◦ 예를 들어, 개체 위치를 일시적으로 사용할 수 없기 때문에 복사본을 30초 이내에 삭제할 수 없습니다. ◦ 백그라운드 삭제 대기열은 유휴 상태입니다.

삭제 수행 방법	사용 시
객체가 즉시 제거됩니다 (동기식 삭제).	S3 클라이언트가 삭제 요청을 하고 다음 조건 중 * 모든 * 가 충족되는 경우: • 모든 사본은 30초 이내에 제거할 수 있습니다. • 백그라운드 삭제 대기열에는 처리할 객체가 포함됩니다.

S3 클라이언트가 삭제 요청을 하면 StorageGRID는 삭제 큐에 개체를 추가하는 것으로 시작합니다. 그런 다음 동기식 삭제 수행 으로 전환됩니다. 백그라운드 삭제 큐에 처리할 개체가 있는지 확인함으로써 StorageGRID는 특히 동시 접속 수가 적은 클라이언트의 경우 삭제 작업을 보다 효율적으로 처리할 수 있으며 클라이언트 삭제 백로그를 방지할 수 있습니다.

객체를 삭제하는 데 필요한 시간입니다

StorageGRID에서 객체를 삭제하는 방법은 시스템이 수행하는 방식에 영향을 미칠 수 있습니다.

- StorageGRID가 동기 삭제를 수행할 때 결과를 클라이언트에 반환하는 데 StorageGRID가 최대 30초가 걸릴 수 있습니다. 즉, StorageGRID에서 삭제할 개체를 큐에 대기할 때보다 복사본이 실제로 더 빠르게 제거되더라도 삭제가 더 느리게 진행되는 것처럼 보일 수 있습니다.
- 대량 삭제 중에 삭제 성능을 면밀히 모니터링하는 경우 특정 수의 개체를 삭제한 후 삭제 속도가 느린 것으로 보일 수 있습니다. 이 변경은 StorageGRID가 삭제를 위해 오브젝트 큐잉에서 동기식 삭제 수행으로 변경될 때 발생합니다. 삭제 속도가 명백히 감소하는 것은 오브젝트 복사본이 더 느리게 제거된다는 의미가 아닙니다. 반면, 공간은 평균적으로 더 빠르게 확보되고 있음을 나타냅니다.

많은 수의 개체를 삭제하는 경우 우선 순위가 공간을 빠르게 확보하는 것이라면 클라이언트 요청을 사용하여 ILM 또는 다른 방법을 사용하여 개체를 삭제하지 않고 개체를 삭제하는 것이 좋습니다. 일반적으로 StorageGRID에서는 동기 삭제를 사용할 수 있으므로 클라이언트에서 삭제할 때 공간이 더 빠르게 확보됩니다.

개체를 삭제한 후 공간을 확보하는 데 필요한 시간은 다음과 같은 여러 요소에 따라 달라집니다.

- 오브젝트 복사본이 동기식으로 제거되는지, 아니면 나중에 제거를 위해 대기하는지(클라이언트 삭제 요청) 여부를 나타냅니다.
- 클라이언트 삭제 및 기타 방법 모두에 대해 개체 복사본이 제거용으로 대기될 때 그리드 내의 개체 수 또는 그리드 리소스의 사용 가능성 등의 기타 요소

S3 버전 오브젝트 삭제 방법

S3 버킷에 대해 버전 관리가 활성화된 경우 StorageGRID는 삭제 요청에 응답할 때 Amazon S3 동작을 따릅니다. 이러한 요청이 S3 클라이언트에서 온 것인지, S3 버킷 라이프사이클의 만료 또는 ILM 정책 요구사항이 있는지 여부에 관계없이 이 동작을 따릅니다.

오브젝트 버전이 지정된 경우 오브젝트 삭제 요청은 오브젝트의 현재 버전을 삭제하지 않고 공간을 확보하지 않습니다. 대신 개체 삭제 요청은 개체의 현재 버전으로 0바이트 삭제 마커를 만들어서 이전 버전의 개체를 "비최신"으로 만듭니다. 개체 삭제 표식은 현재 버전이고 현재 버전이 아닌 경우 만료된 개체 삭제 표식이 됩니다.

개체가 제거되지 않았더라도 StorageGRID 는 개체의 현재 버전을 더 이상 사용할 수 없는 것처럼 동작합니다. 해당 개체에 대한 요청은 404 NotFound 를 반환합니다. 그러나 현재 개체 데이터가 제거되지 않았으므로 개체의 현재 버전이 아닌 버전을 지정하는 요청은 성공할 수 있습니다.

버전 지정된 개체를 삭제할 때 공간을 확보하거나 삭제 표식을 제거하려면 다음 중 하나를 사용합니다.

- * S3 클라이언트 요청 *: S3 오브젝트 삭제 요청에서 객체 버전 ID를 (`DELETE /object?versionId=ID` 지정합니다.) 이 요청은 지정된 버전의 오브젝트 복사본만 제거합니다(다른 버전은 계속 공간을 소모함).
- * 버킷 수명 주기 * : 버킷 수명 주기 `NoncurrentVersionExpiration` 구성에 작업을 사용합니다. 지정된 NoncurrentDays 수가 충족되면 StorageGRID에서 현재 버전이 아닌 개체 버전의 모든 복사본을 영구적으로 제거합니다. 이러한 개체 버전은 복구할 수 없습니다.

  `NewerNoncurrentVersions`버킷 수명 주기 구성의 작업은 버전 `S3` 버킷에 보존되는 비최신 버전 수를 지정합니다. 지정된 것보다 더 많은 비최신 버전이 있으면 `NewerNoncurrentVersions` StorageGRID는 `NoncurrentDays` 값이 경과되었을 때 이전 버전을 제거합니다. `NewerNoncurrentVersions`임계값은 `ILM`에서 제공하는 수명주기 규칙을 재정의합니다. 즉, `ILM`이 삭제를 요청할 경우 임계값 내에 버전이 있는 현재 개체가 `NewerNoncurrentVersions` 보존됩니다.

  만료된 개체 삭제 표식을 제거하려면 `Expiration`,, `Days` 또는 `Date` 태그 중 하나와 함께 작업을 `ExpiredObjectDeleteMarker` 사용합니다.

- * ILM *"활성 정책의 클론을 생성합니다": 새 정책에 두 가지 ILM 규칙을 추가합니다.
  - 첫 번째 규칙: "비현재 시간"을 참조 시간으로 사용하여 객체의 현재 버전과 일치시킵니다. 에서 "ILM 규칙 생성 마법사의 1단계(세부 정보 입력)""버전 관리가 활성화된 S3 버킷의 이전 개체 버전에만 이 규칙을 적용하시겠습니까?"라는 질문에 대해 * 예 * 를 선택합니다.
  - 두 번째 규칙: * Ingest Time * 을 사용하여 현재 버전과 일치시킵니다. "비현재 시간" 규칙은 * Ingest Time * 규칙 위의 정책에 나타나야 합니다.

    만료된 오브젝트 삭제 마커를 제거하려면 * Ingest Time * 규칙을 사용하여 현재 삭제 마커와 일치시킵니다. 삭제 표시자는 * 시간 간격 * / * 일 * 이 경과하고 현재 삭제 작성기가 만료되었을 때만 제거됩니다(최신 버전이 아님).

- * 버킷에서 오브젝트 삭제 *: "모든 개체 버전을 삭제합니다"버킷에서 삭제 마커를 포함하여 테넌트 관리자를 사용합니다.

버전이 지정된 개체가 삭제되면 StorageGRID는 개체의 현재 버전으로 0바이트 삭제 표식을 만듭니다. 버전이 지정된 버킷을 삭제하려면 먼저 모든 오브젝트 및 삭제 마커를 제거해야 합니다.

- StorageGRID 11.7 이하 버전에서 생성된 삭제 표식은 S3 클라이언트 요청을 통해서만 제거할 수 있으며, ILM, 버킷 라이프사이클 규칙에 의해 제거되거나 버킷 작업의 오브젝트 삭제 에 의해 제거되지 않습니다.
- StorageGRID 11.8 이상에서 생성된 버킷의 삭제 마커는 ILM, 버킷 라이프사이클 규칙, 버킷 작업의 오브젝트 삭제 또는 명시적 S3 클라이언트 삭제로 제거할 수 있습니다.

관련 정보

- "S3 REST API 사용"
- "예 4: S3 버전 오브젝트에 대한 ILM 규칙 및 정책"