PURESTORAGE®

WHITE PAPER

Cloning Oracle Databases with FlashArray Volume Snapshots

A practical guide to Oracle Database cloning with Pure Storage® FlashArray™ volume snapshots.

Contents

Executive Summary

Modern enterprises rely on rapid, reliable database cloning to support development, testing, analytics, and disaster recovery across multiple Oracle Database environments. However, legacy storage systems often struggle with prolonged cloning times, high resource consumption, and performance bottleneck challenges that increase complexity and administrative overhead.

Pure Storage® FlashArray™ products offer a transformative solution with advanced volume snapshot technology for near-instant Oracle Database cloning. By leveraging thin-provisioned, metadata-based snapshots, FlashArray drastically reduces storage overhead while ensuring consistent, immutable copies that do not impact production performance. Cloning cycles that once took hours or days can now be completed in minutes, accelerating critical workflows and minimizing downtime.

Beyond speed and efficiency, FlashArray systems simplify complex cloning processes through streamlined automation and an intuitive interface. This alleviates the burden on IT and database administrators, reducing errors and improving agility. The result is a cost-effective, high-performance Oracle Database environment that supports both production excellence and rapid development needs.

Solution Overview

This solution uses the FlashArray volume snapshot copy and clone features to create efficient, reliable clones of Oracle databases. These clones can either be presented as new volumes or used to overwrite existing volumes already attached to a host, replicating the exact state of the database at the time the snapshot was taken. By attaching these cloned volumes to the appropriate Oracle host, administrators can quickly and seamlessly spin up fully functional, identical database copies. This method is ideal for operational scenarios where rapid, efficient, and scalable cloning is critical to meeting business demands.

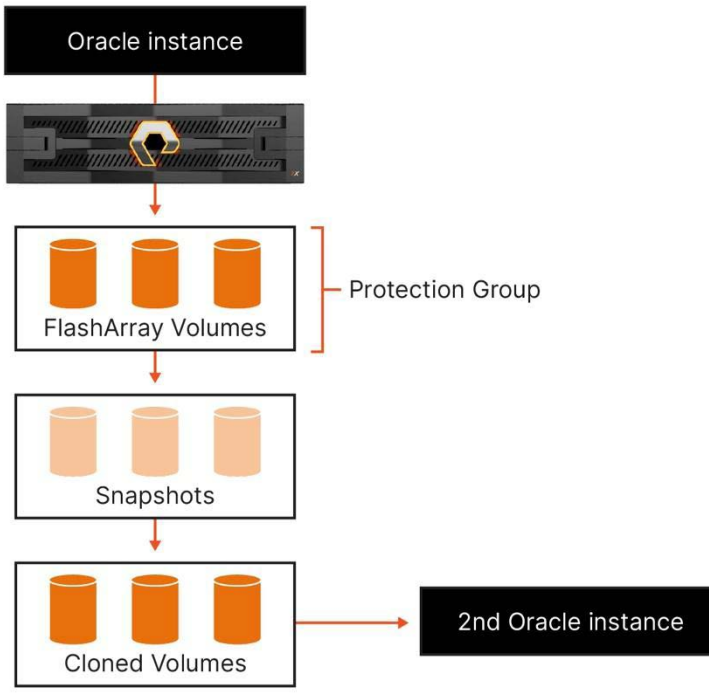

FIGURE 1 The cloning solution for Oracle Database uses FlashArray volume snapshots to create efficient and reliable clones of Oracle databases.

FlashArray snapshots create point-in-time, metadata-based copies of Oracle databases, capturing all essential components, including datafiles, redo logs, and control files. These snapshots use thin provisioning, enabling space-efficient clones that require minimal additional storage. Once attached to a target host, the cloned database functions as a fully operational, production-identical instance, ready for immediate use.

Key use cases include:

- **Development and testing**: FlashArray snapshots enable database administrators to provision production-like clones rapidly, ensuring developers and testers work with accurate, up-to-date data between higher and lower environments.
- **Disaster recovery**: Immutable, point-in-time snapshots ensure data consistency for recovery databases in the event of data loss or corruption. By quickly creating clones for recovery, organizations minimize downtime and restore operations rapidly, maintaining business continuity.
- **Analytics and reporting**: Read-only clones enable analytics teams to run complex queries and generate reports without impacting production performance. Frequent updates to reporting environments ensure data remains accurate and relevant, supporting timely decision-making.
- **Database migration and scaling**: FlashArray cloning simplifies migrations and scaling by allowing administrators to create consistent database copies for new environments. Clones reduce downtime and risk during migrations and provide additional instances for scaling workloads to meet growing demands.

Solution Benefits

FlashArray volume snapshots for Oracle Database cloning deliver the following key advantages:

- **Rapid cloning**: Create fully functional database clones in minutes, significantly reducing time compared to traditional methods and accelerating workflows.
- **Space efficiency**: Thin-provisioned snapshots consume minimal additional storage by referencing existing data blocks, reducing storage costs.
- **Non-disruptive operations**: Cloning occurs independently of production systems, ensuring no impact on live performance.
- **Data integrity**: Immutable, point-in-time snapshots guarantee consistency and reliability, ideal for disaster recovery and backups.
- **Versatile use cases**: The solution supports development, testing, disaster recovery, analytics, migration, and scaling with ease.
- **Simplified management**: Automate tasks via scripting or use the intuitive FlashArray interface, reducing administrative overhead.
- **Uninterrupted cloning**: Clone databases without stopping or pausing them, ensuring continuous availability and seamless workflows.

Technology Overview

The following sections provide an overview of the technologies that are used in this solution for cloning Oracle databases with FlashArray volume snapshots.

Pure Storage FlashArray

FlashArray is a unified block and file storage solution that delivers an effortless, consistent, and efficient data management experience. Known for its advanced data reduction capabilities, it ensures optimal storage efficiency without compromising performance. Key features of FlashArray include the Evergreen® business model, which allows for seamless upgrades to increase capacity and performance without the need for new storage product purchases, offering long-term value. Additionally, FlashArray is designed with sustainability in mind, reducing direct carbon usage in data storage systems by up to 80 percent compared to competitive all-flash systems, and even more when compared to traditional magnetic disk storage.[1]

FlashArray is tailored to meet diverse business needs and workloads through several distinct offerings:

- **FlashArray//C**™: An all-quad-level cell FlashArray providing consistent performance with 2–4ms latency, ideal for capacity-oriented workloads
- **FlashArray//X**™: Delivers ultra-low latency, as low as 150µs, designed for critical applications and business operations
- **FlashArray//XL**™: Offers enterprise-grade performance and scalability, catering to the most demanding workloads
- **FlashArray//E**™: Offers an ideal solution for Oracle Database environments that require the performance of all-flash storage but at a more economical price point; designed to optimize storage efficiency without compromising the reliability and speed needed for critical operations
- **Pure Cloud Block Store**™: A cloud-native block storage solution powered by Purity software, available on your preferred cloud platform

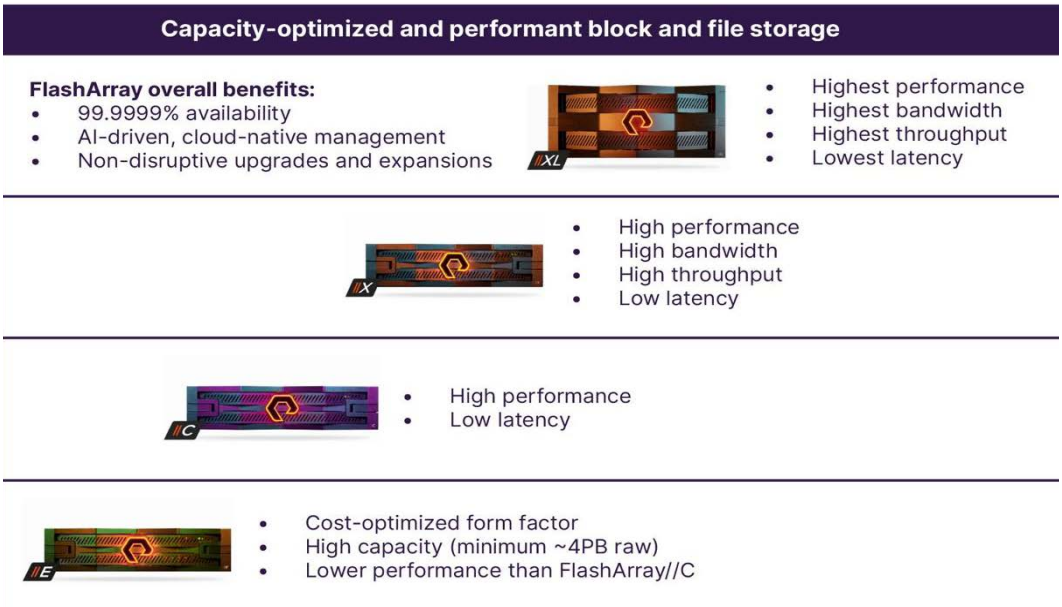

FIGURE 2 FlashArray is tailored to meet diverse business needs and workloads through several distinct offerings.

FlashArray Volumes, Snapshots, and Protection Groups

Snapshots are point-in-time images of a volume. A volume itself is a logical storage unit within a FlashArray. Both snapshots and volumes use metadata pointers to reference data blocks on the FlashArray. A protection group is designed to manage and automate the protection of multiple related storage objects, such as volumes. Snapshots can use protection groups to act on multiple volumes concurrently.

Volumes

A FlashArray volume is a block storage object that can be presented to a host, where it appears as a block device. Volumes store user data on a file system managed by the host's operating system. When a FlashArray volume is exported to a host, it allows applications like Oracle Database to read and write data to this storage.

In FlashArray systems, the internal mapping of data for volumes differs from the way it is presented to the user. Rather than directly containing user data, FlashArray volumes maintain metadata pointers that map to data blocks stored in the array's media pool. When a host writes data to a FlashArray volume, the data itself is stored in the media pool, and the volume's metadata is updated to reflect the location of each new data block. This separation means that the user sees a traditional block storage volume, but the actual storage is managed through metadata pointers, which optimizes storage efficiency and enables features like instant snapshots.

This unique architecture allows FlashArray to create snapshots that reference existing data blocks without duplicating them, making snapshots highly space-efficient.

Snapshots

Volume snapshots provide a fast, secure, and less-disruptive means to capture data at a particular point in time. When a volume snapshot is taken, FlashArray makes a copy of a volume's metadata such that both the volume and the snapshot point to the same data blocks, without copying the individual data blocks. The data blocks that are referenced by the volume and snapshot become read-only and cannot be changed, which makes the snapshot immutable.

If an Oracle Database host requests a volume write that would change any data block that is referenced by a snapshot, that request is redirected to a new data block. FlashArray updates the volume metadata to point to the new data block, while the previous data block is preserved and referenced by the snapshot.

Protection Groups

While a volume is a logical storage unit within a FlashArray environment, a protection group is designed to manage and automate the protection of multiple related storage objects, such as volumes and hosts.

Protection groups offer several advantages, including snapshot management and consistency. For example, they allow for the scheduling of snapshots. When a snapshot is taken, all volumes within the protection group are snapped together to create a point-in-time consistency group across all included volumes.

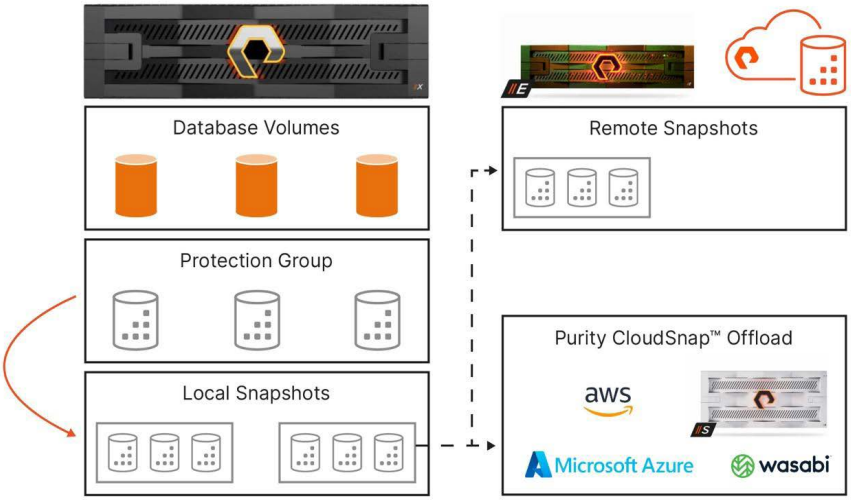

FIGURE 3 High-level overview of FlashArray protection groups, volumes, and snapshots.

Database and storage administrators can use protection groups and snapshots to create consolidated recovery points for Oracle databases. If an Oracle host uses multiple volumes to store user database data, a storage administrator can create a protection group that includes all the user database volumes. Regular snapshots can be scheduled for the protection group, which creates instant recovery points for the user database data. If problems with a database occur, the storage administrator and database administrator can roll a volume back to a previous snapshot instantly, which reduces downtime when compared with restoring a full database backup.

Oracle Database

Oracle databases work seamlessly with FlashArray, ensuring compatibility and enhanced operational efficiency for database administrators. FlashArray is fully compatible with native Oracle tools, such as Oracle Recovery Manager (Oracle RMAN) and Automatic Storage Management (ASM), allowing Oracle administrators to continue using familiar workflows without disruption. FlashArray snapshots complement Oracle RMAN backup and recovery operations, providing an additional layer of speed and reliability, while also integrating seamlessly with ASM. This combination enables streamlined management of Oracle Database storage, optimizing performance and ensuring efficient database operations.

Cloning Oracle Databases with FlashArray Snapshots

Cloning an Oracle database involves creating an exact replica of the database at a specific point in time. This process is widely used for database refreshes, enabling development and testing environments to work with data that mirrors production systems. Cloning is also essential for disaster recovery, where identical copies can be deployed as recovery instances in the event of a failure, and for analytics and reporting, where clones can be used to offload resource-intensive queries without impacting production performance.

Cloning solutions that leverage FlashArray volume snapshot copy/clone procedures ensure minimal downtime, reduced administrative overhead, and seamless support for scenarios that require frequent database replication.

Prerequisites and Considerations

This section describes the foundational elements necessary for a smooth and effective cloning process using FlashArray. By addressing key storage and database configurations upfront, organizations can avoid common pitfalls and optimize the cloning procedure to align with their operational requirements and resources.

Storage Type

Understanding whether databases reside on file system–based storage or ASM is crucial for configuring snapshots and preventing performance or compatibility issues. This section describes how each storage type influences snapshot cloning workflows.

Regardless of the storage type, it must be confirmed that the assigned volumes are accessible and that the Oracle binary versions match those of the source environment.

Verifying Assigned Volumes

Figure 4 shows volumes (for example, cbora2-data and cbora2-fra) assigned to the host group (SingaporeUCS-Prod), including their serial numbers.

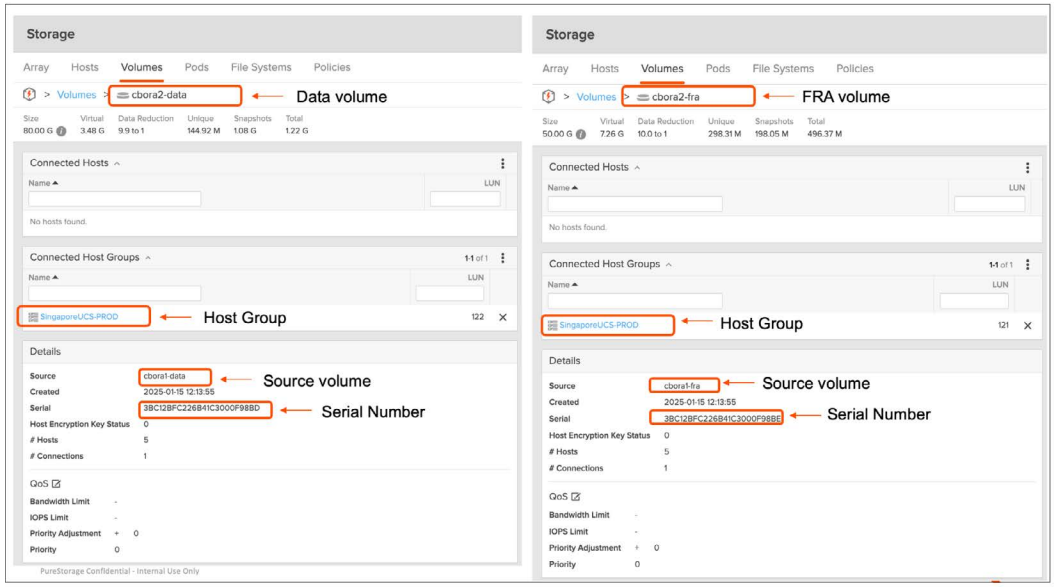

FIGURE 4 Volumes assigned to host groups, including their serial numbers.

Running the command **lsblk -o name,model,serial** on the target host confirms these specific volumes are recognized and correctly mapped.

```
[root@cbora2 ~]#
[root@cbora2 ~]#
[root@cbora2 ~]# lsblk -o name,model,serial
NAME        MODEL            SERIAL
sdd         FlashArray       624a93703bc12bfc226b41c300751634
└─sdd1
sdb         FlashArray       624a93703bc12bfc226b41c3000f98bd
└─sdb1
sr0         VMware SATA CD00 00000000000000000001
sde         Virtual disk
└─sde1
sdc         FlashArray       624a93703bc12bfc226b41c3000f98be
└─sdc1
sda         Virtual disk
├─sda2
│ ├─ol-swap
│ ├─ol-home
│ └─ol-root
└─sda1
```

Matching Oracle Binaries

Using SQL queries (such as **SELECT host_name FROM v$instance;** and **SELECT banner, banner_full FROM v$version**;) confirms that the Oracle Database installation matches the source database's version. This prevents incompatibility issues when cloning.

```
oracle@cbora1.localdomain:/home/oracle [orcl]> sqlplus / as sysdba
SQL*Plus: Release 19.0.0.0.0 - Production on Thu Jan 30 11:16:44 2025
Version 19.5.0.0.0
Copyright (c) 1982, 2019, Oracle.  All rights reserved.
Connected to:
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.3.0.0.0
SQL> select host_name from v$instance;
HOST_NAME
----------------------------------------------------------------
cbora1.localdomain
SQL>
SQL> select banner,banner_full from v$version;
BANNER
--------------------------------------------------------------------------------
BANNER_FULL
--------------------------------------------------------------------------------
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.3.0.0.0
oracle@cbora2.localdomain:/home/oracle [orcl]> sqlplus / as sysdba
SQL*Plus: Release 19.0.0.0.0 - Production on Thu Jan 30 11:17:59 2025
Version 19.3.0.0.0
Copyright (c) 1982, 2019, Oracle.  All rights reserved.
Connected to:
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.3.0.0.0
SQL> select host_name from v$instance;
HOST_NAME
----------------------------------------------------------------
cbora2.localdomain
SQL>
SQL>
SQL>  select banner,banner_full from v$version;
BANNER
--------------------------------------------------------------------------------
BANNER_FULL
--------------------------------------------------------------------------------
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.3.0.0.0
SQL>
```

File System

Before cloning an Oracle database on a file system–based environment, make sure the following conditions are met to ensure a smooth and efficient process:

- **Validated file system paths**: Verify that the file system paths and mount points match your intended directory structure, ensuring the clone can locate datafiles and logs without errors.
- **Initialization file updates**: Copy the init.ora file from the source to the target and adjust paths or parameters to align with the target server's configuration.
- **Resource availability**: Ensure the target host has sufficient CPU, memory, and input/output resources to handle cloning operations alongside production workloads without degradation.
- **Correct Oracle Database permissions**: Confirm that file system permissions (ownership, group memberships, and access modes) are set appropriately for the Oracle Database user and any related service accounts.
- **Granular cloning on virtual machine disks**: When Oracle file systems reside on virtual machine disks and finer cloning control is needed, use a dedicated virtual machine file system datastore per Oracle Database. This approach simplifies snapshot and clone management.

Oracle Automatic Storage Management (ASM)

Before cloning an Oracle database on ASM storage, ensure the following:

- **Persistent device names and permissions**: Configure your system (using UDEV rules or Oracle's ASMLib) so that device names and permissions remain consistent across reboots.
- **Filter driver compatibility**: If using the Oracle ASM Filter Driver, verify compatibility with your Linux kernel version by checking the ASM Filter Driver certification matrix.
- **Renaming disk groups**: When cloning to the same host, rename the disk group to prevent conflicts with the source.
- **VMware environments**: If ASM resides on virtual machine disks and granular cloning is required, dedicate a separate VMware vSphere VMFS datastore for each Oracle Database instance.

Database Type

Different Oracle Database architectures have unique dependencies and requirements. Grasping these distinctions up front helps users tailor their cloning strategies to each environment, ensuring consistency, optimal resource usage, and minimal risk.

Single-Instance Traditional/Standalone

When working with a single-instance Oracle Database environment, verify that the following considerations and prerequisites are in place to ensure a reliable cloning process:

- **Crash-consistent state**: The database should be crash-consistent at snapshot time, meaning all data files, online redo logs, and control files must be in sync to reflect a consistent point in time.
- **Volume requirements**: Provision the necessary volumes for data files and any external files or configurations the database relies on. While there are structural differences between ASM and file system–based environments, additional volumes beyond the primary data areas are not typically required for ASM.
- **Parameter review**: Make certain that initialization parameters, security settings, custom configurations, and database options used in the source are replicated. This preserves the functionality and performance characteristics in the cloned environment.
- **Backup and testing**: Creating a backup of the source database before cloning is highly recommended. Where possible, perform a test run in a non-production environment to validate the cloning workflow and mitigate risks.
- **No hot backup mode required:** As long as the scenario uses Oracle Database 12c Release 2 and above, the database does not need to be placed in hot backup mode for FlashArray snapshot cloning, provided the database is in a restorable state.

The high-level steps for cloning a single instance or standalone database are as follows:

1. **Prepare the source database**: Take a protection-group snapshot of the source database.
2. **Shut down the target database**: Stop the target database to prevent conflicts.
3. **Take the target volumes offline**:
   - Take the ASM disk groups offline.
   - Unmount the file systems.
4. **Copy snapshots to target volumes**: Use FlashArray snapshots to replicate or overwrite the target volumes.
5. **Bring the target volumes online**:
   - Take the ASM disk groups online.
   - Mount the file systems.
6. **Start and verify the cloned database**: Start the database and confirm it is operational.

Single-instance Pluggable Database and Container Database

When performing granular-level cloning of container databases and pluggable databases, the following requirements and considerations must be addressed:

- **ASM configurations**: Each pluggable database must reside on its own ASM disk group (for example, PDB1, PDB2, and PDB3) to enable individual cloning.
- **File system configurations**: Each pluggable database should be configured on a separate file system (for example, /PDB1, /PDB2, and /PDB3) for clear data separation.

The high-level steps for cloning a single-instance pluggable database or container database are as follows:

1. Create the pluggable database on the specified volume; if the volume is ASM, then specify the ASM disk group; if it's a file system, then specify the file system mountpoint.
2. Take a protection-group snapshot of the ASM disk groups or file systems containing the container database or pluggable database data.
3. Shut down the target database, offline the ASM disk group, and unmount the file system.
4. Replicate the snapshot to the target environment by assigning it to the target volumes.
5. Bring the target volumes online:
   - Take the ASM disk groups online.
   - Mount the file systems containing the cloned data.
6. Start the Oracle database.
7. Confirm the pluggable databases are on their respective ASM disk groups or file systems.
8. Open the pluggable database and validate the replicated data.

Oracle Real Application Clusters (Oracle RAC)

Oracle Real Application Clusters (Oracle RAC) is a database clustering solution that enables multiple servers (nodes).

- **Shared storage awareness:** Confirm that all Oracle RAC nodes are aware of, and properly configured to access, the shared storage.
- **Version consistency:** Ensure all Oracle RAC nodes are running the same Oracle Grid Infrastructure and Oracle Database versions to avoid compatibility issues.
- **Network configuration:** Validate that the interconnect and public network configurations are consistent across all nodes.
- **Disk group availability**: Ensure all required ASM disk groups are accessible from every Oracle RAC node.

The high-level steps for cloning an Oracle RAC database are as follows:

1. Ensure the source Oracle RAC database is in a crash-consistent state and all nodes are aware of the shared ASM disk groups.
2. Take snapshots of the ASM disk groups containing the database files.
3. Replicate the ASM snapshots to the target Oracle RAC environment.
4. Confirm all target Oracle RAC nodes have the same Oracle Grid Infrastructure and Oracle Database versions as the source.
5. Take the ASM disk groups online for on all target Oracle RAC nodes.
6. Adjust initialization parameters and reconfigure cluster resources as needed for the target environment.
7. Start the database on the target Oracle RAC cluster and verify functionality across all nodes.

Initial Steps: Creating Snapshots

Volume snapshots are the backbone of the cloning process. This section outlines the critical steps to capture clean, reliable snapshots on FlashArray, forming the starting point for any successful clone. This includes describing how best to prepare and execute these snapshots across various storage and database configurations.

By Storage Type

Each storage approach has unique mechanics for snapshot creation. By following targeted guidance, readers can ensure that snapshots are created efficiently, remain consistent, and align with best practices for their specific storage setup.

File System–based Storage

1. Identify the source database host volumes from the FlashArray user interface.
2. In the example shown in Figure 5, the volumes **ora1-data** and **ora1-fra** are connected to the host group **SYD-Demo-Cluster1**, which includes the three hosts **ESX01**, **ESX02**, and **ESX03**.

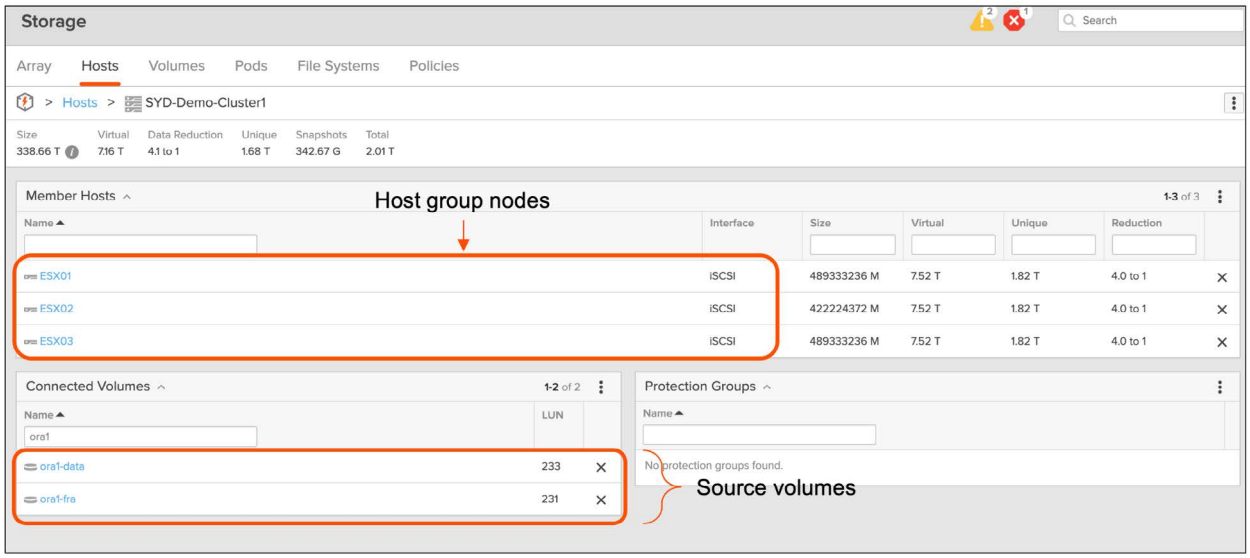

FIGURE 5 File system-based storage snapshots.

3. Confirm the source volume serial numbers on the array match the volumes on the source host. Figure 6 shows that the volumes match:
   - The volume with a serial number ending in **16C19** is logical volume **oradatavg-oradata**, mounted on **/U01**.
   - The volume with a serial number ending in **16C1A** is logical volume **orafravg-orafra**, mounted on **/U02**.

This is confirmed via the **lsblk** command output, which matches the user interface.

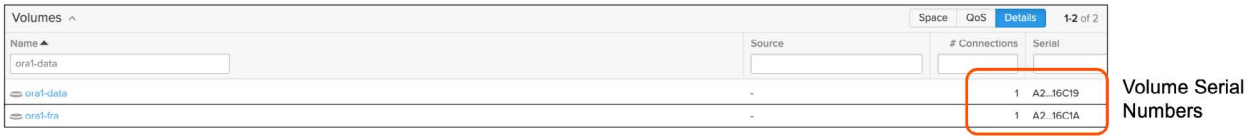

FIGURE 6 Confirm the source volume serial numbers on the array match the volumes on the source host.

```
[root@ora1 ~]#
[root@ora1 ~]# lsblk -o name,model,serial
NAME                  MODEL            SERIAL
sda                   Virtual disk
├─sda1
└─sda2
  ├─ol-root
  ├─ol-swap
  └─ol-home
sdb                   FlashArray       624a9370a21265762db64ece00016c19
└─sdb1
  └─oradatavg-oradata
sdc                   FlashArray       624a9370a21265762db64ece00016c1a
└─sdc1
  └─orafravg-orafra
sdd                   FlashArray       624a9370a21265762db64ece00016c1b
└─sdd1
  └─orafravg1-orafra1
sr0                   VMware SATA CD00 00000000000000000001
[root@ora1 ~]#
[root@ora1 ~]# df -h
Filesystem                      Size    Used    Avail   Use%    Mounted on
devtmpfs                        5.8G    0       5.8G    0%      /dev
tmpfs                           5.8G    0       5.8G    0%      /dev/shm
tmpfs                           5.8G    74M     5.7G    2%      /run
tmpfs                           5.8G    0       5.8G    0%      /sys/fs/cgroup
/dev/mapper/ol-root             49G     12G     38G     24%     /
/dev/sda1                       1014M   321M    694M    32%     /boot
/dev/mapper/ol-home             40G     3.0G    37G     8%      /home
/dev/mapper/oradatavg-oradata   296G    24G     257G    9%      /u01
/dev/mapper/orafravg-orafra     99G     671M    93G     1%      /u02
tmpfs                           1.2G    12K     1.2G    1%      /run/user/42
tmpfs                           1.2G    0       1.2G    0%      /run/user/54321
```

TABLE 1 shows the preparation and creation commands of the protection group.

Identifying Volumes		
purevol	= purevol list ora1 -data ora1-fra	# Identify the source volumes
df -k	= /dev/mapper/oradatavg-oradata 296G 24G 257G 9% /U01	# File system mounted on /U01
df -k	= /dev/mapper/orafravg-orafra 99G 671M 93G 1% /U02m	# File system mounted on /U02
Creating the Protection Group		
purepgroup	= puregroup create orapg	# Create orapg protection group
Adding the Volumes to the Protection Group		
purevol	= purevol add --pgroup orapg ora1-data ora1-fra	# Add volumes to protection group
Listing the Protection Group		
purepgroup	= purepgroup list orapg	# List orapg protection group contents
Taking a Protection Group Snapshot		
purepgroup	= purepgroup snap orapg	# Take a protection group snapshot
Listing the Last Three Protection Group Snapshots		
purepgroup	= purepgroup list –snap orapg –limit 3	# List the three latest snapshots

TABLE 2 Protection group preparation commands.

Figure 7 shows the user interface displaying the protection group name, volumes in the group, and the last three snapshots.

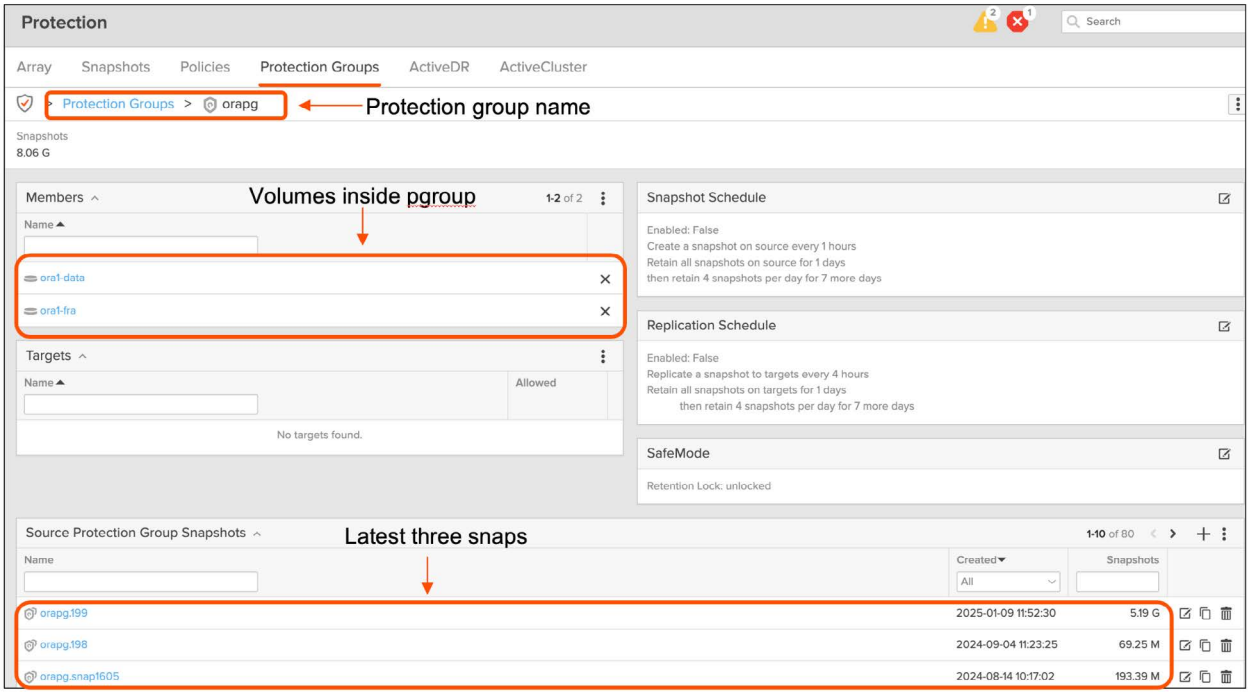

FIGURE 7 The Oracle protection group, its component volumes, and the most recent snapshots.

4. To create a snapshot from the user interface, click the **+** icon under **Source Protection Group Snapshots**. Assigning a suffix to the snapshot name is optional.

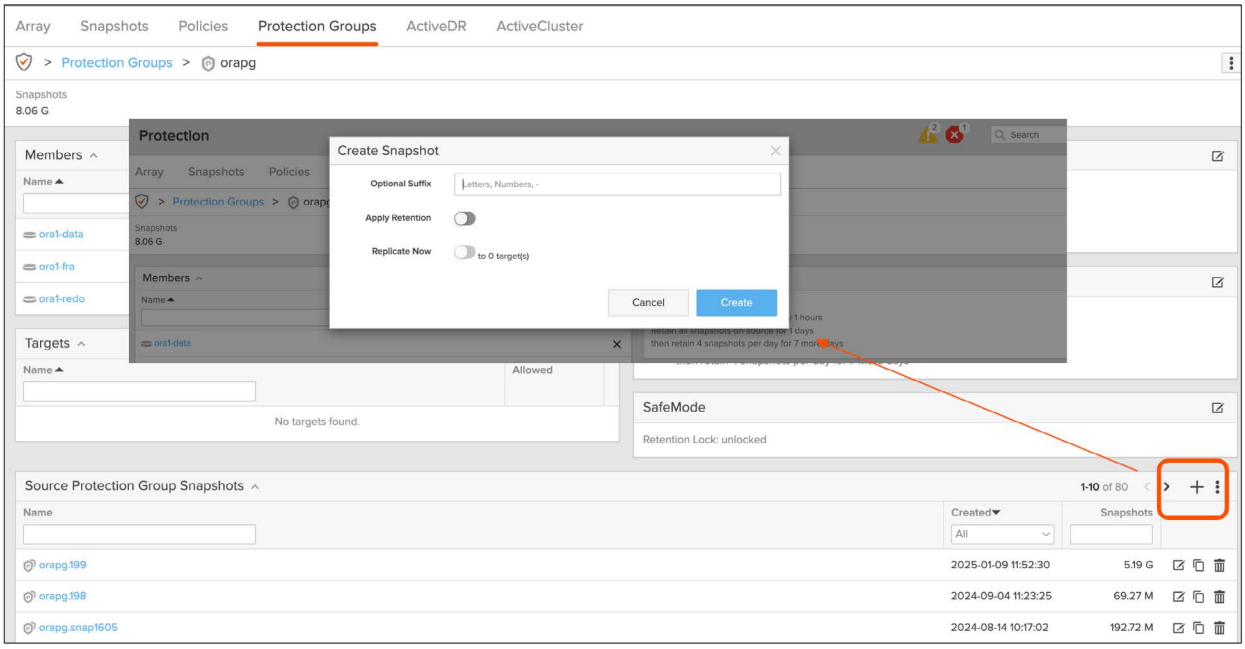

FIGURE 8 Creating a snapshot from the user interface.

ASM-based Storage

The process of capturing protection group snapshots for ASM is the same as for the file system, as all volumes appear identical from the array's perspective. Table 2 outlines the commands for the protection group's preparation.

The most important points to remember are to identify the correct volumes for the source database and to add those volumes to the protection group. The identification process is the same as for file systems:

1. Make sure the disk serial numbers on the host match the volumes on the array. For example, the following command will print the disk device and serial from the array:

```
# lsblk -o name,serial
```

2. Make sure the correct disks are allocated to the respective ASM disk groups.
3. Make sure the correct volumes on the array that are assigned to the host are in the protection group.

Command	Example	Description
Identifying the Volumes on the Source		
purevol	= purevol list cbora1-data cbora1-fra	# Identify the source volumes
asmcmd	= lsdg	# List the ASM disk group
Code:		

```
ASMCMD> lsdg
State    Type    Rebal  Sector  Logical_Sector  Block  AU       Total_MB  Free_MB  Req_mir_free_MB  Usable_file_MB  Offline_disks  Voting_files  Name
MOUNTED  EXTERN  N         512             512   4096  4194304     81916    78296                0           78296              0             N  DATA/
MOUNTED  EXTERN  N         512             512   4096  4194304     51196    43584                0           43584              0             N  FRA/
```

Creating the Protection Group		
purepgroup	= puregroup create asmora	# Create asmora protection group
Adding the Volumes to the Protection Group		
purevol	= purevol add --pgroup asmora cbora1-data cbora1-fra	# Add volumes to protection group
Listing the Protection Group		
purepgroup	= purepgroup list asmora	# List asmora protection group contents
Takinga Protection Group Snapshot		
purepgroup	= purepgroup snap asmora	# Take a protection group snapshot
List the Last Three Protection Group Snapshots		
purepgroup	= purepgroup list –snap asmora –limit 3	# List the three latest snapshots

TABLE 3 Protection group preparation commands.

By Database Type

Whether dealing with simple standalone instances or more complex container- or cluster-based environments, taking snapshots correctly is critical. This section focuses on best practices for each database type, ensuring cloning workflow reliability.

Single-instance Traditional/Standalone and Container Database/Pluggable Database

Once the initial source and target environments are set up, we are ready to clone the database. The following workflow outlines the required steps.

Databases' architecture designs and layouts need to be considered to support the requirements listed in this section. For instance, we might need to copy the **init.ora** files from the source to the target if the directory structures are different.

Note: We have used a Linux-based file system for this document (EXT/XFS). While this approach will also work for ZFS and other Unix-based file systems, this document is based around Linux.

File System

1. Verify the target volumes' file systems are unmounted.
2. Verify the target database is shut down.
3. Create the crash-consistent protection-group snapshot of the source volumes.
4. Copy/overwrite the target volumes with the source volume snapshots.
5. Mount the target file systems.
6. Start the database on the target server.

ASM

1. Verify that the target volumes' ASM disk groups are offline.
2. Verify that the target database is shut down.
3. Create the crash-consistent protection-group snapshot of the source volumes.
4. Copy/overwrite the target volumes with the source volume snapshots.
5. Bring the target ASM disk groups online.
6. Start the database on the target server.

The only difference between the process for a file system and ASM is the unmount and offline commands; the rest of the steps are identical.

Pure Storage snapshots conform to the requirements for crash-consistent snapshots described in Oracle's MOS document 604683.1. That is, crash-consistent snapshots include time stamps of all volumes that demonstrate they are identical and write ordering is maintained. Crash-consistent snapshots, also referred to as "point-in-time" snapshots, capture the state of the datafiles on disk; that is, they do not capture updates in-memory/mid-flight.

For application-consistent snapshots, the following points apply:

- The database is quiesced, which causes a brief pause to database input/output operations.
- Any application-consistent snapshot can be rolled forward to achieve a more granular point-in-time snapshot.

For application-consistent snapshots, do the following:

- Put the database in backup mode.
- Take the protection-group snapshot.
- Take the database out of backup mode.

Refer to the Pure Storage best practices guide for help setting up Oracle Database on Pure Storage.

Oracle RAC

Cloning an Oracle RAC database on a FlashArray involves creating a snapshot-based copy of the database and setting up the cloned environment efficiently.

Cloning Scenarios

This section demonstrates workflows of the snapshot-based cloning process. It provides practical steps, tips, and considerations for a range of configurations.

Cloning Standalone Instances

Ideal for environments needing quick database refreshes or migrations, these scenarios describe how to handle the most common cloning tasks—whether remaining on the same server or moving to a new one.

To the Same Host

1. Take a protection group snapshot on the source.
2. Copy the source snapshot volumes to the new clone volumes on the same host.
3. Attach the new clone volumes to the host.
4. Mount the new clone volumes to the host.
5. Modify any necessary database parameters, such as instance name and file locations, to avoid conflicts with the source database.
6. Start the Oracle Database instance.
7. Verify the clone database.

Logical Volume Manager

Note: If you're using Logical Volume Manager and cloning to the same host, the command **vg importclone** takes care of renaming the volume group and changing the associated universally unique identifier with **PV** and **VG**.

1. Clone the volume, and then attach the clone to the same host and rescan.
2. Make sure the new logical unit number shows up when you run the multipath **-ll** command if you are using multipathing.
3. Import the clone using the commands in Table 3 to change the Logical Volume Manager names.
4. Start the Oracle Database instance.
5. Verify the clone database.

Command	Example	Description
Importing the Clone		
vgimportclone	= vgimportclone -n *<clonedvgname>*/dev/mapper/clonelun	# Import the volume group clone
pvscan	= pvscan -cache	# scan for the physical volume
vgchange	= vgchange -ay *<clonedvgname>*	# execute a change command on the volume group
pvdisplay	= vpvdisplay	# Should show all volumes

TABLE 4 Cloning to the same host with Logical Volume Manager.

Automatic Storage Management (ASM)

The problem is that cloning an ASM logical unit number and mounting it onto the same host would result in two logical unit numbers with the same ASM header information that points to the same disk group name and the same disk name. Similar to a duplicate universally unique identifier error, such a case would result in cloned Logical Volume Manager volumes on the same host.

Oracle generally doesn't support mounting the copy of an ASM logical unit number onto the same server because doing so is error-prone, and admins might accidentally rename or destroy the wrong disk group. This task can be accomplished by using the **renamedg** command, which allows for changing the disk group information at the header-level on the cloned logical unit number. In this example, we are using Oracle ASMLib.

Note: If you do not want to execute the **renamedg** command, and instead want to see what the command would do, you can use the argument **check=TRUE**, which would list the changes but not execute the actual rename command.

```
Cloning ASM to the same host

# Rename using Oracle ASM (using ASMLIB ); this changes the disk name. For example:

oracleasm renamedisk -f <volume name> <new name>
#  example command usage

oracleasm                    = oracleasm renamedisk -f /dev/mapper/data1 DEVDATA1             #  rename disk
group

# Rename using renamedg to modify disk group information as an Oracle Grid Infrastructure user; for example:

renamedg phase=both dgname=original_dg newdgname=new_dg

renamedg                     = renamedg dgname=DATA newdgname=DEVDATA confirm=true asm_diskstring='/dev/mapper/
asmdata          #  rename diskgroup

renamedg                     = renamedg dgname=FRA newdgname=DEVFRA confirm=true asm_diskstring='/dev/mapper/
asmfra        #  rename diskgroup

# Start an ASM disk group:

sqlplus                      = sqlplus / as sysasm
# Login to SQLPLUS as sysasm

sqlplus                      = alter diskgroup DEVDATA mount;                                          # Offline
disk group DATA

sqlplus                      = alter diskgroup DEVFRA  mount;                                            #
Offline disk group FRA
```

To a Different Host

High-level refresh steps include:

1. Shut down the database on the target.
2. Take a protection-group snapshot on the source.
3. Unmount the target file systems or offline the target ASM disk group.
4. Refresh the target volumes with the snapshots of the source volumes.
5. Remount the file systems on the target or mount the ASM disk group.
6. Restart the database.

Cloning Pluggable Databases

Organizations leveraging container and pluggable databases benefit from streamlined resource usage and administration. This subsection outlines the nuances of cloning pluggable databases in various locations, ensuring they maintain consistency and performance.

There are multiple ways to clone pluggable Oracle databases, all of which involve certain key considerations:

- Oracle-native:
- No additional storage required for NOCOPY
- Built-in consistency
- Database-aware
- Limited by storage efficiency
- Pure Storage:
- Storage-efficient
- Fast creation
- Multiple clone refreshes
- Host/storage flexibility
- Might require additional Oracle steps

In this document, we use the Pure Storage method for fast storage and efficient snapshots to create instant pluggable database clones.

To the Same Host

Pluggable database cloning to the same host follows the same guidelines as for same-host cloning with ASM and file systems; see the description for cloning scenarios to the same host earlier in this document.

To a Different Host

The following steps outline the process of cloning a pluggable database on a separate ASM disk group or file system to a different host:

1. Create the pluggable database on the **/u02** file system of the source database.

```
SQL> create pluggable database devpdb admin user admin identified by "oracle12345" create_file_dest = '/u02/
oradata/devpdb';

Pluggable database created.
```

2. Connect to the pluggable database on the source database. To do this, set the session to the pluggable container you want to connect to.

```
SQL> alter session set container = devpdb;

Session altered
```

3. Confirm the pluggable database is running on **/u02**; we can see the datafiles for devpdb are on the /u02 file system.

```
SQL>  SQL> show parameter db_create_file_dest;
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
db_create_file_dest                  string      /u02/oradata/devpdb
SQL>
SQL> select name from v$datafile;
NAME
--------------------------------------------------------------------------------
/u02/oradata/devpdb/ORCL/2CC1D52950FC7119E065000000000001/datafile/o1_mf_system_
msk8w60v_.dbf
/u02/oradata/devpdb/ORCL/2CC1D52950FC7119E065000000000001/datafile/o1_mf_sysaux_
msk8w612_.dbf
/u02/oradata/devpdb/ORCL/2CC1D52950FC7119E065000000000001/datafile/o1_mf_undotbs
1_msk8w613_.dbf
```

4. Now let's add some data to the pluggable database. We will create a table called demo1 and insert data from **dba_objects**, and we will then validate this by printing out the number of lines from the table.

```
SQL> create table demo1 as select * from dba_objects;
SQL> select count(*) from demo1;
```

```
SQL> create table demo1 as select * from dba_objects;
Table created.
SQL> select count(*) from demo1;
    COUNT(*)
————————————
           72355
```

5. We will now create a protection group snapshot of the source volumes that make up the database and the pluggable database; these will be the **/u01** and **/u02** file systems. The underlying FlashArray volumes are **pdbvol1** and **pdbvol2**. To create the snapshot, you can either click the **+** icon on the **Source Protection Group** section, as in Figure 9, or you can use the command-line interface. The following example shows the command running from the array and also from the Linux shell.

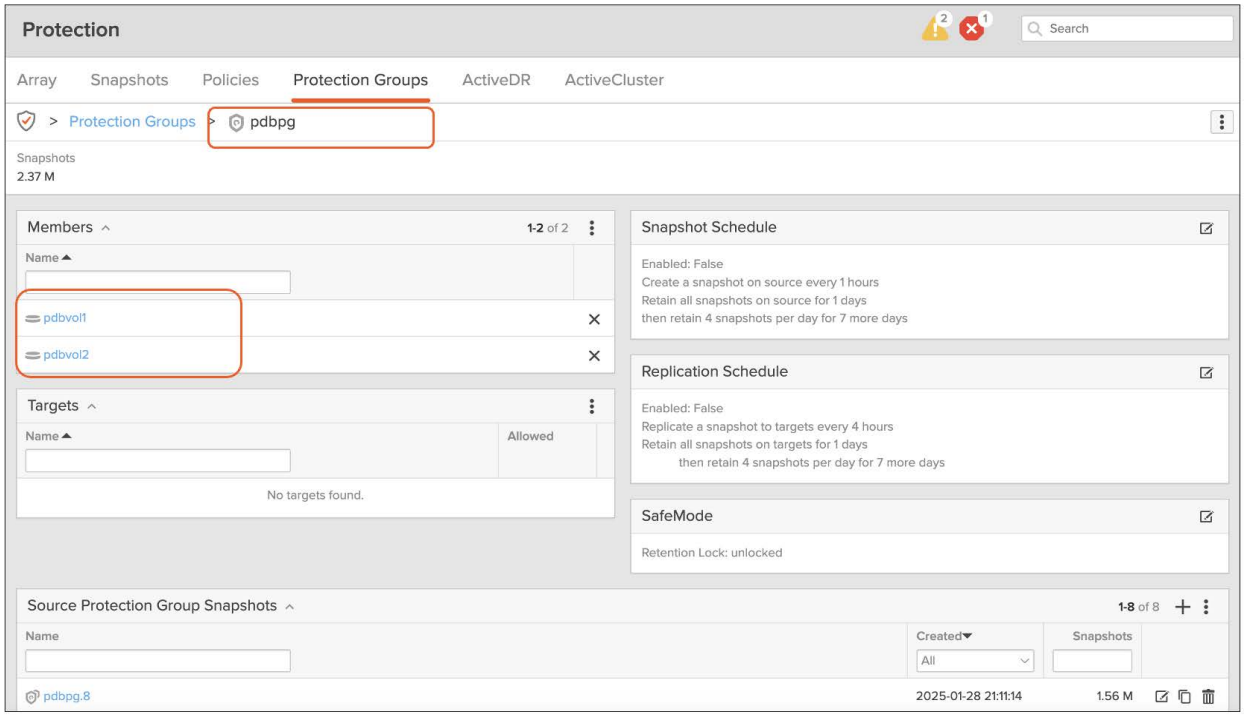

FIGURE 9 To create the snapshot, you can click the **+** icon on the **Source Protection Group** section.

```
purepgroup snap pdbpg
ssh pureuser@192.168.111.130 "purepgroup snap pdbpg"
```

6. Figure 10 shows the volume snapshots have been created, one for each volume. In this example, the suffix **pdpdev** has been provided:
   - pdbpg.pdbdev.pdbvol1
   - pdbpg.pdbdev.pdbvol2

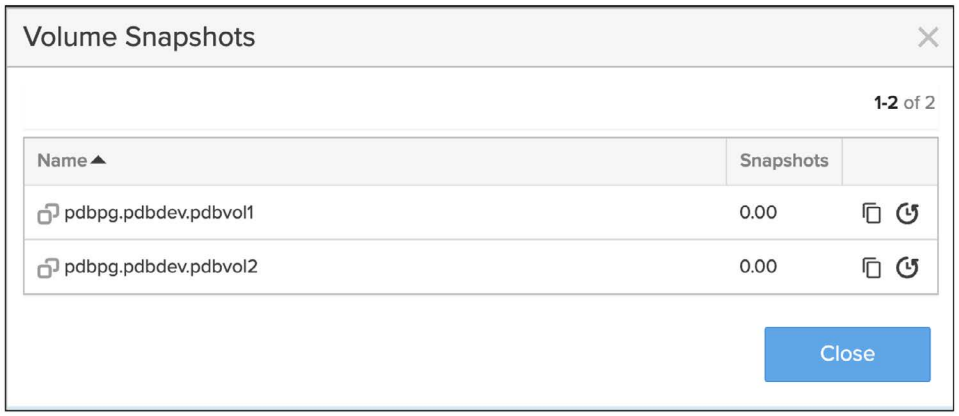

FIGURE 10 Showing that the volume snapshots have been created, one for each volume.

7. Next, copy the snapshots to the target volume. In this example, the target volumes are **pdbclonevol1** (**/u01**) and **pdbclonevol2** (/**u02**).

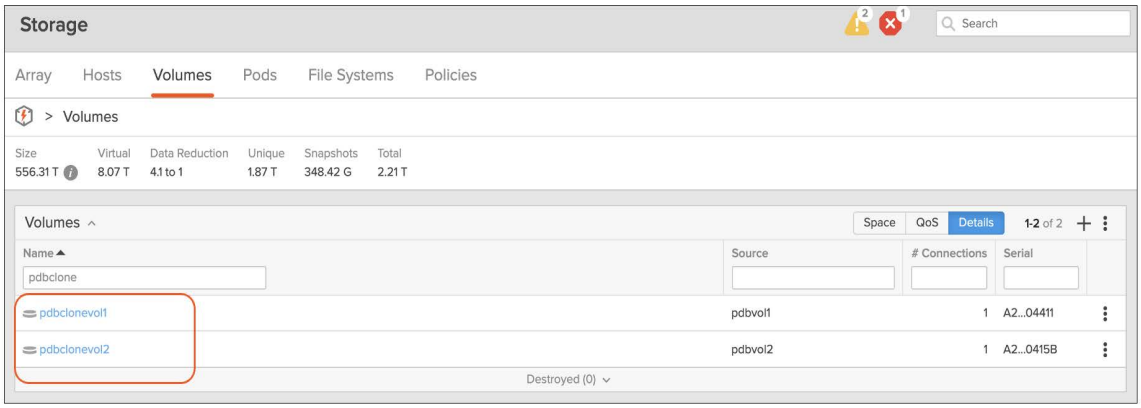

FIGURE 11 Copy the snapshots to the target volume.

8. To overwrite the target volumes, select the required protection-group snapshot, click the **Copy Snapshot** icon, and then provide the volume name to overwrite.

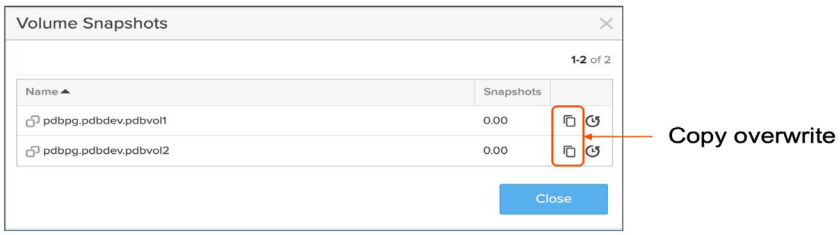

FIGURE 12 Overwriting target volumes.

9. Enter the name of the volumes to overwrite with the snapshot.

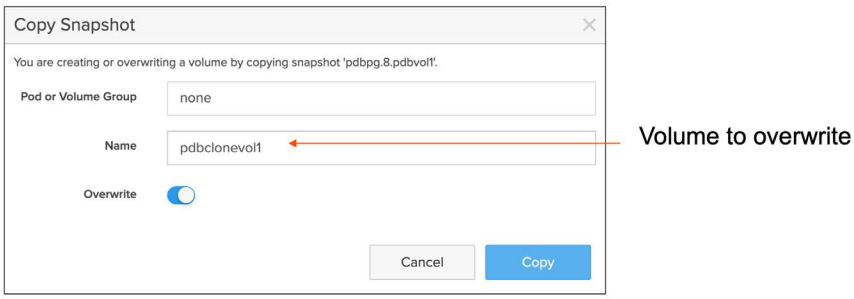

FIGURE 13 Entering the name of the volume to overwrite.

10. Confirm the overwrite will replace the contents of the target volumes. This is a destructive process, so make sure you have selected the correct volumes.

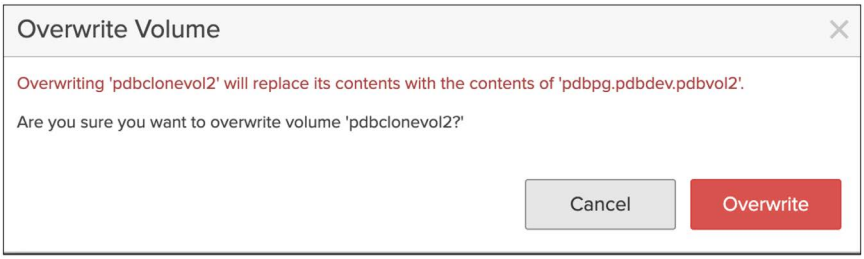

FIGURE 14 Confirm the overwrite will replace the contents of the target volumes.

11. Confirm the target server can see the clone volumes with the correct serial numbers; in this example, **4411** and **415B**. Figure 15 shows the user-interface view of the target volumes, the source volumes, and the serial numbers.

FIGURE 15 Showing the user-interface view of the target volumes, the source volumes, and the serial numbers.

This command identifies the raw volumes that are received from storage to server.

```
oot@pdbclone ~]#
[root@pdbclone ~]# lsblk -o name,serial
NAME        SERIAL
sda
├─sda1
└─sda2
  ├─ol-root
  ├─ol-swap
  └─ol-home
sdb         624a9370a21265762db64ece00904411
└─sdb1
sdc         624a9370a21265762db64ece0090415b
└─sdc1
sr0         00000000000000000001
```

12. Mount the volumes:

```
mount /dev/sdb1 /u01
mount /dev/sdc1 /u02
```

```
[root@pdbclone ~]#
[root@pdbclone ~]#
[root@pdbclone ~]# mount /dev/sdb1 /u01
[root@pdbclone ~]# mount /dev/sdc1 /u02
[root@pdbclone ~]#
[root@pdbclone ~]# df -h
Filesystem              Size    Used    Avail   Use%    Mounted on
devtmpfs                7.7G    0       7.7G    0%      /dev
tmpfs                   7.8G    0       7.8G    0%      /dev/shm
tmpfs                   7.8G    9.9M    7.7G    1%      /run
tmpfs                   7.8G    0       7.8G    0%      /sys/fs/cgroup
/dev/mapper/ol-root     50G     8.3G    42G     17%     /
/dev/mapper/ol-home     42G     2.9G    39G     8%      /home
/dev/sda1               1014M   290M    725M    29%     /boot
tmpfs                   1.6G    8.0K    1.6G    1%      /run/user/42
tmpfs                   1.6G    56K     1.6G    1%      /run/user/54321
tmpfs                   1.6G    0       1.6G    0%      /run/user/1000
/dev/sdb1               50G     12G     39G     24%     /u01
/dev/sdc1               50G     911M    50G     2%      /u02
```

13. Log in as an Oracle administrator, and then start the database.

```
sqlplus / as sysdba
SQL> startup
```

14. Confirm the pluggable databases are visible, and then open the pluggable database in read/write mode.

```
SQL> show pdbs;
SQL> alter pluggable database all open;
```

```
SQL> show pdbs;
    CON_ID      CON_NAME        OPEN MODE       RESTRICTED
----------      ------------    ----------      ----------
                2 PDB$SEED      READ ONLY       NO
                3 PDB1          MOUNTED
                5 DEVPDB        MOUNTED
SQL>
SQL>
SQL> alter pluggable database all open;
Pluggable database altered.
SQL> show pdbs;
    CON_ID      CON_NAME        OPEN MODE       RESTRICTED
----------      --------------  ------------    ----------
        2       PDB$SEED        READ ONLY       NO
        3       PDB1            READ WRITE      NO
        5       DEVPDB          READ WRITE      NO
```

15. Connect to the pluggable database and confirm you can see the demo1 table.

```
SQL> select host_name from v$instance;
SQL> alter session set container = devpdb;
SQL> select count(*) from demo1;
```

```
SQL> select host_name from v$instance;
HOST_NAME
----------------------------------------------------------------
pdbclone.localdomain
SQL>
SQL> select host_name from v$instance;
HOST_NAME
----------------------------------------------------------------
pdbclone.localdomain
SQL>
SQL> alter session set container = devpdb;
Session altered.
SQL>
SQL> select count(*) from demo1;
  COUNT(*)
----------
     72355
SQL>
```

Cloning Oracle RAC on ASM Instances

Replicating a multi-node Oracle RAC environment introduces additional complexity. This section provides insight into managing the cluster interconnect, shared storage, and networking requirements, ensuring a seamless clone with minimal disruption.

To Another Host

Replication to another host has the following prerequisites:

- The source Oracle RAC ASM environment is running on Pure Storage solutions
- The target hosts are configured with required Oracle RAC and Oracle Grid Infrastructure software installed under the same directory structure
- Network and storage connectivity is established
- The same unique identifier and graphical user interface are available on both the source and target database hosts

This process assumes the user will clone all ASM disk groups.

1. Complete the following pre-cloning steps:
   - Check database consistency.
   - Document current configuration settings.
   - Note the ASM disk group configuration.
   - Verify operating system patches and versions on the target environment.
   - Confirm Oracle Grid Infrastructure is up and running.
   - Verify scan listeners are configured.
2. Create protection group snapshots of the source database volumes (including all ASM disk groups).
3. Copy the snapshot volumes to the target environment host.
   - Clone the database volumes.
   - Present the cloned volumes to the target host (map the volumes).
   - Verify the volumes can be seen by the target host.
4. Configure ASM disk groups on the target.
   - Scan for new devices.
   - Create the ASM disk groups using the clone volumes.
   - Verify ASM status.
5. Start the database on the target host.
   - Mount the disk groups.
   - Start Oracle RAC services.
   - Start the database instances.
   - Start the listeners.
   - Verify functionality.
   - Rename the database as required.
6. Complete the validation checklist by ensuring that:
   - All nodes can access the ASM disk groups.
   - The database opens successfully on all nodes.
   - All services are running correctly.
   - Scan listeners are running.
   - Connection to the database is confirmed.
   - Application connections work correctly.

Cloning VMware vSphere Virtualized Databases

For virtualized workloads, proper cloning strategies are essential to maintain performance and resource efficiency. This subsection outlines how to adapt snapshot and cloning practices to VMware environments, covering everything from VMware vSphere Virtual Volumes (vVols) to raw device mapping configurations.

VMware vSphere Virtual Volumes (vVols)

1. Confirm you have a vVol created on the source database server. In the example shown in Figure 16, the cbora1 host does not have any vVols assigned to it; the first step is:
   - Edit the virtual machine settings and select **Add new hard disk**.
   - The new volume size is 79GB.
   - Under the location table, select the vVol datastore to create the vVol on. In the example shown in Figure 16, we use **RedDotX-VVOL**.
   - Click **OK** to create the vVol.

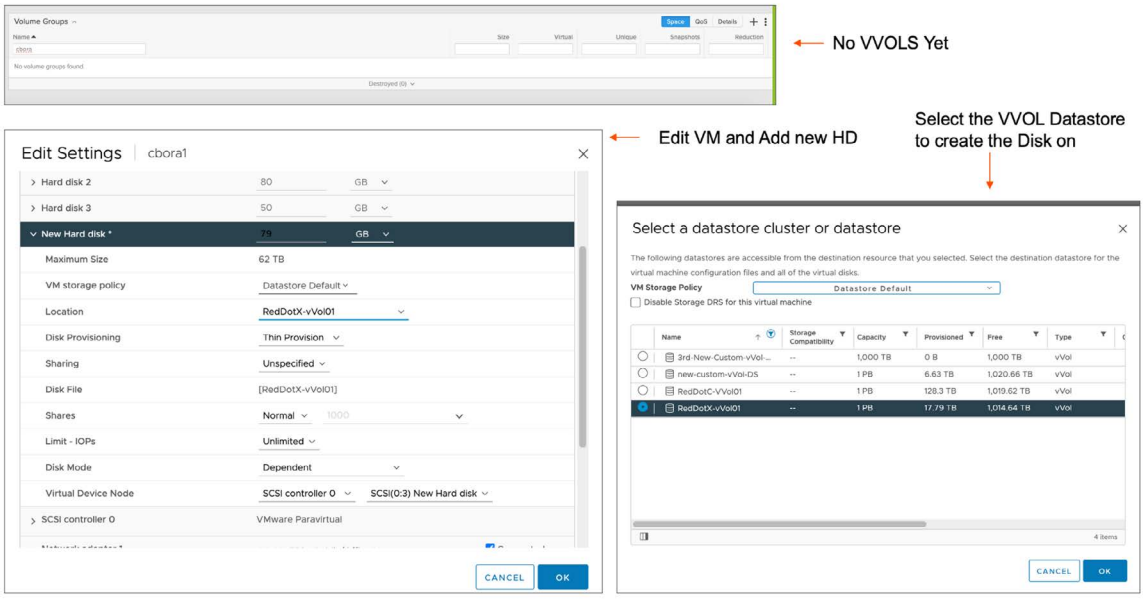

FIGURE 16 An example showing that the cbora1 host does not have any vVols assigned to it.

2. The new vVol will be automatically added to the FlashArray using the Pure Storage plugin. Confirm this with the following:
   - Select the **Volumes** tab in the FlashArray user interface.
   - Under **Volume Groups**, type the name of the host: **cbora1**.
   - The interface displays the new Volume group, called **vvol-cbora1-3a6ce8e5-vg**.

3. When you select the volume group, the user interface will show you the vVols inside the volume group. In this example, the main volume we want is the 79GB volume we created when we added the new disk to the virtual machine.

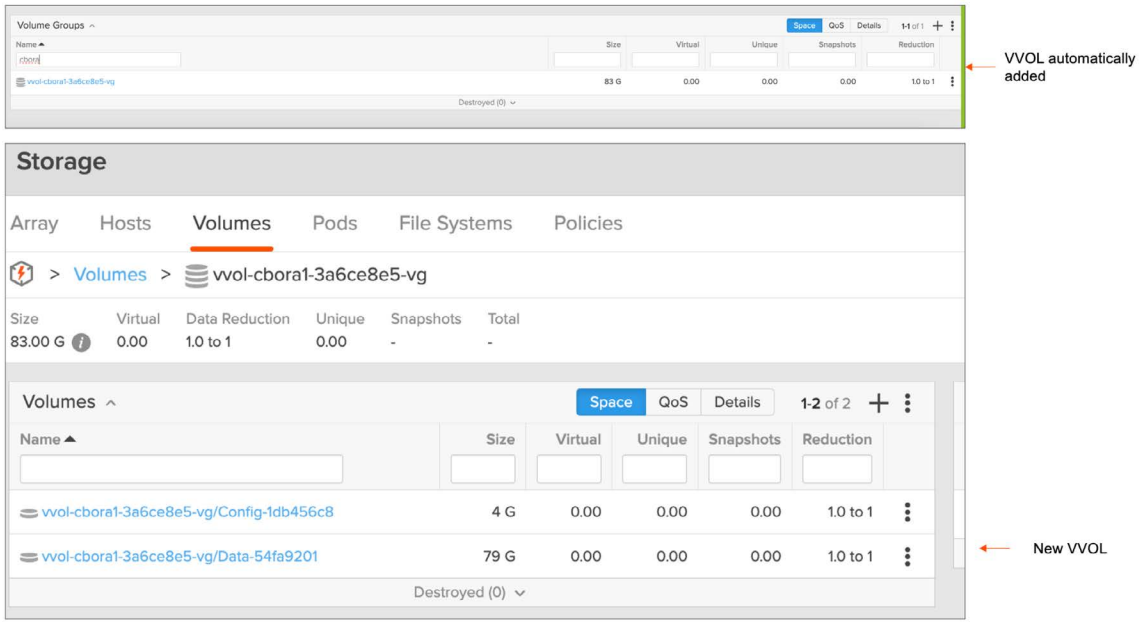

FIGURE 17 The user interface displays the new volume group.

4. Confirm the host operating system can see and mount the volumes for Oracle Database, the host can now see the 79GB disk named **sdd**, and we can see that **/dev/sdd1** is mounted on **/u02**.

```
[root@cbora1 ~]#
[root@cbora1 ~]# lsblk
NAME            MAJ:MIN RM      SIZE    RO TYPE MOUNTPOINT
sdd             8:48    0       79G     0       disk
└─sdd1          8:49    0       79G     0       part /u02
sdb             8:16    0       80G     0       disk
└─sdb1          8:17    0       80G     0       part
sr0             11:0    1       4.4G    0       rom  /run/media/oracle/OL-7.7 Server.x86_64
sdc             8:32    0       50G     0       disk
└─sdc1          8:33    0       50G     0       part
sda             8:0     0       100G    0       disk
├─sda2          8:2     0       99G     0       part
| ├─ol-swap     252:1   0       9G      0       lvm  [SWAP]
| ├─ol-home     252:2   0       40G     0       lvm  /home
| └─ol-root     252:0   0       50G     0       lvm  /
└─sda1          8:1     0       976M    0       part /boot
[root@cbora1 ~]#
[root@cbora1 ~]# df -h
Filesystem              Size    Used    Avail   Use%    Mounted on
devtmpfs                7.7G    0       7.7G    0%      /dev
tmpfs                   7.8G    637M    7.1G    9%      /dev/shm
tmpfs                   7.8G    98M     7.7G    2%      /run
tmpfs                   7.8G    0       7.8G    0%      /sys/fs/cgroup
/dev/mapper/ol-root     50G     35G     16G     69%     /
/dev/mapper/ol-home     40G     7.3G    33G     19%     /home
/dev/sda1               973M    338M    636M    35%     /boot
tmpfs                   1.6G    32K     1.6G    1%      /run/user/54321
/dev/sdd1               79G     14G     66G     17%     /u02
/dev/sr0                4.5G    4.5G    0       100%    /run/media/oracle/OL-7.7 Server.x86_64
```

```
Oracle Profile
ORACLE_SID=orafs
ORACLE_BASE=/u02/app/oracle
DB_UNIQUE_NAME=orafs
ORACLE_HOME=$ORACLE_BASE/product/19c/dbhome_1
PATH=$PATH:$HOME/.local/bin:$HOME/bin:$ORACLE_HOME/bin
export PS1='`/usr/bin/whoami`@${HOSTNAME}:${PWD} [${ORACLE_SID}]> '
export PATH ORACLE_SID ORACLE_BASE ORACLE_HOME DB_UNIQUE_NAME
```

5. Log in to SQL and create a new test table on the source host:

```
create table table1 as select * from dba_objects;
```

```
oracle@cbora1.localdomain:/home/oracle [orcl]> sqlplus / as sysdba
SQL*Plus: Release 19.0.0.0.0 - Production on Wed Jan 29 21:51:42 2025
Version 19.5.0.0.0
Copyright (c) 1982, 2019, Oracle.  All rights reserved.
Connected to:
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.3.0.0.0
SQL>create table table1 as select * from dba_objects;
Table created.
SQL>select count(*) from table1;
  COUNT(*)
  --------
     72373
```

6. Next, clone the vVol by creating a protection-group snapshot and using the snapshot to create a new volume. In this example, we will use the snapshot with the suffix **snap3** to create the clone volume and attach it to host **cbora2**.

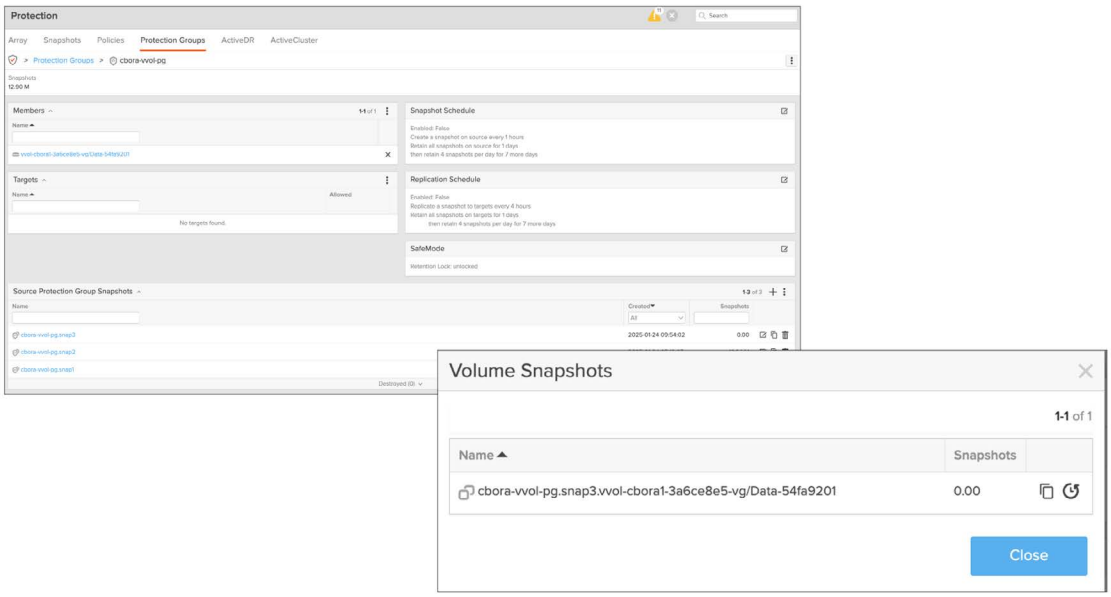

FIGURE 18 This example shows the snapshot with the suffix **snap3** attached to host **cbora2**.

7. Select the virtual machine; in this case, **cbora2**. If you select **vVols**, you see no vVols are attached; using the VMware plugin makes it easy to import the snapshot from the FlashArray:
   - Select **import diskhen search** for the virtual machine that the source vVols are on; in this example, **cbora1**.
   - Expand the hard disk to get a listing of the recent snapshots; in this example, **hard disk 4**.
   - Select the snapshot from the suffix **snap3**, and then import the clone.

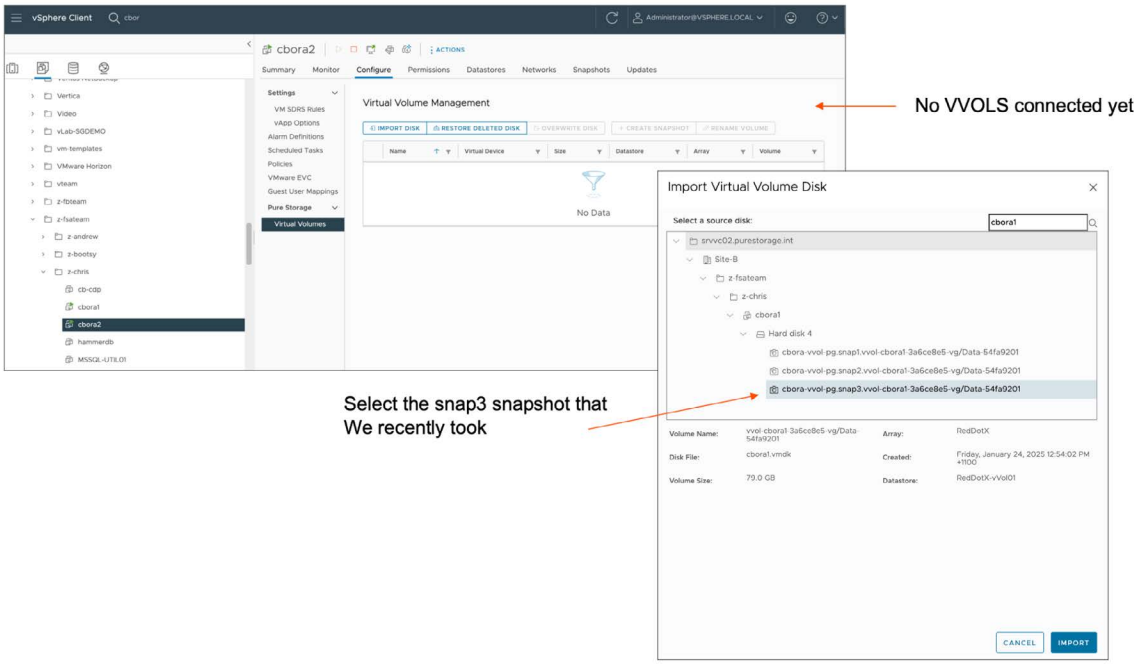

FIGURE 19 Selecting the correct snapshot and importing the clone.

8. The new volume import displays as a vVol.

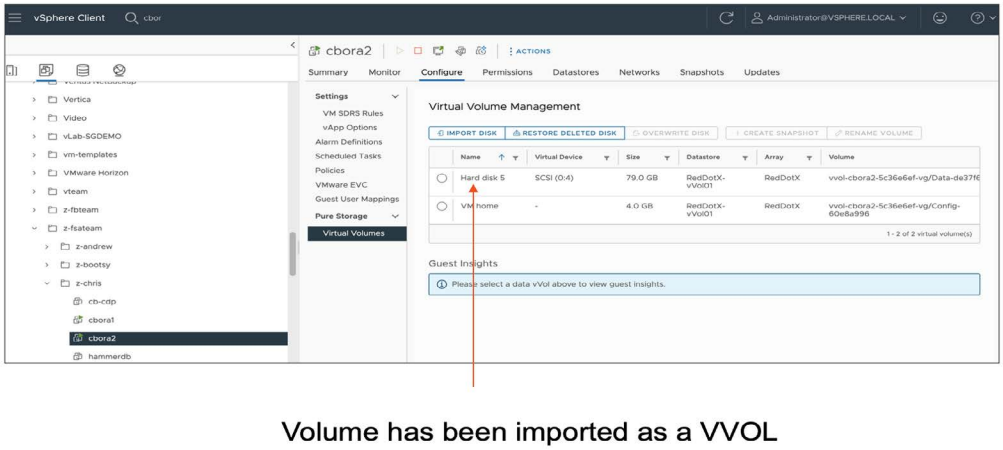

FIGURE 20 The new volume import displays as a vVol.

9. As shown in Figure 20, the new clone disk **/dev/sde** appears with the XFS file system, which has been mounted on **/u02**.

```
[root@cbora2 ~]# lsblk
NAME            MAJ:MIN         RM      SIZE    RO TYPE MOUNTPOINT
sdd             8:48            0       80G     0       disk
└─sdd1          8:49            0       80G     0       part
sdb             8:16            0       80G     0       disk
└─sdb1          8:17            0       80G     0       part
sr0             11:0            1       4.4G    0       rom
sde             8:64            0       79G     0       disk
└─sde1          8:65            0       79G     0       part /u02
sdc             8:32            0       50G     0       disk
└─sdc1          8:33            0       50G     0       part
sda             8:0             0       100G    0       disk
├─sda2          8:2             0       99G     0       part
| ├─ol-swap     252:1           0       9G      0       lvm  [SWAP]
| ├─ol-home     252:2           0       40G     0       lvm  /home
| └─ol-root     252:0           0       50G     0       lvm  /
└─sda1          8:1             0       976M    0       part /boot
[root@cbora2 ~]#
[root@cbora2 ~]# df -h
Filesystem              Size    Used    Avail   Use%    Mounted on
devtmpfs                7.7G    0       7.7G    0%      /dev
tmpfs                   7.7G    637M    7.1G    9%      /dev/shm
tmpfs                   7.7G    106M    7.6G    2%      /run
tmpfs                   7.7G    0       7.7G    0%      /sys/fs/cgroup
/dev/mapper/ol-root     50G     35G     16G     70%     /
/dev/mapper/ol-home     40G     11G     30G     26%     /home
/dev/sda1               973M    440M    534M    46%     /boot
tmpfs                   1.6G    12K     1.6G    1%      /run/user/42
tmpfs                   1.6G    0       1.6G    0%      /run/user/54321
/dev/sde1               79G     14G     66G     17%     /u02
```

10. Make sure the correct Oracle profile is set to point to the correct Oracle system identifier and Oracle home.

```
Oracle Profile
ORACLE_SID=orafs
ORACLE_BASE=/u02/app/oracle
DB_UNIQUE_NAME=orafs
ORACLE_HOME=$ORACLE_BASE/product/19c/dbhome_1
PATH=$PATH:$HOME/.local/bin:$HOME/bin:$ORACLE_HOME/bin
export PS1='`/usr/bin/whoami`@${HOSTNAME}:${PWD} [${ORACLE_SID}]> '
export PATH ORACLE_SID ORACLE_BASE ORACLE_HOME DB_UNIQUE_NAME
```

```
SQL> select name from v$database;
NAME
----
ORAFS
SQL> select count(*) from table1;
  COUNT(*)
----------
     72373
```

Virtual Machine Disk Files

Cloning databases on virtual machine disks can be challenging if you are after a granular level of cloning, such as per-database. Most existing or legacy VMware environments have multiple databases on a single virtual machine disk with this type of configuration. When we take a protection group snapshot of the datastore volumes, we snapshot all the databases.

If we are looking to only snapshot and clone a single database, then that database needs to be on its own virtual machine disk datastore. The following example walks through setting up a database per virtual machine disk and cloning that database.

1. Identify the storage volumes on the FlashArray that make up the source database volumes.
2. Create the volumes of the clone databases, either from the snapshots of the source volumes or from scratch.
3. Attach the clone volumes to the VMware ESX host or cluster.
4. Copy the source datastore volume to the clone datastore volume.
5. Create the clone datastore with a new signature.
6. Set up the virtual machine for the clone database, and then attach clone volumes.
7. Start up the clone database.

8. Identify the source volumes that make up the virtual machine disk datastore.

   – In the example shown in Figure 21, the volume of the source datastore is called **ora-ds-vmdk-cb**, it's connected to **SYD-Demo-Cluster1**, the logical unit number is **175**, and the serial number ends in **C1DE**.

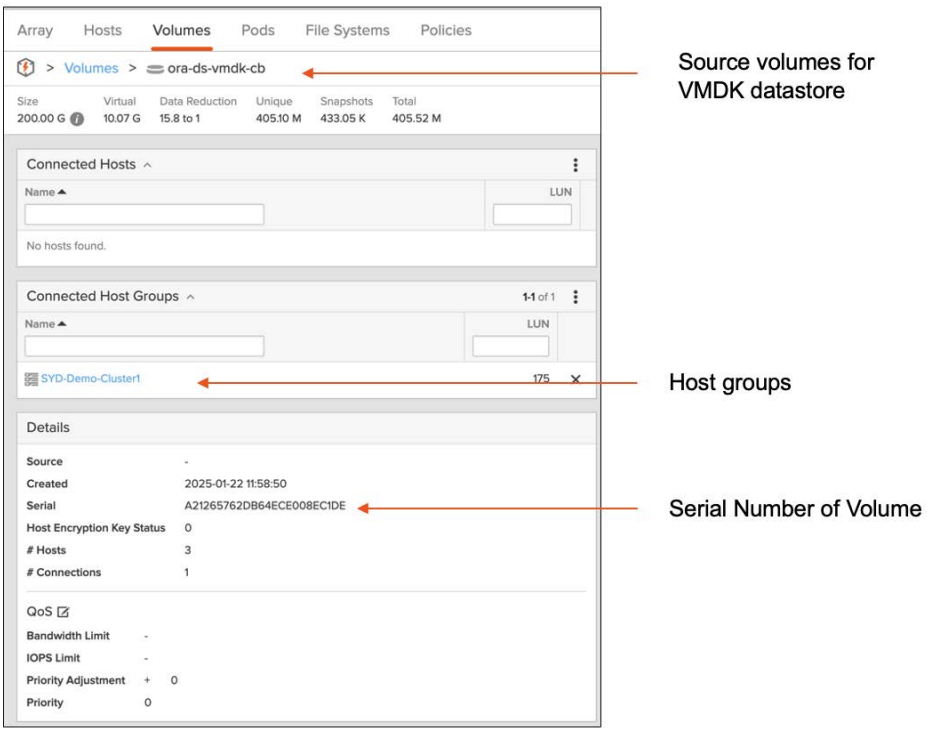

FIGURE 21 Identify the source volumes that make up the virtual machine disk datastore.

   – Confirm the device back end from the VMware ESX environment. Figure 22 shows the correct logical unit number; it matches the serial number from Figure 21.

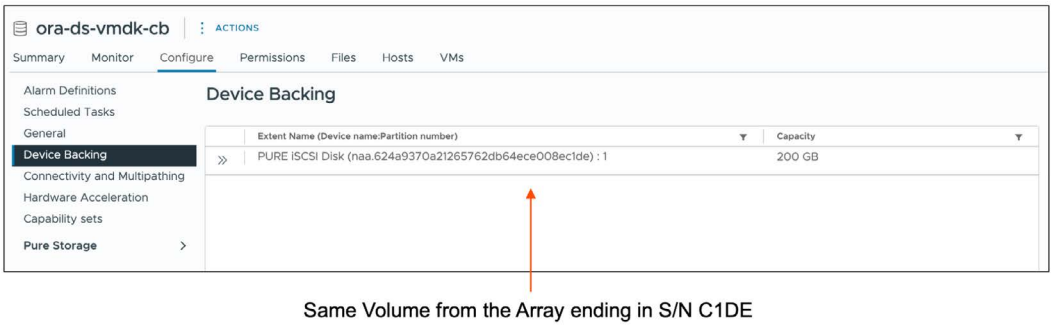

FIGURE 22 Confirm the device back end from the VMware ESX environment.

9. Confirm the Oracle virtual machine, **ora5**, is using the correct virtual machine disk. Figure 23 shows a 100GB disk created on the virtual machine disk called **ora-ds-vmdk-cb**, which is mounted on the ora5 Linux host.

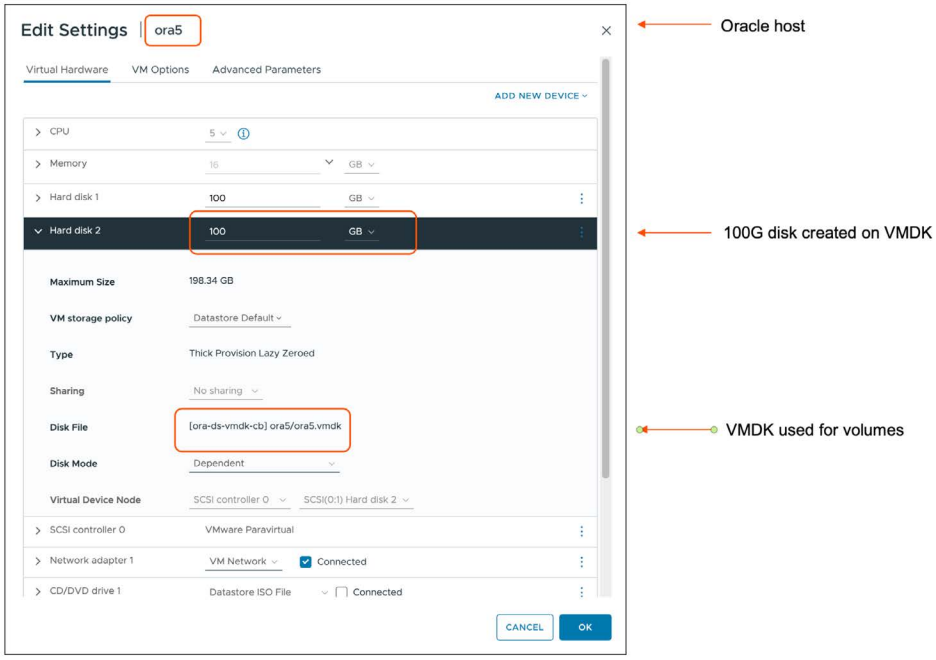

FIGURE 23 Confirm the Oracle virtual machine, **ora5**, is using the correct virtual machine disk.

10. Create the protection group for the virtual machine disk datastore volumes, and then add the datastore volume **ora-ds-vmdk-cb** to the protection group.

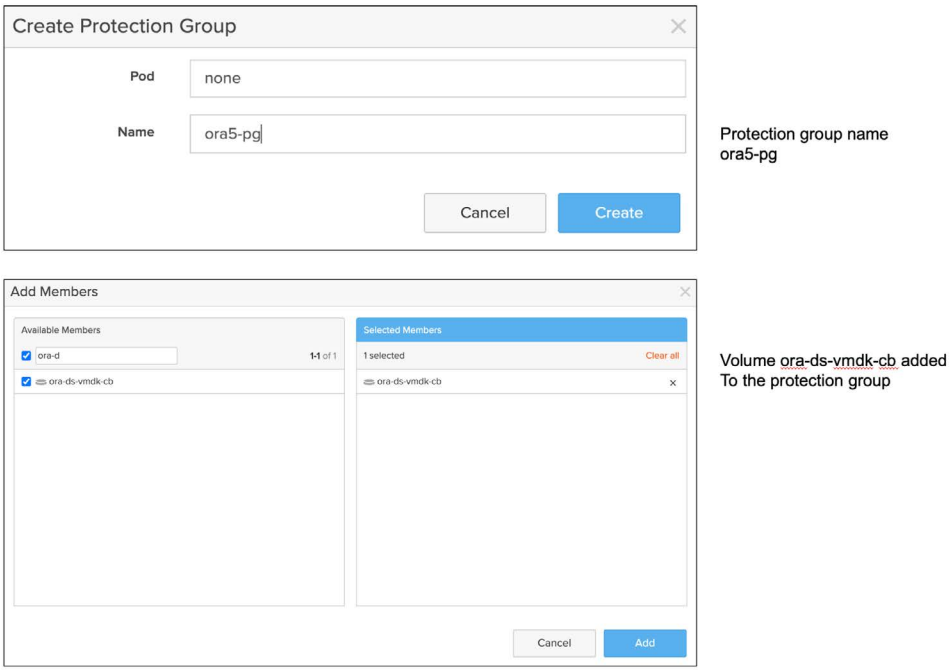

FIGURE 24 Creating a protection group and adding the volume to it.

11. Take the protection-group snapshot; the example in Figure 25 uses the suffix **snap1** to create the protection-group snapshot; the snapshot name is **ora5-pg.snap1.ora-ds-vmdk.cb**.

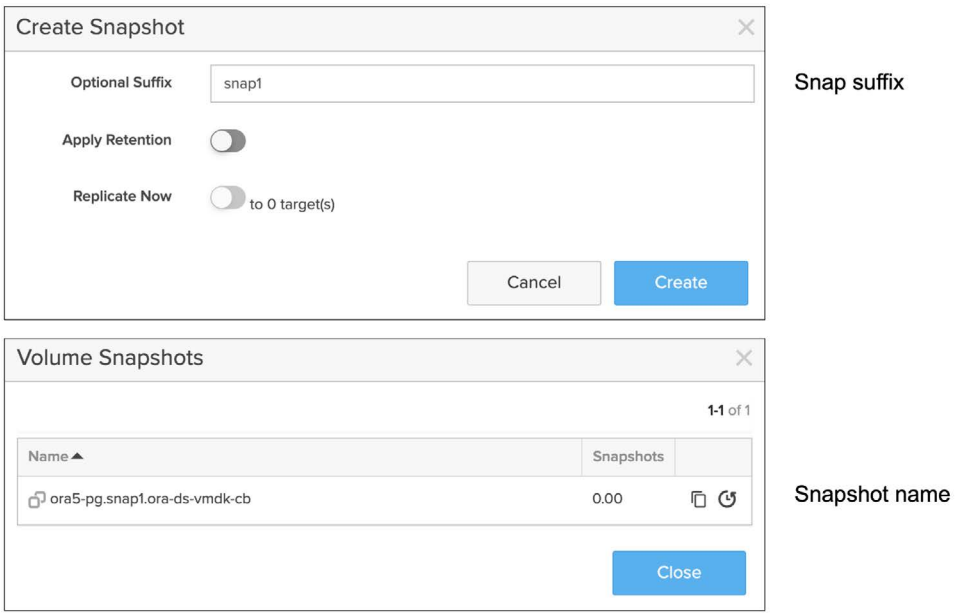

FIGURE 25 Take the protection group snapshot.

12. Create a new clone volume of the datastore volumes from the protection-group snapshot. Note that the volume is called **ora-df-vmdk-clone-db**; this will be the new name for the clone datastore. The volume is connected to the **SYD-Demo-Cluster** with serial number **C553**.

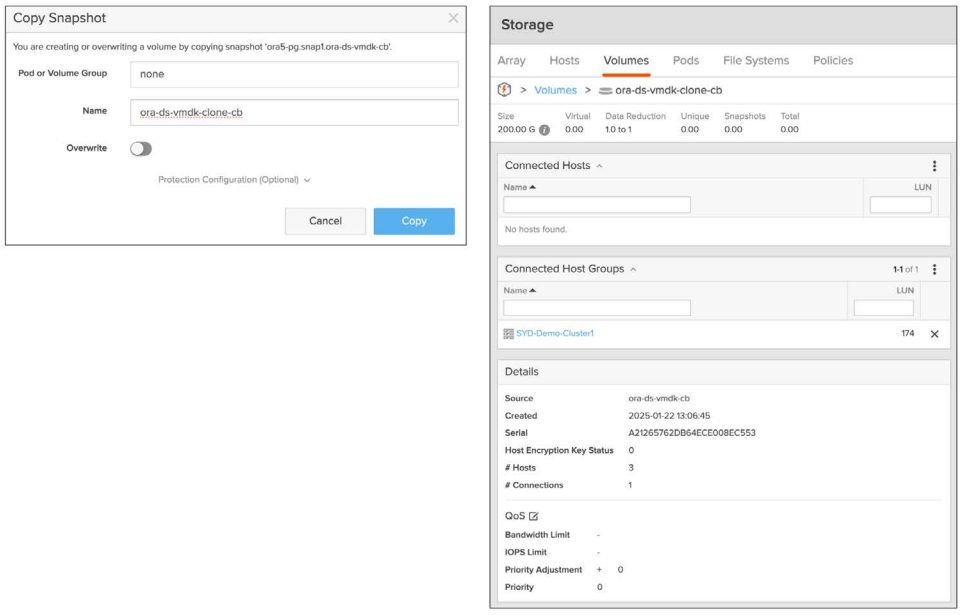

FIGURE 26 Create a new clone volume of the datastore volumes from the protection-group snapshot.

13. Once the clone volume is attached to the host, scan for new volumes using the VMware ESX cluster.

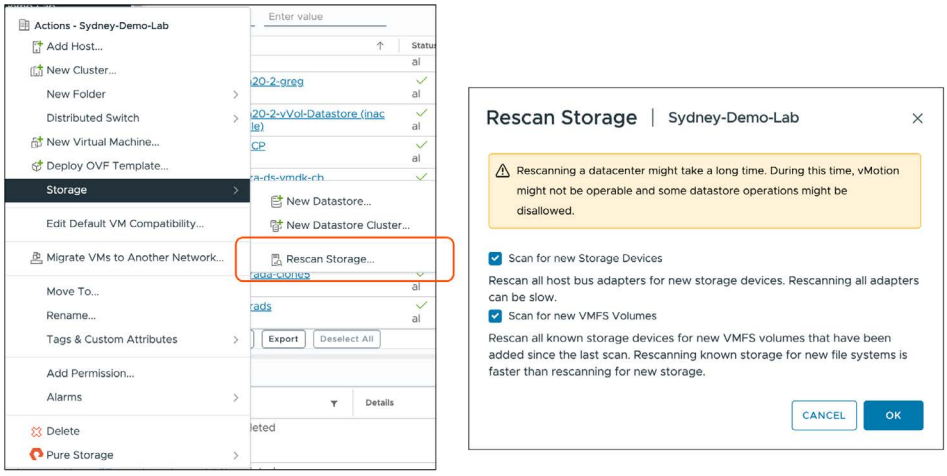

FIGURE 27 Scan for new volumes using the VMware ESX cluster.

14. Create a new datastore from the clone volume:

- Select **New Datastore**.
- Select **VMFS**.
- Enter the new clone datastore name, **ora-ds-vmdk-clone-cb using LUN 174 S/N C553**.

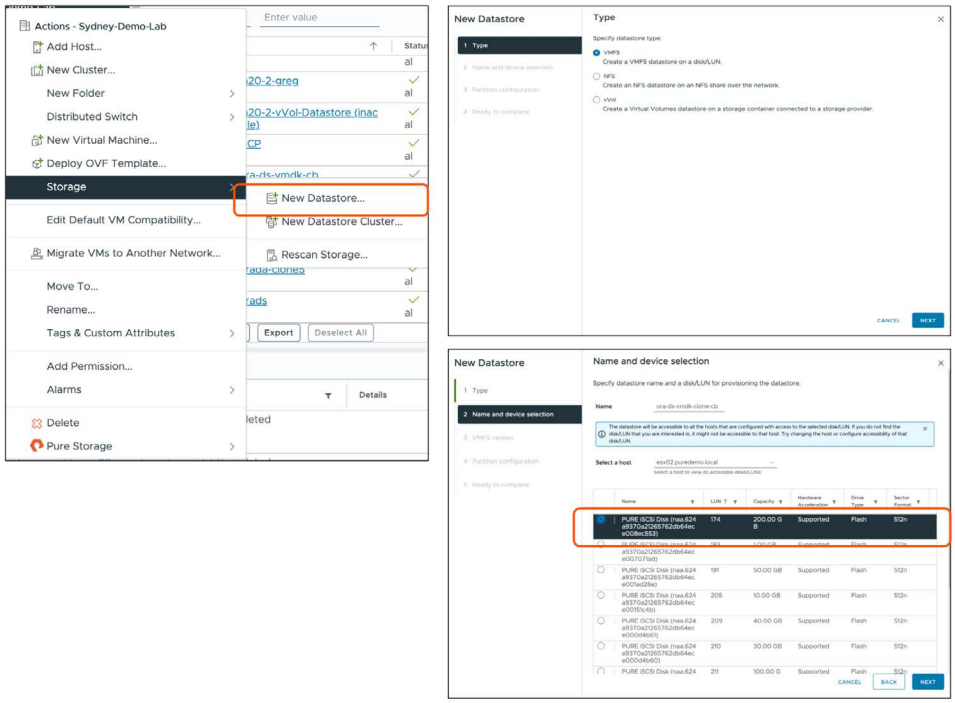

FIGURE 28 Create a new datastore for the clone volume.

15. Because we are cloning an existing datastore, we need to write a new signature to the datastore; otherwise, we will get a signature conflict error.

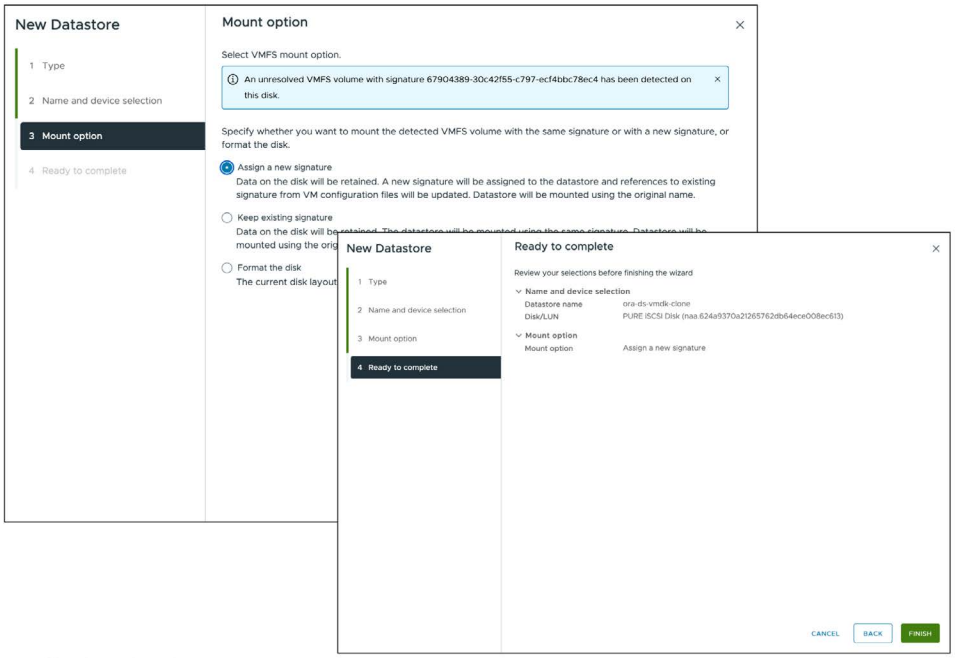

FIGURE 29 Write a new signature to the datastore to avoid a signature conflict error.

16. The new clone datastore is imported with the new name allocated to it. We can now use this datastore to present the clone disk to the same virtual machine or to another virtual machine. In the example shown in Figure 30, we present the clone disk to a new host called **ora5-clone**.

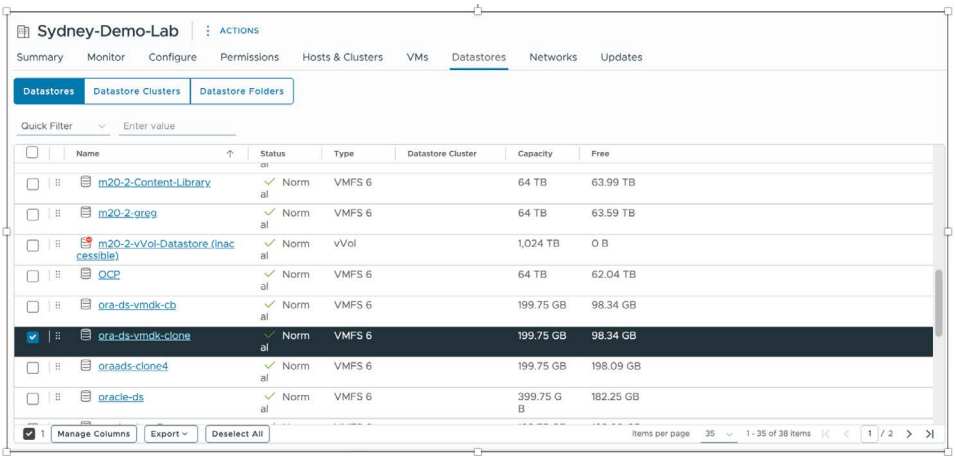

FIGURE 30 The new clone datastore is imported with the new name allocated to it.

17. Select the new **ora5-clone** virtual machine, click **Actions**, and then click **Edit settings**.

18. Select **Add device**.

19. Select **Existing hard drive**.

20. You will see the new clone datastore. Select the datastore and then the **ora5-vmdk** disk to attach.

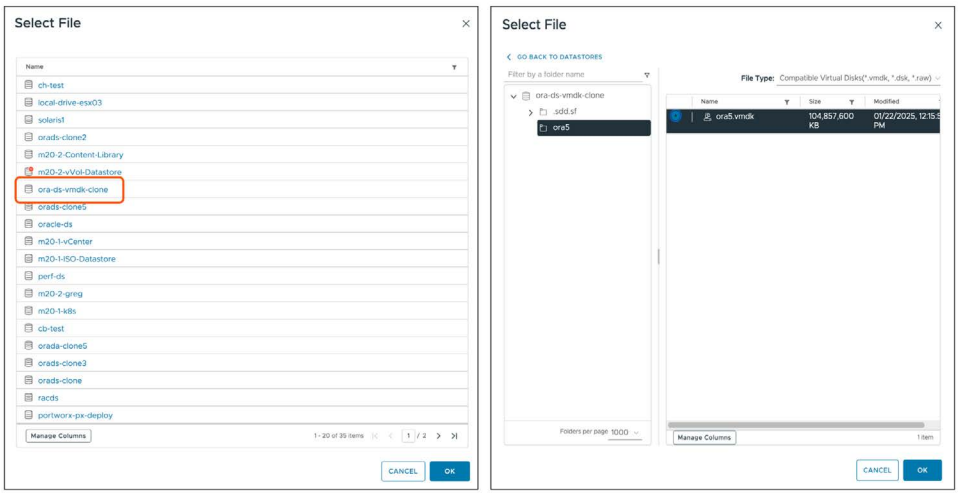

FIGURE 31 Showing the new clone datastore.

21. The new disk is now attached to the ora5-clone virtual machine. We can confirm this from the **Edit Settings** tab, which also shows that the operating system can see the new 100GB clone volume.

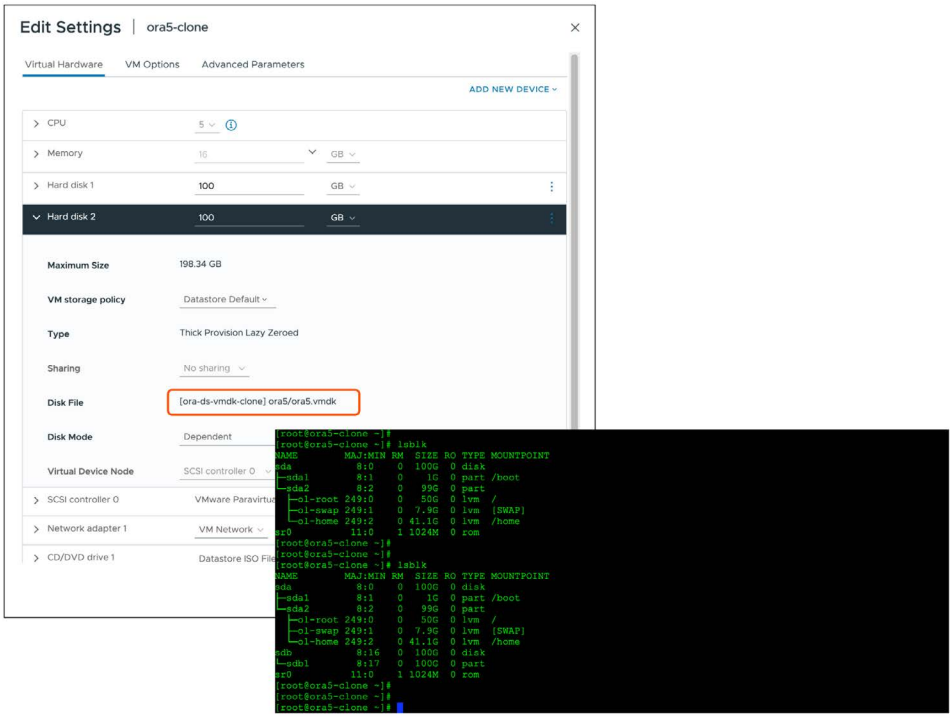

FIGURE 32 The new disk is now attached to the ora5-clone virtual machine, as seen in the **Edit Settings** tab.

22. Mount the volume, start up the Oracle database, and confirm that the table we created on ora5 is present.

```
mkdir /u01
mount /dev/sdb1 /u01
su - oracle
sqlplus / as sysdba
select count(*) from tb1;
[oracle@ora5-clone ~]$ df -h
Filesystem              Size    Used    Avail   Use%    Mounted on
devtmpfs                7.7G    0       7.7G    0%      /dev
tmpfs                   7.8G    0       7.8G    0%      /dev/shm
tmpfs                   7.8G    26M     7.7G    1%      /run
tmpfs                   7.8G    0       7.8G    0%      /sys/fs/cgroup
/dev/mapper/ol-root     50G     7.4G    43G     15%     /
/dev/sda1               1014M   314M    701M    31%     /boot
/dev/mapper/ol-home     42G     39M     42G     1%      /home
tmpfs                   1.6G    4.0K    1.6G    1%      /run/user/42
tmpfs                   1.6G    44K     1.6G    1%      /run/user/1000
/dev/sdb1               100G    11G     90G     11%     /u01
/dev/sdc1               50G     743M    50G     2%      /u02
tmpfs                   1.6G    0       1.6G    0%      /run/user/54321
```

```
[oracle@ora5-clone ~]$ sqlplus / as sysdba
SQL*Plus: Release 19.0.0.0.0 - Production on Thu Jan 30 08:52:18 2025
Version 19.3.0.0.0
Copyright (c) 1982, 2019, Oracle.  All rights reserved.
Connected to:
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.3.0.0.0
SQL>
SQL> select count(*) from tb1;
  COUNT(*)
----------
     72373
```

Raw Device Mappings

1. Identify the storage volumes on the FlashArray that make up the source database volumes and add those volumes to a protection group.
2. Take the protection-group snapshot of the source database volumes.

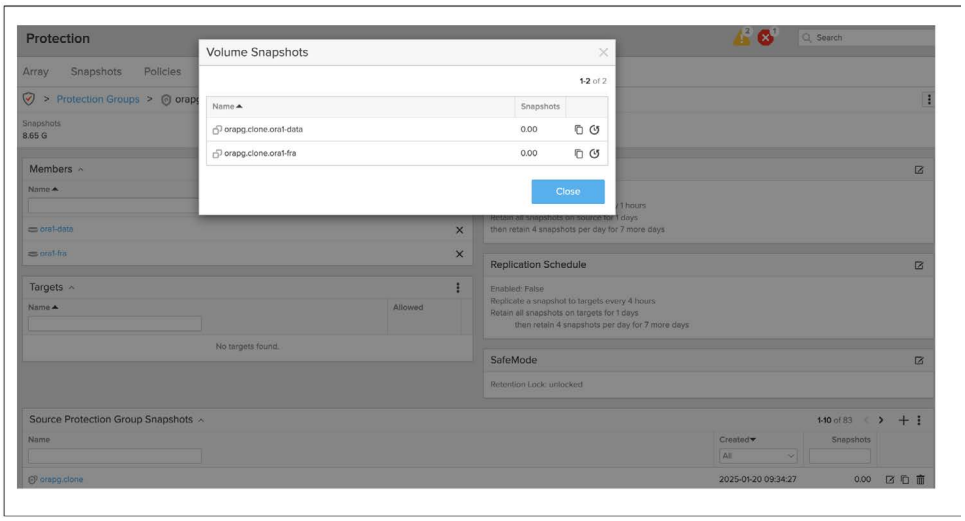

FIGURE 33 The copy-to data and file-recovery area with a new volume.

3. Copy the individual snapshots to the new clone volumes; in this example, we copy **orpag-clone-ora1-data** to **ora4-data** and **ora4-fra**.

```
purevol copy snapshot-name ora4-data
purevol copy snapshot-name ora4-fra
```

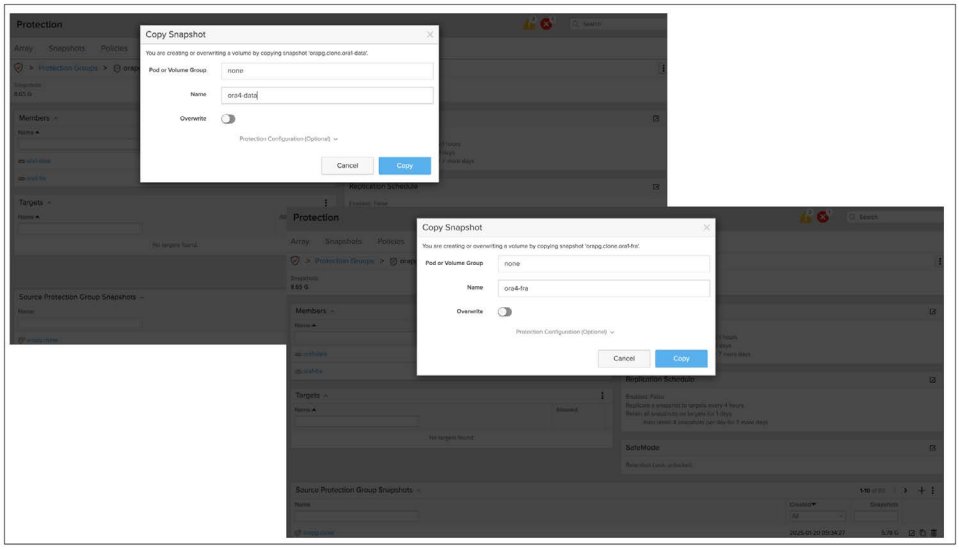

FIGURE 34 Copy the individual snapshots to the new clone volumes.

4. Take note of the new volumes and the serial numbers allocated with the new volumes; these will be used in VMware to allocate volumes to the virtual machine.

5. Connect the new clone volumes to the VMware ESX cluster host group.

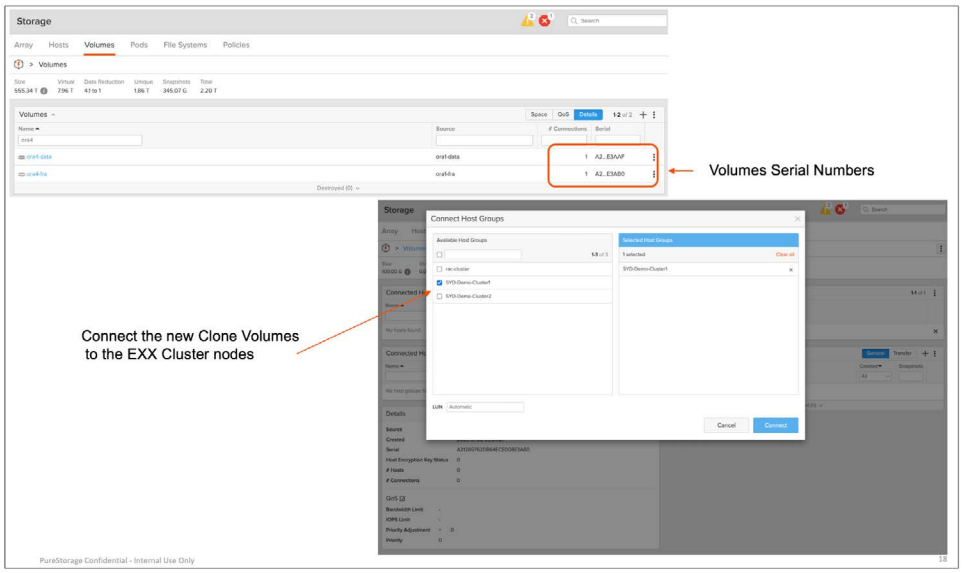

FIGURE 35 Connect the new clone volumes to the ESXi cluster host group.

6. Log in to the VMware vSphere environment, and then select the new virtual machines to use as the target for cloned volumes; in the example shown in Figure 36, this is the **ora4** virtual machine.

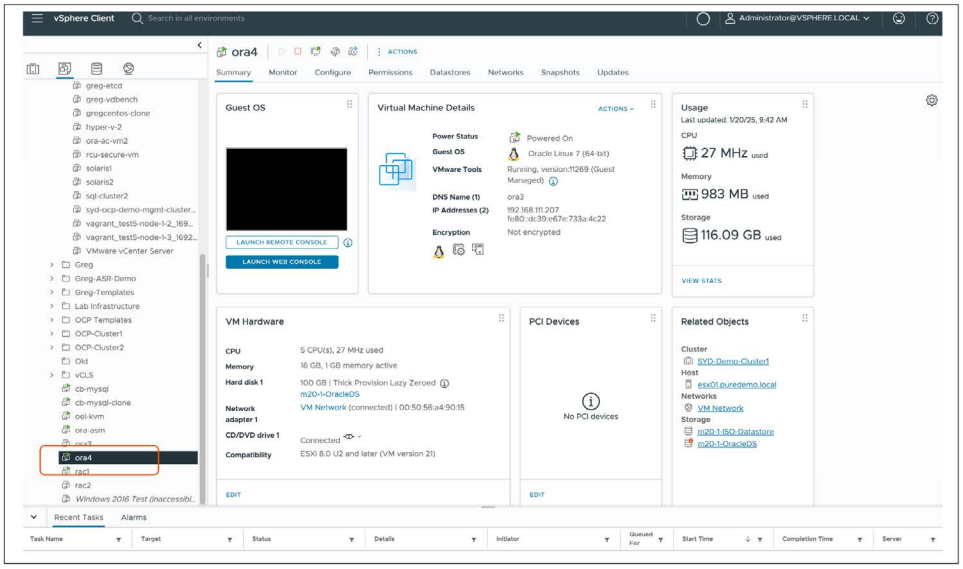

FIGURE 36 Select the new virtual machines to use as the target for cloned volumes.

7. Create a Secure Shell connection to the Oracle virtual machine and confirm there are no clone volumes attached. In this example, we can see that only the boot volume is shown:

```
su -
# lslbk
```

```
[root@ora4 ~]#
[root@ora4 ~]# lsblk
NAME                  MAJ:MIN RM      SIZE    RO TYPE MOUNTPOINT
sda                   8:0     0       100G    0       disk
├─sda1                8:1     0       1G      0       part /boot
└─sda2                8:2     0       99G     0       part
  ├─ol-root           249:0   0       50G     0       lvm /
  ├─ol-swap           249:1   0       7.9G    0       lvm [SWAP]
  └─ol-home           249:2   0       41.1G   0       lvm /home
sr0                   11:0    1       4.1G    0       rom/run/media/oracle/OL-7.5 Server.x86_64
```

8. Select the **ora4** virtual machine in the vSphere console.
9. Right-click the **Actions** tab, and then select **Edit settings**.
10. Select **Add new device**, and then select **RDM**.
11. Figure 37 highlights the new clone volumes needed, with the serial numbers confirmed previously (**e3ab0** and **e3aaf**).

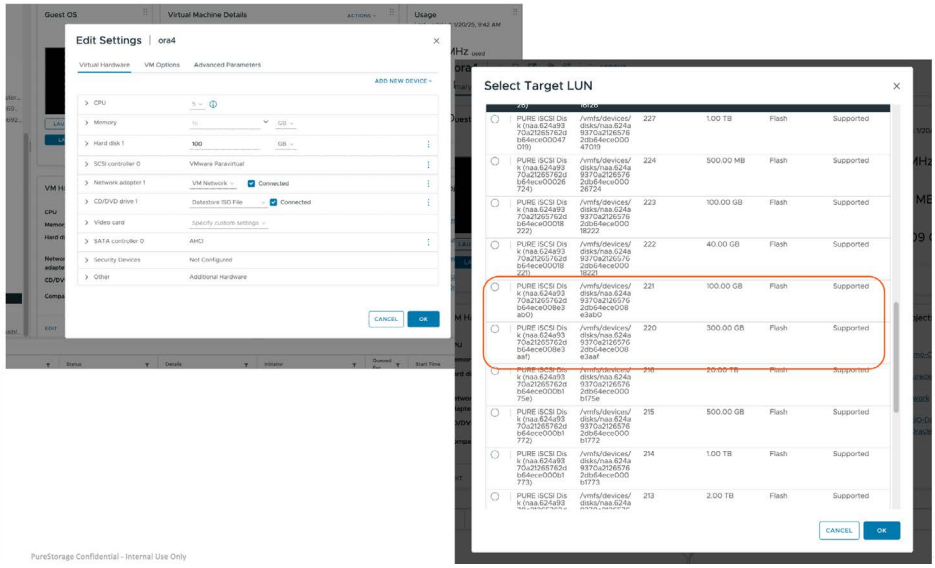

FIGURE 37 Showing the new clone volumes needed.

12. We can now see the new hard disks allocated to the virtual machine (the 300GB and 100GB disks). We can also see the output of the **lsblk** command, which shows us the new volumes attached with the correct serial numbers. Because they are clones, we can also see the Logical Volume Manager names.

```
# lsblk -o name,serial
```

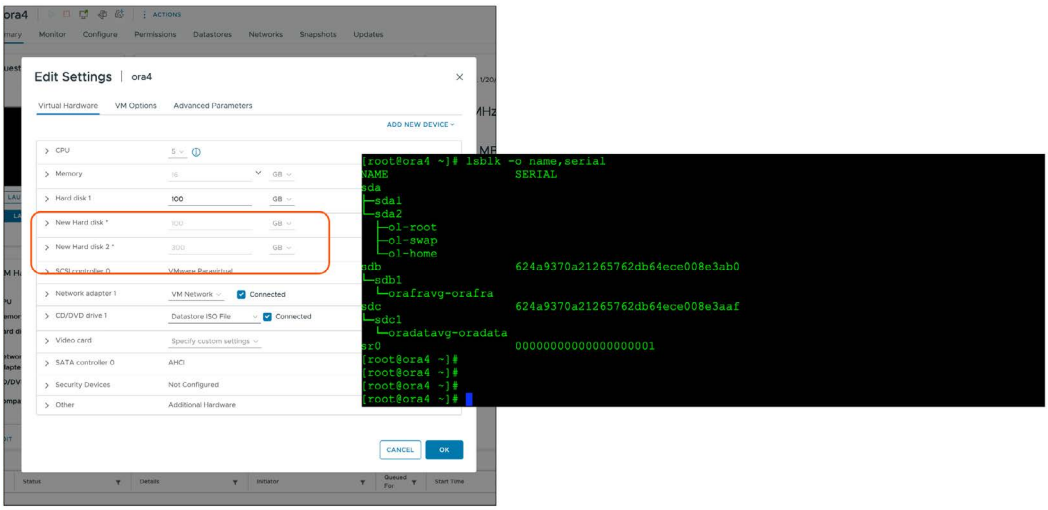

FIGURE 38 The raw volume details.

```
[root@ora4 ~]#
[root@ora4 ~]# lsblk -o name,serial
NAME                    SERIAL
sda
├─sda1
└─sda2
  ├─ol-root
  ├─ol-swap
  └─ol-home
sdb                     624a9370a21265762db64ece008e3ab0
└─sdb1
  └─orafravg-orafra
sdc                     624a9370a21265762db64ece008e3aaf
└─sdc1
  └─oradatavg-oradata
sr0                     00000000000000000001
[root@ora4 ~]#
```

13. On the Oracle host, mount the new clone volumes.

14. Confirm the volumes are mounted and the file systems are consistent with the source host.

15. Switch to the Oracle user, source the **.bash_profile**, and then run the **sqlplus** command to start Oracle:

```
sqlplus / as sysdba
```

```
# mount /dev/mapper/oradatavg-oradata /u01
# mount /dev/mapper/orafravg-orafra /u02
[root@ora4 ~]# df -h
Filesystem                      Size    Used    Avail   Use%    Mounted on
devtmpfs                        7.7G    0       7.7G    0%      /dev
tmpfs                           7.8G    0       7.8G    0%      /dev/shm
tmpfs                           7.8G    18M     7.7G    1%      /run
tmpfs                           7.8G    0       7.8G    0%      /sys/fs/cgroup
/dev/mapper/ol-root             50G     6.9G    44G     14%     /
/dev/mapper/ol-home             42G     43M     42G     1%      /home
/dev/sda1                       1014M   329M    686M    33%     /boot
tmpfs                           1.6G    36K     1.6G    1%      /run/user/54321
/dev/sr0                        4.2G    4.2G    0       100%    /run/media/oracle/OL-7.5 Server.x86_64
tmpfs                           1.6G    0       1.6G    0%      /run/user/1000
/dev/mapper/oradatavg-oradata   296G    24G     257G    9%      /u01
/dev/mapper/orafravg-orafra     99G     671M    93G     1%      /u02
[root@ora4 ~]#
```

16. Confirm Oracle Database starts up, and then select the mode of the database to confirm it is in read/write mode.

17. Select the name of the database; in this example, **orcl**, which is the name of the source database.

Note: Depending on your environment and what you trying to do, the name can be left for the purpose of testing or you can change the name.

18. To change the database name, use the **NID** command provided by Oracle, and then complete the following steps:

- Start in mount mode.
- Run the **NID** command with the new target database name.

```
[oracle@ora4 ~]$
[oracle@ora4 ~]$ sqlplus / as sysdba
SQL> startup mount;
Total System Global Area   4966053832 bytes
Fixed Size                    8906696 bytes
Variable Size              1157627904 bytes
Database Buffers           3791650816 bytes
Redo Buffers                  7868416 bytes
Database Mounted.
SQL>
oracle@ora4 ~] nid TARGET=sys/oracle123 DBNAME=devdb
DBNEWID: Release 19.0.0.0.0 - Production on Thu Jan 30 09:20:43 2025
Copyright (c) 1982, 2019, Oracle and/or its affiliates.  All rights reserved.
Connected to server version 19.3.0
Control Files in database:
    /u01/app/oracle/oradata/ORCL/controlfile/o1_mf_gj74s43x_.ctl
    /u02/fast_recovery_area/ORCL/controlfile/o1_mf_gj74s44y_.ctl
Change database ID and database name ORCL to DEVDB? (Y/[N]) => Y
Proceeding with operation
Changing database ID from 1538741700 to 1095587148
Changing database name from ORCL to DEVDB
    Control File /u01/app/oracle/oradata/ORCL/controlfile/o1_mf_gj74s43x_.ctl - modified
    Control File /u02/fast_recovery_area/ORCL/controlfile/o1_mf_gj74s44y_.ctl - modified
    Datafile /u01/app/oracle/oradata/ORCL/datafile/o1_mf_system_gj74nsbv_.db - dbid changed wrote new name
    Datafile /u01/app/oracle/oradata/ORCL/datafile/o1_mf_tsdemo_gqovfmhx_.db - dbid changed wrote new name
    Datafile /u01/app/oracle/oradata/ORCL/datafile/o1_mf_sysaux_gj74pjjq_.db - dbid changed wrote new name
    Datafile /u01/app/oracle/oradata/ORCL/datafile/o1_mf_undotbs1_gj74qmng_.db - dbid changed wrote new name
    Datafile /u01/app/oracle/oradata/ORCL/datafile/o1_mf_iops_gqlwmt7r_.db - dbid changed wrote new name
    Datafile /u01/app/oracle/oradata/ORCL/datafile/o1_mf_users_gj74qnrc_.db - dbid changed wrote new name
    Datafile /u01/app/oracle/oradata/ORCL/datafile/o1_mf_temp_gq74sd4y_.db - dbid changed wrote new name
    Control File /u01/app/oracle/oradata/ORCL/controlfile/o1_mf_gj74s43x_.ctl - dbid changed, wrote new name
    Control File /u02/fast_recovery_area/ORCL/controlfile/o1_mf_gj74s44y_.ctl - dbid changed, wrote new name
    Instance shut down
Database name changed to DEVDB
Modify parameter file and generate a new password file before restarting
Database ID for database DEVDB changed to 1095587148
All previous backups and archived redo logs for this database are unusable
Database is not aware of previous backups and archive logs in Recovery Area
Database has been shutdown, open database with RESETLOGS option
Successfully changed database name and ID
DBNEWID - Completed successfully
```

```
SQL> alter database open resetlogs;

Database altered.

SQL>

SQL> exit.

[oracle@ora4 ~]$ sqlplus / as sysdba

SQL*Plus: Release 19.0.0.0.0 - Production on Thu Jan 30 09:13:11 2025

Version 19.3.0.0.0

Copyright (c) 1982, 2019, Oracle.  All rights reserved.

Connected to:

Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production

Version 19.3.0.0.0

SQL>

SQL> select open_mode from v$database;

OPEN_MODE

--------------------

READ WRITE

SQL>

SQL>

SQL> select name from v$database;

NAME

---------

DEVDB

SQL>
```

Conclusion

Using FlashArray snapshots for Oracle Database cloning offers several key advantages. First, it minimizes downtime by allowing for crash-consistent copies of Oracle Database volumes without interrupting database operations, ensuring minimal downtime and maintaining continuous access to critical data. Second, it improves efficiency by enabling database administrators to rapidly create point-in-time snapshots, which accelerates the provisioning of new database environments for testing, development, and backup purposes. This streamlined process significantly reduces the time required to create copies of Oracle databases. Additionally, FlashArray snapshots simplify data management, making it easier to create and restore snapshots. This ease of management ensures consistency across multiple database instances. Finally, FlashArray snapshots offer enhanced flexibility, enabling the cloning of entire Oracle databases or specific data volumes.

Additional Resources

- Learn more about Pure Storage solutions for Oracle Database.
- Take Oracle Database primary storage for a test drive in the FlashArray virtual lab.

1 See www.purestorage.com/content/dam/pdf/en/misc/esg/2023-esg-pure-report.pdf.

PS2734-01-en 03/25