Tiering

Enterprise applications

NetApp
April 25, 2024

Table of Contents

Tiering

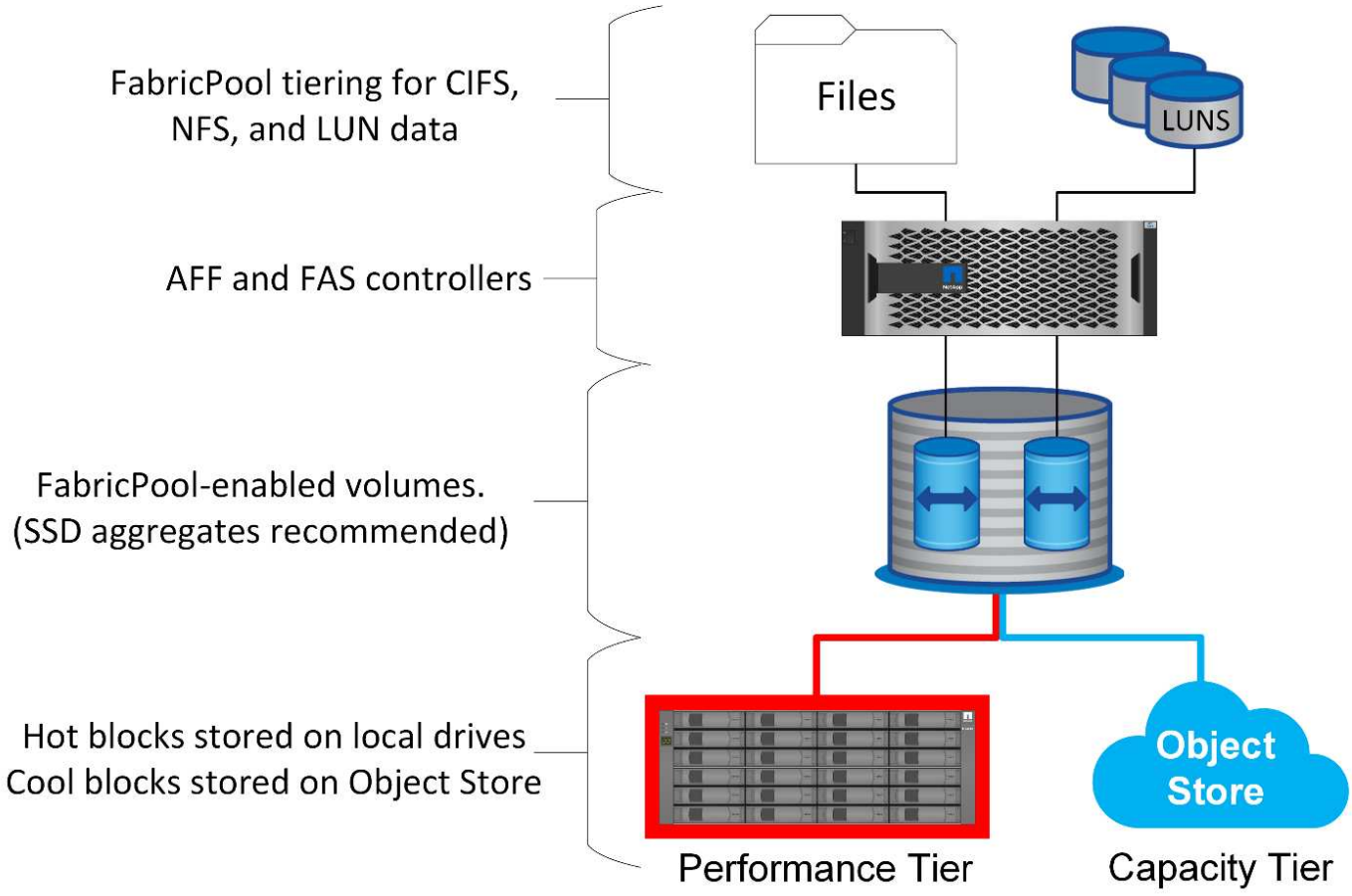
Oracle database FabricPool tiering overview

Understanding how FabricPool tiering affects Oracle and other databases requires an understanding of low-level FabricPool architecture.

Architecture

FabricPool is a tiering technology that classifies blocks as hot or cool and places them in the most appropriate tier of storage. The performance tier is most often located on SSD storage and hosts the hot data blocks. The capacity tier is located on an object store and hosts the cool data blocks. Object storage support includes NetApp StorageGRID, ONTAP S3, Microsoft Azure Blob storage, Alibaba Cloud Object Storage service, IBM Cloud Object Storage, Google Cloud storage, and Amazon AWS S3.

Multiple tiering policies are available that control how blocks are classified as hot or cool, and policies can be set on a per-volume basis and changed as required. Only the data blocks are moved between the performance and capacity tiers. The metadata that defines the LUN and file system structure always remains on the performance tier. As a result, management is centralized on ONTAP. Files and LUNs appear no different from data stored on any other ONTAP configuration. The NetApp AFF or FAS controller applies the defined policies to move data to the appropriate tier.



Object store providers

Object storage protocols use simple HTTP or HTTPS requests for storing large numbers of data objects. Access to the object storage must be reliable, because data access from ONTAP depends on prompt servicing of requests. Options include the Amazon S3 Standard and Infrequent Access options, and Microsoft Azure Hot and Cool Blob Storage, IBM Cloud, and Google Cloud. Archival options such as Amazon Glacier and Amazon Archive are not supported because the time required to retrieve data can exceed the tolerances of host operating systems and applications.

NetApp StorageGRID is also supported and is an optimal enterprise-class solution. It is a high-performance, scalable, and highly secure object storage system that can provide geographic redundancy for FabricPool data as well as other object store applications that are increasingly likely to be part of enterprise application environments.

StorageGRID can also reduce costs by avoiding the egress charges imposed by many public cloud providers for reading data back from their services.

Data and metadata

Note that the term "data" here applies to the actual data blocks, not the metadata. Only data blocks are tiered, while metadata remains in the performance tier. In addition, the status of a block as hot or cool is only affected by reading the actual data block. Simply reading the name, timestamp, or ownership metadata of a file does not affect the location of the underlying data blocks.

Backups

Although FabricPool can significantly reduce storage footprints, it is not by itself a backup solution. NetApp WAFL metadata always stays on the performance tier. If a catastrophic disaster destroys the performance tier, a new environment cannot be created using the data on the capacity tier because it contains no WAFL metadata.

FabricPool can, however, become part of a backup strategy. For example, FabricPool can be configured with NetApp SnapMirror replication technology. Each half of the mirror can have its own connection to an object storage target. The result is two independent copies of the data. The primary copy consists of the blocks on the performance tier and associated blocks in the capacity tier, and the replica is a second set of performance and capacity blocks.

Tiering policies

Oracle database FabricPool tiering policies

Four policies are available in ONTAP which control how Oracle data on the performance tier become a candidate to be relocated to the capacity tier.

Snapshot-only

The `snapshot-only tiering-policy` applies only to blocks that are not shared with the active file system. It essentially results in tiering of database backups. Blocks become candidates for tiering after a snapshot is created and the block is then overwritten, resulting in a block that exists only within the snapshot. The delay before a `snapshot-only` block is considered cool is controlled by the `tiering-minimum-cooling-days` setting for the volume. The range as of ONTAP 9.8 is from 2 to 183 days.

Many datasets have low change rates, resulting in minimal savings from this policy. For example, a typical

database observed on ONTAP has a change rate of less than 5% per week. Database archive logs can occupy extensive space, but they usually continue to exist in the active file system and thus would not be candidates for tiering under this policy.

Auto

The `auto` tiering policy extends tiering to both snapshot-specific blocks as well as blocks within the active file system. The delay before a block is considered cool is controlled by the `tiering-minimum-cooling-days` setting for the volume. The range as of ONTAP 9.8 is from 2 to 183 days.

This approach enables tiering options that are not available with the `snapshot-only` policy. For example, a data protection policy might require 90 days of certain log files to be retained. Setting a cooling period of 3 days results in any log files older than 3 days to be tiered out from the performance layer. This action frees up substantial space on the performance tier while still allowing you to view and manage the full 90 days of data..

None

The `none` tiering policy prevents any additional blocks from being tiered from the storage layer, but any data still in the capacity tier remains in the capacity tier until it is read. If the block is then read, it is pulled back and placed on the performance tier.

The primary reason to use the `none` tiering policy is to prevent blocks from being tiered, but it could become useful to change the policies over time. For example, let's say that a specific dataset is extensively tiered to the capacity layer, but an unexpected need for full performance capabilities arises. The policy can be changed to prevent any additional tiering and to confirm that any blocks read back as IO increases remain in the performance tier.

All

The `all` tiering policy replaces the `backup` policy as of ONTAP 9.6. The `backup` policy applied only to data protection volumes, meaning a SnapMirror or NetApp SnapVault destination. The `all` policy functions the same, but is not restricted to data protection volumes.

With this policy, blocks are immediately considered cool and eligible to be tiered to the capacity layer immediately.

This policy is especially appropriate for long-term backups. It can also be used as a form of Hierarchical Storage Management (HSM). In the past, HSM was commonly used to tier the data blocks of a file to tape while keeping the file itself visible on the file system. A FabricPool volume with the `all` policy allows you to store files in a visible and manageable yet consuming nearly no space on the local storage tier.

Oracle databases and FabricPool retrieval policies

The tiering policies control which Oracle database blocks are tiered from the performance tier to the capacity tier. Retrieval policies control what happens when a block that has been tiered is read.

Default

All FabricPool volumes are initially set at `default`, which means the behavior is controlled by the `cloud-retrieval-policy. `The exact behavior depends on the tiering policy used.

- `auto`– only retrieve randomly read data

- `snapshot-only`– retrieve all sequentially or randomly read data
- `none`– retrieve all sequentially or randomly read data
- `all`– do not retrieve data from the capacity tier

On-read

Setting `cloud-retrieval-policy` to on-read overrides the default behavior so a read of any tiered data results in that data being returned to the performance tier.

For example, a volume might have been lightly used for a long time under the `auto` tiering policy and most of the blocks are now tiered out.

If an unexpected change in business needs required some of the data to be repeatedly scanned in order to prepare a certain report, it may be desirable to change the `cloud-retrieval-policy` to `on-read` to ensure that all data that is read is returned to the performance tier, including both sequentially and randomly read data. This would improve performance of sequential I/O against the volume.

Promote

The behavior of the promote policy depends on the tiering policy. If the tiering policy is `auto`, then setting the `cloud-retrieval-policy `to `promote` brings back all blocks from the capacity tier on the next tiering scan.

If the tiering policy is `snapshot-only`, then the only blocks that are returned are the blocks that are associated with the active file system. Normally this would not have any effect because the only blocks tiered under the `snapshot-only` policy would be blocks associated exclusively with snapshots. There would be no tiered blocks in the active file system.

If, however, data on a volume was restored by a volume SnapRestore or file-clone operation from a snapshot, some of the blocks that were tiered out because they were only associated with snapshots may now be required by the active file system. It may be desirable to temporarily change the `cloud-retrieval-policy` policy to `promote` to quickly retrieve all locally required blocks.

Never

Do not retrieve blocks from the capacity tier.

Tiering strategies

Oracle database full file FabricPool tiering

Although FabricPool tiering operates at the block level, in some cases it can be used to provide file-level tiering.

Many applications datasets are organized by date, and such data is generally increasingly less likely to be accessed as it ages. For example, a bank might have a repository of PDF files that contain five years of customer statements, but only the most recent few months are active. FabricPool can be used to relocate older datafiles to the capacity tier. A cooling period of 14 days would ensure the more recent 14 days of PDF files remain on the performance tier. Furthermore, files that are read at least every 14 days would remain hot and therefore remain on the performance tier.

Policies

To implement a file-based tiering approach, you must have files that are written and not subsequently modified. The `tiering-minimum-cooling-days` policy should be set high enough so that files that you might need remain on the performance tier. For example, a dataset for which the most recent 60 days of data is required with optimal performance warrants setting the `tiering-minimum-cooling-days` period to 60. Similar results can also be achieved based on the file access patterns. For example, if the most recent 90 days of data is required and the application is accessing that 90-day span of data, then the data would remain on the performance tier. By setting the `tiering-minimum-cooling-days` period to 2, you get prompt tiering after the data becomes less active.

The `auto` policy is required to drive tiering of these blocks because only the `auto` policy affects blocks that are in the active file system.

> Any type of access to data resets the heat map data. Virus scanning, indexing, and even backup activity that reads the source files prevents tiering because the required `tiering-minimum-cooling-days` threshold is never reached.

Oracle partial file FabricPool tiering

Because FabricPool works at the block level, files that are subject to change can be partially tiered to object storage while also remaining partially on performance tier.

This is common with databases. Databases that are known to contain inactive blocks are also candidates for FabricPool tiering. For example, a supply chain management database might contain historical information that must be available if needed but is not accessed during normal operations. FabricPool can be used to selectively relocate the inactive blocks.

For example, datafiles running on a FabricPool volume with a `tiering-minimum-cooling-days` period of 90 days retains any blocks accessed in the preceding 90 days on the performance tier. However, anything that is not accessed for 90 days is relocated to the capacity tier. In other cases, normal application activity preserves the correct blocks on the correct tier. For example, if a database is normally used to process the previous 60 days of data on a regular basis, a much lower `tiering-minimum-cooling-days` period can be set because the natural activity of the application makes sure that blocks are not relocated prematurely.

The `auto` policy should be used with care with databases. Many databases have periodic activities such as end-of-quarter process or reindexing operations. If the period of these operations is greater than the `tiering-minimum-cooling-days` performance problems can occur. For example, if end-of-quarter processing requires 1TB of data that was otherwise untouched, that data might now be present on the capacity tier. Reads from the capacity tier is often extremely fast and may not cause performance problems, but the exact results will depend on the object store configuration.

Policies

The `tiering-minimum-cooling-days` policy should be set high enough to retain files that might be required on the performance tier. For example, a database in which the most recent 60 days of data might be required with optimal performance would warrant setting the `tiering-minimum-cooling-days` period to 60 days. Similar results could also be achieved based on the access patterns of files. For example, if the most recent 90 days of data is required and the application is accessing that 90- day span of data, then the data would remain on the performance tier. Setting the `tiering-minimum-cooling-days` period to 2 days would tier the data promptly after the data becomes less active.

The `auto` policy is required to drive tiering of these blocks because only the `auto` policy affects blocks that are

in the active file system.

> Any type of access to data resets the heat map data. Therefore, database full table scans and even backup activity that reads the source files prevents tiering because the required `tiering-minimum-cooling-days` threshold is never reached.

Oracle database archive log tiering

Perhaps the most important use for FabricPool is improving the efficiency of known cold data, such as database transaction logs.

Most relational databases operate in transaction log archival mode to deliver point-in-time recovery. Changes to the databases are committed by recording the changes in the transaction logs, and the transaction log is retained without being overwritten. The result can be a requirement to retain an enormous volume of archived transaction logs. Similar examples exist with many other application workflows that generate data that must be retained, but is highly unlikely to ever be accessed.

FabricPool solves these problems by delivering a single solution with integrated tiering. Files are stored and remain accessible in their usual location, but take up virtually no space on the primary array.

Policies

Use a `tiering-minimum-cooling-days` policy of a few days results in retention of blocks in the recently created files (which are the files most likely to be required in the near term) on the performance tier. The data blocks from older files are then moved to the capacity tier.

The `auto` enforces prompt tiering when the cooling threshold has been reached irrespective of whether the logs have been deleted or continue to exist in the primary file system. Storing all the potentially required logs in a single location in the active file system also simplifies management. There is no reason to search through snapshots to locate a file that needs to be restored.

Some applications, such as Microsoft SQL Server, truncate transaction log files during backup operations so that the logs are no longer in the active file system. Capacity might be saved by using the `snapshot-only` tiering policy, but the `auto` policy is not useful for log data because there should rarely cooled log data in the active file system.

Oracle with FabricPool snapshot tiering

The initial release of FabricPool targeted the backup use case. The only type of blocks that could be tiered were blocks that were no longer associated with data in the active file system. Therefore, only the snapshot data blocks could be moved to the capacity tier. This remains one of the safest tiering options when you need to ensure performance is never affected.

Policies - local snapshots

Two options exist for tiering inactive snapshot blocks to the capacity tier. First, the `snapshot-only` policy only targets the snapshot blocks. Although the `auto` policy includes the `snapshot-only` blocks, it also tiers blocks from the active file system. This might not be desirable.

The `tiering-minimum-cooling-days` value should be set to a time period that makes data that might be required during a restoration available on the performance tier. For example, most restore scenarios of a

critical production database include a restore point at some time in the previous few days. Setting a `tiering-minimum-cooling-days` value of 3 would make sure that any restoration of the file results in a file that immediately delivers maximum performance. All blocks in the active files are still present on fast storage without needing to recover them from the capacity tier.

Policies - replicated snapshots

A snapshot that is replicated with SnapMirror or SnapVault that is only used for recovery should generally use the FabricPool `all` policy. With this policy, metadata is replicated, but all data blocks are immediately sent to the capacity tier, which yields maximum performance. Most recovery processes involve sequential I/O, which is inherently efficient. The recovery time from the object store destination should be evaluated, but, in a well-designed architecture, this recovery process does not need to be significantly slower than recovery from local data.

If the replicated data is also intended to be used for cloning, the `auto` policy is more appropriate, with a `tiering-minimum-cooling-days` value that encompasses data that is expected to be regularly used in a cloning environment. For example, a database's active working set might include data read or written in the previous three days, but it could also include another 6 months of historical data. If so, then the `auto` policy at the SnapMirror destination makes the working set available on the performance tier.

Oracle database backup tiering

Traditional application backups include products such as Oracle Recovery Manager, which create file-based backups outside the location of the original database.

```
`tiering-minimum-cooling-days` policy of a few days preserves the most
recent backups, and therefore the backups most likely to be required for
an urgent recovery situation, on the performance tier. The data blocks of
the older files are then moved to the capacity tier.
```

The `auto` policy is the most appropriate policy for backup data. This ensures prompt tiering when the cooling threshold has been reached irrespective of whether the files have been deleted or continue to exist in the primary file system. Storing all the potentially required files in a single location in the active file system also simplifies management. There is no reason to search through snapshots to locate a file that needs to be restored.

The `snapshot-only` policy could be made to work, but that policy only applies to blocks that are no longer in the active file system. Therefore, files on an NFS or SMB share must be deleted first before the data can be tiered.

This policy would be even less efficient with a LUN configuration because deletion of a file from a LUN only removes the file references from the file system metadata. The actual blocks on the LUNs remain in place until they are overwritten. This situation can create a lengthy delay between the time a file is deleted and the time that the blocks are overwritten and become candidates for tiering. There is some benefit to moving the `snapshot-only` blocks to the capacity tier, but, overall, FabricPool management of backup data works best with the `auto` policy.

This approach helps users manage the space required for backups more efficiently, but FabricPool itself is not a backup technology. Tiering backup files to object store simplifies management because the files are still visible on the original storage system, but the data blocks in the object store destination depend on the original storage system. If the source volume is lost, the object store data is no longer useable.

Oracle database and object store access interruptions

Tiering a dataset with FabricPool results in a dependency between the primary storage array and the object store tier. There are many object storage options that offer varying levels of availability. It is important to understand the impact of a possible loss of connectivity between the primary storage array and the object storage tier.

If an I/O issued to ONTAP requires data from the capacity tier and ONTAP cannot reach the capacity tier to retrieve blocks, then the I/O eventually times out. The effect of this timeout depends on the protocol used. In an NFS environment, ONTAP responds with either an EJUKEBOX or EDELAY response, depending on the protocol. Some older operating systems might interpret this as an error, but current operating systems and current patch levels of the Oracle Direct NFS client treat this as a retriable error and continue waiting for the I/O to complete.

A shorter timeout applies to SAN environments. If a block in the object store environment is required and remains unreachable for two minutes, a read error is returned to the host. The ONTAP volume and LUNs remain online, but the host OS might flag the file system as being in an error state.

Object storage connectivity problems `snapshot-only` policy is less of a concern because only backup data is tiered. Communication problems would slow data recovery but would not otherwise affect data being actively used. The `auto` and `all` policies allow tiering of cold data from the active LUN, which means that an error during object store data retrieval could affect database availability. A SAN deployment with these policies should only be used with enterprise-class object storage and network connections designed for high availability. NetApp StorageGRID is the superior option.

Copyright information

Trademark information