ORACLE

Exadata Exascale Best Practices for Database Snapshots and Clones

Database snapshot and clone best practices for Exadata Exascale–on-premises, Exadata Database Service on Exascale Infrastructure, and Exadata Database Service on Cloud@Customer

June, 2025, Version 1.0

ORACLE

Table of contents

List of figures

List of tables

Introduction

Many customers require the ability to quickly and easily create space-efficient snapshots and clones of their databases for many purposes, including increasing developer agility and productivity, robust testing with production size and quality data, application, schema, and configuration changes, and better production support. Exadata has long had the capability to help customers with these requirements. Now, with Exadata Exascale (Exascale), new capabilities and innovations are available to simplify management and increase the storage efficiency of running multiple databases. Exascale enables clones for all read-write and read-only Oracle Database 23ai databases—PDBs and CDBs alike—providing greater source database flexibility, reducing the administration of operating large volumes of database clones, reducing storage costs by efficiently using storage resources and accelerating the time to delivery new databases for any use case.

Exascale is Oracle Exadata's new intelligent data architecture. Introduced in Exadata System Software 24ai, Exascale is available for on-premises Exadata deployments, Exadata Cloud@Customer, and Exadata Database Service on Exascale Infrastructure (ExaDB-XS) in Oracle Cloud Infrastructure (OCI). Exascale will soon be available on Exadata Database Service on Dedicated Infrastructure (ExaDB-D) on OCI and Oracle's multicloud partners.

Exascale transforms compute and storage resources management on Exadata platforms by decoupling and streamlining storage management. This innovation enables new capabilities, including a new approach to database snapshots and clones. You can discover more background on Exadata Exascale here:

Blog: Exadata Exascale - World's Only Intelligent Data Architecture for Cloud

Documentation: Exadata Exascale Users Guide

Common Customer Requirements for Database Snapshots and Clones

Before moving to the mechanics of database snapshots and clones with Oracle Database and Exascale, let's discuss the most common customer use cases (or requirements), which are grouped as follows:

- **Developer agility:** Developers often request separate and isolated databases for their development work before merging code and database changes into a common integration database environment. For example, a developer may request a clone of a banking application database to test new business logic and schema changes before promoting the changes through their software development lifecycle process.
- **Production-like shared development environments:** Many organizations create scaled-down non-production database environments that simulate production using the same data model and code. However, some customers view storage capacity as expensive, and the volume of data compared to production is reduced (typically using custom scripts or with seed/generated data), resulting in a smaller database. While this achieves the purpose of a production-like data model and code, production performance can be difficult to assess, even on similar hardware.
- **Test databases for software update testing:** Before applying software updates, release updates, or upgrading databases, many organizations test such changes on a copy of the production database or test systems.
- **Data Sharing:** Making data available to other users for secure and high-performance platforms, providing new or innovative insights.
- **Quasi-production:** Secure offload of point-in-time reporting, what-if scenario-modeling, data audit, and production support operations to a production-like platform isolated from the source production workload.
- **Point-in-time "fallback":** Many organizations require the ability to view data as it was at a specific time as a fast fallback or database restore option. This need may be due to logical data corruption or external events,

including ransomware attacks. Some regulatory requirements mandate the ability to restore point-in-time database copies.

Introduction to Exascale

Exascale is a new software architecture in Exadata that transforms how storage and compute resources are managed. By combining Exadata's best capabilities with cloud concepts such as resource pooling and resource elasticity, Exascale significantly increases Exadata's agility. Unlike Automatic Storage Management (ASM), which manages slices of storage on a database/VM cluster-by-cluster basis, Exascale manages storage holistically within the storage servers for all databases and clusters, reducing the need for storage resource siloing and database server-driven storage management operations.

Three key concepts of Exascale must be understood in the context of database snapshots and clones.

- **Storage Pools**: Exascale collects storage in a storage pool that multiple databases, Oracle Grid Infrastructure (GI) clusters, or VMs can use simultaneously.
- **Vaults**: Secure logical collections of files (database, redo, backup, etc.) belonging to a database, GI, or VM cluster. Vaults are analogous to the function DATA, RECO, and SPARSE disk groups in ASM as a single entity – i.e., a single vault houses all file types that Exascale manages. Vaults may also incorporate storage of different persistent media types – HDD (High Capacity storage servers) and Capacity-Optimized Flash (Extreme Flash storage servers).
- **Files**: Everything in Exascale is represented as a file. Moreover, content type and redundancy (2-way mirroring – normal redundancy; and 3-way mirroring – high redundancy), which are implicitly set when a file is written to an ASM disk group are now attributes of each file. This enables Exascale to store files of different content types and redundancy efficiently without creating storage silos.

Introduction to Database Snapshots and Clones on Exascale

Exascale brings new database, file, and volume capabilities to Exadata to help organizations implement their database management requirements. Specifically, it reimagines read-only snapshots and read-write clones of any read-only or read-write database with no upstream dependencies, such as a read-only copy. In addition, Oracle Database includes capabilities often used with Exascale Snapshots and Clones to help achieve the required customer outcome. Exascale uses redirect-on-write (ROW) techniques to create and maintain clones quickly and extremely space-efficiently.

Redirect-on-write is a storage snapshot technique that writes new data to a new location on disk, instead of overwriting the original data.

How it works:

- The original extent contains the point-in-time data for a snapshot.
- New data is written to a new location (extent) on the disk without overwriting the original extent. Unchanged data is read from the snapshot.
- Changed data is written to a new extent in Exascale storage, with the original version remaining in the snapshot.

Exascale Snapshot: a read-only point-in-time copy of a file. The source file for a snapshot can be a regular file, a clone, or another snapshot.

Exascale Clone: a point-in-time copy of a file that is readable and writable. The source file for a clone can be a regular file, a snapshot, or another clone. Clones are created as thin clones (preserving the space-efficient property of the file

clone) or full clones (a byte-for-byte copy of the source file). Clones implicitly create snapshots of files or can use an existing snapshot as the basis of the clone.

Storage Reclamation

With Exascale, storage is automatically managed by the Exadata Storage Servers. When a file, snapshot, or clone no longer references an extent, Exascale automatically reclaims that extent for future use.

Core Oracle Database and Exadata Concepts

The following technologies should be familiar:

- **Container Database (CDB):** A container database is an Oracle database that contains one or more pluggable databases. A CDB centralizes database resource management enabling greater utilization and administration efficiency.
- **Pluggable Database (PDB):** A pluggable database is a portable collection of schemas, schema objects, and non-schema objects that appear and behave as a separate database. PDBs are contained within a CDB, which manages shared resources.
- **PDB Clones**: An Oracle Database 23ai Container Database (CDB) can have its Pluggable Databases (PDBs) cloned (thin or full) within the same CDB. On Exascale, PDB Clones can use the new snapshot and clone capabilities.
- **Remote PDB Clones**: A PDB can be cloned (thin or full) into a different CDB. These CDBs may be in the same GI cluster, a different GI cluster on the same physical infrastructure, or CDBs on a different infrastructure. For thin clones, the source and target CDB must be able to access the same Exascale vault.
- **Refreshable PDB Clones**: PDB clones are created at a point in time in the target CDB to be refreshed from the source PDB at regular intervals.
- **PDB Snapshot Carousel**: A PDB snapshot carousel is a library of PDB snapshots. These snapshots can be created manually or automatically at regular intervals. Users can subsequently use PDB snapshots to create read-write PDB Clones.
- **Recovery Manager (RMAN)**: RMAN is the backup and restoration utility for Oracle databases. It can restore a database backup to a different location, thereby cloning a database. Typically, RMAN is used to duplicate a production database as a non-production database.
- **Data Pump**: Data Pump is the import/export utility for Oracle databases. Data Pump can create an export (logical backup) of a database that can be imported into another database on the same or a different infrastructure or GI cluster.
- **Data Guard and Active Data Guard**: Data Guard replicates changed data from a primary database to one or more standby databases. Typically, a standby database is configured as a data protection and failover environment if the primary database becomes unavailable. Standby databases can also be open read-only (with Active Data Guard) to enable query offloading from the primary database.

Maximum Availability Architecture Recommendations

Exascale enables the creation of snapshots and clones from any file stored within it using redirect-on-write technology. For best results, the Exadata Maximum Availability Architecture (MAA) team recommends that snapshots and clones should not be created on a production environment, including primary databases or standby databases to be used for disaster recovery directly, for the following reasons:

1. **Safety**: Creating read-write clones of production databases in the same environment i.e., within the same GI cluster , may lead to applications, users, or administrators connecting to the incorrect database.
2. **Isolate resource consumption**: While running multiple databases and workloads in the same GI cluster is possible, mixing production and non-production workloads in the same GI cluster is not recommended. Clones should, therefore, be created in a separate environment to isolate resource consumption between databases with disparate use cases.

Creating point-in-time snapshots for the "fallback" use case on production systems is a valid MAA use case and does not conflict with the recommendation of mixing production with test use cases.

Security

It is recommended that data in databases used for development or testing be masked using Oracle Enterprise Manager, application-specific masking utilities, or other mechanisms before users and applications connect to use the database. Alternatively, a masked cloned database can be the source for subsequent clones to centralize data masking operations.

Common Use Cases for Database Snapshots and Clones

The following table summarizes the more common use cases of database snapshots and clones.

Use case	Source	Type of snapshot or clone
Create a read-only point-in-time reference to a PDB	• Any PDB • Primary database only	PDB Snapshot
Create space-efficient copies of an individual PDB	• Any PDB or a snapshot on a PDB Snapshot Carousel • Primary or standby database	PDB Thin Clone
Create full copies of an individual PDB	• Any PDB or a snapshot on a PDB Snapshot Carousel • Primary or standby database	PDB Full Clone
Create copies (space-efficient or full) of a previously cloned PDB (thin or full)	• Any existing PDB clone or a snapshot on a PDB Snapshot Carousel • Primary or standby database	PDB Thin Clone or PDB Full Clone
Create automatic read-only point-in-time PDB snapshots for future use	• Any PDB can be configured with a PDB Snapshot Carousel • Primary database only	PDB Snapshot Carousel
Create space-efficient copies of a CDB, including all its PDBs	• Any CDB • Primary or standby database	CDB Clone using gDBClone. The CDB must be mounted and configured in ARCHIVELOG mode

Table 1 - Summary of database snapshot and clones on Exascale

All snapshots and/or clones created in the same operation are point-in-time consistent, and all files in a snapshot and/or clone operation must be in the same Exascale Vault.

Creating Snapshots and Clones of Pluggable Databases

On Exascale, snapshots and clone operations for pluggable databases are integrated with the existing `CREATE` and `ALTER PUGGABLE DATABASE` commands. This simplifies the creation of PDB snapshots and clones by encapsulating the related storage operations in a single database SQL statement.

PDB Snapshots

A snapshot of a PDB is a space-efficient, read-only point-in-time reference to a PDB that can be used to create a PDB clone. Snapshots are quickly created using the `ALTER PLUGGABLE DATABASE SNAPSHOT` command. A name may be given to the snapshot; otherwise, a name will be generated automatically.

By default, snapshots are inconsistent. This enables the snapshot creation to be completed quickly without impacting workloads running in the PDB. When a clone (thin or full) is created from a PDB snapshot, it is automatically recovered using redo. Once the PDB Clone is recovered it can be opened read-write.

For example, to create a snapshot of a PDB named PDB1 from a snapshot BEFORE_APP_UPGRADE use the following from within the CDB$ROOT:

1. Manually create a PDB snapshot.

```
SQL> ALTER PLUGGABLE DATABASE PDB1 SNAPSHOT BEFORE_APP_UPGRADE;

Pluggable database BEFORE_APP_UPGRADE altered.
```

2. To display information related to the snapshot, query the DBA_PDB_SNAPSHOTS view.

```
SQL> select con_name, snapshot_name, snapshot_scn, scn_to_timestamp(snapshot_scn) from
dba_pdb_snapshots;

CON_NAME       SNAPSHOT_NAME            SNAPSHOT_SCN SCN_TO_TIMESTAMP(SNAPSHOT_SCN)
_____________ _____________________ _______________ ___________________________________

PDB1           BEFORE_APP_UPGRADE          285455107 31-MAR-25 11.00.15.000000000 PM
```

IMPORTANT—As the snapshot requires redo to be applied during the clone operation, you must ensure that the required redo is retained or restored from backup on your system to recover the cloned files to a consistent state.

PDB Thin Clones

To create a space-efficient thin clone of a PDB, use the `CREATE PLUGGABLE DATABASE <new_pdb_name> FROM <source_pdb_name> SNAPSHOT COPY` command. The SNAPSHOT COPY keywords tell the database to use the space-efficient Exascale capabilities to create the clone.

To create a PDB Thin Clone of an existing PDB, use the following steps from within the CBD$ROOT:

1. Create a thin clone PDB_THIN_CLONE1 from PDB1

```
SQL> CREATE PLUGGABLE DATABASE PDB_THIN_CLONE1 FROM PDB1 SNAPSHOT COPY;

Pluggable database PDB_THIN_CLONE1 created.
```

2. Open the PDB thin clone PDB_THIN_CLONE1 on all instances in the cluster

```
SQL> ALTER PLUGGABLE DATABASE PDB_THIN_CLONE1 OPEN INSTANCES=ALL;

Pluggable database PDB_THIN_CLONE1 altered.
```

Following on from the example in the previous section, to create a PDB thin clone from an existing PDB snapshot, use the following steps from within the CBD$ROOT:

1. Create a new PDB thin clone from an existing snapshot

```
SQL> CREATE PLUGGABLE DATABASE PDB_CLONE_BEFORE_APP_UPGRADE FROM PDB1 USING SNAPSHOT
BEFORE_APP_UPGRADE SNAPSHOT COPY;

Pluggable database PDB_CLONE_BEFORE_APP_UPGRADE created.
```

2. Open the PDB thin clone PDB_THIN_CLONE1 on all instances in the cluster

```
SQL> ALTER PLUGGABLE DATABASE NEW_PDB OPEN INSTANCES=ALL;

Pluggable database PDB_CLONE_BEFORE_APP_UPGRADE altered.
```

PDB Full Clone

To create a full clone of a PDB, use the `CREATE PLUGGABLE DATABASE <new_pdb_name> FROM <source_pdb_name>` command. The absence of the SNAPSHOT COPY keywords tells the database to create a complete block-for-block copy of the PDB.

For example, to create a full clone of PDB1, use the following steps from the CDB$ROOT;

1. Create a full clone PDB_FULL_CLONE1 from PDB1

```
SQL> CREATE PLUGGABLE DATABASE PDB_FULL_CLONE1 FROM PDB1;

Pluggable database PDB_FULL_CLONE1 created.
```

2. Open the PDB full clone PDB_FULL_CLONE1 on all instances in the cluster

```
SQL> ALTER PLUGGABLE DATABASE PDB_FULL_CLONE1 OPEN INSTANCES=ALL;

Pluggable database PDB_FULL_CLONE1 altered.
```

The complete syntax for PDB clones is available in the Oracle Database documentation, under: '<u>Cloning a PDB</u>'.

It is possible to create a PDB full clone from an existing snapshot or PDB Thin Clone. In both cases, the SNAPSHOT COPY keywords should be omitted for creating full clones.

PDB Snapshot Carousel

The PDB Snapshot Carousel automates creating read-only snapshots of a PDB. It is configured by setting the number of snapshots allowed on the PDB carousel and the interval at which snapshots are created. Manual snapshots of a PDB may also be created on the carousel using the ALTER PLUGGABLE DATABASE SNAPSHOT command.

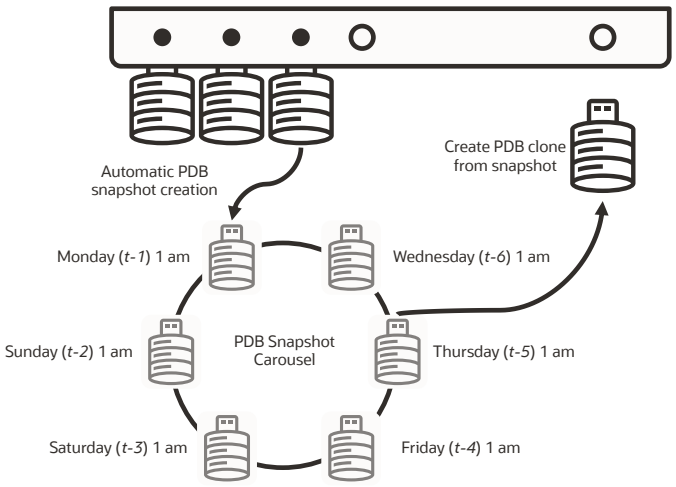

Figure 1 - PDB Snapshot Carousel

To configure the PDB Carousel for a PDB, the following can be used:

1. Alter your session to enter the PDB

```
SQL> alter session set container=PDB1;

Session altered.
```

2. Enable automatic snapshots of the PDB

```
SQL> alter pluggable database snapshot mode every 24 hours;

Pluggable database MODE, EVERY altered.
```

PDB Carousel configuration and commands are available in the Oracle Database documentation here: Administering PDB Snapshots

Creating Space-Efficient Clones of a Container Database

gDBClone is a simple-to-use but powerful utility that coordinates and executes the end-to-end space-efficient thin clone (known as a snapshot in gDBClone) or full clone of a container database and all PDBs it contains. This includes the file clone operations, invocation of RMAN to ensure the consistency of the clone files, creation of instance-related files (password files, files, etc.), conversion of a single instance database to RAC or vice versa as required, and configuration (registration with GI).

gDBClone usage is documented in the following My Oracle Support article gDBClone Powerful Database Clone/Snapshot Management Tool (Doc ID 2099214.1).

Consider you have an RAC-enabled CDB called CDB1DB1 with multiple PDBs that you would like to create a thin clone for a development project named DEVCDB1. The simplest way of achieving this would be to use gDBClone, providing a source and target (the name given to the clone CDB) as follows:

```
gDBClone.bin snap -sdbname <source DB name> -tdbname <Target Database Name>
```

```
$ /opt/gDBClone/gDBClone.bin snap -sdbname CDB1DB1 -tdbname DEVCDB1
```

This command will not only thin clone the CDB1DB1 database data files for use by the DEVCDB1 database, it will also make the data files consistent, create the DEVCDB1 database instances, manage password and TDE wallet files, copy and modify the spfile, register the database instances with CRS, and execute other related actions automatically.

gDBClone can also convert databases from or to RAC and modify the SGA size. It can also create full copies of the CDB using the 'gDBClone clone' command, enabling a database to be cloned between different Exascale vaults.

Backup and Recovery

RMAN seamlessly supports the backup and recovery of PDB and CDB clones. It treats PDB and CDB clones as regular databases, ensuring they are backed up and recovered like any other Oracle database.

When a database is restored and recovered on Exascale, its files are written as full copies, entirely independent of any source snapshots or clones they were created from. This ensures a reliable and consistent restoration, treating all databases with the same integrity.

RMAN does not backup PDB Snapshots.

Using the Standby Database for Snapshots and Clones

The following should be understood when working with Exascale snapshots and clones in a standby environment.

- PDB Snapshots created on the primary database (manually or automatically) are not replicated to the standby database(s).
- PDB Thin Clones on the primary database—using the SNAPSHOT COPY command—are created as full clones on the standby database(s).
- Oracle Database 23ai uses PDB Recovery Isolation to automatically copy and recover the data files from the primary database. A summary of the steps taken by the standby is as follows:
    - When Active Data Guard media recovery discovers that a clone[1] was created of a PDB in the primary CDB, it performs the following steps:
        - Temporarily marks the PDB as disabled and continues with CDB media recovery.
        - Automatically recovers the PDB in a separate background session.
        - Enables the PDB after the PDB recovery is completed.
        - Merges the isolated PDB recovery session with the Active Data Guard media recovery session.

  PDB Recovery Isolation ensures that media recovery of the standby CDB continues unimpeded by one or more cloned PDBs that are inconsistent with the rest of the CDB. This enables the standby CDB and other PDBs to be used for read-only purposes and continues with redo recovery while PDB clones are synchronized on standby.
- Environments not using Active Data Guard must copy the cloned data files from the primary to the standby or use the STANDBYS=NONE clause in the CREATE PLUGGABLE DATABASE SNAPSHOT COPY command. [2]
- When cloning a PDB from a standby CDB to another read-write CDB , redo apply on the standby must be paused temporarily to ensure the PDB is in a consistent state. This can be done easily by editing the standby

[1] PDB Recovery Isolation automates the recovery of newly created "hot" clones–clones of an online PDB–in the standby database. For cold PDB clones, or PDBs that have been plugged in from a PDB archive or similar, the administrator is required to copy the required datafiles to the standby database.
[2] For full details see Oracle Support Document 1916648.1 (Making Use Deferred PDB Recovery and the STANDBYS=NONE Feature with Oracle Multitenant): https://support.oracle.com/epmos/faces/DocumentDisplay?id=1916648.1

database state with the Data Guard manager command-line utility (dgmgrl) to set the state to 'apply-off,' creating the PDB clone, and then resetting the standby database state to 'apply-on.' For example:

```
standby_host1$ dgmgrl edit database <standby> set state='apply-off’;

standby_host1$ sqlplus /@target_cdb as sysdba

SQL> CREATE PLUGGABLE DATABASE PDB1TEST FROM PDB1@STANDBY_DB_LINK SNAPSHOT COPY;

Pluggable database created.

SQL> exit

standby_host1$ dgmgrl edit database <standby> set state='apply-on';
```

Common Snapshot/Clone Scenarios

Using a secondary standby database as a Snapshot/Clone source (recommended)

The MAA recommended approach to creating clones (and snapshots) from a production database is to create a secondary Data Guard standby database on non-production infrastructure to act as a source. This enables the production primary and disaster recovery databases to continue being used for their intended purposes without adding typically non-production use cases to these environments. In the event of a disaster, the secondary standby database and any snapshots and clones have no impact on the production system's failover, availability, or performance.

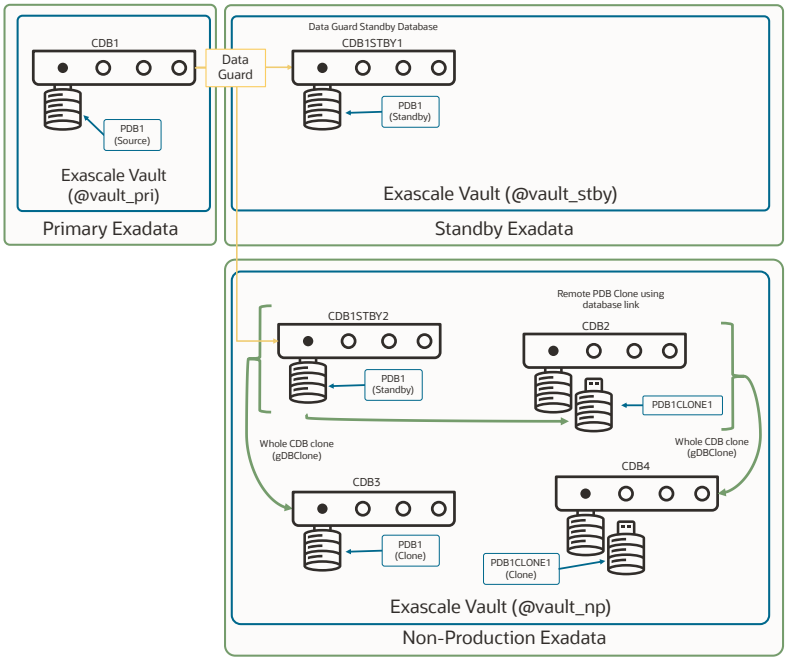

Figure 2 - Using a secondary standby database for snapshots and clones

Standalone CDB as a source of snapshots/clones (recommended)

The most common source of database snapshots and clones on Exascale is a “standalone” database. The “standalone” source database can be either a PDB or a CDB (including all its PDBs). Typically, PDB snapshots and clones are used when one or more PDBs need to be created from the same source PDB, such as using the same masked source for multiple developers. CDB clones would typically be used when a collection of PDBs in the same CDB need to be cloned together.

Source databases are created using any method that can be used to copy or create a 23ai Oracle Database—e.g., an RMAN backup of a production database that is restored (and renamed) to a non-production infrastructure, a refreshable PDB clone of a production PDB in a non-production CDB and infrastructure, using DBCA to create a new database with application seed data injected, or using Data Pump Export/Import.

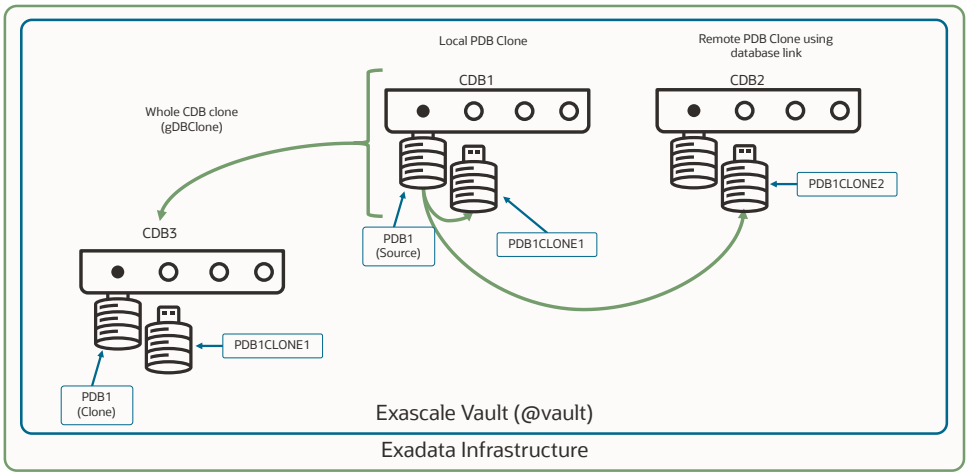

Figure 3 - Using a standalone database for snapshots and clones

Using primary and standby databases as a Snapshot/Clone source

Many customers with primary and standby databases configured in production require clones and snapshots created from either the primary or the DR standby database.

Again, while it is possible to create clones of standby CDBs or individual PDBs (using Remote PDB Clone), it is recommended to use a secondary standby database to minimize resource sharing and ensure greater isolation from what are typically production and disaster recovery databases. See Maximum Availability Architecture Recommendations

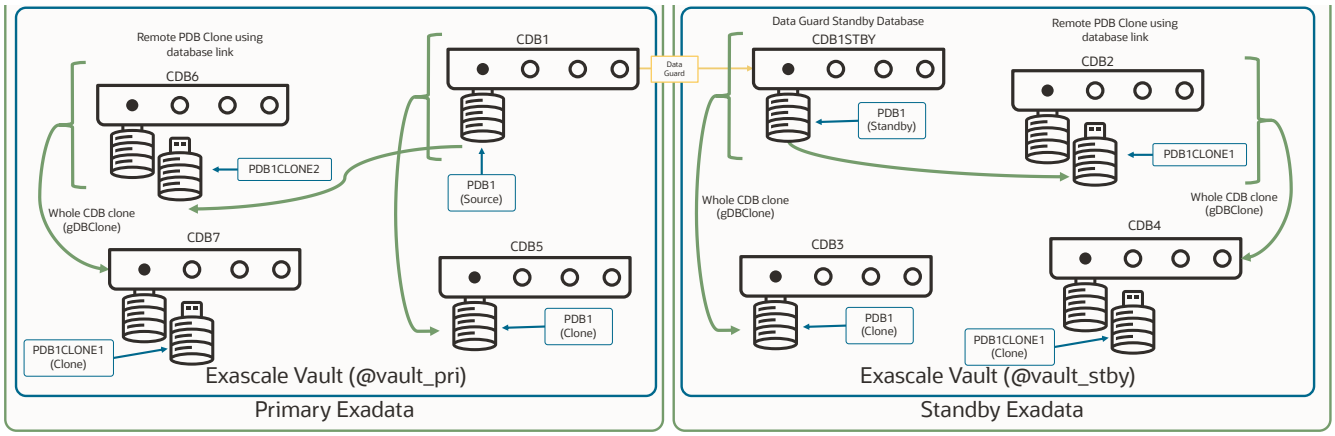

Figure 4 - Using a primary and standby database for snapshots and clones

Summary

Exadata Exascale's reimagined database snapshot and clone capabilities are extremely powerful, flexible, and simple. They enable organizations to increase the overall value of their Exadata investment by utilizing Exadata for more database workloads. Coupled with the latest generation of Exadata, X11M, more databases can be consolidated on Exadata than ever before by using space-efficient database snapshots and clones for all application development lifecycle stages.

ORACLE

Connect with us

Call +**1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

blogs.oracle.com facebook.com/oracle twitter.com/oracle