Technical Report

Deploying NetApp E-Series Copy Services with Oracle and SQL Server Databases

E-Series Snapshot and Replication Technology

Mitch Blackburn and Mike Bostock, NetApp
December 2016 | TR-4458

Abstract

This guide is intended for storage administrators and database administrators to help them successfully deploy NetApp® E-Series and EF-Series copy services with Microsoft SQL Server and Oracle databases.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction to E-Series and EF-Series Copy Services

This technical report (TR) gives you guidance and recommendations for how and when to use the NetApp E-Series and EF-Series storage system copy services features. You need to protect your data, and that protection must occur with minimal interruptions. To fulfill these requirements, the E-Series and EF-Series offer the following copy services features:

- NetApp Snapshot® point-in-time images
- Snapshot volumes
- Volume copy
- Synchronous mirroring
- Asynchronous mirroring

All these features are available at no additional cost when you purchase an E-Series or EF-Series. You can use these features individually, or you can combine them to design a high-availability disaster recovery solution for your critical data protection needs.

Note: E-Series alone is used to reference features found in NetApp SANtricity® Storage Manager 11.20 discussed in this document.

2 E-Series and EF-Series Snapshot Images

A Snapshot image is a point-in-time logical copy of a user volume or a set of volumes. You can use a Snapshot image to create a read/write volume that:

- Is accessible from the host
- Requires only a small percentage of the base volume capacity
- Allows rollback of the base volume to restore its data to a previous point in time

Snapshot image benefits also include use for testing and training as well as protection against user error and data corruption.

Snapshot technology is widely used to test applications as well as upgrades, application patches, OS upgrades, and OS patches before corporatewide rollout. It is possible to create an identical Snapshot volume to support an in-depth test environment.

Additionally, during code and application development, you can create parallel environments to speed development of multiple solutions or competing solutions. The use of Snapshot volumes for testing has the advantage of creating a virtual clone of a volume without having to touch the actual volume itself.

By using consistency groups, you can consistently back up applications that are spread across multiple volumes, such as a database, for use in a secondary environment.

To maintain the point-in-time image of the base volume, Snapshot images use a repository to keep the original data blocks for those blocks that change in the base volume after the Snapshot image is taken. The repository is automatically created when the Snapshot group is created and is, by default, 40% of the size of the base volume. Snapshot images are logical copies of the base volume, so they do not take any capacity from the storage system other than the capacity that the repository requires.

Table 1 provides a glossary of Snapshot terms that are used throughout this document.

Table 1) Glossary of Snapshot terms used in the SANtricity storage management software.

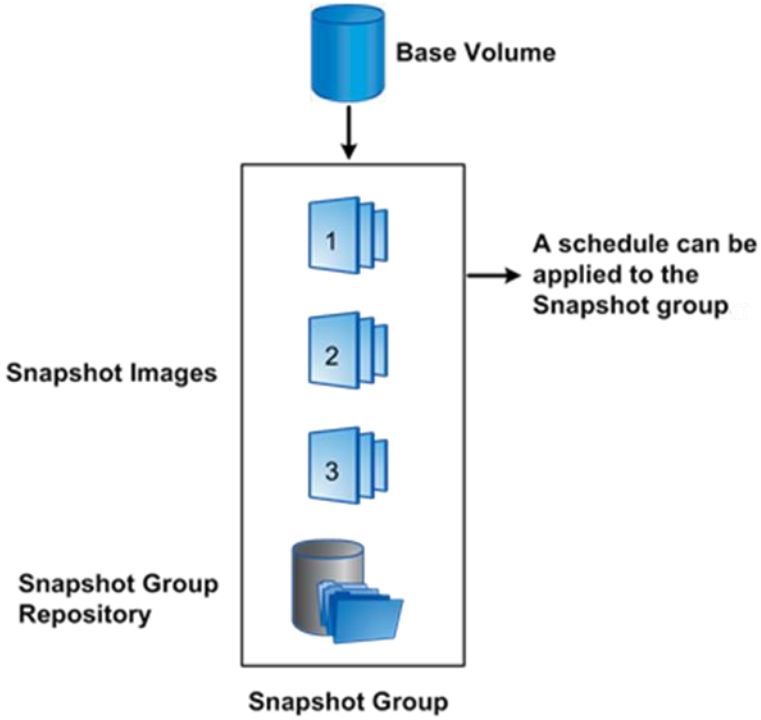
SANtricity Term	Definition and Use Case
Base volume	A normal data volume that is typically mapped to a host system through LUN mapping. A base volume can be a standard volume or a thin-provisioned volume.

SANtricity Term	Definition and Use Case
	• **Use case:** The base volume provides block-level data access through the `SCSI READ` and `SCSI WRITE` commands.
Snapshot image	A logical point-in-time image of the content of a base volume, created at a specific moment. A Snapshot image is not directly read/write accessible to hosts. • **Use case:** The Snapshot image is used for rolling back the base volume to a known good image that was created at a specific point in time.
Snapshot group	A collection of Snapshot images of a single base volume. • **Use case:** The Snapshot group allows the creation of a sequence of Snapshot images of a given base volume without affecting system performance. It accomplishes this goal by using a single copy-on-write operation for each write operation to the repository.
Snapshot group repository	The storage provided to each Snapshot group for tracking both which data blocks of the base volume have been overwritten and the preserved (that is, preoverwrite) content of those blocks. • **Use case:** The Snapshot group repository stores the copy-on-write information that is required to maintain the Snapshot images.
Snapshot volume	A standard volume that allows the host to access the Snapshot image within a Snapshot group (read-only or read/write). • **Use case:** The Snapshot volume can be used to create a test environment or a reporting environment so that the production environment is not affected.
Consistency group	A collection of volumes that are treated as a single entity when a Snapshot image is created. Each of these volumes has its own Snapshot image, but all the images are created at the same point in time. • **Use case:** Consistency groups are used by applications that span multiple volumes.

2.1 Snapshot Images

Figure 1 represents how Snapshot images are created from a base volume at specific points in time.

Figure 1) Snapshot image and Snapshot group.



Snapshot images are collected into a Snapshot group that is associated only with a unique base volume. When the first Snapshot image of the base volume is created, if a Snapshot group does not yet exist, the storage management software automatically also creates a Snapshot group in which to place that Snapshot image. The first Snapshot image is an exact logical copy of the base volume at the point in time when the image was created. Thereafter, when the system generates another Snapshot image, it captures only the data changes that occurred after the previous Snapshot image was made.

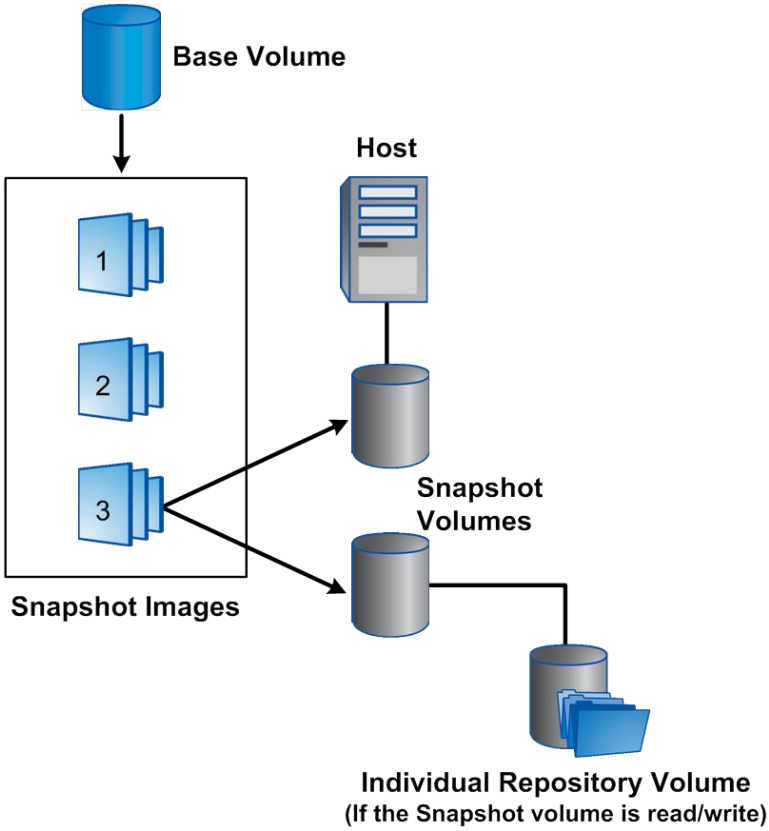
Each Snapshot group includes a Snapshot group repository in which to save Snapshot images. Every Snapshot image is created in the context of exactly one Snapshot group. Up to four Snapshot groups can exist per base volume. When a base volume is in a consistency group, the consistency group is counted toward the limit of four Snapshot groups. Each Snapshot group can in turn contain 32 Snapshot images. As a storage administrator, you can create multiple Snapshot groups to support different schedules for creating Snapshot images or to support more than 32 Snapshot images for the base volume.

The base volume that is associated with a Snapshot group can reside in a disk pool or in a volume group. If the base volume resides in a volume group, the repository members for any associated Snapshot group can reside in a disk pool or in a volume group. If, however, the base volume resides in a disk pool, all repository members for any associated Snapshot group must reside in the same disk pool as the base volume.

2.2 Snapshot Volumes

A Snapshot volume is a logical entity that is structured on top of a Snapshot image, as illustrated in Figure 2.

Figure 2) Snapshot volume.



Snapshot images are not directly read/write accessible to hosts; to map a Snapshot image to a host, you must create a Snapshot volume from the Snapshot image. You can create up to four Snapshot volumes from the same Snapshot image.

Every Snapshot volume has its own repository to keep track of data changes that occur on the base volume. A one-to-one relationship exists between a Snapshot volume and its associated repository volume. The repository is not accessible for host I/O; it is used only internally to manage the Snapshot point-in-time image.

The Snapshot volume has the same characteristics as the base volume (such as RAID level and I/O characteristics). However, the Snapshot volume is viewed as a separate standard volume by the storage system; therefore, it can be mapped to a host, read from, or written to. A Snapshot volume can be designated as read-only or read/write. If the Snapshot volume is read-only, then no associated repository is required. Read-only Snapshot volumes are generally designated for a backup application. Read/write Snapshot volumes are mapped to a host application for testing or experimental purposes.

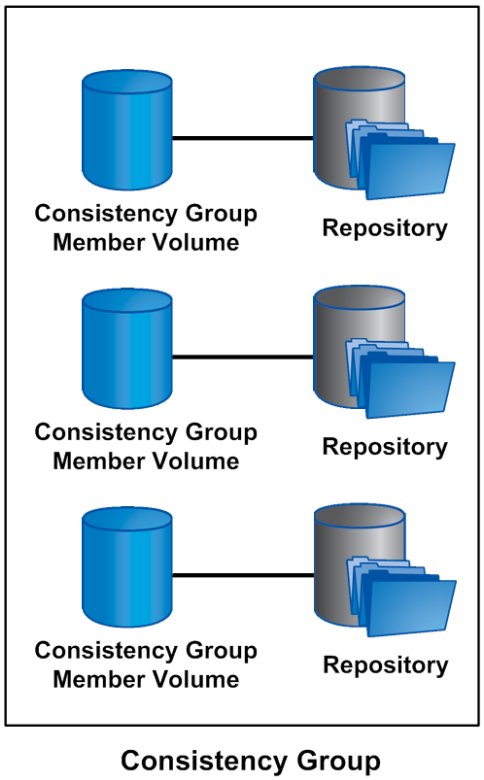
A Snapshot volume can also be created from a consistency group, generating an exact point-in-time copy of all member volumes in that consistency group.

2.3 Consistency Groups

A consistency group is a convenience entity that allows a collection of base volumes to have consistent behavior and interaction relative to Snapshot images and Snapshot volumes. Each volume is referred to as a *member volume* of the consistency group. A member volume is either a standard volume in a disk pool or volume group or a thin-provisioned volume in a disk pool. When a member volume is added to a consistency group, the system automatically creates an associated member repository and a Snapshot group for it.

When a Snapshot image or Snapshot volume is created at the consistency group level, all associated member volumes in the consistency group are affected. A member volume in a consistency group receives an associated repository when it is added to the consistency group, as shown in Figure 3. If a Snapshot volume is created at the consistency group level, the member volume also receives another repository. A consistency group can have Snapshot images that are created manually or by applying a schedule. All Snapshot images in a consistency group have the same timestamp.

Figure 3) Consistency group.

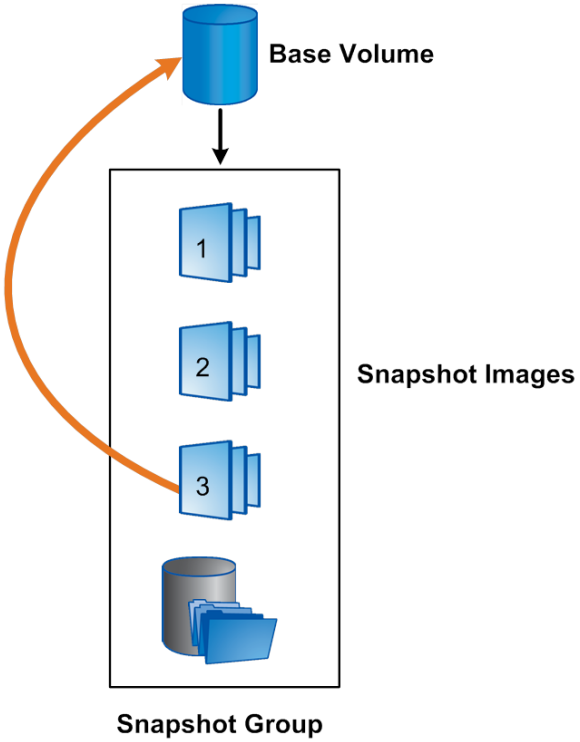


The purpose of a consistency group is to make simultaneous Snapshot images of multiple volumes, which promotes consistent copies of a collection of volumes at a particular point in time. For example, you can create a consistency group to make a synchronized Snapshot image of several volumes that are in different volume groups or disk pools in a storage system. The consistency group mechanism is ideal for use with applications that span multiple volumes, such as a database application with the logs on one volume and the database on another volume.

2.4 Rollback

It is sometimes necessary to return the data in a base volume to a previous point in time, possibly because of data corruption or user error on the base volume. This process is called a *rollback* from a previous Snapshot image. Snapshot rollback provides an instantaneous restore of a full volume from a Snapshot image, as shown in Figure 4.

Figure 4) Rollback from Snapshot image to base volume.



When you select a Snapshot image for rollback onto a base volume, all other Snapshot images (newer and older) in the Snapshot group are still preserved. A background process sweeps through the base volume's logical block accesses (LBAs) to find copy-on-write data in the rollback Snapshot image (or beyond) to be restored to the base volume. Rollback writes to the base volume might cause copy-on-write actions on the Snapshot group.

Regardless of how much of the repository capacity is currently used, the base volume is accessible for both reads and writes during the rollback session, and all previously written data is also available immediately. Therefore, the associated repository must be large enough to contain all changes that might occur while the rollback operation is still processing. The data transfers from the repository back to the base volume continue as a background operation until the rollback is complete.

Note: When a rollback has been initiated, you cannot stop or cancel it.

2.5 I/O Handling to Snapshot Volumes

Write Requests

After the Snapshot image is created, ensuing writes to the base volume cause copy-on-write (COW) actions to occur. With these COW actions, the required images of the affected sectors are saved in the Snapshot group repository before the base volume's sectors are overwritten with the new data. This process occurs only for the first overwrite of each base volume sector after the creation of the Snapshot image.

In a Snapshot group, the creation of a new Snapshot image causes the following changes:

- The predecessor Snapshot image's content, including both its index and the captured data, is effectively frozen within the Snapshot group repository. That is, it is not changed thereafter, regardless of any ongoing write activity on the base volume.

- A new, and initially empty, index is created for the new Snapshot image. As base volume writes begin to occur, this new index is used when COW operations are carried out for the Snapshot group. Not only is the new index updated to track newly saved base-volume data, but that new data is also added to the Snapshot group repository. In some cases, a given 64KB extent (called a *cluster*) of the base volume has been saved in the Snapshot group repository for a predecessor Snapshot image. In those cases, a rewrite of that cluster after the creation of the new Snapshot image causes another image of the cluster to be saved in the Snapshot group repository.

As a result of these two points, the number of COW operations for a base volume write operation (when a Snapshot group exists for the base volume) is limited to at most one. Specifically, if the most recently created Snapshot index in the associated Snapshot group repository indicates that the affected cluster has already been saved in a previous COW for that Snapshot image, then no additional COW is required. Otherwise, a COW action is performed. This point holds true regardless of the number of predecessor Snapshot images that might exist. Only the current (newest) Snapshot image's index is used for determining whether a COW is needed when a base volume write request is processed.

Read Requests

With Snapshot groups, reading data associated with a Snapshot image starts out with the index for the relevant Snapshot image. That index is checked to determine whether the requested cluster is in that Snapshot image's portion of the Snapshot group repository. If so, the requested data is fetched from the Snapshot group repository. However, if the index indicates that the cluster is not present, the index of the next (that is, the *successor*, or later) Snapshot image within the Snapshot group is checked. This check is required because the base volume's cluster might have been written after the creation of the successor Snapshot image. In that case, the base volume data is saved to that successor Snapshot image's portion of the Snapshot group repository.

However, the data that is saved there is actually the same content that must be returned when the predecessor Snapshot image is accessed. In fact, if a successor Snapshot image is found that does contain the requested cluster, it is necessary to check the index of every successor within the Snapshot group. And they must be checked in the order in which they were created. If the requested cluster is not either in the Snapshot group repository of the requested Snapshot image or in any of its successors, then the data is retrieved from the base volume. It is retrieved from the base volume because clearly it has not been modified since the requested Snapshot image was created or since any of that Snapshot image's successors were created.

When reading data from older Snapshot volumes, it is often necessary to check the indexes of potentially many successor Snapshot images. The performance impact of doing so is mitigated by the fact that extensive caching occurs for Snapshot index information, thereby reducing the likelihood that any actual media accesses are required to carry out the index checks.

Table 2 summarizes how host I/O read and write requests are handled to a Snapshot volume.

Table 2) I/O handling to a Snapshot base volume.

I/O Type	Handling
Reads to the base volume	A host I/O read to the base volume is passed through the base proxy with minimal interruption.
Writes to the base volume	When a write is destined for a base volume that has an associated Snapshot image, the write request first encounters the base proxy: • If it is the first time that the base volume data has been modified, then the repository reads the appropriate LBAs from the base volume and writes them to the repository. • If a Snapshot image already exists and it is the first time that the base volume data has changed since the last Snapshot image was created, then the repository reads the appropriate LBAs from the base volume and writes them to the Snapshot group

I/O Type	Handling
	repository. After the base proxy has been notified that this operation is complete, the host proceeds with the write operation in a normal fashion. • Upon a second write request or iterative write requests to the same LBAs with no new Snapshot images having been made, no data movement is required to preserve the Snapshot image.
Reads to the Snapshot volume	Reads to the Snapshot volume occur as normal read requests, although they might be redirected to either the associated repository or the base volume, depending on the location of the most current data. If multiple Snapshot images exist of a given base volume within the same Snapshot group, then traversal of all the associated Snapshot image indexes might be required to locate the correct data.
Writes to the Snapshot volume	All writes to the Snapshot volume are passed to an associated user-created repository volume for that Snapshot volume. A corresponding pointer is also updated to reflect the position of this new data for future read requests to the Snapshot volume. Snapshot volumes might be mounted as read-only, in which case a specific Snapshot volume repository does not need to be used to store subsequent write modifications to the Snapshot volume itself.

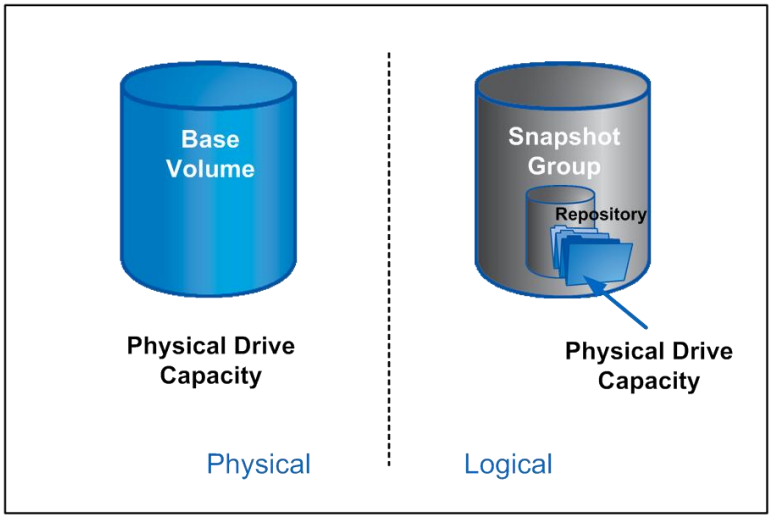
A single Snapshot group can contain up to 32 Snapshot images of a given base volume. Therefore, it is possible, albeit very unlikely, that the required capacity for the Snapshot group repository could be up to 32 times the capacity of the original base volume. For this scenario to occur, the entire contents of the base volume would have to change between each subsequent Snapshot image being created.

In addition to the COW clusters being copied to the Snapshot group repository, an additional small amount of metadata is associated with each Snapshot image. The metadata typically does not exceed 0.02% of the capacity of the original base volume and is therefore negligible. Accounting for Snapshot group repository capacity is largely a function of understanding the ratio of cluster overwrites as it relates to the base volume and the associated Snapshot images.

Snapshot Copy-on-Write Technology

The Snapshot feature uses COW technology to create a durable point-in-time image that is useful for backup and testing. The Snapshot repository holds data that is saved from the base volume during COW activity, as illustrated in Figure 5.

Figure 5) Snapshot COW technology.



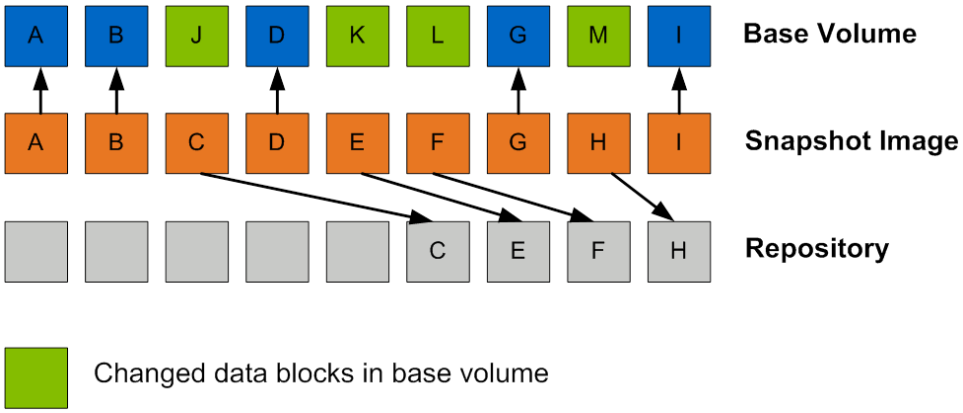
Original data blocks must first be written to the repository before they can be updated in the base volume, causing a slight performance impact while a Snapshot image or a Snapshot volume is enabled. Therefore, if a Snapshot image or a Snapshot volume is no longer needed, it should be disabled. If a Snapshot image is disabled, all COW activity from the base volume ceases, and the Snapshot volume becomes inaccessible for I/O operations, thus eliminating the performance impact.

If the Snapshot volume is needed later, it can be recreated, which causes a new point-in-time image to be made, and the Snapshot volume becomes accessible for I/O operations again. Disabling the Snapshot volume is useful in situations that call for regular or periodic backups. The Snapshot volume can be disabled, but the configuration and structural elements (such as the repository, the Snapshot warning thresholds, and so on) are retained. The base volume can return to full performance for the period between backups. Then, at the time of the next backup, a new point-in-time image is created by using the previously defined Snapshot volume.

When the repository of a Snapshot volume is nearing the full point, an alert is issued to the storage system. By default, this alert is issued when the repository is 75% full; however, as a storage administrator, you can adjust this threshold up or down as needed. When the alert is issued, you can increase the capacity of the repository at that time. In that regard, you can configure each Snapshot volume independently.

Figure 6 exemplifies the COW technology. After the original data block has been copied to the repository, the block is overwritten on the base volume. The actual Snapshot image is created by combining the repository and the base volumes.

Figure 6) COW example.



2.6 Technical Specifications

Table 3 presents the technical specifications for the Snapshot functionality in SANtricity Storage Manager for the different E-Series controller models.

Table 3) Technical specifications for Snapshot images per E-Series controller model.

Snapshot Specification	E2700	E5500/EF550	E5600/EF560
Snapshot images per Snapshot group	32	32	32
Snapshot images per storage system	512	2,048	2,048
Snapshot groups per base volume	4	4	4
Snapshot groups per storage system	256	1,024	1,024
Snapshot volumes per Snapshot image	4	4	4

Snapshot Specification	E2700	E5500/EF550	E5600/EF560
Snapshot volumes per storage system	256	1,024	1,024
Base volumes per consistency group	32	64	64
Consistency groups per storage system	16	32	32

For detailed information about using Snapshot images and Snapshot volumes, go to the SANtricity Storage Manager online help.

3 E-Series and EF-Series Volume Copy

By performing a byte-by-byte copy from a source volume to a target volume, the volume copy feature creates a full clone of a volume. When the copy is complete, the target volume is identical to the source volume. Because the target volume is a real volume, rather than a logical one, you can use it as part of a disaster recovery solution.

Both volumes must be on the same storage system. The copy pair is the association between the source volume and the target volume for a single volume copy operation.

If a volume is recopied from the source to the same target volume, the data in the target volume is overwritten. Therefore, if you need a second clone of the source volume, you should use a different target volume for the copy function.

As the storage requirements for a volume change, you can use the volume copy feature to copy data from one volume to another. This process is an easy way to migrate data to larger capacity or faster disks within the same storage system. You can also use the volume copy feature to move data from a thin-provisioned volume to a standard volume.

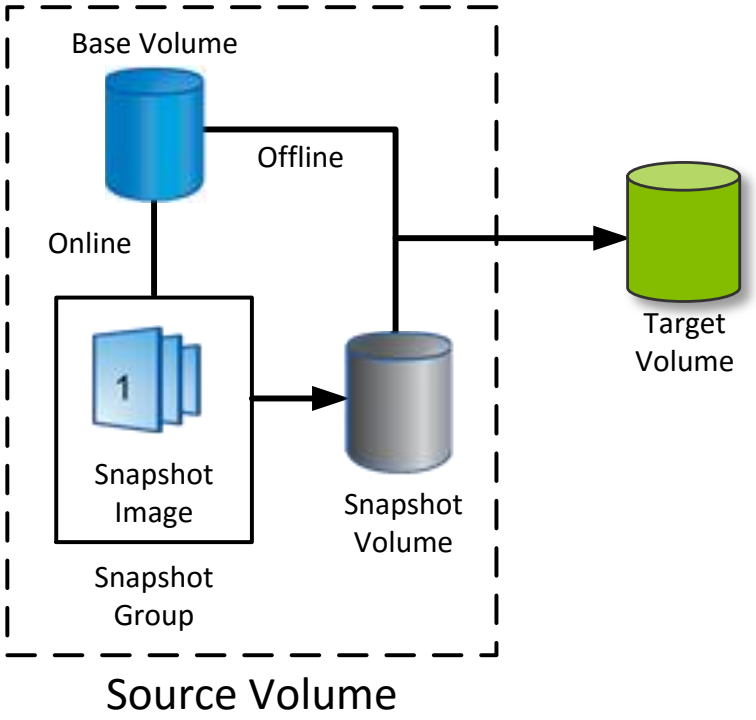
You can also use volume copy to create a backup of a volume by copying data from one volume to another volume in the same storage array. You can use the clone volume for testing, training, or development.

There are two ways to copy a volume: offline and online. All write operations are suspended to a source volume during a volume copy operation. With this approach, the data on the source volume remains consistent until the procedure is completed. After the volume copy operation is completed, the source volume automatically becomes read-/write-accessible again. Following are the differences between the two copy methods:

- An offline copy suspends all write operations to the source volume while the copy is being made. For journaled file systems, such as Linux XFS, NetApp WAFL® (Write Anywhere File Layout), and NTFS, the volume is inaccessible to both reads and writes while the copy is being made.
- An online copy is possible by first creating a Snapshot image of the volume to be copied and then using that logical Snapshot volume as the source for the volume copy. By using a Snapshot image in this way, you can read and write to the base volume to be copied, even while it is being copied.

Figure 7 provides a graphical view of the relationship.

Figure 7) Volume copy.



For detailed information about using volume copy, go to the SANtricity Storage Manager online help.

4 E-Series and EF-Series Synchronous and Asynchronous Mirroring

The SANtricity mirroring features provide storage-based data replication, which enables mirroring of data volumes from one storage system to another either synchronously or asynchronously by using either the FC or iSCSI network protocol.

SANtricity mirroring features maintain a copy of data that is physically distant from the site where the data is used. If a disaster occurs at the primary site, such as a massive power outage or a flood, the data can be quickly accessed from the remote location. Accessing the data from a remote storage system is much faster than uploading off-site tape backups. Also, the data that was in use at the time of the disaster does not differ at the remote site as much as it might from a tape backup that is several days old.

E-Series and EF-Series block-replication technology provides many benefits, such as:

- Block-level updates, which reduce bandwidth and time requirements by replicating only the blocks that have changed
- Crash-consistent data that is maintained at a disaster recovery site
- The ability to test disaster recovery plans without affecting production and replication
- Mirroring between dissimilar NetApp E-Series storage systems
- The use of a standard IP or FC network for replication

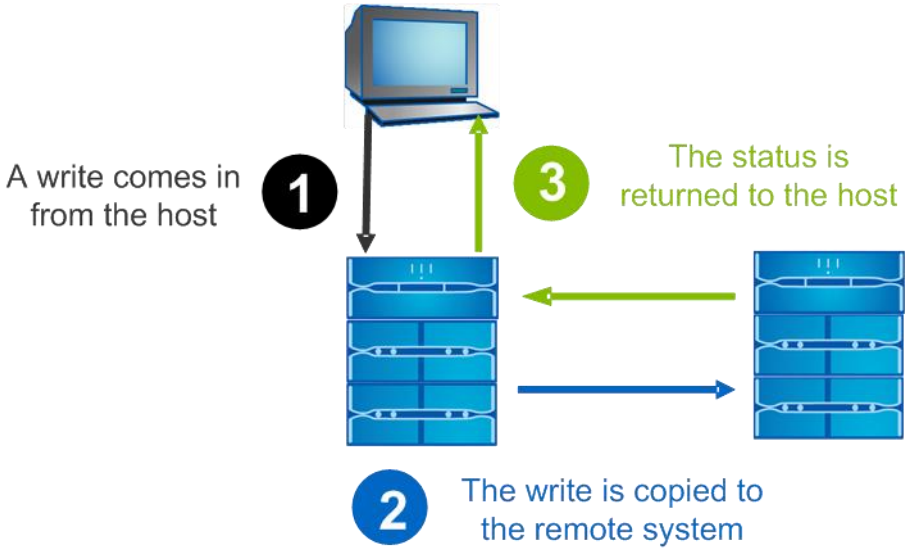
4.1 Synchronous Mirroring

The SANtricity synchronous mirroring feature is used for online, real-time data replication between remote storage arrays. Any new data that is written to the local (or primary) system is immediately transferred to

the remote (or secondary) system. A very fast connection between the local and remote storage systems is required to maintain the mirror relationship without unduly reducing local I/O.

With synchronous mirroring, new data is sent to a remote system before the status of any incoming write operation is returned, as Figure 8 shows. Thus data at the two sites is identical. You can configure the local system such that if it cannot complete a transfer, it either breaks the mirror relationship or rejects I/O. Because synchronous mirroring transfers are time-sensitive, testing has shown that they can be supported at distances of up to only 10 kilometers and only on an FC network.

Figure 8) Synchronous mirroring.



4.2 Asynchronous Mirroring

The SANtricity asynchronous mirroring feature provides a controller-level, firmware-based mechanism for data replication between primary and secondary sites. Asynchronous mirroring transfers data to the remote site only at set intervals, so local I/O is not affected nearly as much by slow network connections. Because local I/O is not affected by network latency, iSCSI is a viable connection alternative with asynchronous mirroring.

With asynchronous mirroring, a write operation first occurs on the local storage system, and then the status of the operation is returned to the host. Rather than keeping the local and remote volumes in constant alignment, the process logs and tracks all changes on the local volume. At set intervals, all changes are sent to the remote system as a background process; see Figure 9. Because this transfer is not tied to the local I/O, it does not affect application performance. Therefore, the feature can use slower connections, such as iSCSI, and run across longer distances between the local and remote storage systems. Asynchronous mirroring can be supported over distances of up to 12,000 kilometers on either an FC or an iSCSI network.

Figure 9) Asynchronous mirroring.

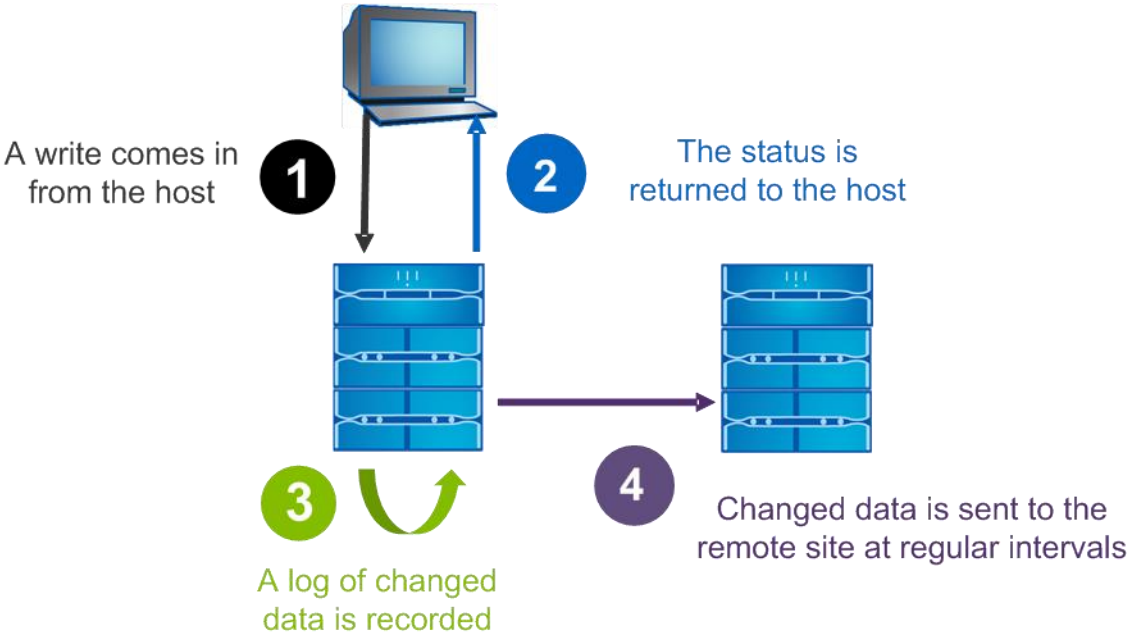


Table 4 presents the technical specifications for the asynchronous mirroring functionality in SANtricity Storage Manager for the different E-Series controller models.

Table 4) Technical specifications for asynchronous mirroring per E-Series controller model.

Asynchronous Mirroring Specification	E2700	E5500/EF550	E5600/EF560
Maximum mirrored pairs per base volume	1	1	1
Maximum mirrored pairs per storage array	32	128	128
Maximum mirrored pairs per async group	32	64	64
Maximum mirrored consistency groups	4	4	4

4.3 Additional Information

Although the local and remote volumes are almost never perfectly matched, the differences between them are a function of the time interval between transfers. The differences are thus consistent and predictable.

Both synchronous and asynchronous mirrored volumes can be accessed at all times. However, secondary volumes are strictly read-only, as far as hosts are concerned.

Note: For a database environment, it is not possible to use the database as read-only. The relational database management system software must write to the data and the log files to bring the database online.

If you need to access the data on the secondary volume for full use without breaking the mirror relationship, you can create a Snapshot volume from the secondary volume. You must then map it to the host that uses the data.

To use mirroring between a pair of storage systems, you must use some of their host ports: either FC or iSCSI. FC connections must go through a fabric, and the last host port of each controller is made

unavailable for any other I/O. Only asynchronous mirroring can occur over iSCSI connections, but you can use any iSCSI host port for mirroring.

Note: You cannot asynchronously mirror between two systems with different drive security levels (one with FDE-capable drives and the other without). In that situation, you must use synchronous mirroring.

For detailed information about using synchronous and asynchronous mirroring, go to the SANtricity Storage Manager online help. For further information about SANtricity asynchronous mirroring, see TR-4327: SANtricity Asynchronous Mirroring Feature for NetApp E-Series Storage Systems.

5 Copy Services Use Cases

The use cases presented here are not meant to represent a complete disaster recovery solution. They are meant to illustrate how different parts of such a solution could be implemented by using the E-Series or EF-Series copy services functionality. Therefore, these use cases don't show the last step of saving a backup to a long-term storage location such as to NetApp StorageGRID® services.

5.1 Consistency

SQL Server

Because of the SQL Server architecture, without the use of the Microsoft Volume Shadow Copy Service (VSS), you can recover backup Snapshot images and asynchronous mirrored copies only to the last point in time that an image was taken. You cannot roll forward this point-in-time image by using logs to the last committed write.

As an example, if a backup image is taken at 1 p.m. and then two hours later a database file is lost, you can restore the database only to the 1 p.m. point in time. All other writes after 1 p.m. are lost.

This effect is similar to a server going down (crashing) and then starting back up at a particular point in time. The database is fully consistent at that point in time.

Oracle

In contrast, the Oracle database architecture makes use of archived redo logs if the database is in `ARCHIVELOG` mode. These archived redo logs allow the database to be rolled forward through all the archived redo logs that were created since the backup Snapshot image was taken. If the online redo log is still available, you can roll forward the database through all recorded transactions and then roll it back to the last commit point. The database can then be considered to be fully recovered and consistent.

If all online redo logs have been lost, then you must open the database by using `RESETLOGS`. You can recover the database only through the last archive log, and all writes contained in the online redo log are lost. The database is fully consistent to the point in time of the last archive log.

5.2 Backup by Using Snapshot Images

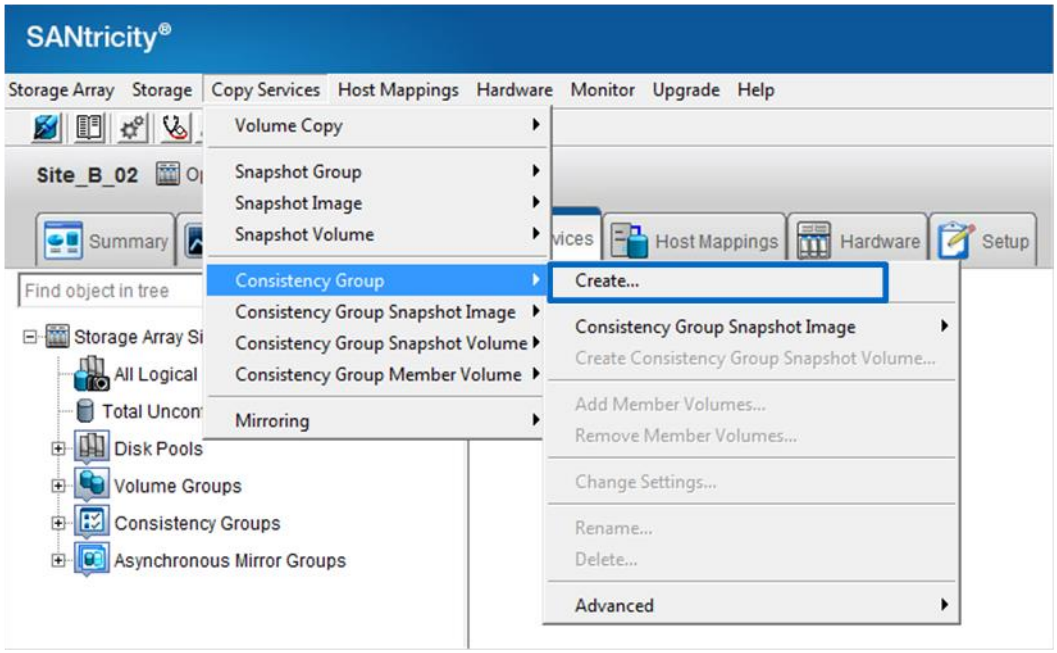
When you work with a database, you must have all the Snapshot images of the data volumes and the log volumes occur at the same time; you can do so by using consistency groups.

Create Consistency Groups

You can create a consistency group as an empty object so that you can either add member volumes later or add member volumes when you create the consistency group. The following procedure describes the steps for adding the member volumes when the consistency group is created.

To create a consistency group and add member volumes to it, complete the following steps:

1. From the Copy Services menu in the AMW Storage & Copy Services tab, select Consistency Group > Create.



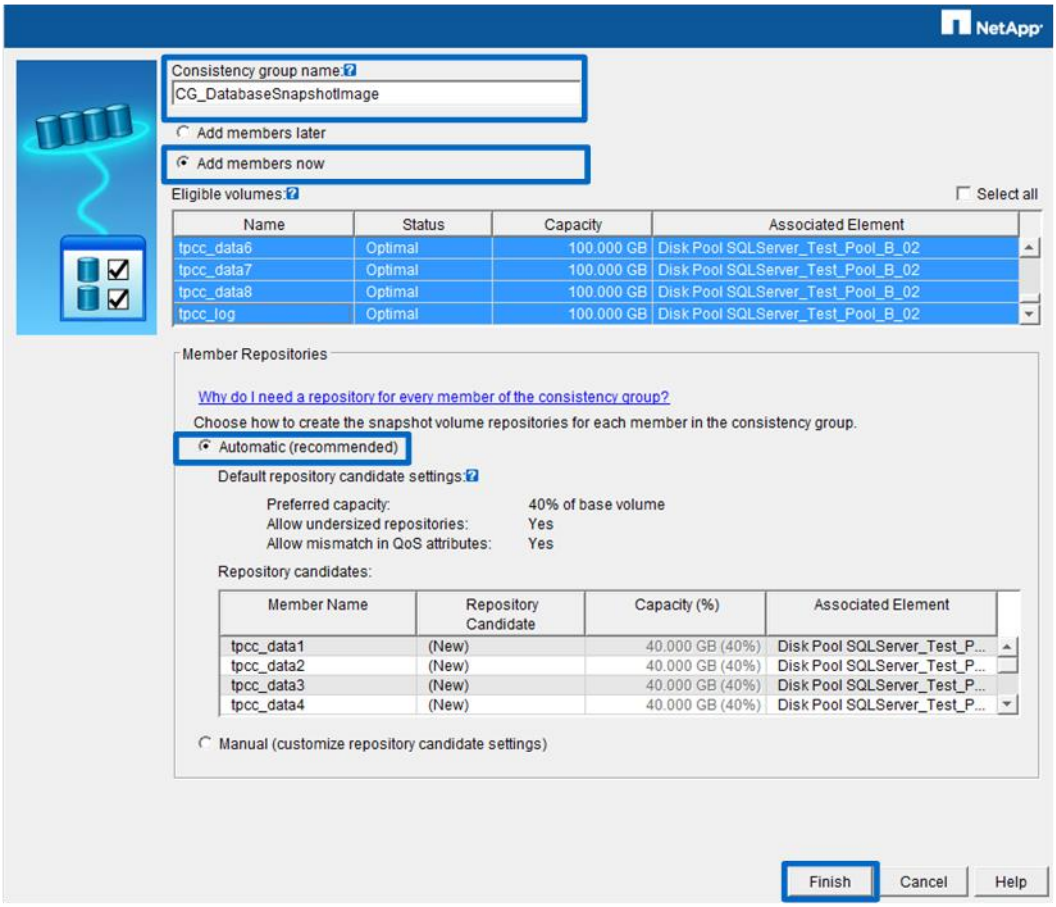
2. Configure the consistency group settings:
   a. Enter a name for the consistency group.

   Note: Consistency group names must not exceed 30 characters and cannot contain spaces. Names can contain letters, numbers, underscores (_), hyphens (-), and pound signs (#).

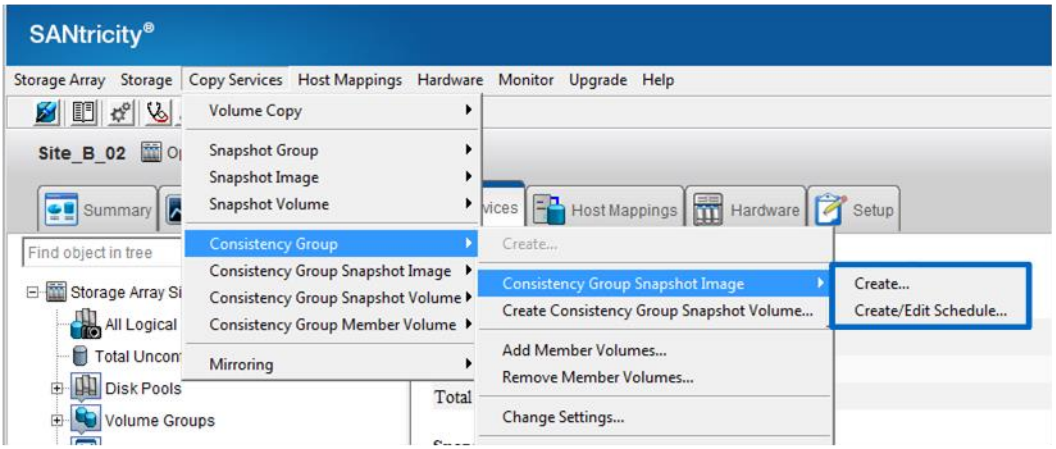
   b. Select Add Members Now. From the Eligible Volumes pane, select the volumes to add as members of the consistency group.
   c. Select Automatic (Recommended) to create the member repository.

NetApp recommends using the automatic method for creating Snapshot image repositories. The manual option allows you to specify all the customizable settings for the repositories. The manual method is considered advanced, however, and only those administrators who understand drive redundancy and optimal drive configurations should use this method. For instructions about how to set the repository parameters, see the SANtricity Storage Manager online help.

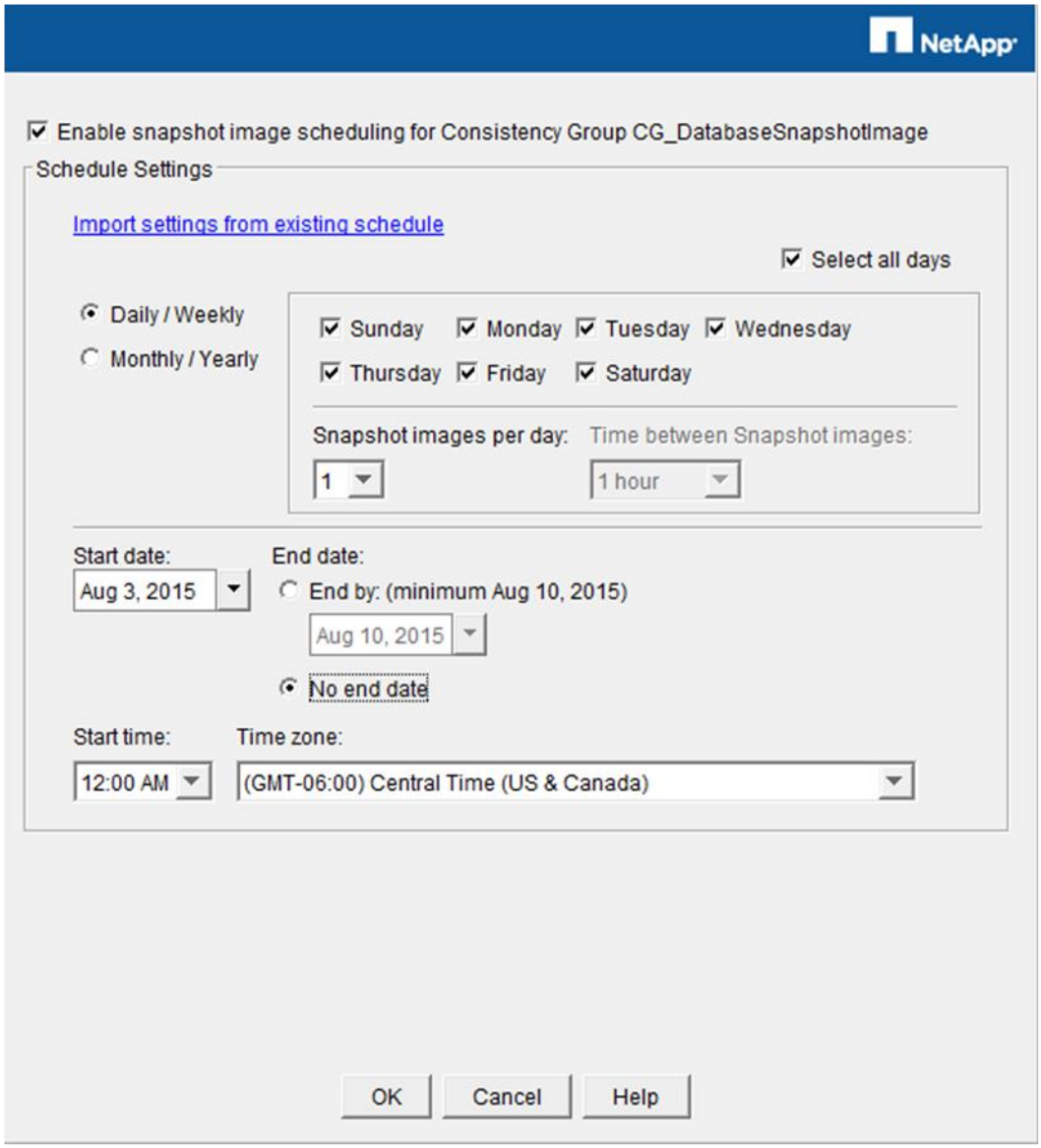
   d. Click Finish to create the consistency group.



3. After the consistency group is created, create the first Snapshot image. From the Copy Services menu, select Consistency Group > Consistency Group Snapshot Image > Create. Alternatively, set a schedule for creating Snapshot images of the consistency group. From the Copy Services menu, select Consistency Group > Consistency Group Snapshot Image > Create/Edit Schedule.



4. Set the schedule as desired and click OK.



SQL Server

For a SQL Server database, you must create only one consistency group with all the data files and the log file in it. You do not have to include any of the `tempdb` data files because `tempdb` is recreated each time that SQL Server is started.

Oracle

Generally, because of the way that Oracle Automatic Storage Management (ASM) is set up by default, Oracle uses more than one consistency group. Typically, it uses one consistency group for the data files and one for the online and archived redo log files. If the database is in `ARCHIVELOG` mode, you might find it useful to create a third Oracle ASM disk group to separate the online logs from the archive logs.

See Appendix A: Oracle Script for Snapshot Image Backup, for a script to:

1. Put the Oracle database in hot backup mode.
2. Take the Snapshot images by using SANtricity CLI commands.

3. Then take the database back out of hot backup mode.

5.3 Restore from a Snapshot Image

Snapshot images are useful any time that a volume needs to be rolled back to a known good data state at a specific point in time. For example, before you perform a risky operation on a volume, you can create a Snapshot image to enable the undo capability for the entire volume.

In a database environment, you should take the database offline before you roll back to a previous point in time. The rollback is initiated for a Snapshot image of a consistency group, which allows all or selected member volumes of the consistency group to be rolled back to a previous state.

Note: You can also roll back data by creating a Snapshot volume of a Snapshot image. With this method, you can retrieve deleted or modified files from that Snapshot volume (the base volume remains undisturbed).

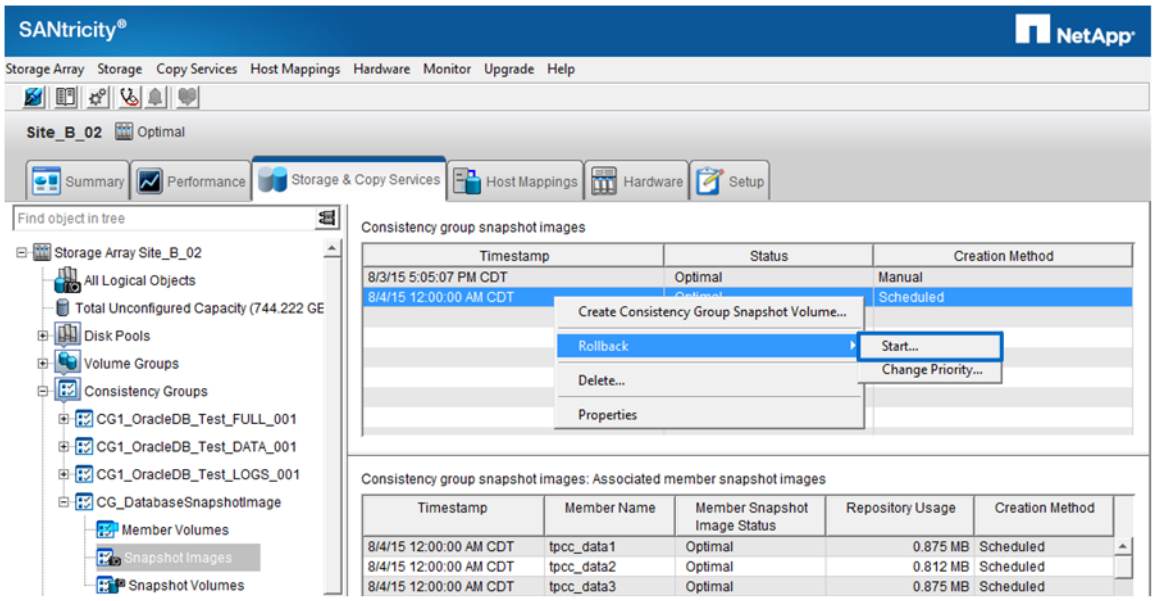
The rollback operation changes the content of a base volume to match the content that is saved in a Snapshot image. For prerequisites and detailed guidelines of the restore process from a Snapshot image, go to the SANtricity Storage Manager online help.

Start Rollback Operation

To start a rollback operation, complete the following steps:

Note: The database to be restored should be offline before you start this operation.

1. In the AMW Storage & Copy Services tab, right-click either the Consistency Group Snapshot image or the Snapshot image of a base volume to use in the rollback and select Rollback > Start.

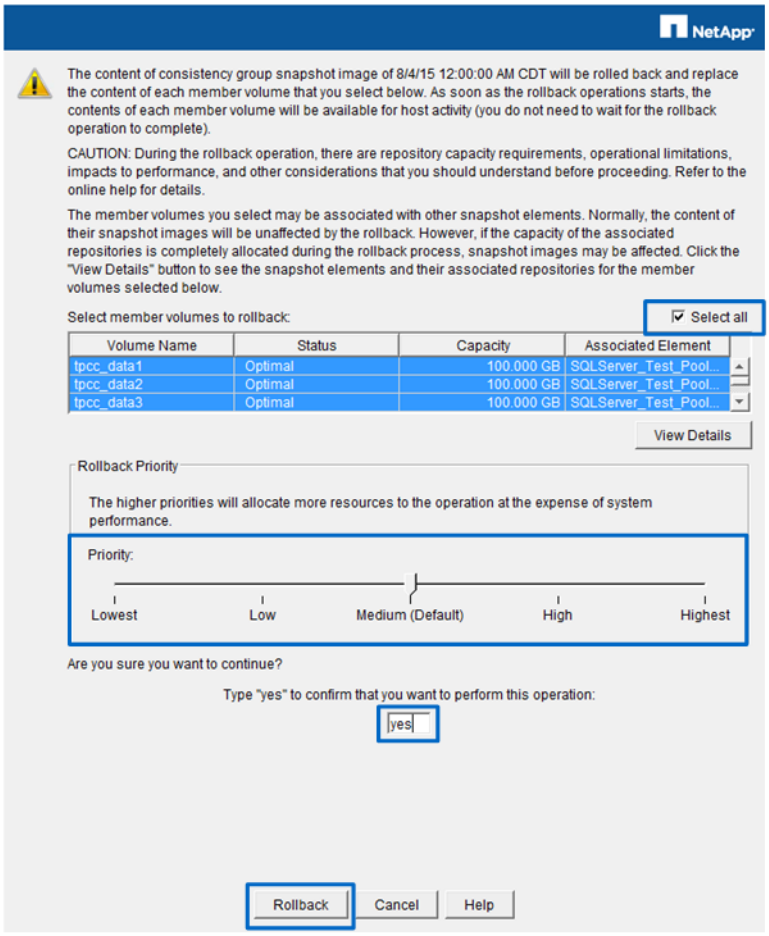


2. The system displays the Confirm Rollback Consistency Group Snapshot Image dialog box. The rollback in this example is from a scheduled consistency group Snapshot image:
   a. Select the member volumes to roll back. Generally, choose Select All to roll back all base volumes to a previous point in time.
   b. In the Rollback Priority pane, drag the slider to set a priority for the rollback operation. Five priority rates are available: lowest, low, medium, high, and highest.

   Note: If the priority is set at the lowest rate, I/O activity is prioritized, and the rollback operation takes a longer time to complete. If the priority is set at the highest rate, the rollback operation

is prioritized, but I/O activity for the storage system might be affected. The difference in the lowest to highest priority can mean hours to minutes for the operation to complete.

c. To confirm and start the rollback operation, type `yes` in the text box and click Rollback.

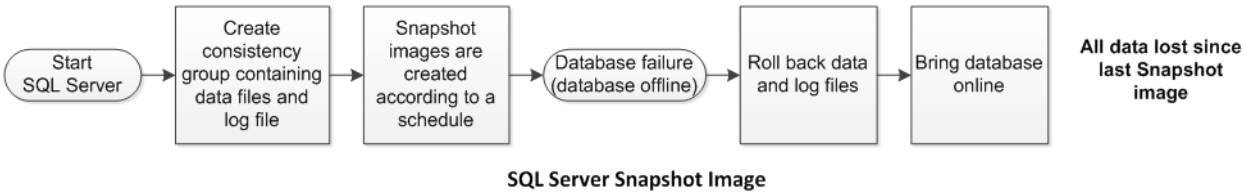


Note: After the rollback operation completes, you can bring the database back online.

SQL Server

Figure 10 is a flowchart of the backup and restore process by using Snapshot images for a SQL Server database. When you bring the SQL Server database back online, it is consistent only to the point in time of the last Snapshot image.

Figure 10) Flowchart of SQL Server database restore with Snapshot image.

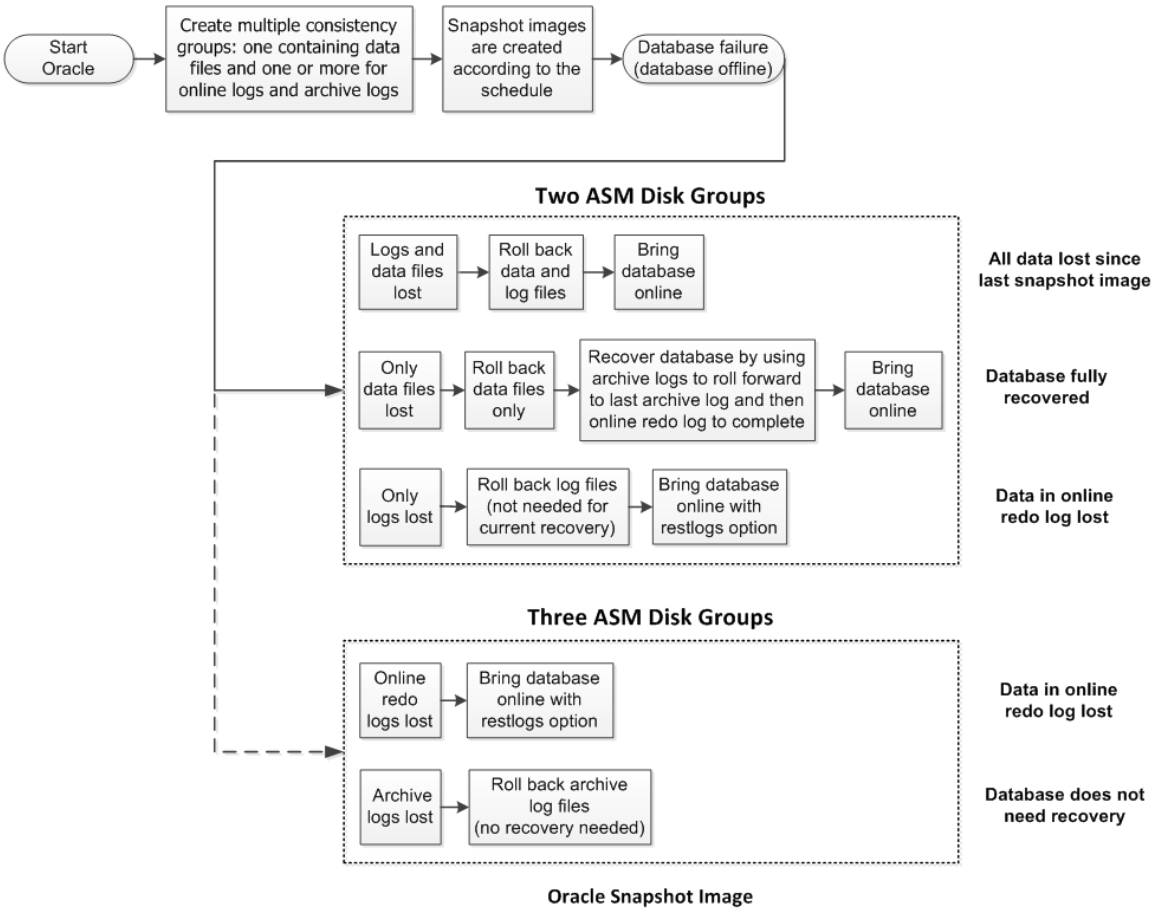


Oracle

Figure 11 is a flowchart of the backup and restore process by using Snapshot images for an Oracle database. Multiple consistency groups are used to facilitate bringing the database back in a consistent state based on the loss of data files, log files, or both.

To minimize data loss, it can be advantageous to separate Snapshot images of online redo logs and archived redo logs, as shown in Figure 11 for three ASM disk groups, but it is not required.

Figure 11) Flowchart of Oracle database restore with Snapshot image.



5.4 Asynchronous Mirroring to a Disaster Recovery Site

Asynchronous mirroring is database-friendly in the sense that you can create an asynchronous mirror group (consistency group) that contains all the database volumes that are required for a specific database. Using asynchronous mirroring to a secondary site is very similar to using Snapshot images for disaster recovery. The volumes to be replicated all need to be put into an asynchronous mirror group (think consistency group) and then scheduled to replicate on some defined schedule.

Activate Asynchronous Mirroring

Before you can activate asynchronous mirroring:

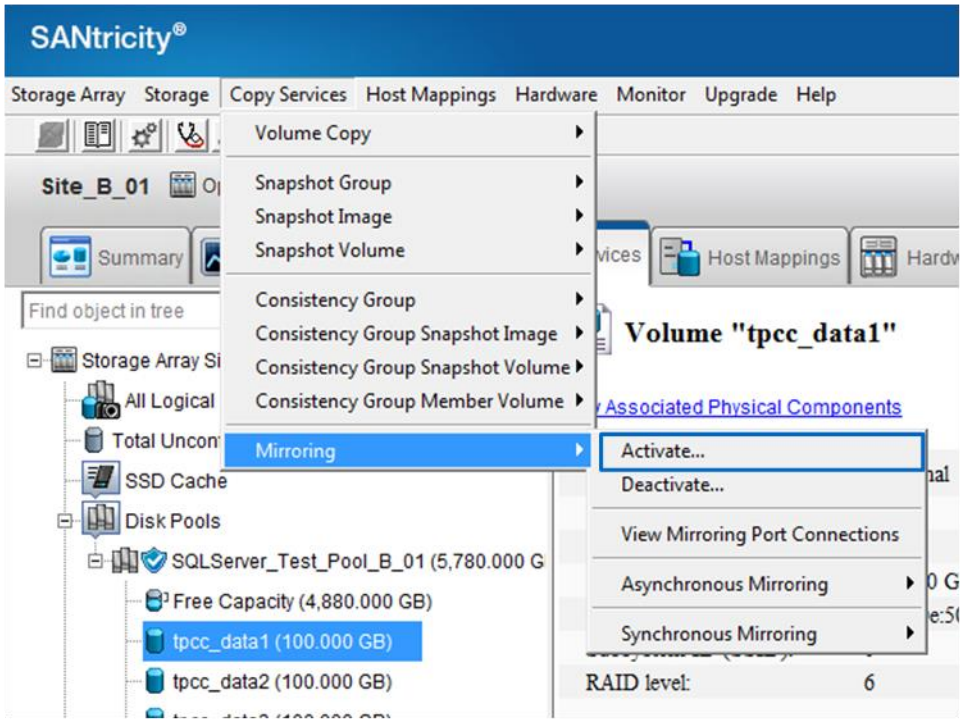
- Asynchronous mirroring using FC requires the use of port 4 of both controller A and controller B of each array. An iSCSI implementation does not require dedicated ports.

- Adequate space must exist for the mirror repository volumes on both arrays. By default, each array creates a repository that uses 20% of the volume size for each volume that is being mirrored. More space might also be required.
- The necessary mirrored volumes must exist on the secondary array, and their size must be equal to or greater than the size of the volumes on the primary array.
- Volumes (LUNs) on the secondary array should not be mounted.

Note: For a complete list of restrictions and guidelines, go to the SANtricity Storage Manager online help.

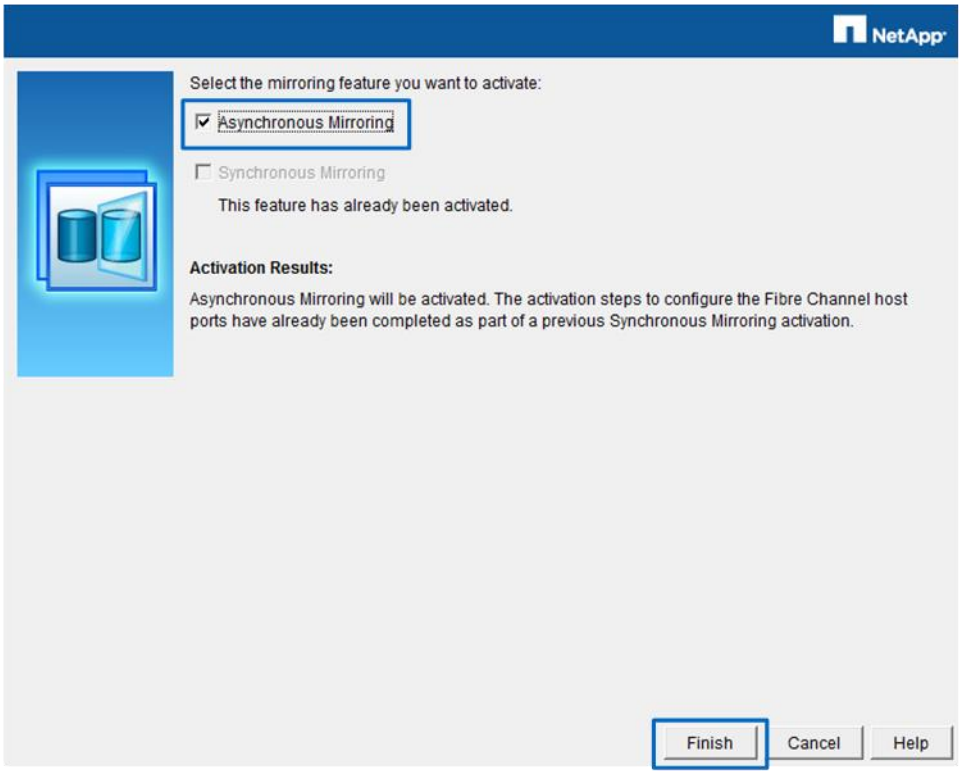
To active asynchronous mirroring, perform the following steps on both the primary and the secondary array:

1. Select Copy Services > Mirroring > Activate.



2. Select Asynchronous Mirroring and click Finish.

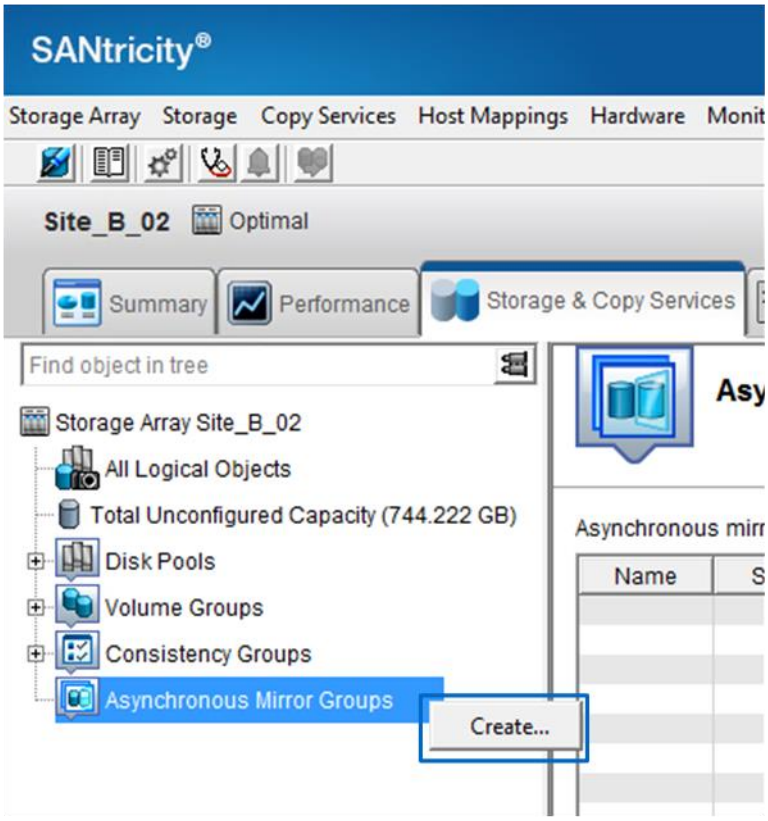
   Note: Both asynchronous and synchronous mirroring can be active simultaneously.



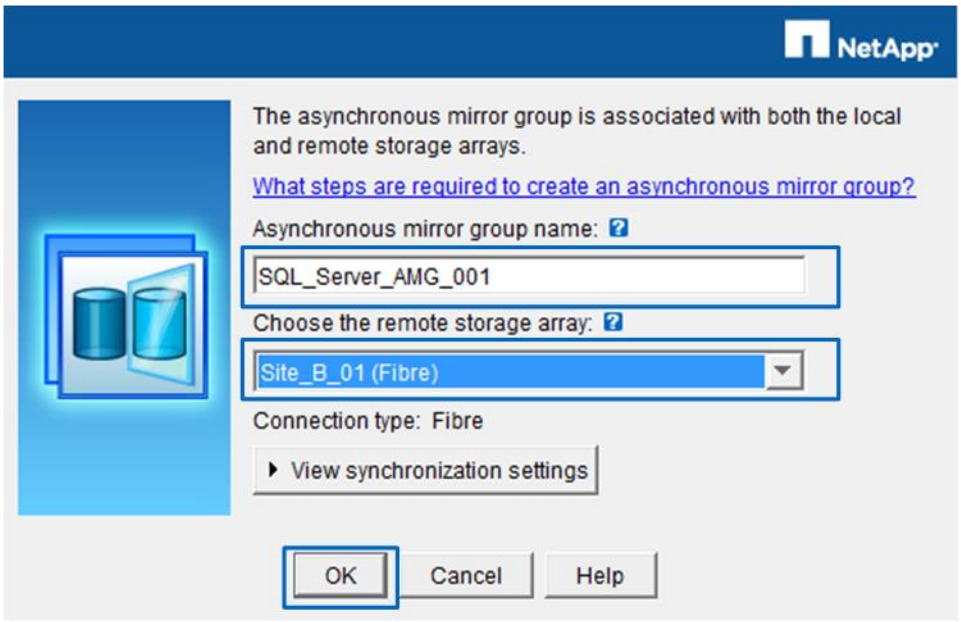
3. Click OK to acknowledge the activation of asynchronous mirroring.

Create Mirrored Volume Pairs

1. On the local (primary) storage array, right-click Asynchronous Mirror Groups in the object tree and click Create.

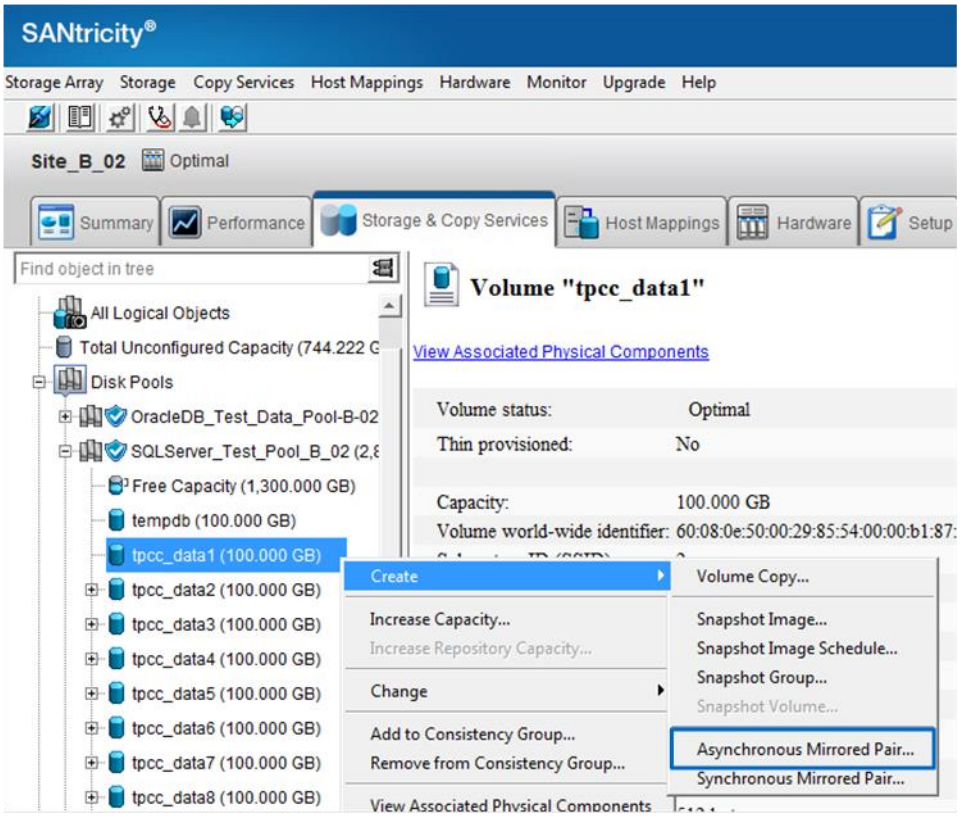


2. Provide a name for the asynchronous mirror group, select the remote storage array from the drop-down list, and click OK.

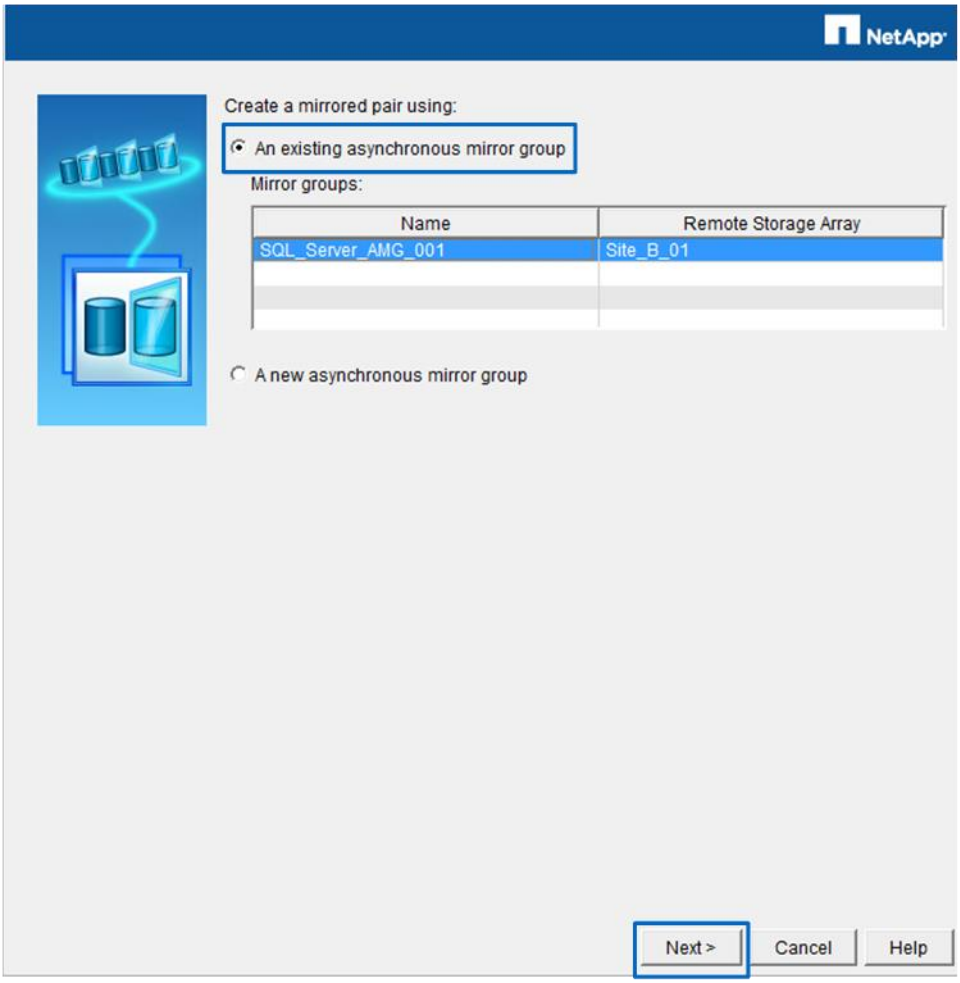


3. Click No. Add all the mirrored pairs by using the same method.

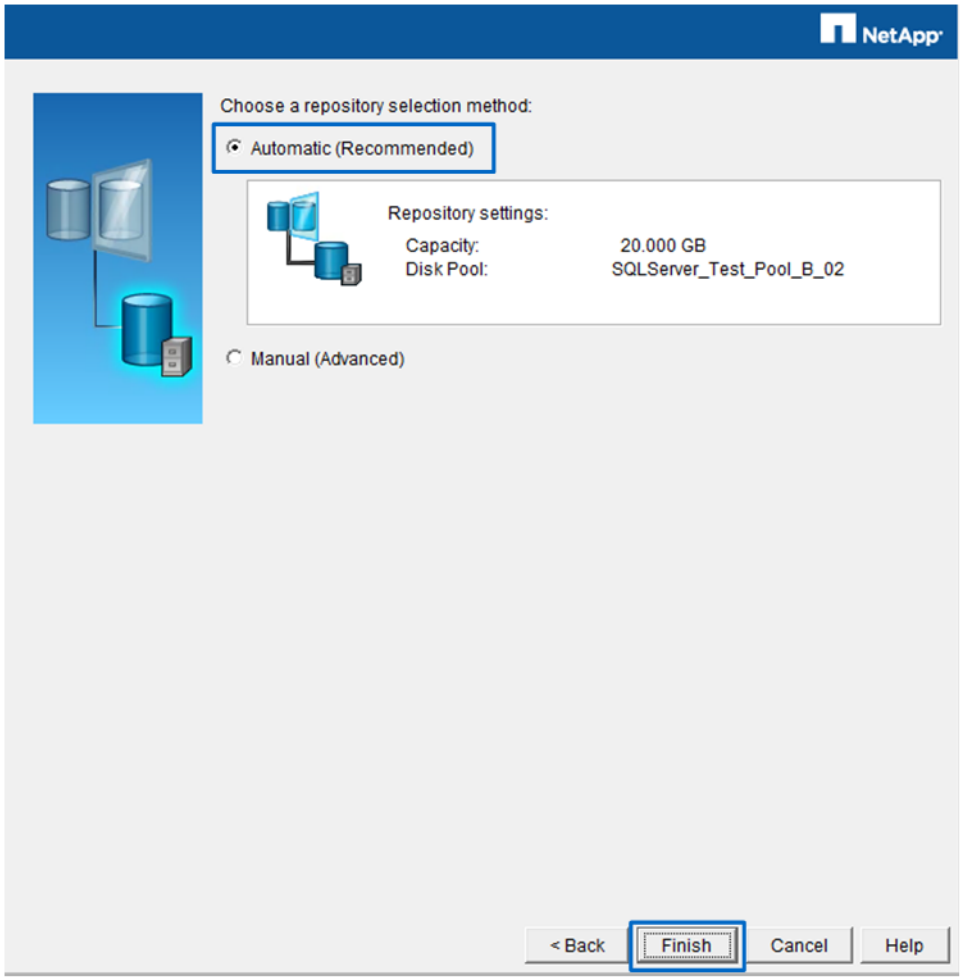
4. In the object tree, right-click the first volume to be added to the asynchronous mirror group. Select Create > Asynchronous Mirrored Pair.



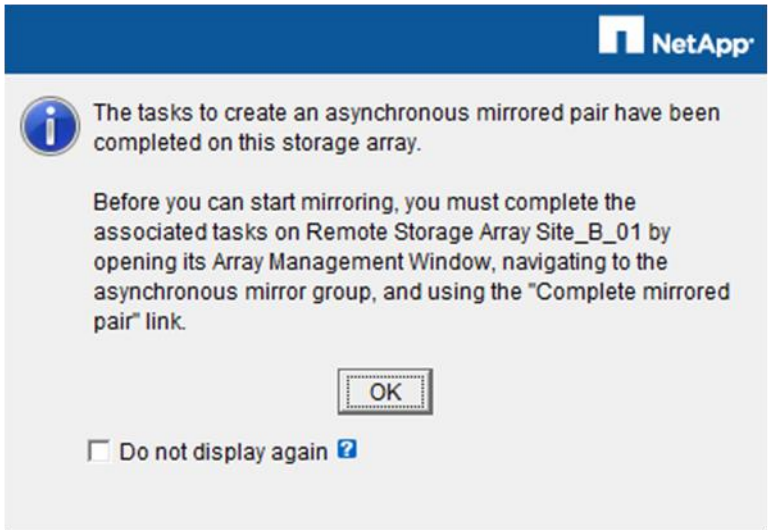
5. Under Create a Mirrored Pair Using, select An Existing Asynchronous Mirror Group. Select the mirror group that you created earlier and click Next.



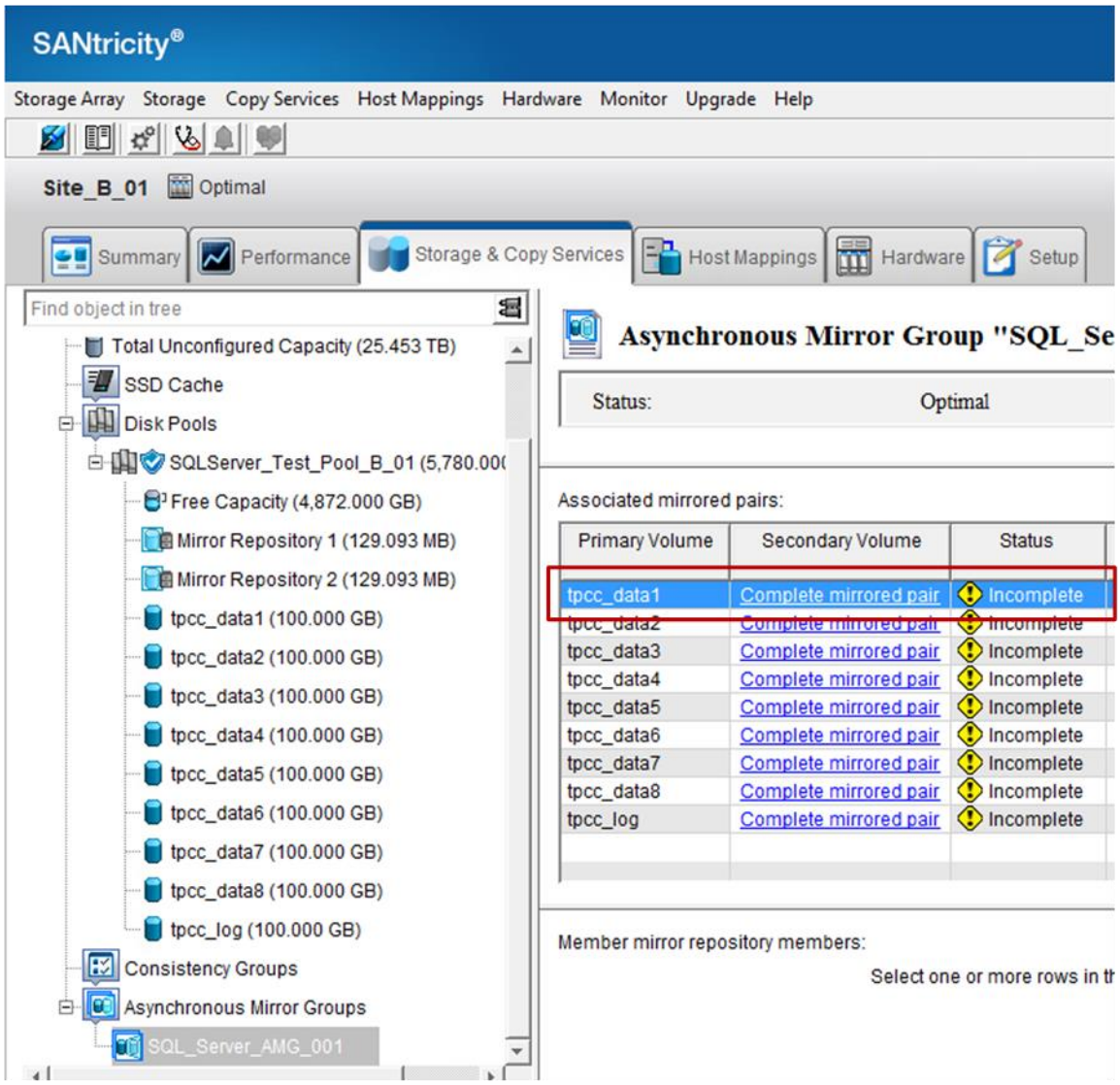
6. Select the Automatic (Recommended) method of repository selection and click Finish.



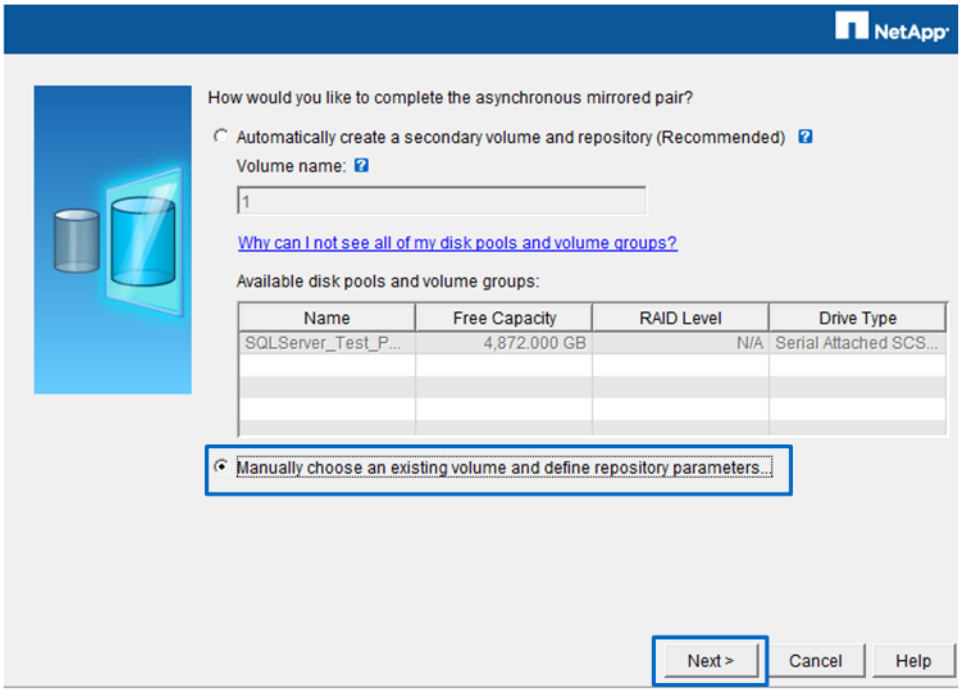
7. To acknowledge the message to complete the associated tasks on the remote storage array, click OK. Repeat steps 4 through 7 for each volume to be added.



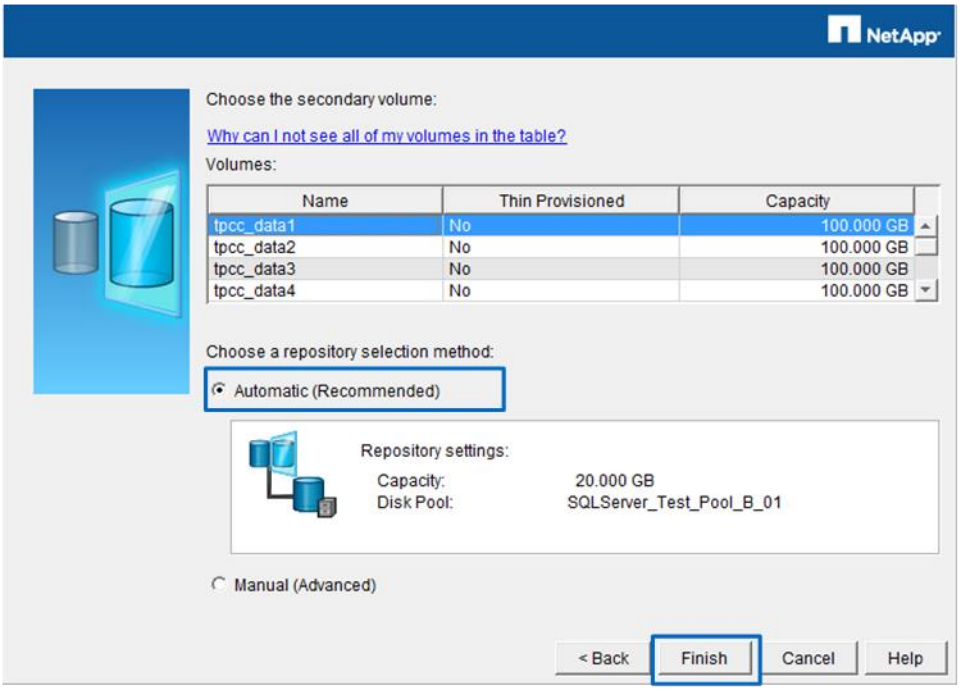
8. On the remote (secondary) storage array, in the object tree, select the asynchronous mirror group with the name that you entered earlier for the primary array. In the right pane, select Complete Mirrored Pair for the primary volume that you selected earlier.



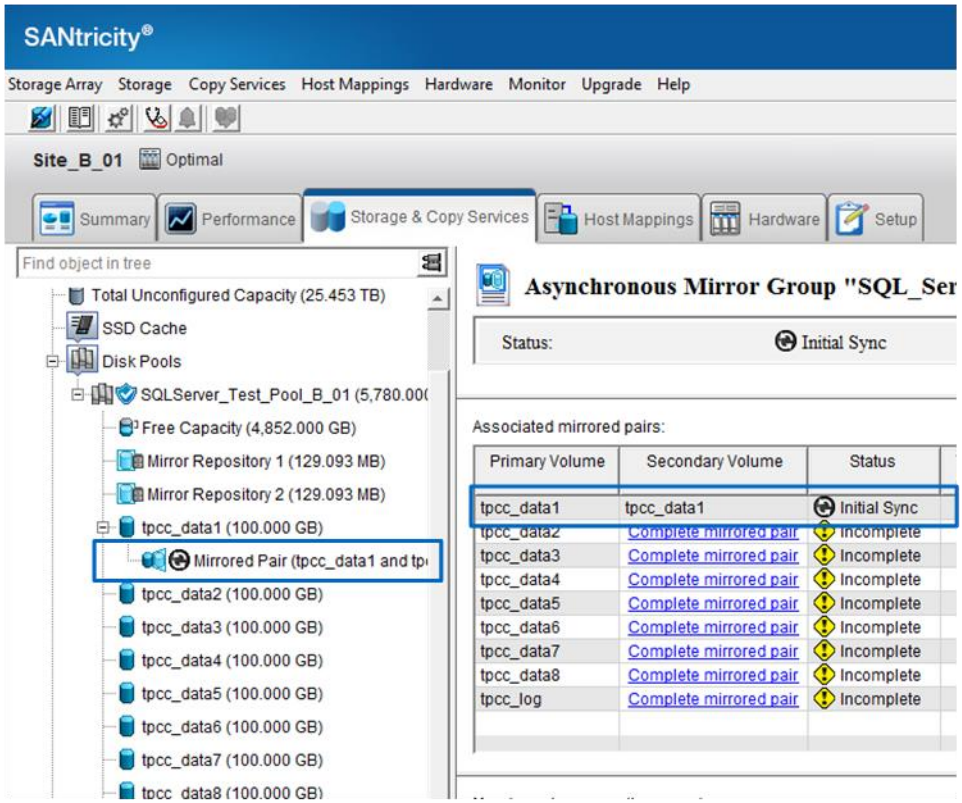
9. Select Manually Choose an Existing Volume and Define Repository Parameters. Click Next.



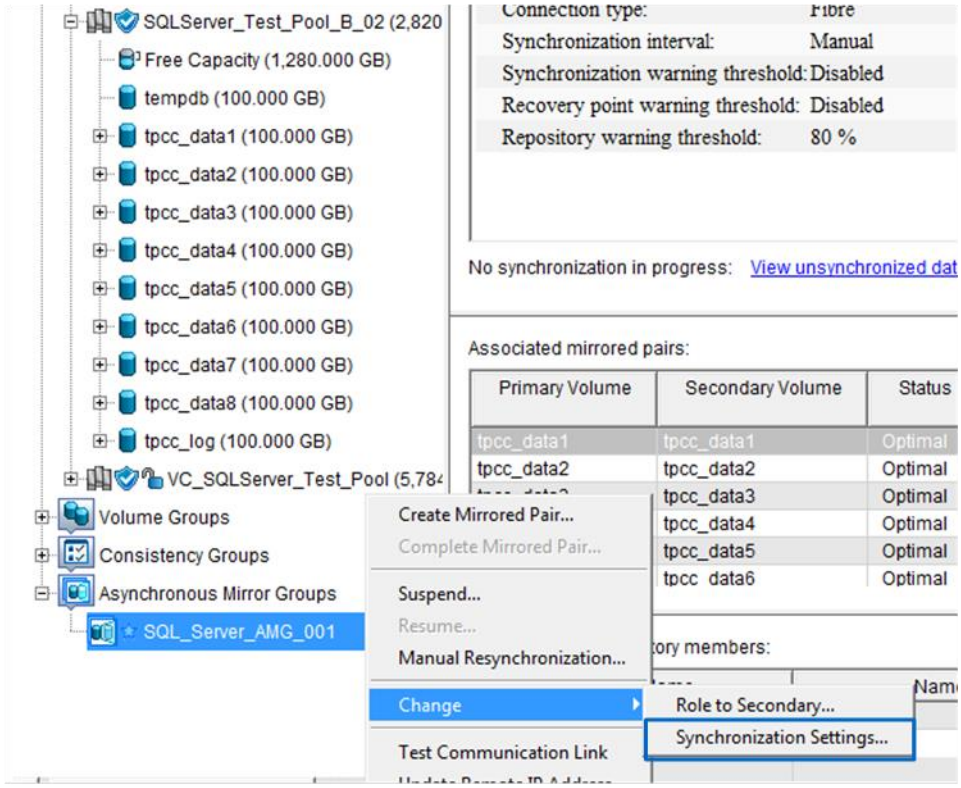
10. Select the volume that was created for this purpose earlier and select Automatic (Recommended) for repository creation. Click Finish.



11. The asynchronous mirrored pair immediately begins the synchronization process.

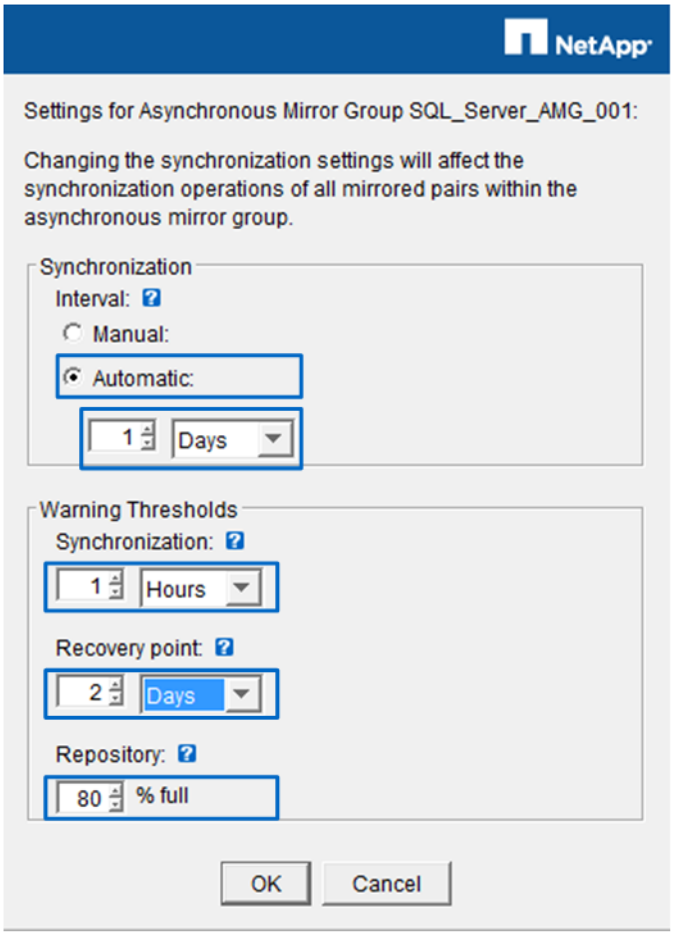


12. To set the synchronization settings, right-click the asynchronous mirror group in the object tree and select Change > Synchronization Settings.



13. Determine the synchronization interval that is correct for your environment. The minimum interval for asynchronous mirroring is 10 minutes. Set warning thresholds.

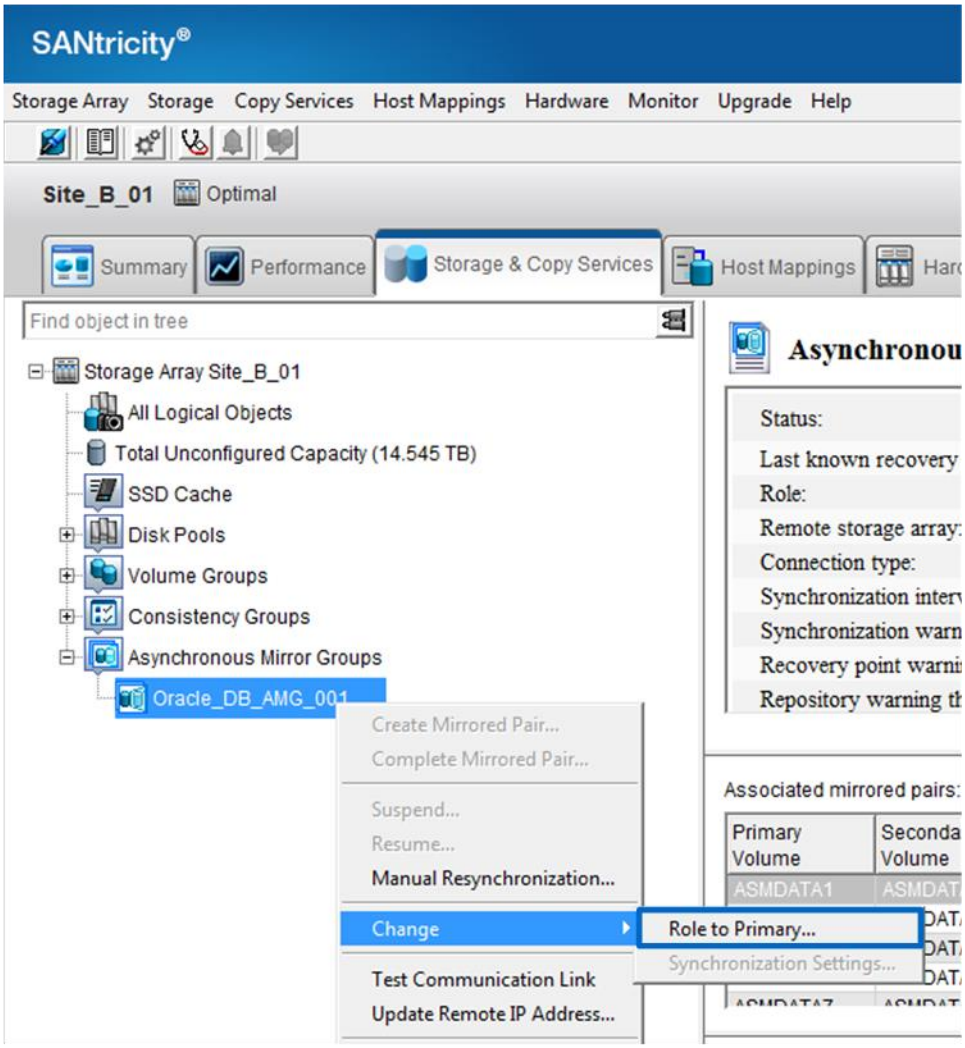
    Note: For further information, go to the SANtricity Storage Manager online help.



Activate the Secondary Site

If it becomes necessary to promote the secondary site to the primary site, perform the following steps:

1. Right-click the asynchronous mirror group that is currently being mirrored. Select Change > Role to Primary.



2. Map all promoted volumes to the secondary host.
3. Bring the database online in the appropriate manner for Oracle or SQL Server.

Note: When the original primary site is available again:

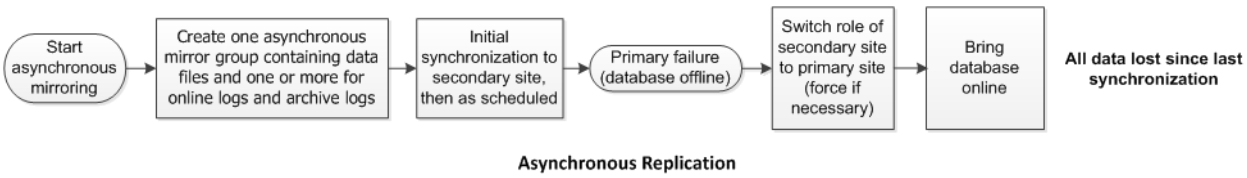
a. Reverse the roles as needed.
b. Manually resynchronize the stored changes.
c. Make sure that all database volumes are mapped.
d. Bring the database back online.

SQL Server

Asynchronous mirroring with the use of asynchronous mirror groups functions in a way that is very similar to how Snapshot images use consistency groups. Figure 12 is a flowchart that shows the synchronization and recovery process for a database.

Note: This process is valid for both Oracle and SQL Server databases.

Figure 12) Flowchart of database restore with asynchronous mirroring.



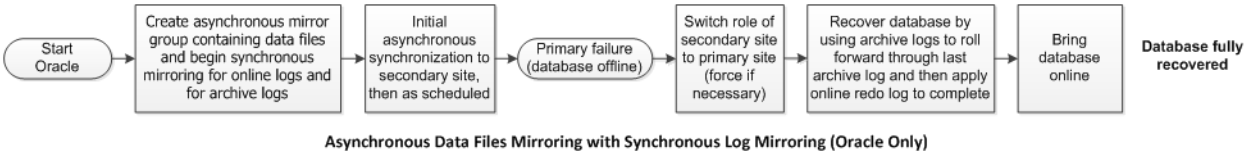
Oracle

A script for backing up Oracle databases while using asynchronous mirroring is provided in Appendix B: Oracle Scripts for Asynchronous Mirroring. This script puts the database in hot backup mode, performs a manual synchronization of the database, and then takes the database out of hot backup mode. A database backed up in this way is recoverable only to the last synchronization point, as shown in Figure 12.

A second script is also provided in Appendix B to perform just the archiving of the log file. These two scripts can be scheduled independently to deliver a better RPO. For example, the datafile mirror update might be performed once per day, while the log mirror is updated every 15 minutes.

Because of the Oracle architecture, another method to mirror the database is to blend both asynchronous mirroring and synchronous mirroring. Synchronous mirroring is discussed in detail in section 5.5, Synchronous Mirroring to a Disaster Recovery Site. A flowchart of this process is shown in Figure 13.

Figure 13) Flowchart of asynchronous and synchronous mirroring used for database restore.



This method has two important advantages:

- The database is fully recoverable and consistent.
- The performance impact of synchronously replicating the entire database is reduced.

5.5 Synchronous Mirroring to a Disaster Recovery Site

Because all volumes and all writes from all volumes that are being mirrored are applied on the secondary array, there is no concept of a consistency group with synchronous mirroring.

Activate Synchronous Mirroring

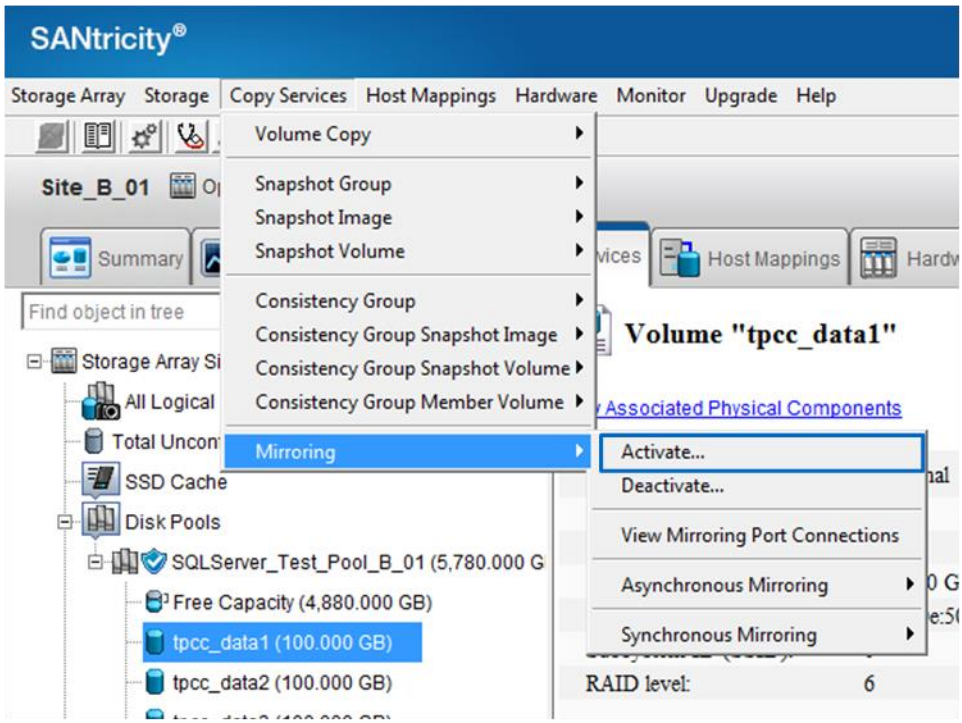
Before you can activate synchronous mirroring:

- Because synchronous mirroring on an EF-Series array is a premium feature, it requires a premium feature key.
- The FC host ports must already be activated. Synchronous mirroring requires the use of port 4 of both controller A and controller B of each array.
- Adequate space must exist for two mirror repository volumes on both arrays.
- The necessary mirrored volumes must exist on the secondary array, and their size must be equal to or greater than the size of the volumes on the primary array.
- Volumes (LUNs) on the secondary array should not be mounted.

Note: For a complete list of restrictions and guidelines, go to the SANtricity Storage Manager online help.

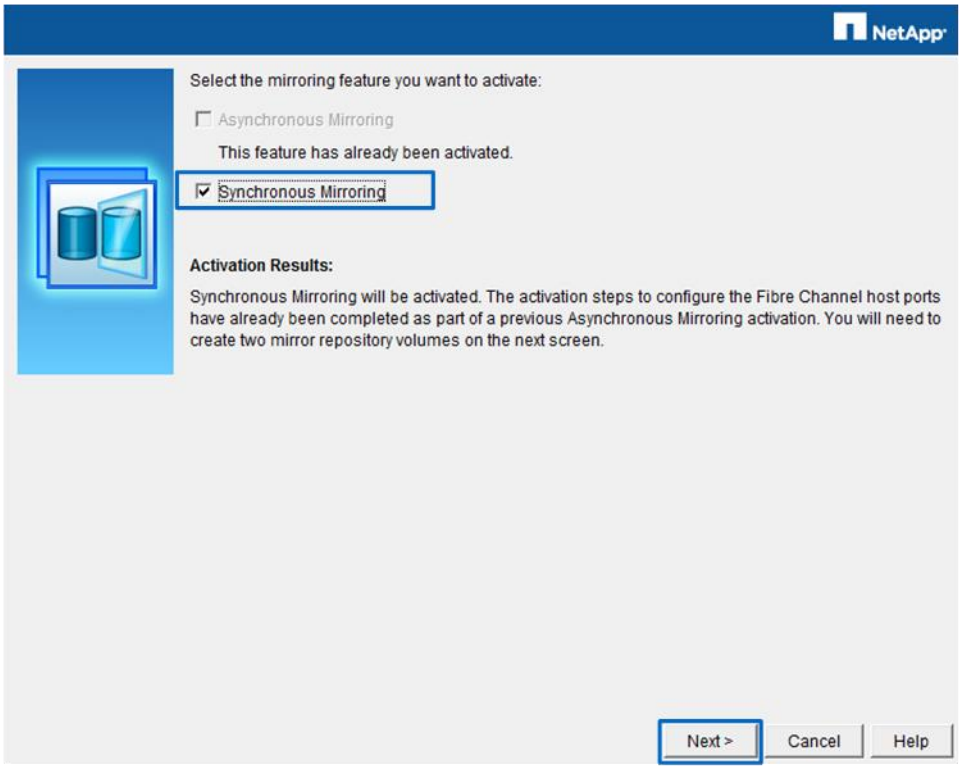
To activate synchronous mirroring, perform the following steps on both the primary and the secondary array:

1. Select Copy Services > Mirroring > Activate.

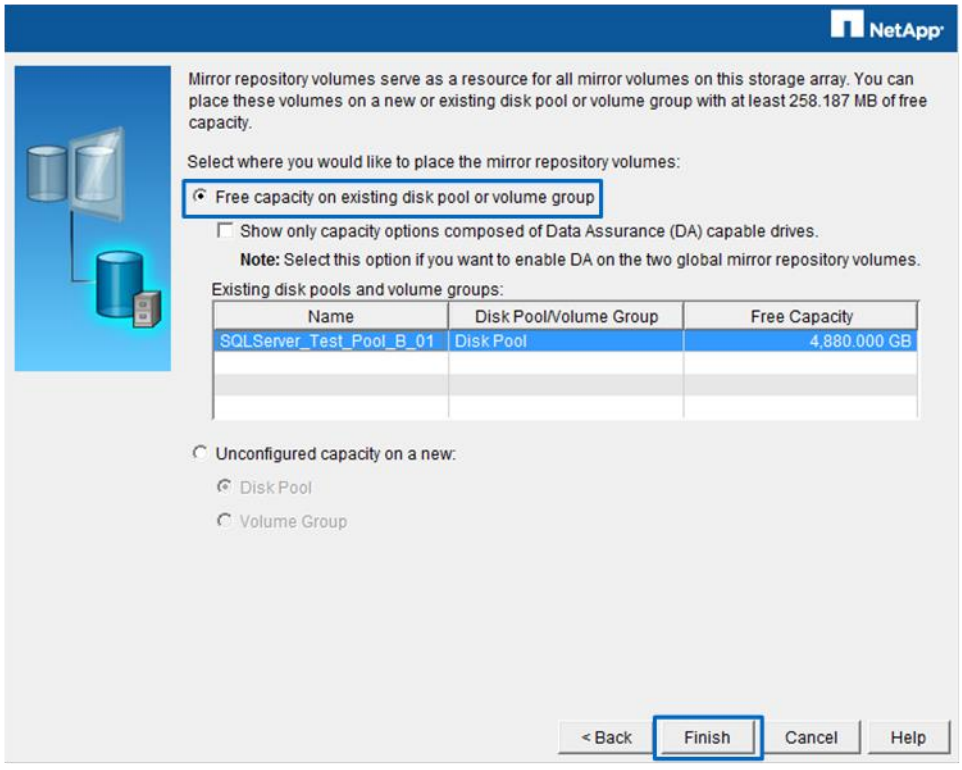


2. Select Synchronous Mirroring and click Next.

   Note: Both synchronous and asynchronous mirroring can be active simultaneously.

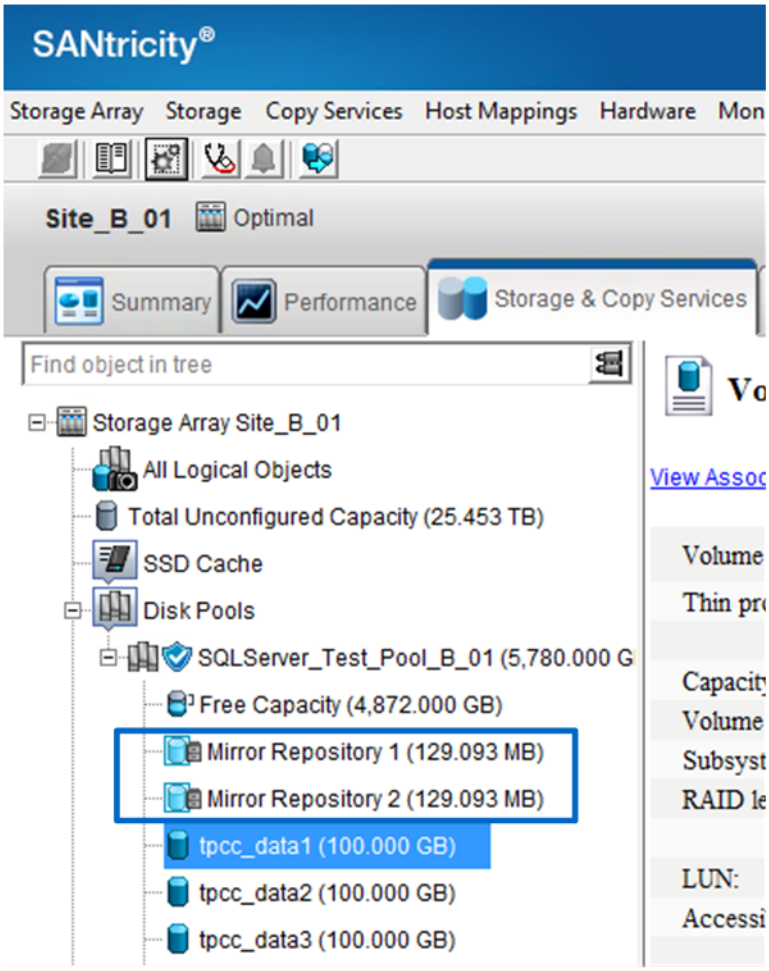


3. Select where to place the mirror repository volumes and click Finish.



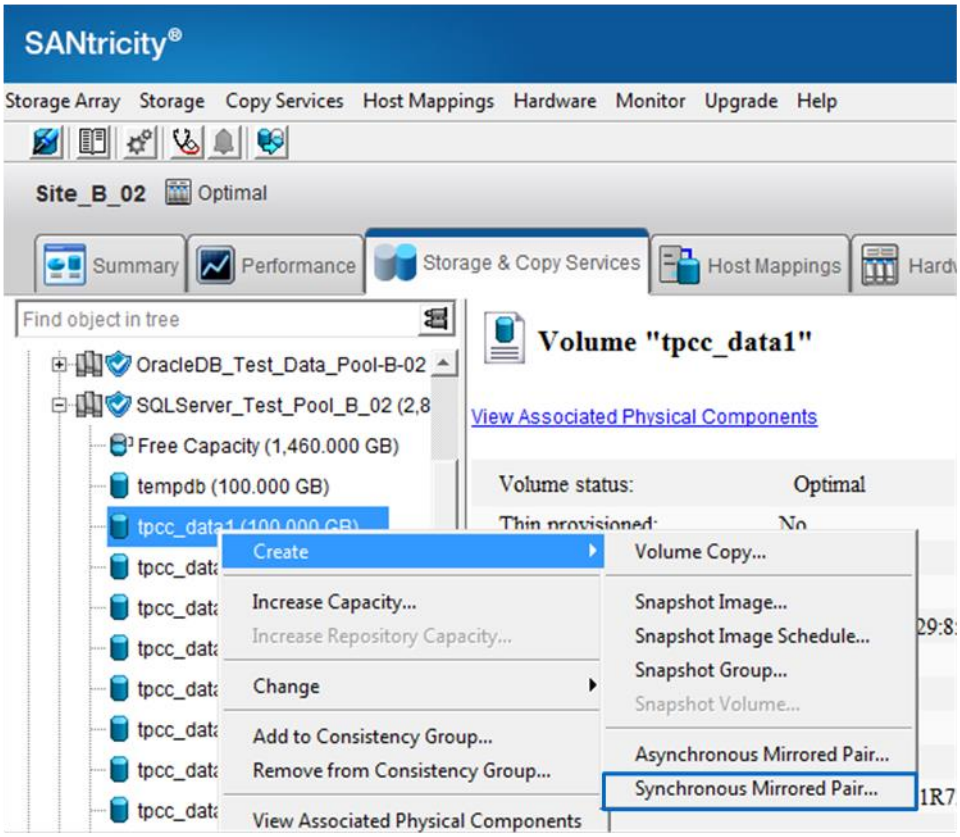
4. Click OK to acknowledge the message and to activate mirroring on the secondary array if it has not already been completed.

5. Note that two new mirror repositories have been created.

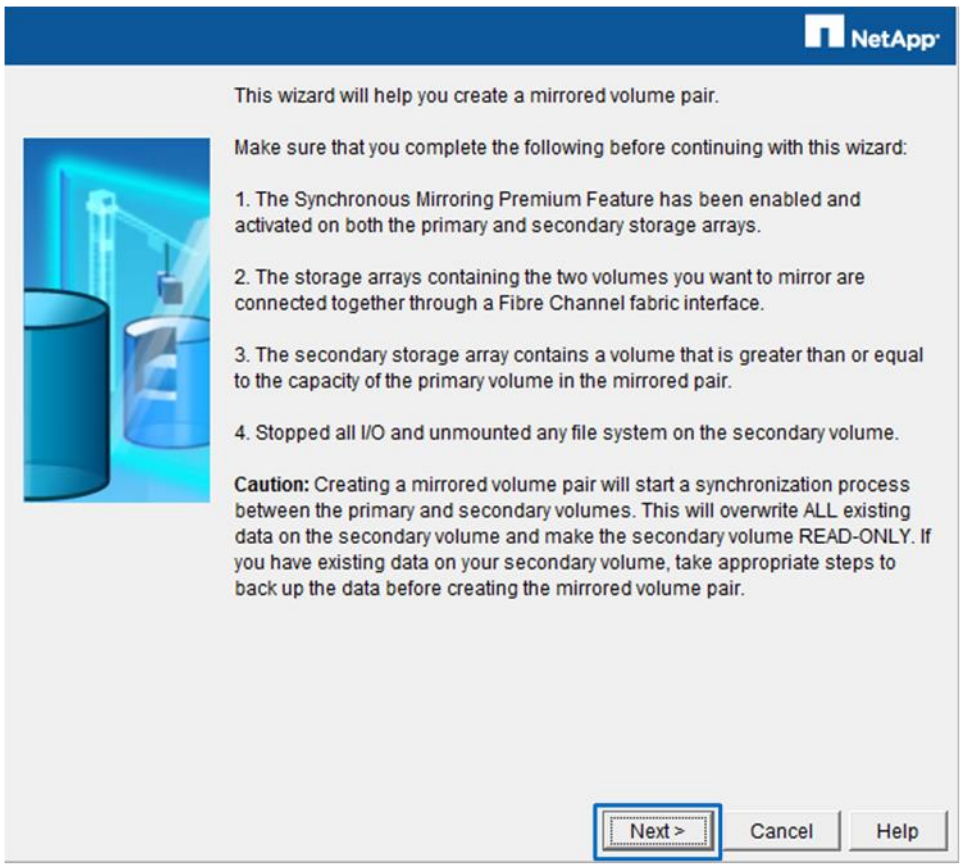


Create Mirrored Volume Pairs

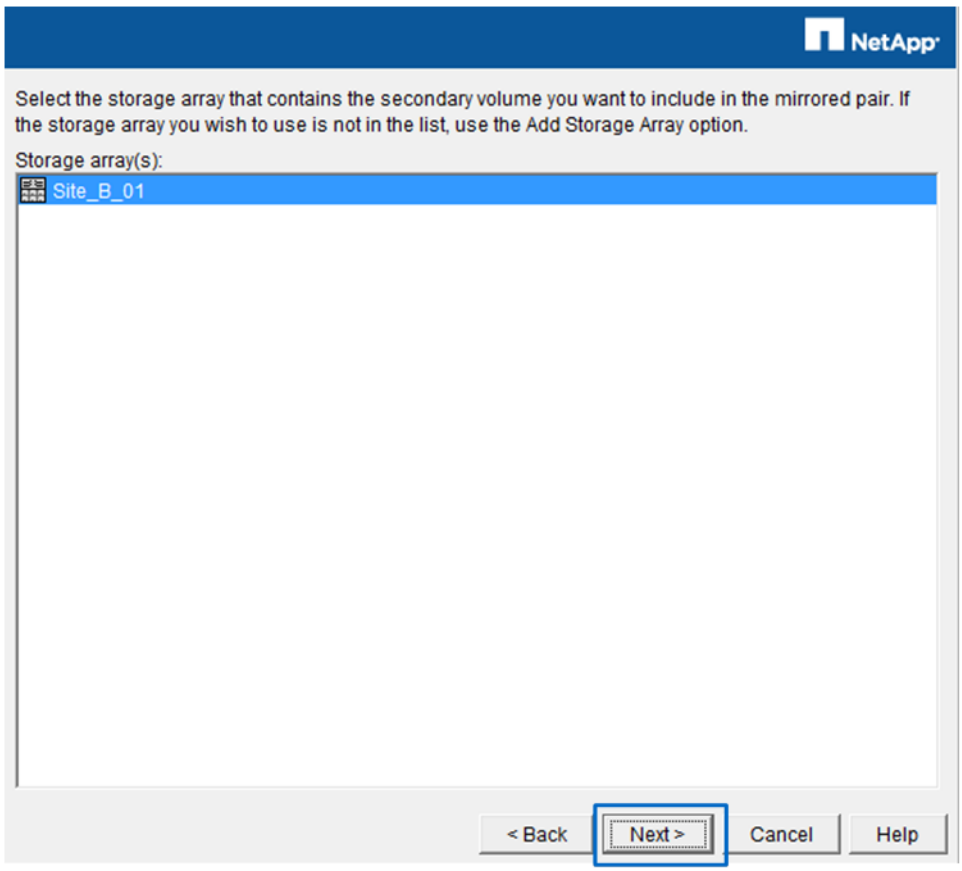
1. In the object tree, right-click the volume to be mirrored. Select Create > Synchronous Mirrored Pair.



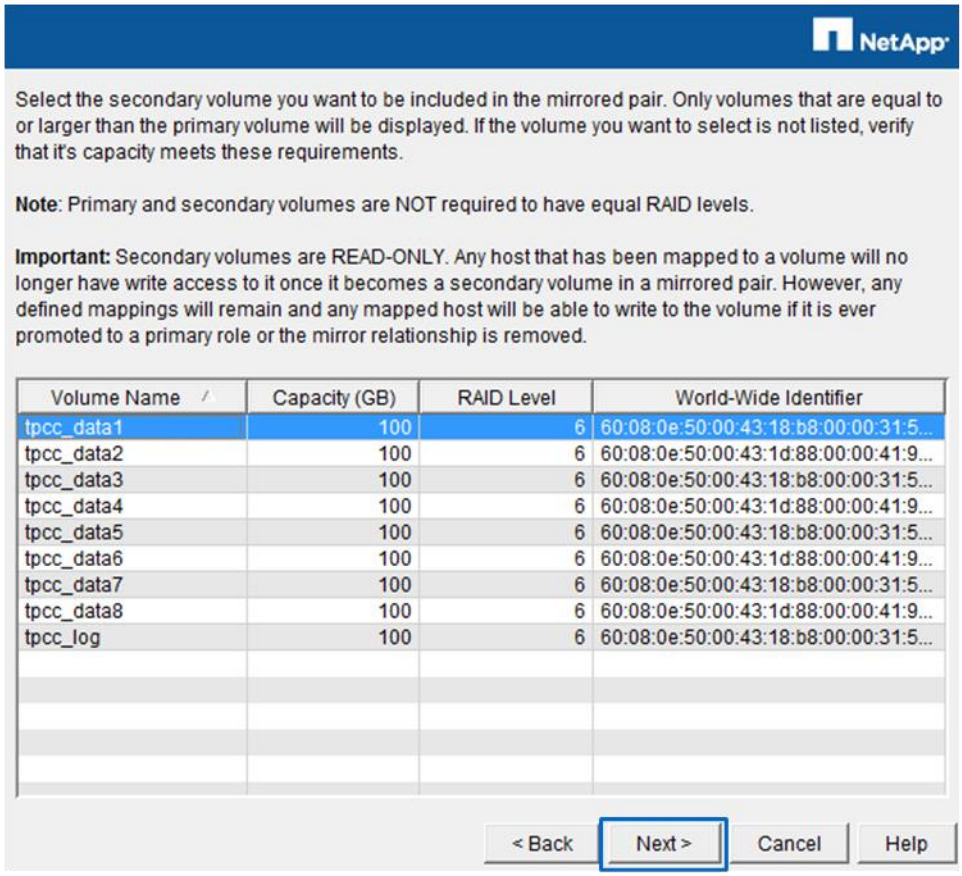
2. Read the prerequisites and the warnings. Click Next.



3. Select the storage array for the secondary site from the list shown and click Next.

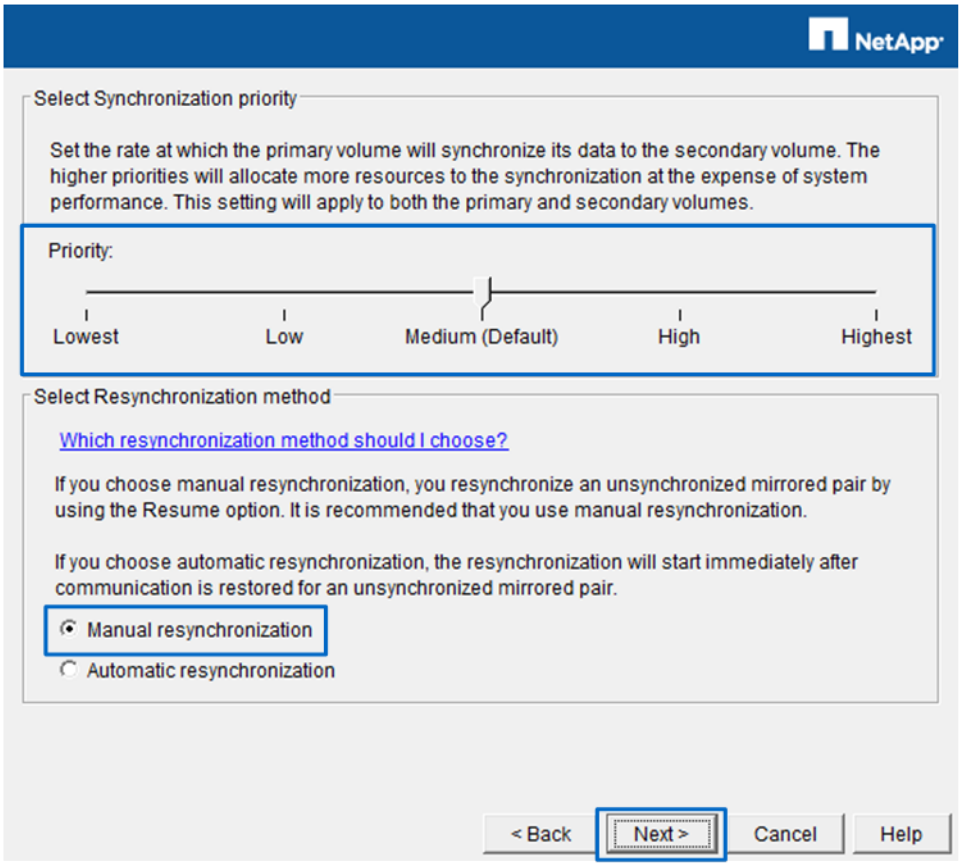


4. Select the secondary volume from the list shown and click Next.

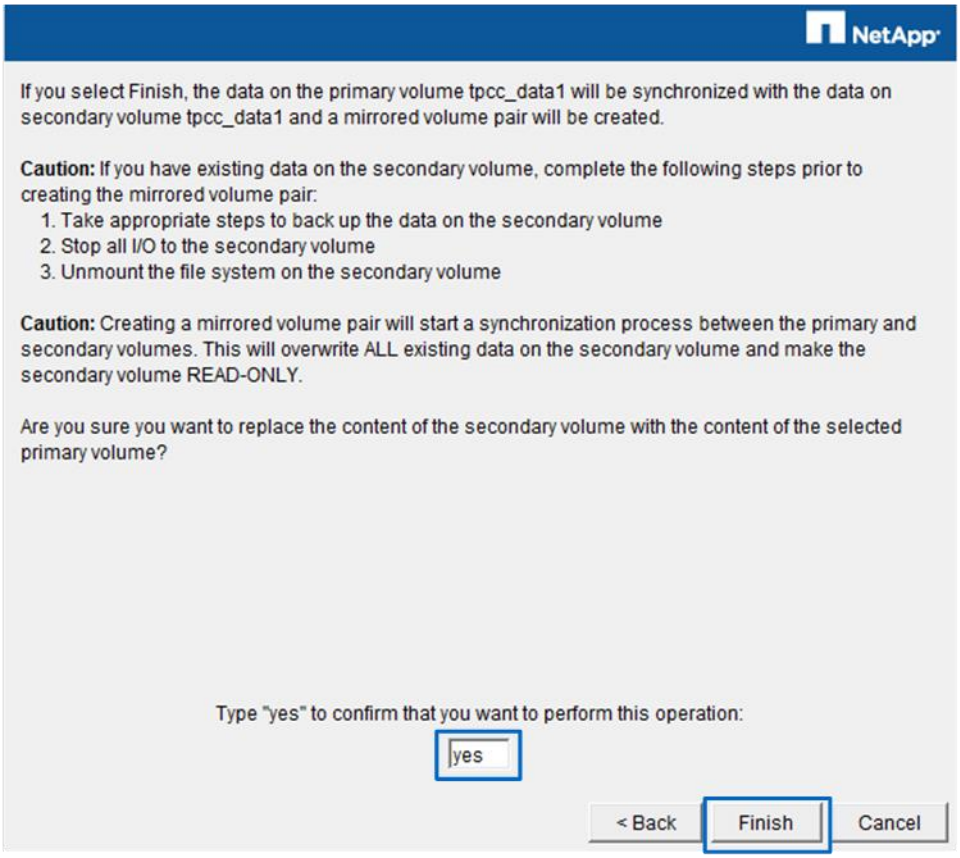


5. Select the synchronization priority and resynchronization method. Click Next.

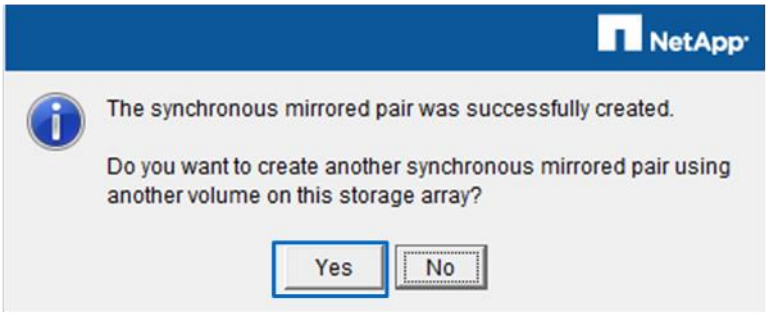
   Note: The synchronization priority has a direct effect on performance. The synchronization may take several hours using the default priority, but only a few minutes using the highest priority.



6. Read the cautions, type `yes` in the text box, and click Finish.



7. If there are more volumes to be mirrored, click Yes.

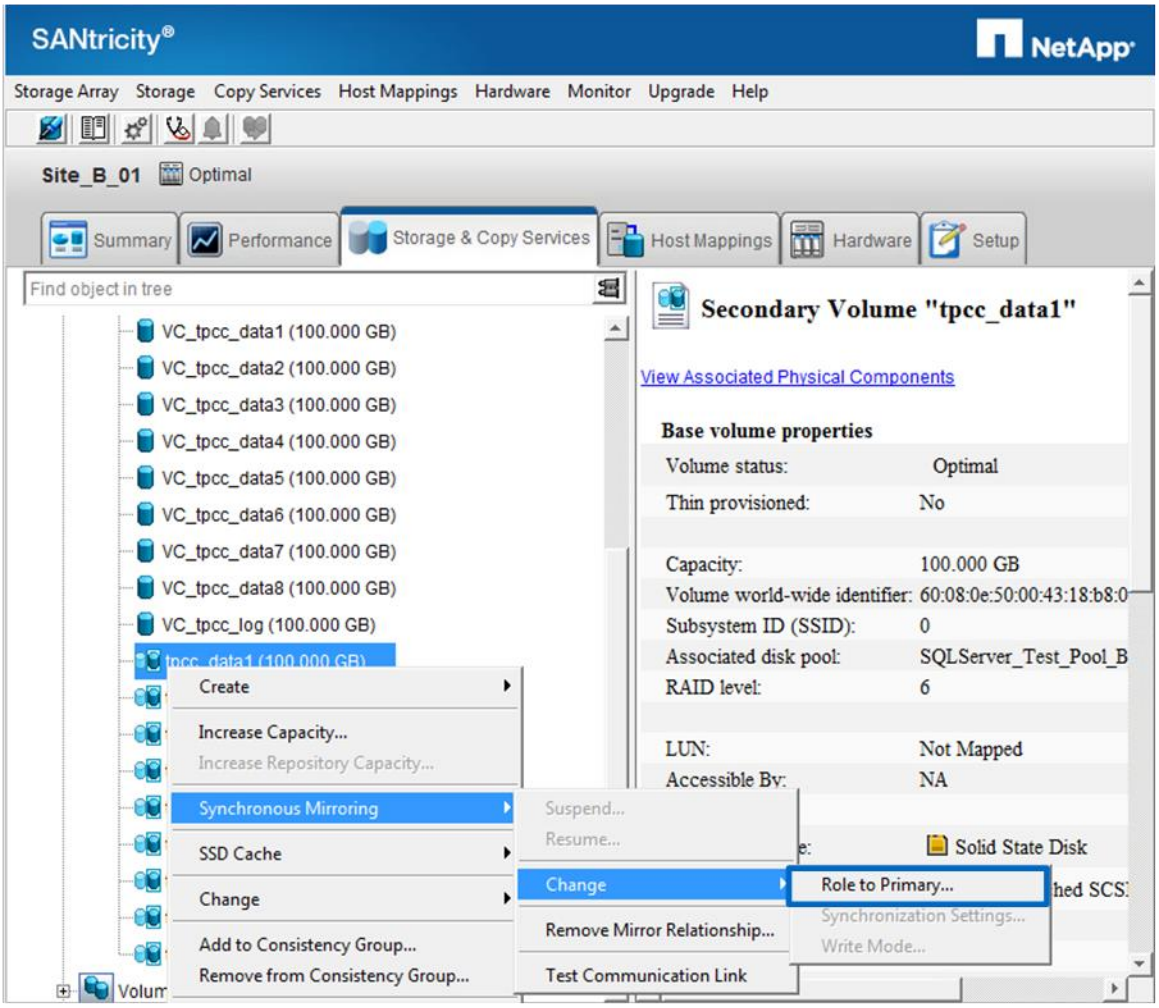


8. Select the next volume to be mirrored, click Next, and repeat steps 1 to 7.

Activate the Secondary Site: Role Reversal

If it becomes necessary to promote the secondary site to the primary site, perform the following steps:

1. Right-click a volume that is currently being mirrored. Select Synchronous Mirroring > Change > Role to Primary. Perform this step for all the volumes of the database.



2. Map all promoted volumes to the secondary host.
3. Bring the database online in the appropriate manner for Oracle or SQL Server.

Note: When the original primary site is available again:

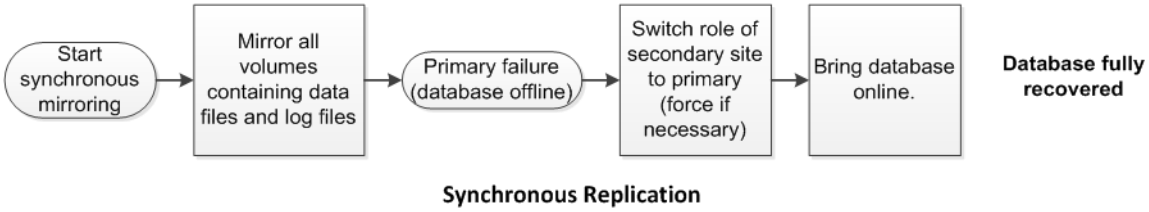
a. Reverse the roles as needed.
b. Manually resynchronize the volumes.
c. Make sure that all database volumes are mapped.
d. Bring the database back online.

SQL Server and Oracle

With synchronous mirroring, all data is mirrored immediately so that the database is consistent at all times. Figure 14 is a flowchart showing the synchronization and recovery process for a database.

Note: This process is valid for both Oracle and SQL Server databases.

Figure 14) Flowchart of database restore with synchronous mirroring.



5.6 Snapshot Volume and Volume Copy to Make the Secondary Site Useful

Before you use the Snapshot volume and volume copy features on the secondary site, you must prepare the mount points for the two database copies. The mirrored copy already has mount points stored as part of the database information. Therefore, you should prepare two additional sets of mount points, assuming that the copies are mounted to the same server as for the mirrored copy.

As an example for Windows, if the original mount point is `C:\MSSQL\Mirror_Copy_Mount_Point`, the Snapshot volume mount point might be `C:\MSSQL\SV_Mount_Point`.

After you mount the new volumes, you then have to change the database name and attach it. If it is a SQL Server database, you can run the following commands while you use `SQLCMD`:

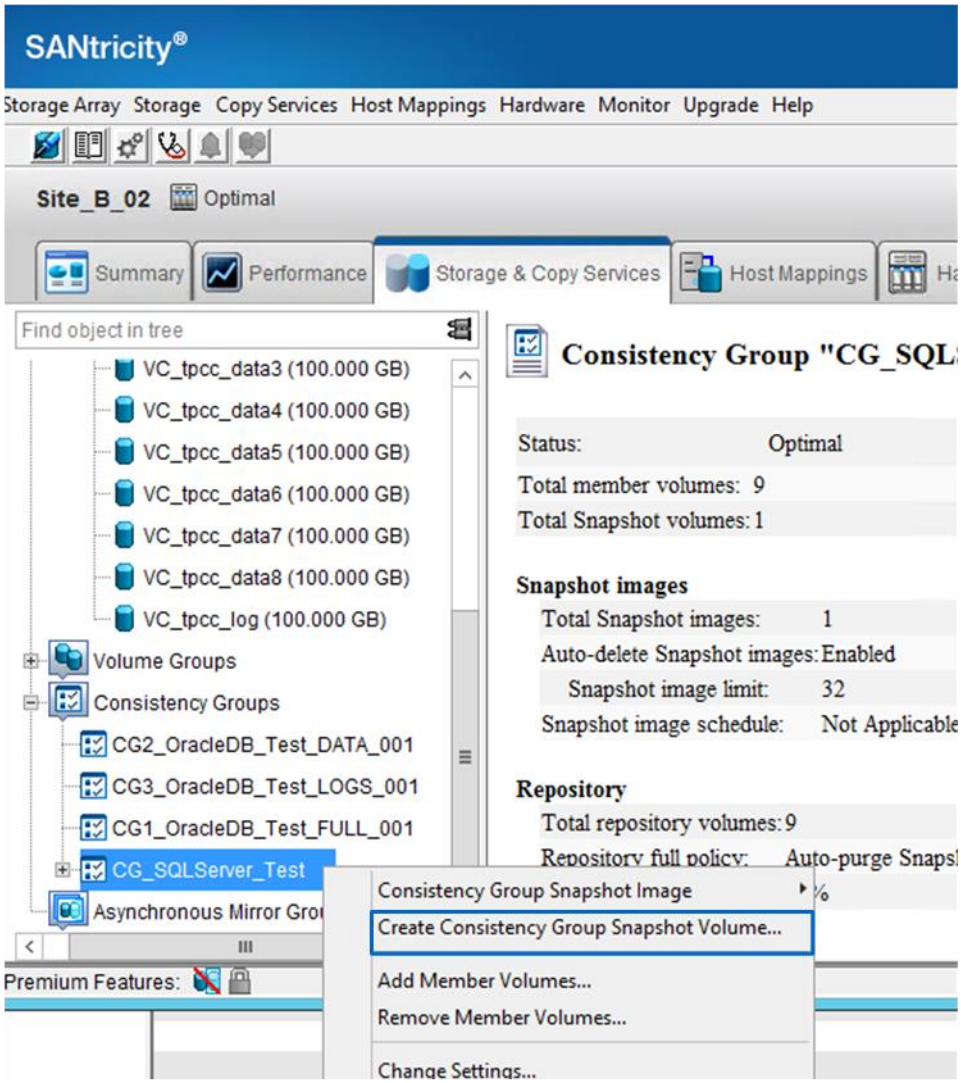
```
-- Rename database and attach to correct mount points
CREATE DATABASE [SV_tpcc] ON
(FILENAME = N'C:\MSSQL\SV_tpcc_data1.mdf'),
(FILENAME = N'C:\MSSQL\SV_tpcc_log.ldf'),
(FILENAME = N'C:\MSSQL\SV_tpcc_data2.ndf'),
(FILENAME = N'C:\MSSQL\SV_tpcc_data3.ndf'),
(FILENAME = N'C:\MSSQL\SV_tpcc_data4.ndf'),
(FILENAME = N'C:\MSSQL\SV_tpcc_data5.ndf'),
(FILENAME = N'C:\MSSQL\SV_tpcc_data6.ndf'),
(FILENAME = N'C:\MSSQL\SV_tpcc_data7.ndf'),
(FILENAME = N'C:\MSSQL\SV_tpcc_data8.ndf'),
(FILENAME = N'C:\MSSQL\SV_tpcc_data1.mdf')
FOR ATTACH
```

Note: You can perform a similar function for an Oracle database by editing the database `control` file.

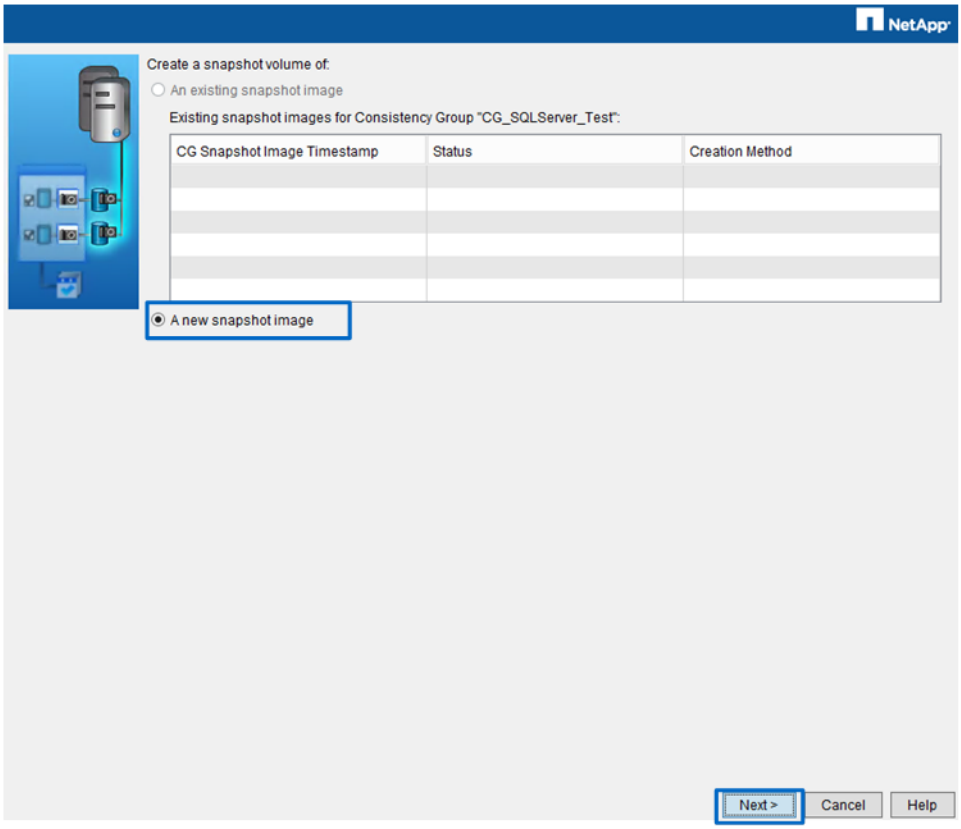
Create a Snapshot Volume of a Database

For a host to be able to read from an existing consistency group Snapshot image or be able to write to it, you must create a consistency group Snapshot volume from that Snapshot image. Perform the following steps:

1. Create a consistency group as described in section 5.2, Backup by Using Snapshot Images. Use all the mirrored volumes required for the database.
2. Right-click the consistency group with the volumes for the database. Select Create Consistency Group Snapshot Volume.



3. Verify that a new Snapshot image is used and click Next.



4. Configure the consistency group Snapshot volume settings:
   a. Enter a name for the consistency group Snapshot volume.

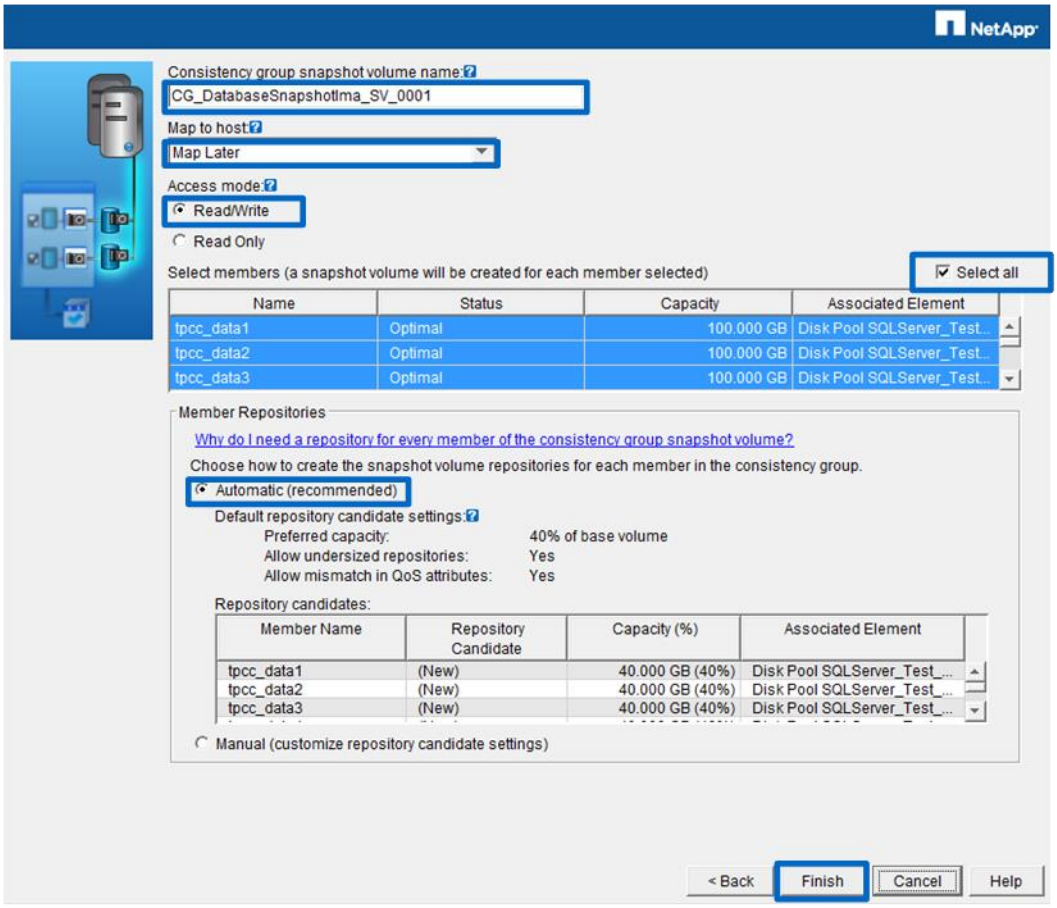
   Note: Consistency group Snapshot volume names must not exceed 30 characters and cannot contain spaces. Names can contain letters, numbers, underscores (_), hyphens (-), and pound signs (#).

   b. Select Map Later so that the Snapshot volume is not mapped to a host.
   c. Optional: Use SSD cache if it is available.
   d. Under Access Mode, select Read/Write for the new Snapshot volume.

   Note: To mount a database, Read/Write is required.

   e. Select members of the consistency group to be included in the consistency group Snapshot volume. By default, all members of the consistency group are selected.
   f. Select the Automatic (Recommended) option to create the repositories.
   g. Click Finish.

   Note: NetApp recommends using the automatic method for creating Snapshot image repositories. The manual option allows you to specify all the customizable settings for the repositories. The manual method is considered advanced, however, and only those administrators who understand drive redundancy and optimal drive configurations should use this method. For instructions about how to set the repository parameters, see the SANtricity Storage Manager online help and specifically the help topic "Creating a Consistency Group Repository (Manually)."



5. Map the volume to the host.
6. Mount the volumes and change the mount points.
7. Bring the database online with the new name and file locations.

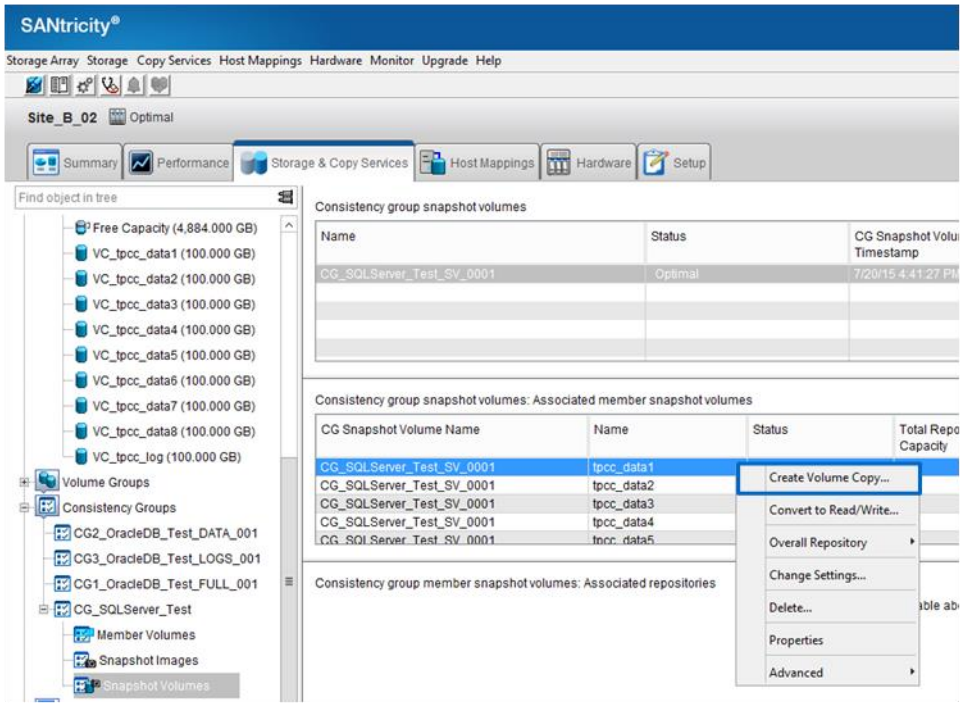
Create a Volume Copy of a Database

You can perform an offline volume copy only by using the Snapshot volumes that were created within a consistency group.

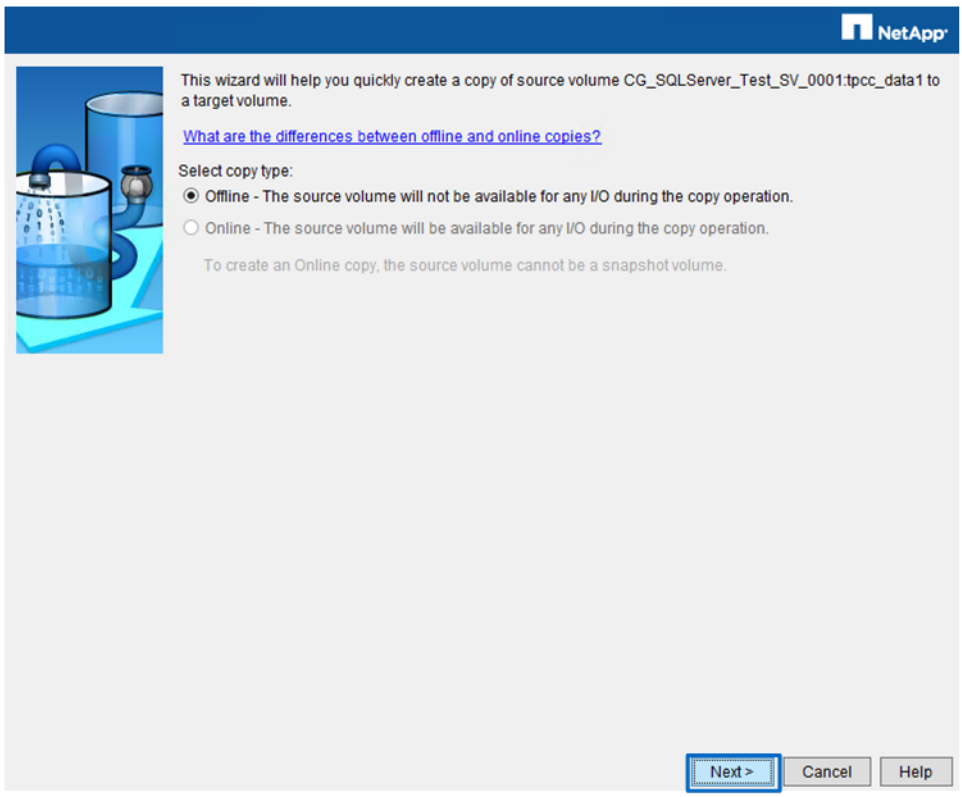
Note: Before you start the volume copy of the database volumes, target volumes of equal or greater capacity must all exist.

To create a volume copy of a database:

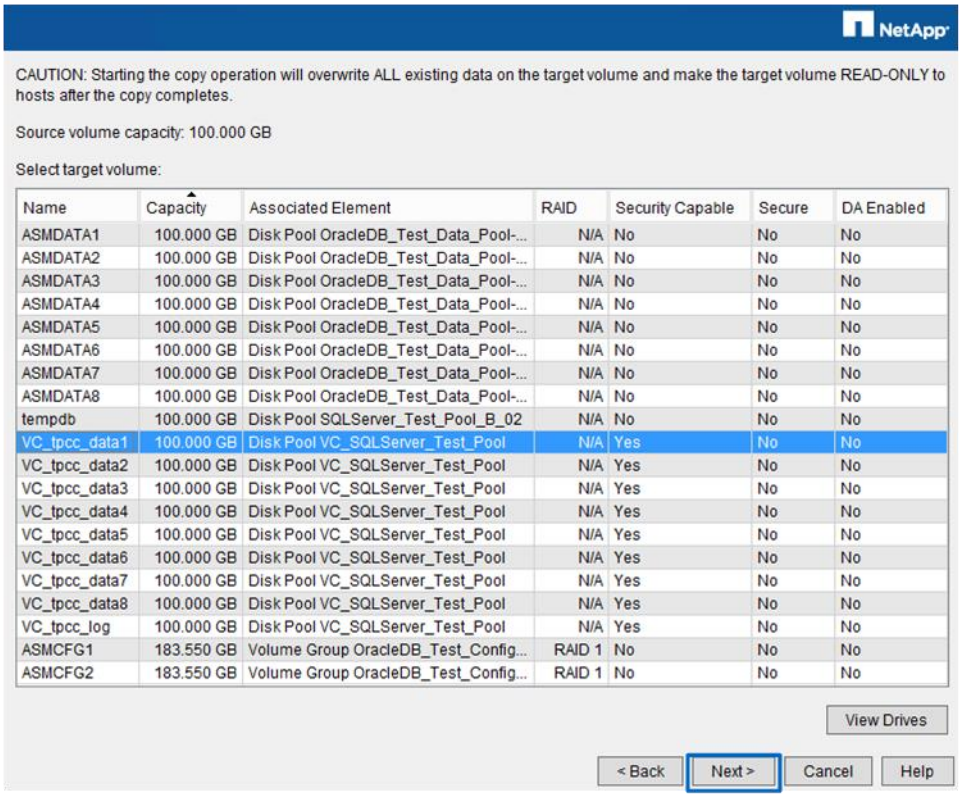
1. Expand the consistency group that you used to create the Snapshot volumes and select Snapshot Volumes in the object tree. In the properties pane of the Snapshot volumes, right-click the first Snapshot volume in the list and select Create Volume Copy.



2. An offline volume copy is already selected. Click Next.

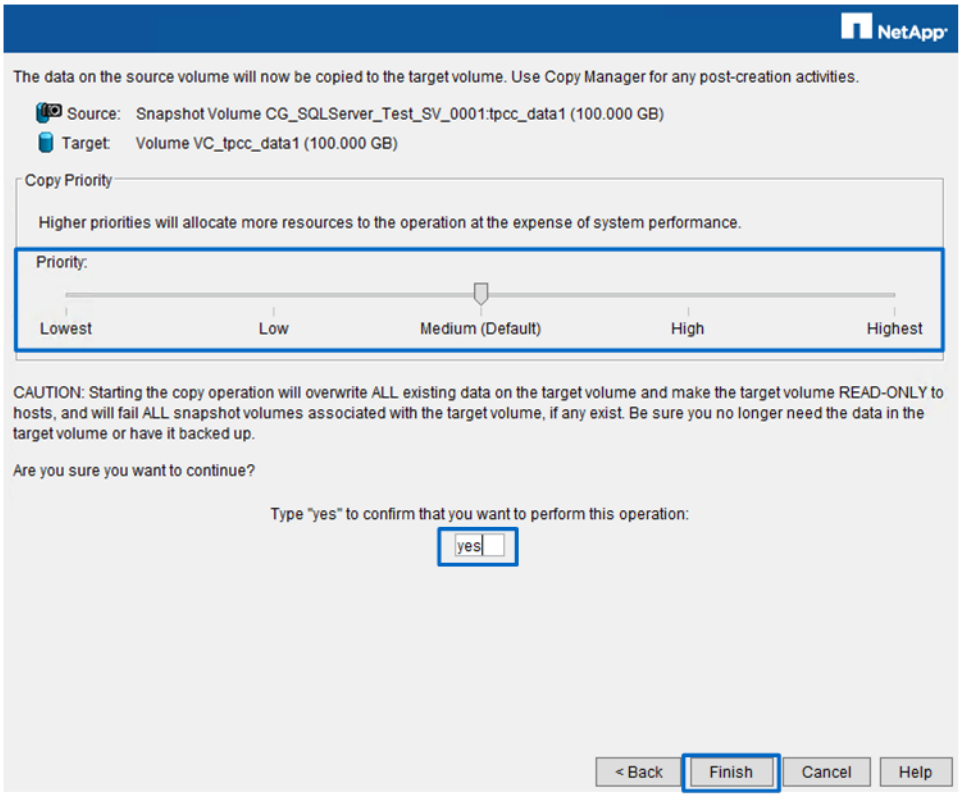


3. From the list, select the target volume that you prepared earlier for the volume copy. Click Next.

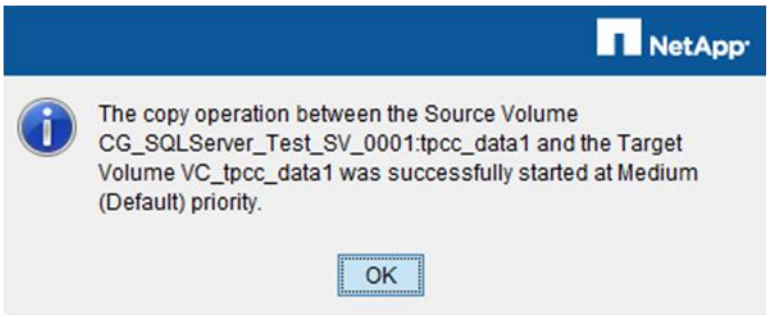


4. Select the priority for the volume copy, type `yes` to confirm, and click Finish.

   Note: Even if no work is currently being processed on the array, a lower priority takes a considerable amount of time. The difference in the lowest to highest priority can mean hours to minutes for the operation to complete. You cannot change the priority after the copy begins.

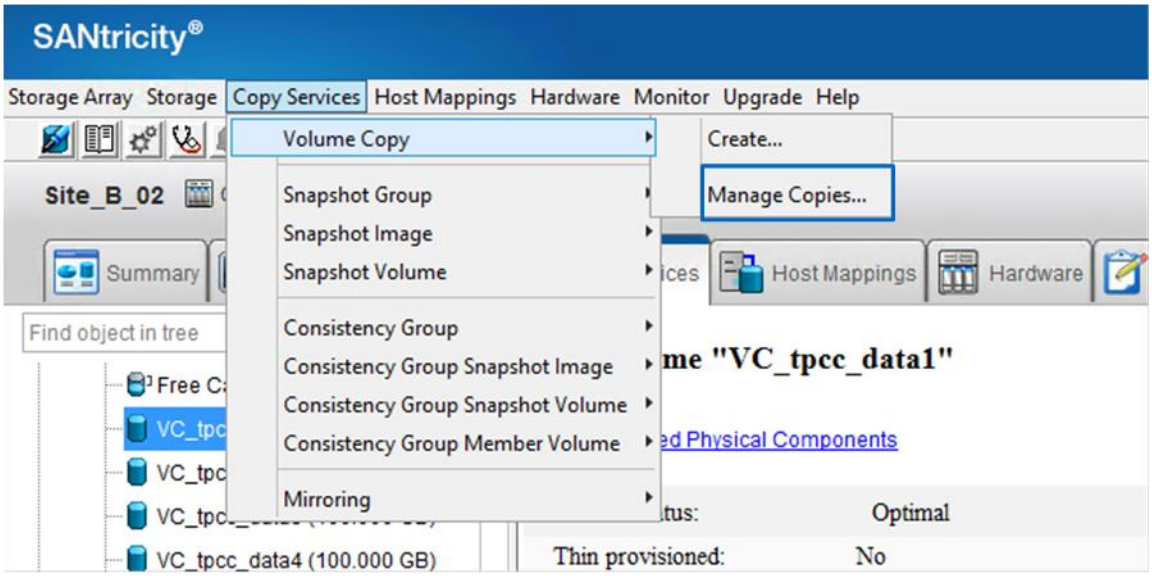


5. Click OK to acknowledge the message.



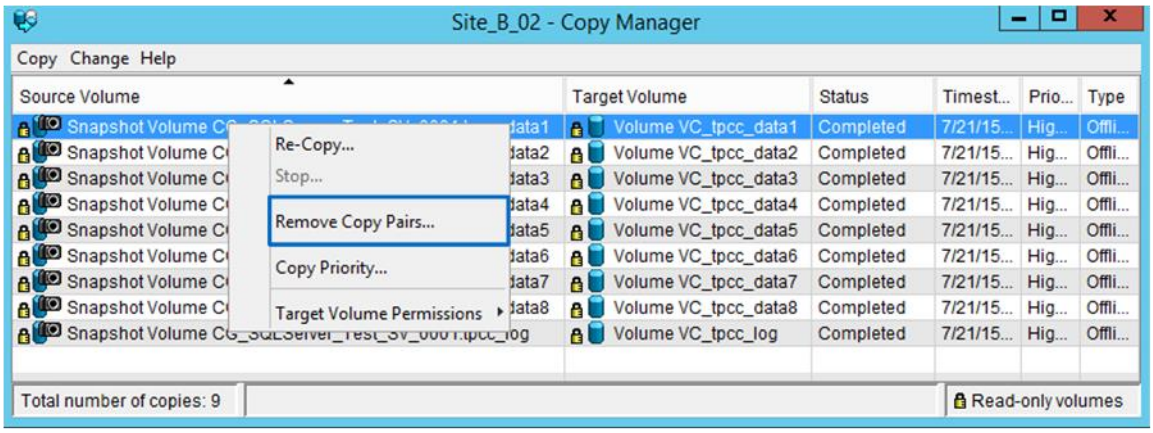
Note: You must repeat steps 3 through 5 for all the volumes of the database.

6. When all the volume copies for the database have completed, select one of the copied volumes from the object tree. From the Copy Services tab, select Volume Copy > Manage Copies.

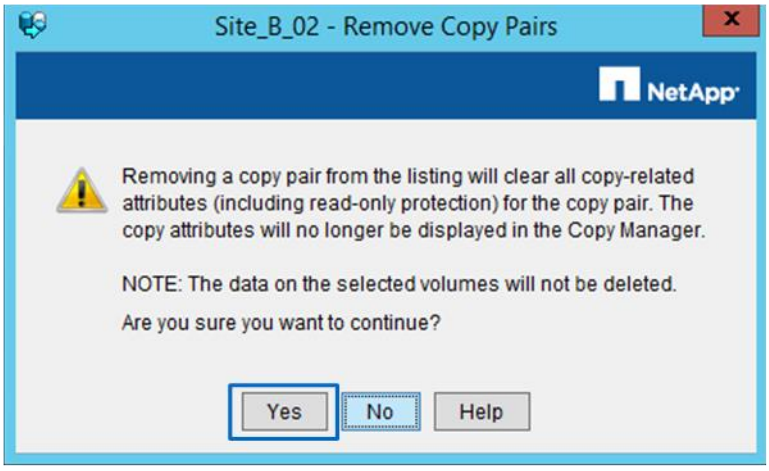


7. Select one or more of the source volume–target volume pairs, right-click, and select Remove Copy Pairs.

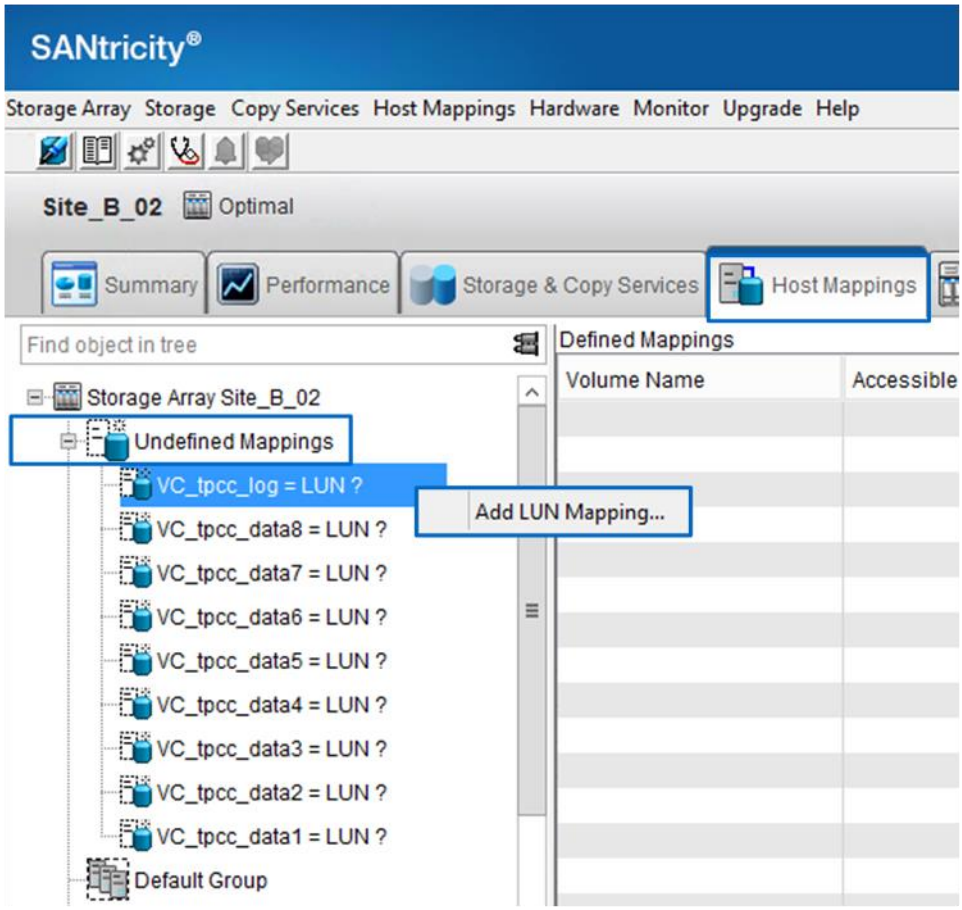
   Note: You must remove the read-only status before you can use volumes to mount a database.



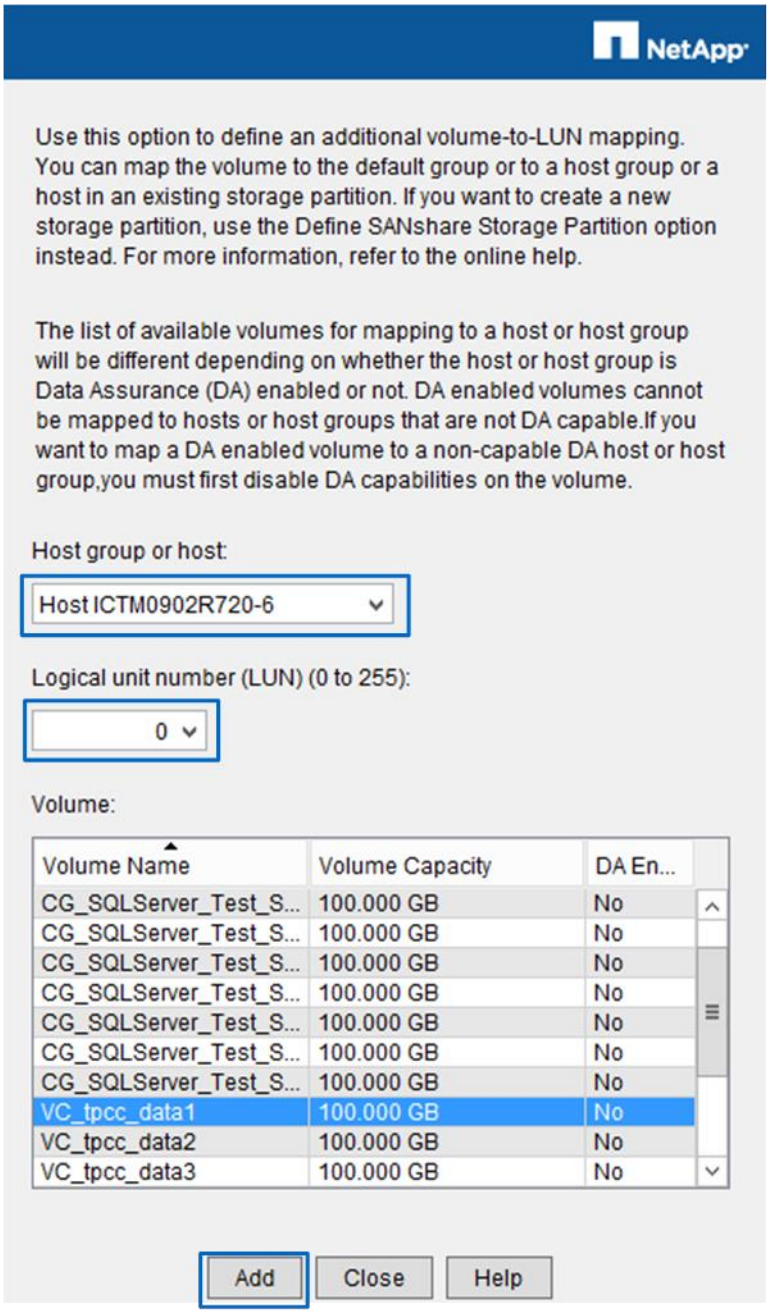
8. To continue, click Yes.



9. Click the Host Mappings tab and then select Undefined Mappings in the object tree. Locate the volumes being used for the database and right-click the first one. Select Add LUN Mapping.



10. Select the host to which to map the volume, the LUN, and the volume name. Click Add. Continue this process until you have added all the volumes.

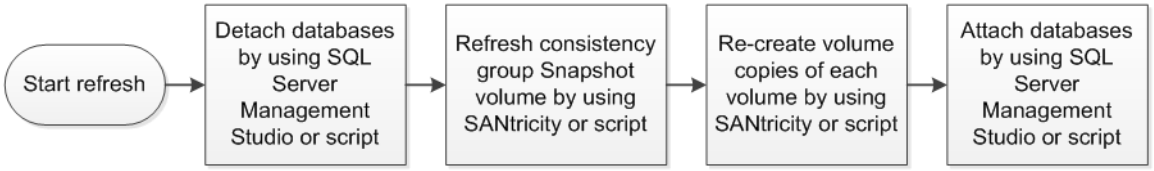


Refresh a Snapshot Volume and Volume Copy

Appendix C: PowerShell Script for Snapshot Volume or Volume Copy Refresh, provides a script to automatically refresh the Snapshot volume and volume copy of the mirrored database. Figure 15 shows a flowchart of the steps to perform the refresh for SQL Server.

Note: For Oracle, you must shut down the databases, perform the refresh operation, and then restart the databases.

Figure 15) SQL Server Snapshot volume and volume copy refresh process.

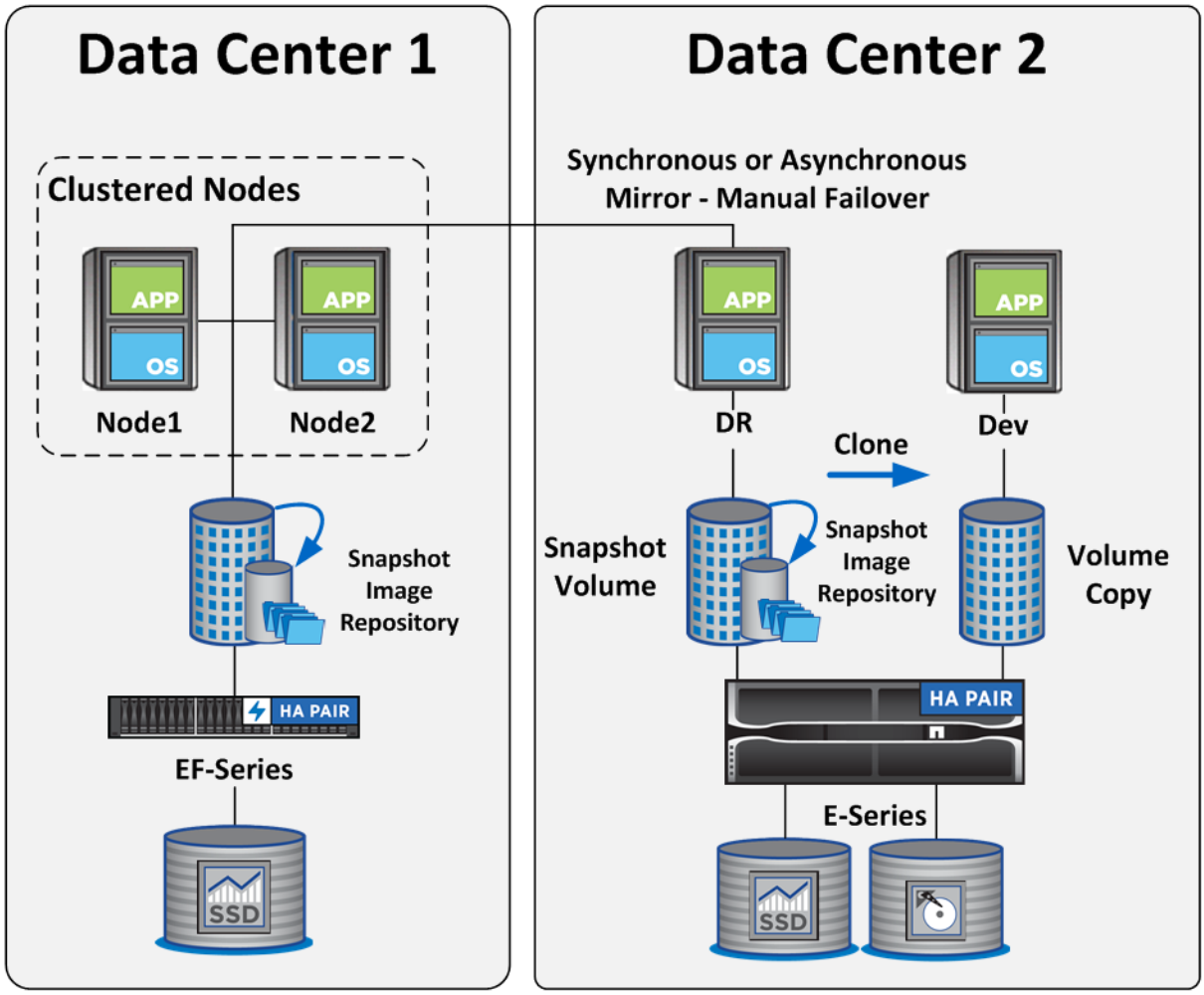


5.7 Putting It All Together

Figure 16 shows what putting all the use cases together might look like to create a complete database high-availability (HA) disaster recovery (DR) solution. In this example:

- Snapshot images are used in the primary data center to take a database backup that can be recovered quickly for high availability.
- Asynchronous or synchronous mirroring is used to copy the database to a secondary site for disaster recovery.
- At the secondary site, the mirrored copy can be turned into a Snapshot volume that can be used to take a permanent backup for archiving.
- Volume copy can then be used to turn the Snapshot volume into a physical copy (clone) for development or other uses.

Figure 16) High-availability disaster recovery solution that uses NetApp E-Series and EF-Series copy services.



6 Sizing Considerations

Because E-Series employs copy-on-write technology, as described previously, it is important to consider how many changes are occurring in the database that is using Snapshot images, Snapshot volumes, or asynchronous mirroring. The following information can help you determine the capacity requirements of an E-Series storage array.

6.1 Repository Sizing for Snapshot Volumes and Asynchronous Mirroring

You should use the information presented here as a preliminary reference point. Every configuration is different and needs to be carefully analyzed to obtain proper repository sizing. After you have determined the initial sizing and have implemented it, you should routinely review the sizing parameters.

Primary Considerations

Because the repository volume size is based on a percentage of the total capacity of base volumes, you must consider several factors, including:

- How much write activity is posted to the base volume during the lifecycle of the Snapshot image or Snapshot volume? As the amount of data that is rewritten to the base volume increases, so does the number of original data blocks that are copied to the repository volume.
- How long is the Snapshot lifecycle? The longer the retention period of the Snapshot image or Snapshot volume, the greater the likelihood that more of the original data blocks change. As a result, more data blocks are copied to the repository volume.

The default repository sizes might be appropriate for some configurations:

- 20% for asynchronous mirror pairs
- 40% for Snapshot images and Snapshot volumes
- 40% for consistency groups
- 40% for consistency group Snapshot volumes

Note: You should include some overhead when you size repositories for the Snapshot volume metadata; typically it is only about 0.02% of the base volume.

Sizing Adjustments

In some cases, you might have to adjust the default settings:

- If the data changes extensively at the base volume because of heavy write activity, set the repository percentage higher.
- If the base volume is predominantly used for reads and very few write changes occur, set the repository percentage lower.
- If the Snapshot lifecycle is prolonged, set the repository percentage higher. The longer the lifecycle, the greater the likelihood that more data blocks change. As a result, more blocks are written to the repository volume.
- If the Snapshot lifecycle is short, with minimal data block changes, set the repository percentage lower.

Example

For a 100GB base volume in which 30% of the data blocks are projected to change over the Snapshot lifecycle:

- 30% (0.30) of 100GB = 30GB
- Add 0.02% (0.0002) of 100GB for overhead = 0.02GB (20MB)
- Repository size = 30.02GB

6.2 Synchronous Mirroring

When you activate the synchronous mirroring feature, two repository volumes are created for the remote mirroring. The capacity is approximately 128MB for each volume and is not a user-configurable option.

7 Performance Considerations

7.1 Snapshot Images and Snapshot Volumes

The Snapshot feature uses copy-on-write (COW) technology to create a durable point-in-time image that is useful for backup and testing. The Snapshot repository holds data that is saved from the base volume during COW activity, as discussed earlier in this document. Original data blocks must first be written to the repository before they can be updated in the base volume, causing a mild performance impact while a Snapshot image or a Snapshot volume is enabled. Therefore, if a Snapshot image or a Snapshot volume

is no longer needed, you should delete it or disable it. If a Snapshot volume is disabled, all COW activity from the base volume ceases, and the Snapshot volume becomes inaccessible for I/O operations, thus eliminating the performance impact.

7.2 Asynchronous Mirroring

With asynchronous mirroring, a write operation first occurs on the local storage system, and then the status of the operation is returned to the host. Rather than keeping the local and remote volumes in constant alignment, the process logs and tracks all changes on the local volume. This tracking of all changes causes a slight performance impact on the array.

At set intervals, all changes are sent to the remote system as a background process. This step also causes a performance impact on the array as the changes are read from disk. Because this transfer is not tied to the local I/O, however, it does not affect the production application performance on the array.

For more information about the effect on database performance of asynchronous mirroring, see TR-4259: Best Practice Guide for Microsoft SQL Server with NetApp EF-Series.

7.3 Volume Copy

The copy priority that you use for a volume copy can have a significant impact on the time that it takes for the copy to complete, ranging from a few minutes to several hours. Higher priorities allocate more resources to the operation, at the expense of system performance. Priorities range from lowest to highest, with the default being medium.

7.4 Synchronous Mirroring

When the data on the primary volume and the secondary volume that are participating in synchronous mirroring is no longer synchronized, the current owner of the primary volume performs a copy operation. This operation resynchronizes the data between the mirrored pair.

The current owner of the primary volume performs this operation in the background. At the same time, the controller processes local I/O writes to the primary volume and processes associated remote writes to the secondary volume. Because the resynchronization diverts controller processing resources from I/O activity, it can have a performance impact on the host application. The synchronization priority defines how much processing time is allocated for synchronization activities relative to system performance.

The lowest priority rate supports system performance, but the resynchronization takes longer. The highest priority rate supports the resynchronization, but system performance might be compromised. The guidelines shown in Table 5 roughly approximate the differences between the five priorities. Volume size and host I/O rate loads affect the synchronization time comparisons.

Table 5) Priority rate synchronization time comparison.

Priority Rate for Full Synchronization	Time Elapsed Compared with Highest Synchronization Rate
Lowest	Approximately eight times as long as at the highest priority rate
Low	Approximately six times as long as at the highest priority rate
Medium	Approximately four times as long as at the highest priority rate
High	Approximately twice as long as at the highest priority rate
Highest	Equal

For more information about the effect on database performance of synchronous mirroring, see TR-4259: Best Practice Guide for Microsoft SQL Server with NetApp EF-Series.

Conclusion

NetApp E-Series copy services provide high-availability and disaster recovery capabilities to the E-Series platform when working with databases. They can be particularly useful when the version of the database management system in use does not provide the full range of functionality required. Examples highlighted in this document are:

- By employing Snapshot images to back up an Oracle database, you can shorten the recovery time objective. With Snapshot images, you do not incur the time penalty from moving backup data files into their required production location.
- You can use asynchronous and synchronous mirroring either separately or in concert to provide disaster recovery at a secondary site.
- By employing Snapshot volumes at the secondary site, you can perform a physical database backup without affecting the primary site.
- By employing volume copy, you can create a database clone for testing and development activities without affecting the production database application.

Appendix A: Oracle Script for Snapshot Image Backup

```
#!/bin/bash
##
# Sample script to:
# Put the DB in Hot Backup mode
# Issue SMcli commands to create new snapshot image of datafiles
# Take the DB out of Hot Backup mode
# Archive current log file
# Issue SMcli commands to create new snapshot image of archive log
# Use cron to schedule script to run  daily
##
# Set up the Database Environment for oracle user e.g. ORACLE_HOME, ORACLE_SID, PATH etc.
source /home/oracle/.bash_profile

# Creates a new log for every day.
LOG=/u01/backups/NTAP1/dailybackup/logs/backup_DB_`date +%d-%b-%Y`.log
touch $LOG
date >> $LOG
echo " " >> $LOG
echo "***************" >> $LOG
echo " " >> $LOG

# Put the DB in Hot Backup mode, log output
echo "Putting the Database in Backup Mode." >> $LOG
cd /u01/backups/NTAP1/dailybackup
sqlplus "/as sysdba" <<EOF
set time on;
spool tmp1.log;
alter database begin backup;
select count(*) from v\$backup where status='ACTIVE';
spool off;
exit;
EOF

# Copy the sql output the the main log
echo " " >> $LOG
cat /u01/backups/NTAP1/dailybackup/tmp1.log >> $LOG
rm /u01/backups/NTAP1/dailybackup/tmp1.log
echo " " >> $LOG
echo "Database has been put in Backup Mode." >> $LOG
echo " " >> $LOG

##
# Use SMcli to take snapshots of the DATA and LOG groups
# Required pre-requisites
# User must be able to run SMcli via sudo
```



```
# Add the storage in by IP to be able to use the user friendly name
# SMcli -A 10.113.84.210 10.113.84.211
# Validate the user friendly name
# SMcli -d -i
##
echo "Creating SnapShot images of the datafiles." >> $LOG
echo "SMcli -n Site_B_02 -c 'create cgSnapImage consistencyGroup="CG1_OracleDB_DATA_001";'"
>>$LOG
sudo SMcli -n Site_B_02 -c 'create cgSnapImage consistencyGroup="CG1_OracleDB_DATA_001";' >>$LOG
echo " " >> $LOG
echo "Database has been backed up." >> $LOG
echo "Bringing the Database out of Backup Mode and switching logs." >> $LOG
echo " " >> $LOG

# Take the DB out of hot backup mode and archive current log
echo "Exiting hot backup, rolling the archive logs." >> $LOG
cd /u01/backups/NTAP1/dailybackup
sqlplus "/as sysdba" <<EOF
set time on;
spool tmp2.log;
alter database end backup;
select count(*) from v\$backup where status='ACTIVE';
alter system archive log current;
spool off;
exit;
EOF

echo "Creating SnapShot images of the archive log volume." >> $LOG
echo "SMcli -n Site_B_02 -c 'create cgSnapImage consistencyGroup="CG1_OracleDB_LOGS_001";'"
>>$LOG
sudo SMcli -n Site_B_02 -c 'create cgSnapImage consistencyGroup="CG1_OracleDB_LOGS_001";' >>$LOG

# Copy the sql output the the main log
cat /u01/backups/NTAP1/dailybackup/tmp2.log >> $LOG
rm /u01/backups/NTAP1/dailybackup/tmp2.log
echo " " >> $LOG
echo "Database is out of Backup Mode now." >> $LOG
echo " " >> $LOG
echo " " >> $LOG
echo "Backup Completed for `date +%d-%b-%Y`." >> $LOG
echo " " >> $LOG
echo "################" >> $LOG
#####################################################
```

Appendix B: Oracle Scripts for Asynchronous Mirroring

The following two scripts deliver asynchronous mirroring of a database. They can be scheduled independently to deliver a better RPO. For example, the datafile mirror update might be performed once per day, while the log mirror is updated every 15 minutes.

```
#!/bin/bash
##
# Simple script to put the DB in Hot Backup mode
# Issue SMcli commands to update Async RVM
# Take the DB out of Hot Backup mode
# Archive current log file
# Issue SMcli commands to update Async RVM of archive log
# Use cron to schedule script to run  daily
##
# Set up the Database Environment for oracle user e.g. ORACLE_HOME, ORACLE_SID, PATH etc.
source /home/oracle/.bash_profile

# Creates a new log for every day.
LOG=/u01/backups/NTAP1/dailyarvm/logs/backup_DB_`date +%d-%b-%Y`.log
touch $LOG
date >> $LOG
echo " " >> $LOG
echo "***************" >> $LOG
```



```
echo " " >> $LOG

# Put the DB in Hot Backup mode, log output
echo "Putting the Database in Backup Mode." >> $LOG
cd /u01/backups/NTAP1/dailyarvm
sqlplus "/as sysdba" <<EOF
set time on;
spool tmp1.log;
alter database begin backup;
select count(*) from v\$backup where status='ACTIVE';
spool off;
exit;
EOF

# Copy the sql output the the main log
echo " " >> $LOG
cat /u01/backups/NTAP1/dailyarvm/tmp1.log >> $LOG
rm /u01/backups/NTAP1/dailyarvm/tmp1.log
echo " " >> $LOG
echo "Database has been put in Backup Mode." >> $LOG
echo " " >> $LOG

##
# Use SMcli to asynchronously mirror the DATA and LOG groups
# Required pre-requisites
# User must be able to run SMcli via sudo
# Add the storage in by IP to be able to use the user friendly name
# SMcli -A 10.113.84.210 10.113.84.211
# Validate the user friendly name
# SMcli -d -i
##
echo "Update Async RVM  of the datafiles." >> $LOG
echo "SMcli -n Site_B_02 -c 'start asyncMirrorGroup ["CG1_OracleDB_DATA_001"] synchronize;'"
>>$LOG
sudo SMcli -n Site_B_02 -c 'start asyncMirrorGroup ["CG1_OracleDB_DATA_001"] synchronize;' >>
$LOG
echo " " >> $LOG
echo "Database has been backed up." >> $LOG
echo "Bringing the Database out of Backup Mode and archiving current log." >> $LOG
echo " " >> $LOG

# Take the DB out of hot backup mode
cd /u01/backups/NTAP1/dailyarvm
sqlplus "/as sysdba" <<EOF
set time on;
spool tmp2.log;
alter database end backup;
alter system archive log current;
select count(*) from v\$backup where status='ACTIVE';
spool off;
exit;
EOF

echo "Update Async RVM  of the log files." >> $LOG
echo "SMcli -n Site_B_02 -c 'start asyncMirrorGroup ["CG1_OracleDB_LOGS_001"] synchronize;'"
>>$LOG
sudo SMcli -n Site_B_02 -c 'start asyncMirrorGroup ["CG1_OracleDB_LOGS_001"] synchronize;' >>
$LOG


# Copy the sql output the the main log
cat /u01/backups/NTAP1/dailyarvm/tmp2.log >> $LOG
rm /u01/backups/NTAP1/dailyarvm/tmp2.log
echo " " >> $LOG
echo "Database is out of Backup Mode now." >> $LOG
echo " " >> $LOG
echo " " >> $LOG
echo "Backup Completed for `date +%d-%b-%Y`." >> $LOG
echo " " >> $LOG
echo "################" >> $LOG
```



```
#####################################################

#!/bin/bash
##
# Simple script to archive current log file and issue SMcli commands to update Async RVM
##
# Set up the Database Environment for oracle user e.g. ORACLE_HOME, ORACLE_SID, PATH etc.
source /home/oracle/.bash_profile

# Creates a new log for every day.
LOG=/u01/backups/NTAP1/dailyarvm/logs/backup_DB_`date +%d-%b-%Y`.log
touch $LOG
date >> $LOG
echo " " >> $LOG
echo "***************" >> $LOG
echo " " >> $LOG

##
# Use SMcli to asynchronously mirror the LOG groups
# Required pre-requisites
# User must be able to run SMcli via sudo
# Add the storage in by IP to be able to use the user friendly name
# SMcli -A 10.113.84.210 10.113.84.211
# Validate the user friendly name
# SMcli -d -i
##
echo "Switching Log files" >> $LOG
echo " " >> $LOG

# Switch log file
cd /u01/backups/NTAP1/dailyarvm
sqlplus "/as sysdba" <<EOF
set time on;
spool tmp2.log;
alter system archive log current;
select count(*) from v\$backup where status='ACTIVE';
spool off;
exit;
EOF

echo "Update Async RVM  of the log files." >> $LOG
echo "SMcli -n Site_B_02 -c 'start asyncMirrorGroup ["CG1_OracleDB_LOGS_001"] synchronize;'"
>>$LOG
sudo SMcli -n Site_B_02 -c 'start asyncMirrorGroup ["CG1_OracleDB_LOGS_001"] synchronize;' >>
$LOG


# Copy the sql output the the main log
cat /u01/backups/NTAP1/dailyarvm/tmp2.log >> $LOG
rm /u01/backups/NTAP1/dailyarvm/tmp2.log
echo " " >> $LOG
echo "Database is out of Backup Mode now." >> $LOG
echo " " >> $LOG
echo " " >> $LOG
echo "Backup Completed for `date +%d-%b-%Y`." >> $LOG
echo " " >> $LOG
echo "################" >> $LOG
#####################################################
```


Appendix C: PowerShell Script for Snapshot Volume or Volume Copy Refresh

Preparation of Environment

The PowerShell scripts used to refresh the Snapshot volume and volume copy make use of the NetApp SANtricity Web Services Proxy 1.2 and the NetApp PowerShell Toolkit.

To download the E-Series Web Services API, go to the NetApp Support site Software Download page. To download the NetApp PowerShell Toolkit, go to the NetApp Support site ToolChest.

The following steps were described earlier and are a prerequisite to performing the refresh:

1. Create the folders to be used as mount points on the Windows server.
2. Create a consistency group of the database base volumes.
3. Create a consistency group Snapshot volume.
4. Create a volume copy of each Snapshot volume.
5. Remove the Snapshot volume-volume copy pairs.
6. Map all the volumes to the host.
7. Attach databases.

Running the Scripts

Initial Setup

1. Open Windows PowerShell and change to the directory where the following scripts have been saved.
2. Run `Add-ArrayToProxyManagement.ps1 -ManagementIpAddress "Array IP1","Array IP2" -Url https://ServerName:ServerPort for Web Services/devmgr/`.

   Note: `Example:./ Add-ArrayToProxyManagement.ps1 -ManagementIpAddress "10.113.84.144”,”10.113.84.145” -Url https://ICTM0902R720-6:8082/devmgr/`
3. Validate with `Get-NeStorageSystem -cred $cred`.

Normal Workflow

1. Open Windows PowerShell and change to the directory where the following scripts have been saved.
2. Run `Run-SV_VC_Refresh.ps1`, which uses the default consistency group, CG_SQLServer_Test. Or use `Run-SV_VC_Refresh.ps1 -CGName "CG_name"`.

   Note: The default name can be modified in the script.

The PowerShell script is made up of several subscripts, as shown in the following examples. The main script, `Run-SV_VC_Refresh.ps1`, calls the other PowerShell scripts in the following order:

- `Set-ScriptingEnvironment.ps1`
- `Invoke-SqlScript.ps1` (runs `Unmount.sql`)
- `Remove-AllVolumeCopyPairs.ps1`
- `Recreate-CGSnapshotVolume.ps1`
- Creation of volume copies performed in main script
- `Invoke-SqlScript.ps1` (runs `Mount.sql`)

Add-ArrayToProxyManagement.ps1

```
<#
.SYNOPSIS
    A script that will add a storage array to be managed by the SANtricity PowerShell toolkit.
.DESCRIPTION
    This script will add a storage array so it can be managed by the SANTricity PowerShell
toolkit. This script only needs to be run once.  Subsequent executions are harmless in that
management connections do not accumulate.  The cmdlets use the SANtricity WebServices Proxy to
interact with the storage.  This is a convient way to centralize array managmenet while
provivding the flexibility of executing PowerShell scripts in many locations.  Future versions of
the SANtricity OS will allow PowerShell scripts to be executed directly against the storage
system.  However, all currently available versions of SANtricity OS only support proxy
managmenet.  Once the array is added, it will continue to be known by the SANtricity Web Services
Proxy and be available to the PowerShell Toolkit without having to re-run this script.

    The script will set 2 variables that will be available for the scope of the PowerShell (or
PowerShell ISE) window.  They are called $ss which is a reference to the StorageSystem object
and $cred which is a reference to the Credential to the WebService Proxy.

    The SANtricity WebServices Proxy runs as a web server container and will listen on http port
8080 and https port 8443.  If these ports are already in use, then you will need to configure the
ports in C:/Program Files/NetApp/SANTricity Web Services Proxy/wsconfig.xml.  After editing
wsconfig.xml, you will need to restart the service (Stop-Service NetAppWebServicesProxy ; Start-
Service NetAppWebServicesProxy)

.PARAMETER ManagmentIpAddresses
    An array of no more than 2 strings that are the managment ip addresses for the storage array
you wish to add to proxy managment.
.PARAMETER Url
    The url to the host running the SANtricity Web Services Proxy.  For example,
https://myHost:8443/devmgr/
.PARAMETER ProxyUser
    The name of the user on the SANtricity Web Services Proxy.  The default user is rw.
.PARAMETER ProxyPwd
    The password for the user on the SANtricity Web Services Proxy.  The default user is rw.

.NOTES
    Prerequisites
    1. NetApp.SANtricity.PowerShell PowerShell module.  Verify with command Get-Module -
ListAvailable
    2. For the SANtricity PowerShell toolkit to work correctly, the SANtricity Web Services Proxy
will need to be installed in the environment.  The url for that points to the webapi is
configurable through Set-ScriptingEnvironment.ps1

.EXAMPLE

./ Add-ArrayToProxyManagement.ps1 -ManagementIpAddress "1.1.1.1”,”1.1.1.2” -Url
https://ICTM0902R720-6:8082/devmgr/
    Add the array with management addresses of 1.1.1.1 and 1.1.1.2 to be managed by the
SANtricity WebServices Proxy.
#>
param([Parameter(Mandatory=$true)][string[]]$ManagementIpAddresses,
      [Parameter(Mandatory=$true)][string]-Url https://ServerName:ServerPort for Web
Services/devmgr/",
      [string]$user="rw",
      [string]$pwd="rw")

if (-not $Url.EndsWith("/")) { $Url+="/" }
$global:cred = Get-NeProxyCredential -Url $Url -ProxyUser $user -ProxyPwd $pwd

$(New-NeStorageSystem -Cred $cred -ControllerAddresses $ManagementIpAddresses) |Out-Null
Start-sleep 2 # this allows the proxy to contact the array and load it's data to memory.
$global:ss = Get-NeStorageSystem -Credential $cred |? { $_.ip1 -eq $ManagementIpAddresses[0] }

# this is a fancy trick that allows me to specify both parameters when
# calling a cmdlet (splatting)
$global:c = @{
             Credential= $cred
             SystemId = $ss.id
            }
```



```
Write-Host "Array [$($ss.Name)] Id [$($ss.Id)]"
```

Run-SV_VC_Refresh.ps1

```
<#
.SYNOPSIS
    A script that will orchestrate failover sequence for SANtricity Storage supporting an MSSQL
workload.
.DESCRIPTION
    The script will first source and set the environment to work with the default storage system
as specified by script parameter defaults in Set-ScriptingEnvironment.ps1.

    The next step of the script is to unmount the SQLServer database.  This is done by running
the script called Invoke-SqlScript with the Unmount operation.  For this to be successfull an sql
script called Unmount.sql must exist in the same directory as this script.  To extend this
example you will need to write an sql script that is tuned to your specific environment.

    Now the script will recreate the consistency group snapshot volume.  This is accomplished
with a script called Recreate-CGSnapshotVolume.ps1.  This script does not leverage powershell
cmdlets to suspend and resume (the recipe to the recreate action) but rather the legacy
SANtricity command line application, SMcli.  The next release of this script will use the
SANtriicty PowerShell Toolkit to achieve this step.  The output of this script (since it is all
text) is logged to a file called Recreate-cGSnapshotVolume.log.

    With the databases offline and consistency group refreshed, the next step is to make a copy
for each volume in the consistency group.  This step is accomplished in the scope of this script.
This step will take some time depending on the size of the volumes being copied and the current
I/O load on the storage array.  The script displays a copy progress meter so you can guage how
long this step will take.

    Finally, after snapshot volumes are copied, the database is restarted with the Invoke-
SqlScript.ps1 script. This time it is run with the Mount operation.

.PARAMETER CGName
    The name of the consistency group that contains the MSSQL volumes.

.NOTES
    Prerequisites
    1. NetApp.SANtricity.PowerShell PowerShell module.  Verify with command Get-Module -
ListAvailable
    2. SQLServer PowerShell module.  Verify with the same Get-Module command.
    3. For the SANtricity PowerShell toolkit to work correctly, the SANtricity WebApi will need
to be installed in the environment.  The url for that points to the webapi is configurable
through Set-ScriptingEnvironment.ps1

.LINK
    Invoke-SqlScript.ps1
    Set-ScriptingEnvironment.ps1
    Recreate-CGSnapshotVolume.ps1
.EXAMPLE
    ./Run-SV_VC_Refresh.ps1
    Run the script and refresh with the default consistency group, CG_SQLServer_Test.
.EXAMPLE
    ./Run-SV_VC_Refresh.ps1 -CGName "CG_name"
    Run the script and refresh with the named consistency group.
#>
param([string]$CGName="CG_SQLServer_Test")
$elapsed = [System.Diagnostics.Stopwatch]::StartNew()
Write-Host "* * * INFO - Database Refresh started at $(Get-Date)"
if ($ss -eq $null) { .\Set-ScriptingEnvironment.ps1 }

function New-SnapshotVolumeCopy() {
param([NetApp.ESeries.PowerShell.TransferObjects.VolumeEx]$BaseVolume,
      [NetApp.ESeries.PowerShell.TransferObjects.SnapshotGroup]$SnapshotVolume,
      [string]$CGViewRef
)
    $s = Get-NeConsistencyGroupSnapshotVolume @c -ObjectId $cg.CgRef |? {$_.cgviewref -eq
$CGviewRef}
```



```
    $PitView = Get-NePitView @c |? {$_.BaseVol -eq $BaseVolume.id -and
$_.Label.StartsWith($s.Label)}
    $BaseVolumeName = $BaseVolume.Label
    $VCCopyTarget="VC_" + $BaseVolumeName
    $VolumeObject = $BaseVolume
    $CopyTargetObject = Get-NeVolume @c |? {$_.Label -eq $VCCopyTarget }
    New-NeVolumeCopyPair @c -SourceId $PitView.viewRef -TargetId $CopyTargetObject.Id -
CopyPriority "priority4" `
        -DisableSnapshot $true -TargetWriteProtected $false |Out-Null
    Write-Host "* * * INFO - Made volume copy pair $($PitView.Label) ==>
$($CopyTargetObject.Name)"
}

# Detach databases.  Sometimes the SQLServer provider doesn't return from it's "directory", so we
guard against.
$Here = Get-Location
.\Invoke-SqlScript.ps1 -Operation Unmount
Set-Location $Here

# Remove Volume Copy Pairs
Write-Host "* * * Remove previously created volume copy pairs. " -NoNewline
.\Remove-AllVolumeCopyPairs.ps1 |Out-Null
Write-Host -ForegroundColor Green " [Complete]"

# Consistency group and use the script parameter $CGName.
$cg = Get-NeConsistencyGroup @c |? {$_.label -eq $CGName }

# Recreate the consistency group snapshot volume
.\Recreate-CGSnapshotVolume.ps1 -ArrayName $($ss.Name)

# Get the members of the snapshot group in the CG.
$MemberVolumes = Get-NeSnapshotGroup @c |? {$_.ConsistencyGroupRef -eq $cg.CgRef}

# Make a volume copy for each snapshot volume in the consistency group.
foreach ($memberVolume in $MemberVolumes) {
    $baseVolume = Get-NeVolume @c |? {$_.id -eq $memberVolume.BaseVolume}
    $CGSnap = Get-NeConsistencyGroupSnapshotVolume @c -ObjectId $cg.CgRef |? {$_.label -eq
$($CGName + "_0001")}
    New-SnapshotVolumeCopy -BaseVolume $baseVolume -SnapshotVolume $memberVolume -CGViewRef
$CGSnap.cgViewRef
}

# Now that the volume copy's are started, we need to wait for the copy operations to complete.
$PercentComplete = 0
do {
    $VolCopyProgress = Get-NeVolumeCopyProgress @c
    $totalPercent = 0
    foreach ($progress in $VolCopyProgress) {
        if ($progress.PercentComplete -gt 0) {
            $totalPercent += $progress.PercentComplete
        } elseif ($progress.PercentComplete -eq -1) {
            $totalPercent+=100
        }
    }
    $totalPercent /= $VolCopyProgress.count
    $PercentComplete = $totalPercent
    Write-Progress -Activity "Volume Copy Progress" -PercentComplete $PercentComplete
    Start-sleep -Seconds 1
} until ($PercentComplete -eq 100)

# Fire up databases on the volume copies.
.\Invoke-SqlScript.ps1 -Operation Mount
$totalTime = $($elapsed.Elapsed.ToString())
Write-Host  "* * * INFO - Database Refresh finished - total time [$totalTime]"
```

Set-ScriptingEnvironment.ps1

```
# ------------------------------------------------------------------------------ #
# This script will set an environment for testing by configuring the proxy url
# and storage array by setting some global variables.  This means the
```



```
# variables are durable after running this script.
# ------------------------------------------------------------------------ #
param([string]$ArrayName="Site_B_01", [string]$Url="https://ICTM0902R720-6:8082/devmgr/",
        [string]$user="rw", [string]$pwd="rw")
if (-not $Url.EndsWith("/")) { $Url+="/" }
$global:cred = Get-NeProxyCredential -Url $Url -ProxyUser $user -ProxyPwd $pwd

# ARVM config of MP's
$global:ss = Get-NeStorageSystem -Credential $cred |? { $_.name -eq $ArrayName }
# this is a fancy trick that allows me to specify both parameters when
# calling a cmdlet (splatting)
# Get-NeStorageSystem @c is eqal to Get-NeStorageSystem -cred $cred -SystemId $ss.id
$global:c = @{
              Credential= $cred
              SystemId = $ss.id
            }
Write-Host "Array [$($ss.Name)] Id [$($ss.Id)]"
```

Invoke-SqlScript.ps1

```
<#
.SYNOPSIS
    A script that will run commands in an sql file.
.DESCRIPTION
    The name of the sql file must correspond to the
    type of operation.  The type of operation must be "Mount" or "Unmount".  Therefore, mount.sql
    or unmount.sql will be executed with the Invoke-SqlCmd cmdlet.
.PARAMETER Operation
    Choose the kind of operation you wish to run.  Valid values are "Mount" and "Unmount" and
    are enforced by the script.
.NOTES
    Be aware this is script can be a security liability in the hands of someone who wishes
    to do harm.  The script executes with permission of the user who executes the script
    so it is not run with elevated privilege.
.LINK
    Invoke-Sqlcmd
.EXAMPLE
    ./Invoke-SqlScript -Operation Mount
    Run the script and invoke mount.sql in the present directory.
.EXAMPLE
    ./Invoke-SqlScript -Operation Unmount
    Run the script and invoke unmount.sql in the present directory.
#>
Param([ValidateSet("Mount","Unmount")] [string]$Operation="Mount" )
$SQLFile = Get-Item -Path $($Operation + ".sql")
if ($SQLFile.Exists) {
    Invoke-Sqlcmd -InputFile $SQLFile.FullName # todo - catch an error about db's already being
detached.
} else {
    Write-Error "* * * ERROR - File [$($SQLFile.FullName)] does not exist."
}
```

Unmount.sql

```
-- Detach Snapshot Volume and Volume Copy databases before refresh
ALTER DATABASE [SV_tpcc] SET  SINGLE_USER WITH ROLLBACK IMMEDIATE
GO
EXEC master.dbo.sp_detach_db @dbname = N'SV_tpcc'
GO
ALTER DATABASE [VC_tpcc] SET  SINGLE_USER WITH ROLLBACK IMMEDIATE
GO
EXEC master.dbo.sp_detach_db @dbname = N'VC_tpcc'
GO
```

Mount.sql

```
-- Attach databases to correct mount points after refresh
CREATE DATABASE [SV_tpcc] ON
( FILENAME = N'C:\MSSQL\SV_tpcc_data1\tpcc_data1.mdf' ),
```



```
( FILENAME = N'C:\MSSQL\SV_tpcc_log\tpcc_log.ldf' ),
( FILENAME = N'C:\MSSQL\SV_tpcc_data2\tpcc_data2.ndf' ),
( FILENAME = N'C:\MSSQL\SV_tpcc_data3\tpcc_data3.ndf' ),
( FILENAME = N'C:\MSSQL\SV_tpcc_data4\tpcc_data4.ndf' ),
( FILENAME = N'C:\MSSQL\SV_tpcc_data5\tpcc_data5.ndf' ),
( FILENAME = N'C:\MSSQL\SV_tpcc_data6\tpcc_data6.ndf' ),
( FILENAME = N'C:\MSSQL\SV_tpcc_data7\tpcc_data7.ndf' ),
( FILENAME = N'C:\MSSQL\SV_tpcc_data8\tpcc_data8.ndf' )
 FOR ATTACH
GO

CREATE DATABASE [VC_tpcc] ON
( FILENAME = N'C:\MSSQL\VC_tpcc_data1\tpcc_data1.mdf' ),
( FILENAME = N'C:\MSSQL\VC_tpcc_log\tpcc_log.ldf' ),
( FILENAME = N'C:\MSSQL\VC_tpcc_data2\tpcc_data2.ndf' ),
( FILENAME = N'C:\MSSQL\VC_tpcc_data3\tpcc_data3.ndf' ),
( FILENAME = N'C:\MSSQL\VC_tpcc_data4\tpcc_data4.ndf' ),
( FILENAME = N'C:\MSSQL\VC_tpcc_data5\tpcc_data5.ndf' ),
( FILENAME = N'C:\MSSQL\VC_tpcc_data6\tpcc_data6.ndf' ),
( FILENAME = N'C:\MSSQL\VC_tpcc_data7\tpcc_data7.ndf' ),
( FILENAME = N'C:\MSSQL\VC_tpcc_data8\tpcc_data8.ndf' )
 FOR ATTACH
GO
```

Recreate-CGSnapshotVolume.ps1

```
param([ValidateScript({Test-Path $_ })]
      [string]$SMcli='C:\Program Files (x86)\StorageManager\client\SMcli.exe',
      [string]$ArrayName="Site_B_02",
      [string]$CGSnapVolume="CG_SQLServer_Test_SV_0001",
      [string]$CGSnapImage="CG_SQLServer_Test",
      [string]$CGSnapImageRevision="NEWEST",
      [string]$LogFile="RecreateCGSnapshotVolume.log"
)
Write-Host -NoNewline "* * * INFO - Stopping Consistency Group Snapshot volume
[$($CGSnapVolume)]"
& $SMcli -n $ArrayName -c @"
stop cgSnapVolume [\"$($CGSnapVolume)\"] ;
"@ | Out-File -FilePath $LogFile
Write-Host -ForegroundColor Green " [Complete]"

Write-Host -NoNewline "* * * INFO - Create a new snapshot image. [$($CGSnapImage)]"
& $SMcli -n $ArrayName -c @"
create cgSnapImage ConsistencyGroup=\"$($CGSnapImage)\";
"@ |Out-File -Append -FilePath $LogFile
Write-Host -ForegroundColor Green " [Complete]"

Write-Host -NoNewline "* * * INFO - Resume Consistency Group Snapshot volume [$($CGSnapVolume)]"
& $SMcli -n $ArrayName -c @"
resume cgSnapVolume [\"$($CGSnapVolume)\"] cgSnapImage = \"$($CGSnapImage +
":"+$CGSnapImageRevision)\";
"@ | Out-File -Append -FilePath $LogFile
Write-Host -ForegroundColor Green " [Complete]"
```

Remove-AllVolumeCopyPairs.ps1

```
if ($ss -eq $null) { .\Set-ScriptingEnvironment.ps1 }
Get-NeVolumeCopyPair @c |% {
    Remove-NeVolumeCopyPair @c -ObjectId $_.id
}
```

References

The following references were used in this TR:

- NetApp Support site documentation library
  http://support.netapp.com/portal/documentation

- NetApp Support site Software Download page
  http://mysupport.netapp.com/NOW/cgi-bin/software
- NetApp Support site ToolChest
  http://mysupport.netapp.com/tools/index.html
- SANtricity Storage Manager online help
  http://mysupport.netapp.com/NOW/public/eseries/amw/index.html
- TR-4259: Best Practice Guide for Microsoft SQL Server with NetApp EF-Series
  https://fieldportal.netapp.com/search/?searchTerm=tr-4259&includeDocuments=0&includeArchived=0&showThumbnails=1&showDescription=1

Version History

Version	Date	Document Version History
Version 1.0	October 2015	Initial release.
Version 1.0.1	December 2016	Added log replication script to Appendix B.

Refer to the Interoperability Matrix Tool (IMT) on the NetApp Support site to validate that the exact product and feature versions described in this document are supported for your specific environment. The NetApp IMT defines the product components and versions that can be used to construct configurations that are supported by NetApp. Specific results depend on each customer's installation in accordance with published specifications.

Copyright Information

Trademark Information