Google Cloud NetApp Volumes mit Oracle HA implementieren

NetApp database solutions

NetApp
June 25, 2026

Inhalt

Google Cloud NetApp Volumes mit Oracle HA implementieren

Google Compute Engine Instanzen für Google Cloud NetApp Volumes bereitstellen

Google Compute Engine-VMs bereitstellen, um Oracle Database 26ai auf Google Cloud NetApp Volumes iSCSI-Speicher zu hosten. Dieses Verfahren umfasst das Erstellen der primären und Standby-Datenbankhosts sowie der Fast-Start Failover Observer VM, das Konfigurieren von VPC-Firewallregeln für Oracle Net, das Einrichten der Hostnamensauflösung, das Vorbereiten des Betriebssystems und das Erfassen der iSCSI-Initiatornamen für die GCNV-Speicherbereitstellung.

Schritt 1: VMs erstellen

Drei Google Compute Engine VMs in verschiedenen Zonen derselben Region für die Isolierung von zonalen Ausfällen erstellen. Die Cloud Console, `gcloud`, Terraform oder den üblichen Bereitstellungs-Workflow verwenden.

1. Die drei VMs werden mit den in der folgenden Tabelle aufgeführten Spezifikationen erstellt.

   Eine kohlenstoffärmere Region ist hinsichtlich Gesamtbetriebskosten und Nachhaltigkeit vorzuziehen, sofern sie den Anforderungen an Latenz und Compliance entspricht (zum Beispiel `us-west1` vs `us-central1`):

VM	Zone	Maschinentyp	Bootdisk	Netzwerk	Zweck
`oracdb1`	`us-west1-a`	`n4-highmem-8`(Beispiel) oder `c4-standard-*`	OL 10, 50 GB Hyperdisk Balanced (nur Betriebssystem)	`oracle-vpc` / `oracle-subnet`, gVNIC	Primäre Datenbank
`oracdb2`	`us-west1-b`	Gleich wie primär	OL 10, 50 GB Hyperdisk Balanced (nur Betriebssystem)	Dasselbe	Standby-Datenbank
`oradg-obs`	`us-west1-c`	`e2-medium`	OL 10, 20 GB Hyperdisk Balanced	Dasselbe	FSFO Observer (nur Instant Client)

   Der Premium network tier empfiehlt sich, wenn Latenz oder ausgehender Datenverkehr (>~200 GiB/Monat) relevant sind; der Standard network tier bietet niedrigere Gesamtbetriebskosten in Entwicklung/Test.

2. Shielded VM-Funktionen aktivieren und die Boot-Disk-Konfiguration überprüfen:

   Secure Boot, **vTPM** und **Integrity Monitoring** sollten auf allen drei VMs aktiviert sein.

Die Boot-Disk enthält nur das Betriebssystem. `/u01` Grid/DB Homes, Staging und alle ASM-Daten verwenden GCNV iSCSI Volumes (siehe GCNV iSCSI-Volumes bereitstellen)

Keine separate GCE-Datenfestplatte für `/u01` anschließen.

Schritt 2: VPC-Firewall für TCP 1521 konfigurieren

VPC-Firewallregeln erstellen, die TCP/1521 zwischen allen drei VMs für Oracle Net redo transport und Observer-Konnektivität zulassen. Fehlende Regeln unterbrechen die Data Guard-Replikation.

1. Eine VPC-Firewall-Ingress-Regel erstellen, um TCP/1521 zwischen allen drei internen VM-IPs zuzulassen. VPC-Firewall-Regeln oder Firewall-Richtlinien mit derselben Allowlist verwenden:

   Cloud Console: VPC-Netzwerk → Firewall → Regel erstellen `allow-oracle-net-dbhosts` auf `oracle-vpc` — Eingehend, Zulassen, Quellen = drei `/32` IPs, TCP 1521. Ausgehenden Datenverkehr spiegeln, falls erforderlich.

2. Die Konnektivität jeder VM kann überprüft werden, um festzustellen, ob die Firewall-Regeln eingerichtet sind:

```
sudo dnf install -y nmap-ncat

for tgt in <oracdb1-ip> <oracdb2-ip> <oradg-obs-ip>; do
  nc -zv -w 5 "$tgt" 22
  nc -zv -w 5 "$tgt" 1521
done
```

Port	Erwartet	Bedeutung
`22`	Verbunden	SSH-Pfad funktioniert
`1521`	Verbindung abgelehnt	Firewall geöffnet; Grid-Listener startet während Schritt 1: Installieren Sie Oracle Grid Infrastructure (Oracle Restart) auf jedem DB-Host
Entweder	Zeitüberschreitung	Firewall- oder Routing-Probleme beheben

Führe den Befehl von allen drei VMs zu jeder Peer-IP aus.

Schritt 3: Hostnamen, DNS und `/etc/hosts` konfigurieren

Hostname- und DNS-Auflösung auf allen drei VMs so konfigurieren, dass die Vorwärts- und Rückwärts-Namensauflösung für Oracle Net, den Data Guard Broker und den Observer funktioniert.

1. Den Hostnamen festlegen und `/etc/hosts`-Einträge auf allen drei Hosts hinzufügen. Die internen IP-Adressen von GCE einsetzen (sichtbar in der Liste **Compute Engine → VM instances**, Spalte *Internal IP*):

```
# Run on each VM, substituting the local short name (oracdb1, oracdb2,
oradg-obs)
sudo hostnamectl set-hostname <this-host>.example.internal

# Run on every VM (same content)
sudo tee -a /etc/hosts >/dev/null <<EOF

# Oracle DG peers + FSFO Observer
<oracdb1-ip>    oracdb1.example.internal    oracdb1
<oracdb2-ip>    oracdb2.example.internal    oracdb2
<oradg-obs-ip>  oradg-obs.example.internal  oradg-obs
EOF
```

2. Namensauflösung von jedem Host validieren:

```
ping -c 1 oracdb1 && ping -c 1 oracdb2 && ping -c 1 oradg-obs
```

Schritt 4: Das Betriebssystem nur auf den DB-Hosts vorbereiten

Das Betriebssystem auf `oracdb1` und `oracdb2`wird für Oracle Database 26ai vorbereitet, indem das Preinstall-Paket installiert, Benutzer und Gruppen erstellt, iSCSI- und Multipath-Pakete installiert und der iSCSI-Initiator konfiguriert werden. Die Observer-Einrichtung ist in Schritt 4: Installieren Sie Oracle Instant Client auf dem Observer-Host enthalten.

Voraussetzung: Ausgehendes HTTPS zu `yum.oracle.com` (Cloud NAT oder internem Mirror auf privaten Subnetzen).

1. Das Oracle Database Preinstall-Paket installieren, den `grid` Benutzer und die ASM-Gruppen erstellen sowie den `oracle` Benutzer zu den ASM-Gruppen hinzufügen:

```
# Oracle 26ai preinstall (package name varies by repo)
sudo dnf install -y oracle-ai-database-preinstall-26ai \
  || sudo dnf install -y oracle-database-preinstall-26ai \
  || sudo dnf install -y oracle-database-preinstall-23ai

# grid user + asm groups
sudo groupadd -g 54327 asmadmin; sudo groupadd -g 54328 asmdba; sudo
groupadd -g 54329 asmoper
sudo useradd -u 54322 -g oinstall -G dba,oper,asmadmin,asmdba,asmoper
grid
sudo passwd -l grid; sudo passwd -l oracle
sudo usermod -a -G asmdba,asmadmin oracle
```

2. iSCSI-, Multipath- und JDK-Pakete installieren, anschließend THP und die Zeitsynchronisierung überprüfen:

```
sudo dnf install -y iscsi-initiator-utils device-mapper-multipath
sg3_utils \
  java-21-openjdk-headless libxcrypt-compat

# THP and time
cat /sys/kernel/mm/transparent_hugepage/enabled   # expect [never]
timedatectl
chronyc tracking
```

3. SELinux-, Firewall- und iSCSI-Initiator-Einstellungen konfigurieren und anschließend das System neu starten:

> **Sicherheitsstatus (OL 10):** Die folgenden Befehle setzen SELinux auf den permissiven Modus und deaktivieren `firewalld`. Dies ist lediglich eine minimale Testumgebung. Für eine gehärtete SELinux- und Firewall-Konfiguration konsultieren Sie die Sicherheitsrichtlinien Ihrer Organisation.

```
sudo setenforce 0
sudo sed -i 's/^SELINUX=.*/SELINUX=permissive/' /etc/selinux/config
sudo systemctl disable --now firewalld

sudo cp -n /etc/iscsi/iscsid.conf /etc/iscsi/iscsid.conf.orig
sudo sed -i '/^[#[:space:]]*node\.session\.timeo\.replacement_timeout/d'
/etc/iscsi/iscsid.conf
echo "node.session.timeo.replacement_timeout = 120" | sudo tee -a
/etc/iscsi/iscsid.conf
sudo systemctl enable --now iscsid

sudo reboot
```

Schritt 5: Erfassen des iSCSI-Initiatornamens (IQN)

Den iSCSI-Initiatornamen (IQN) von jedem Datenbankhost nach dem Neustart erfassen. Diese IQNs werden verwendet, um die GCNV Hostgruppen in Schritt 2: Erstellen der Hostgruppen zu erstellen.

1. Den IQN von `oracdb1` erfassen und protokollieren:

```
sudo cat /etc/iscsi/initiatorname.iscsi
# InitiatorName=iqn.1994-05.com.redhat:abc123def456
```

2. Wiederholen Sie dies auf `oracdb2` und protokollieren Sie dessen IQN. Es wird eine Hostgruppe pro Host

verwendet, sodass ein Neustart oder eine IQN-Regenerierung eines einzelnen Hosts die GCNV iSCSI-Volume-Sichtbarkeit eines anderen Hosts nicht beeinträchtigen kann.

Geklonte VMs: Wenn beide Hosts die gleiche IQN verwenden, neu generieren auf `oracdb2` (stop `iscsi`, clear `/var/lib/iscsi/nodes/*`, new `InitiatorName` in `/etc/iscsi/initiatorname.iscsi`, restart `iscsid`).

Was kommt als Nächstes?

Um gemeinsam genutzten Speicher für Oracle-Binärdateien und ASM-Festplattengruppen bereitzustellen, gehen Sie zu Bereitstellung von Google Cloud NetApp Volumes iSCSI-Pools, Hostgruppen und Volumes.

Google Cloud NetApp Volumes iSCSI-Speicher für Oracle Database 26ai bereitstellen

Bereitstellung von Google Cloud NetApp Volumes iSCSI-Blockspeicher für die Hochverfügbarkeit von Oracle Database 26ai auf Google Compute Engine. Dieses Verfahren umfasst die Erstellung von GCNV Flex Unified Speicherpools, die Definition von Hostgruppen, die Erstellung von iSCSI-Volumes für jeden Datenbank-Host, die Konfiguration von Linux iSCSI und Multipath, die Partitionierung von ASM-Backing-Devices und das Einbinden des `/u01` Dateisystems.

Schritt 1: GCNV iSCSI Pools erstellen

Zwei Flex Unified Speicherpools werden erstellt, jeweils einer in jeder Datenbankzone, um iSCSI-Volumes für den primären und den Standby-Host bereitzustellen. Jeder Datenbank-Host verwendet Volumes aus dem Pool seiner lokalen Zone.

1. Zwei Speicherpools werden mithilfe der Cloud Console erstellt. Die Spezifikationen in der folgenden Tabelle werden verwendet und der Erstellungsprozess für jede Zone wiederholt:

Poolname	Zone	Wird verwendet von
`oracle-pool-a`	`us-west1-a`	`oracdb1` (primär)
`oracle-pool-b`	`us-west1-b`	`oracdb2` (Standby)

   NetApp Volumes → **Speicherpools** → **Erstellen** für jeden Pool:

   - **Servicelevel:** Flex (nicht Premium)
   - **Typ:** Einheitlich
   - **Zone:** stimmt mit der Datenbank-VM-Zone überein (`us-west1-a`/ `us-west1-b`)
   - **Wichtiger Hinweis:** verbunden mit `oracle-vpc`
   - **Kapazität:** dimensioniert für die Arbeitslast; verwenden Sie benutzerdefinierten Durchsatz/IOPS, wenn Redo, Backup oder Restore den Standardspielraum überschreiten (bis zu 5120 MiB/s oder 160K IOPS pro Pool, gemäß Produktbeschränkungen)

2. Warten Sie, bis beide Pools den `READY` Status erreicht haben, bevor Sie fortfahren. Passen Sie die Poolgrößen an den Speicherbedarf Ihrer Datenbank an (die Größen in Schritt 3: Erstellen der GCNV

iSCSI-Volumes sind Beispiele):

> **Standardmodus (diese Anleitung):** Flex Unified pools verwenden den Standardmodus (`--mode=default`). Erstellen Sie Pools und iSCSI-Volumes mit Cloud Console oder `gcloud netapp`. Volume-Replikation, Snapshots und Klone verwenden Google Cloud APIs (Schritt 3: GCNV-Standby-Initialisierung).

Schritt 2: Hostgruppen erstellen

Erstellen Sie für jeden Datenbank-Host eine eigene Hostgruppe, sodass jede VM nur ihre eigenen Volumes sieht. Die primären und Standby-Hosts dürfen keine GCNV iSCSI-Volumes gemeinsam nutzen, um unabhängigen Speicher beizubehalten.

1. Die Hostgruppe für `oracdb1` wird mithilfe der Cloud Console erstellt:

   NetApp Volumes → Hostgruppen → Erstellen

   - **Name:** `oracdb1-hg`
   - **Region:** `us-west1`
   - **Typ:** iSCSI-Initiator
   - **Betriebssystemtyp:** Linux
   - **Hosts:** Fügen Sie den IQN aus `oracdb1` (dem Wert von `/etc/iscsi/initiatorname.iscsi`) ein.
   - **Beschreibung:** „Oracle primary host oracdb1“
   - **Erstellen**

2. Wiederholen Sie den Vorgang für `oracdb2` mit dem Namen `oracdb2-hg` und dem IQN von `oracdb2`. Der Observer Host benötigt keine GCNV Ressourcen.

Schritt 3: GCNV iSCSI-Volumes erstellen

Fünf GCNV iSCSI-Volumes sind für jeden Datenbank-Host zu erstellen: eines für `/u01` und vier für ASM-Backing-Devices. Die Volumes jedes Hosts müssen im Storage-Pool der lokalen Zone mit der entsprechenden Hostgruppe erstellt werden.

1. Erstellen Sie die fünf Volumes für `oracdb1` in `oracle-pool-a` mit der Hostgruppe `oracdb1-hg`. Die Spezifikationen befinden sich in der folgenden Tabelle:

GCNV iSCSI-Volume	Größe	Verwenden	Multipath-Alias
`ora_<host>_u01`	100 GiB	`/u01` GCNV iSCSI-Volume – Grid/Oracle-Homes, Staging	`/dev/mapper/ora_<host>_u01`
`ora_<host>_data_01`	50 GiB	ASM `+DATA`	`/dev/mapper/ora_<host>_data_01`
`ora_<host>_data_02`	50 GiB	ASM `+DATA` (gestreift)	`/dev/mapper/ora_<host>_data_02`
`ora_<host>_arch_01`	100 GiB	ASM `+RECO`	`/dev/mapper/ora_<host>_arch_01`

GCNV iSCSI-Volume	Größe	Verwenden	Multipath-Alias
ora_<host>_fra_01	100 GiB	ASM +FRA	/dev/mapper/ora_<host>_fra_01

Volume-Namen: Nur Buchstaben, Zahlen und Unterstriche (keine Bindestriche).

Minimales Layout (nur zur Validierung): Zwei LUNs pro Host (*_data, *_reco) mit arch_01p1→+RECO und arch_01p2→+FRA sind für das Labor akzeptabel; in der Produktion werden fünf Volumes pro Schritt 3: Erstellen der GCNV iSCSI-Volumes verwendet.

2. Die fünf Volumes für oracdb2 in oracle-pool-b mit Hostgruppe oracdb2-hg unter Verwendung derselben Spezifikationen erstellen. Für jeden Pool **NetApp Volumes** → **Volumes** → **Create** — iSCSI, korrekter Pool und Hostgruppe, Linux verwenden. Die folgenden Informationen sind zu dokumentieren:
   - iSCSI-Portal-IPs → <ISCSI_PORTAL_1>, <ISCSI_PORTAL_2> (primäre Pool-Portale auf oracdb1; Standby-Pool-Portale auf oracdb2 — sie können unterschiedlich sein)
   - Datenträger-Seriennummer aus der Cloud Console – Verwendung mit der vom Host ermittelten WWID in Schritt 4: Linux iSCSI und Multipath für GCNV iSCSI-Volumes konfigurieren

Schritt 4: iSCSI und Multipath konfigurieren

iSCSI und device-mapper-multipath sind auf jedem Datenbankhost zu konfigurieren, um über beide Storage-Portal-IPs auf die GCNV Volumes zuzugreifen. Diese Schritte sind auf oracdb1 mit den Portal-IPs des primären Pools auszuführen und anschließend auf oracdb2 mit den Portal-IPs des Standby-Pools zu wiederholen. Ist der ausgehende Datenverkehr des Hosts eingeschränkt, ist TCP/3260 von jeder Datenbank-VM zu ihren GCNV iSCSI-Portal-IPs zuzulassen (zusätzlich zu TCP/1521 zwischen den VMs Schritt 2: VPC-Firewall – TCP/1521 in allen drei Zonen auf die Allowlist setzen).

1. Ziele ermitteln, anmelden und Knotenstart beibehalten:

```
sudo iscsiadm --mode discovery --op update --type sendtargets --portal
<ISCSI_PORTAL_1>
sudo iscsiadm --mode discovery --op update --type sendtargets --portal
<ISCSI_PORTAL_2>
sudo iscsiadm --mode node --op update --name node.startup --value
automatic
sudo iscsiadm --mode node -l all
sudo systemctl enable --now iscsid iscsi multipathd
sudo iscsiadm --mode session         # expect 10 sessions (5 GCNV iSCSI
volumes × 2 portals)
sudo lsblk -o NAME,SIZE,WWN,VENDOR,MODEL
```

Nach dem Neustart vor dem Starten von Oracle erneut prüfen:

```
sudo iscsiadm --mode session
sudo multipath -ll
```

2. Mit Standardeinstellungen und Blacklist-Regeln `device-mapper-multipath` konfigurieren:

```
sudo tee /etc/multipath.conf >/dev/null <<'EOF'
defaults {
    find_multipaths     yes
    user_friendly_names yes
}
blacklist {
    devnode "^(ram|raw|loop|fd|md|dm-|sr|scd|st)[0-9]*"
    devnode "^hd[a-z]"
    devnode "^cciss.*"
}
EOF

sudo systemctl enable --now multipathd
sudo multipath -ll
```

3. Füge vom Host ermittelte WWID-Aliase zu `/etc/multipath.conf` hinzu (nicht raten — `multipath.conf` erweitert **keine** Shell-Variablen). WWIDs ermitteln:

```
sudo multipath -ll
for dev in /dev/sd*; do
  [ -b "$dev" ] || continue
  printf '%s: ' "$dev"
  sudo /usr/lib/udev/scsi_id --whitelisted --device="$dev" 2>/dev/null
|| true
  echo
done
```

Füge konkrete Aliase für diesen Host zu `/etc/multipath.conf` hinzu und dann `sudo systemctl restart multipathd`.

An `oracdb1`, anhängen:

```
multipaths {
    multipath { wwid <host-discovered-wwid-for-u01>      alias
ora_oracdb1_u01     }
    multipath { wwid <host-discovered-wwid-for-data-01>  alias
ora_oracdb1_data_01 }
    multipath { wwid <host-discovered-wwid-for-data-02>  alias
ora_oracdb1_data_02 }
    multipath { wwid <host-discovered-wwid-for-arch-01>  alias
ora_oracdb1_arch_01 }
    multipath { wwid <host-discovered-wwid-for-fra-01>   alias
ora_oracdb1_fra_01  }
}
```

Bei `oracdb2` wird dasselbe Muster mit `ora_oracdb2_*` Aliasen verwendet, dann:

```
sudo systemctl restart multipathd
ls -l /dev/mapper/ora_$(hostname -s)_*
```

Schritt 5: ASM-Geräte partitionieren

Die vier ASM-Backing-Devices (außer `u01`) jeweils mit einer GPT-Partition für die Nutzung durch ASM partitionieren und anschließend udev-Regeln für den Besitz durch Grid konfigurieren. Diese Schritte sind auf jedem Datenbank-Host auszuführen.

1. Die vier ASM-Backing-Devices mit GPT partitionieren und die Partitionen überprüfen:

```
HOST=$(hostname -s)        # oracdb1 on the primary, oracdb2 on the
standby
for dev in /dev/mapper/ora_${HOST}_data_01 \
           /dev/mapper/ora_${HOST}_data_02 \
           /dev/mapper/ora_${HOST}_arch_01 \
           /dev/mapper/ora_${HOST}_fra_01; do
  sudo parted -s "$dev" mklabel gpt
  sudo parted -s "$dev" mkpart primary 0% 100%
done
sudo partprobe
sudo systemctl reload multipathd
ls /dev/mapper/ora_${HOST}_*p1     # expect 4 partitions
```

2. udev-Regeln konfigurieren, um die Grid-Zuständigkeit zuzuweisen und die Änderungen auszulösen:

```
HOST=$(hostname -s)
sudo tee /etc/udev/rules.d/99-oracle-asm.rules >/dev/null <<'EOF'
KERNEL=="dm-*", ENV{DM_UUID}=="part?-mpath-*",
ENV{DM_NAME}=="ora_oracdb*_*p?", \
    OWNER="grid", GROUP="asmadmin", MODE="0660"
EOF

sudo udevadm control --reload-rules
for part in /dev/mapper/ora_${HOST}_*p1; do
  dm=$(readlink -f "$part" | xargs basename)
  sudo udevadm trigger --action=change --name-match="/dev/${dm}"
done
sudo udevadm settle
ls -lL /dev/mapper/ora_${HOST}_*p1    # grid:asmadmin 0660
```

Schritt 6: Formatieren und Einbinden `/u01`

Das `ora_<host>_u01`GCNV Volume wird mit XFS formatiert und unter Verwendung der UUID dauerhaft in `/etc/fstab` eingebunden. Das `/u01`Dateisystem enthält Grid home, Oracle home und Staging-Dateien.

1. Das Multipath-Gerät mit XFS formatieren und dessen UUID erfassen:

```
HOST=$(hostname -s)
U01_DEV=/dev/mapper/ora_${HOST}_u01
ls -l "$U01_DEV"

sudo mkfs.xfs -f "$U01_DEV"
U01_UUID=$(sudo blkid -s UUID -o value "$U01_DEV")
```

2. Den UUID-basierten Mount-Eintrag zu `/etc/fstab` hinzufügen und das Dateisystem mounten:

```
sudo mkdir -p /u01
echo "UUID=${U01_UUID} /u01 xfs defaults,_netdev,nofail,x-
systemd.requires=iscsi.service,x-systemd.requires=multipathd.service,x-
systemd.after=iscsi.service,x-systemd.after=multipathd.service 0 0" |
sudo tee -a /etc/fstab
sudo mount -a
```

3. Die Verzeichnisstruktur mit den korrekten Besitzrechten für Grid und Oracle Software erstellen:

```
sudo mkdir -p /u01/app/oraInventory /u01/app/26ai/grid /u01/app/grid \
  /u01/app/oracle/product/26ai/db_1 /u01/stage
sudo chown -R grid:oinstall /u01/app/oraInventory /u01/app/26ai
/u01/app/grid
sudo chown -R oracle:oinstall /u01/app/oracle /u01/stage
sudo chmod -R 775 /u01/app /u01/stage
```

Starten Sie das System einmal neu und bestätigen Sie `/u01`die Mounts, bevor Installation der Oracle Software.

Was kommt als Nächstes?

Um die Binärdateien für Oracle Grid Infrastructure und Database auf den vorbereiteten Hosts zu installieren, zu Die Oracle Grid Infrastructure und die Oracle Database Software installieren auf beiden Hosts wechseln.

Oracle Grid Infrastructure und Oracle Database 26ai auf Google Cloud NetApp Volumes installieren

Oracle Grid Infrastructure mit Oracle Restart und ASM auf Google Cloud NetApp Volumes iSCSI-Speicher für jeden Datenbankhost installieren, danach die Oracle Database 26ai Software installieren. Dieses Verfahren umfasst das Staging von Oracle GoldImages, das Ausführen unbeaufsichtigter Installationen mit Antwortdateien, das Erstellen von ASM-Festplattengruppen auf GCNV Volumes sowie die Vorbereitung sowohl der primären als auch der Standby-Hosts mit identischer Oracle Software vor der Datenbankerstellung.

Schritt 1: Installation der Grid Infrastructure auf jedem DB-Host

Die Oracle Grid Infrastructure GoldImage ist auf jedem Datenbankhost zu installieren, um Oracle Restart und ASM zu ermöglichen. Beide Hosts benötigen jeweils ein eigenes Grid Home, eine eigene ASM-Instanz und eigene Festplattengruppen; Data Guard repliziert Daten über Oracle Net, nicht über gemeinsam genutzten Speicher. Alle Schritte sind auf `oracdb1` vollständig durchzuführen, bevor sie auf `oracdb2` wiederholt werden.

1. Die Oracle GoldImages-, Release Update- und OPatch-Binärdateien werden in `/u01/stage` bereitgestellt:

```
sudo chown oracle:oinstall /u01/stage && sudo chmod 775 /u01/stage
# Upload GoldImages, RU, OPatch to /u01/stage.
```

2. Das Grid GoldImage wird direkt am Ziel-Grid-Home entpackt. Das 26ai GoldImage wird durch direktes Entpacken in das Zielverzeichnis installiert:

```
sudo -u grid bash -c '
cd /u01/app/26ai/grid
unzip -q /u01/stage/LINUX.X64_<RELEASE>_grid_home.zip
'
sudo chown -R grid:oinstall /u01/app/26ai/grid
```

Wenn das Grid GoldImage älter als die Ziel-RU ist, kann das Grid-Verzeichnis während der Einrichtung mithilfe des `gridSetup.sh -applyRU` Flows gepatcht werden, oder es wird ein GoldImage mit der RU gebündelt verwendet. Grid- und Datenbankverzeichnisse sollten sich auf dem gleichen vorgesehenen Patch-Stand befinden.

3. Die `gridSetup`Antwortdatei `/tmp/grid.rsp`wird auf jedem Host erstellt. Der Hostname wird ersetzt und starke Passwörter werden verwendet:

```
HOST=$(hostname -s)

sudo -u grid bash -c "cat > /tmp/grid.rsp <<RSP
oracle.install.responseFileVersion=/oracle/install/rspfmt_crsinstall_res
ponse_schema_v23.0.0
INVENTORY_LOCATION=/u01/app/oraInventory
installOption=HA_CONFIG
ORACLE_BASE=/u01/app/grid
clusterUsage=GENERAL_PURPOSE
OSDBA=asmdba
OSOPER=asmoper
OSASM=asmadmin
storageOption=FLEX_ASM_STORAGE
sysasmPassword=WelcomeOracle1!
asmsnmpPassword=WelcomeOracle1!
diskGroupName=DATA
redundancy=EXTERNAL
auSize=4
diskString=/dev/mapper/ora_${HOST}_*p*
diskList=/dev/mapper/ora_${HOST}_data_01p1,/dev/mapper/ora_${HOST}_data_
02p1
managementOption=NONE
RSP"
sudo -u grid chmod 600 /tmp/grid.rsp
```

4. Im Silent-Modus `gridSetup.sh`werden die Binärdateien kopiert und die Konfiguration bereitgestellt. Zu erwarten sind `Successfully Setup Software with warning(s).`die Exit-Codes 6 (Warnungen) oder 0:

```
sudo -u grid bash -c '
export ORACLE_HOME=/u01/app/26ai/grid
export ORACLE_BASE=/u01/app/grid
cd /u01/app/26ai/grid
./gridSetup.sh -silent -responseFile /tmp/grid.rsp -ignorePrereqFailure
'
```

5. `orainstRoot.sh` und `root.sh` als Root ausführen. Das `root.sh`Skript erstellt die `crsctl`, `srvctl` und `asmcmd` Wrapper und startet OHAS:

```
sudo /u01/app/oraInventory/orainstRoot.sh
sudo /u01/app/26ai/grid/root.sh
```

6. `gridSetup.sh -executeConfigTools` ausführen, um die Konfigurationsassistenten (NETCA, ASMCA, CVU) für die Antwortdatei auszuführen. Dadurch werden die ASM-Instanz und die `+DATA` Festplattengruppe erstellt. Nach NETCA / ASMCA / CVU ist Folgendes zu erwarten: `Successfully Configured Software.`

```
sudo -u grid bash -c '
export ORACLE_HOME=/u01/app/26ai/grid
export ORACLE_BASE=/u01/app/grid
cd /u01/app/26ai/grid
./gridSetup.sh -silent -executeConfigTools -responseFile /tmp/grid.rsp
'
```

7. Die `+RECO` und `+FRA` Festplattengruppen werden mit `asmca` erstellt. Die Single-Shot-Installation erstellt nur `+DATA`:

```
HOST=$(hostname -s)

sudo -u grid bash -c "
export ORACLE_HOME=/u01/app/26ai/grid
export ORACLE_SID=+ASM

\$ORACLE_HOME/bin/asmca -silent -createDiskGroup \
  -diskGroupName RECO \
  -disk /dev/mapper/ora_${HOST}_arch_01p1 \
  -redundancy EXTERNAL -au_size 4

\$ORACLE_HOME/bin/asmca -silent -createDiskGroup \
  -diskGroupName FRA \
  -disk /dev/mapper/ora_${HOST}_fra_01p1 \
  -redundancy EXTERNAL -au_size 4
"
```

8. Der Status der ASM-Festplattengruppen und der Oracle Restart-Ressourcen kann überprüft werden:

```
sudo -u grid ORACLE_HOME=/u01/app/26ai/grid ORACLE_SID=+ASM \
  /u01/app/26ai/grid/bin/sqlplus -s / as sysasm <<'SQL'
SELECT name, total_mb, free_mb, state FROM v$asm_diskgroup ORDER BY
name;
SQL

sudo /u01/app/26ai/grid/bin/crsctl stat res -t
# Expected ONLINE: ora.DATA.dg, ora.RECO.dg, ora.FRA.dg,
ora.LISTENER.lsnr, ora.asm, ora.cssd, ora.evmd.
```

9. Die oben genannten Schritte werden auf `oracdb2` wiederholt. Das `HOST=$(hostname -s)`-Muster in Schritte 3 und 4 und Schritt 7 wählt automatisch die GCNV iSCSI-Geräte dieses Hosts aus.

   Verwenden Sie dieselben ASM-Festplattengruppennamen — Data Guard repliziert über Oracle Net, nicht über den Speicher.

Schritt 2: Installation der Oracle Database auf jedem DB-Host

Die Oracle Database 26ai Software-Home wird auf jedem Datenbankhost mittels einer unbeaufsichtigten, rein softwarebasierten Installation mit dem neuesten Release Update installiert. Alle Schritte werden auf `oracdb1` abgeschlossen, bevor sie auf `oracdb2` wiederholt werden.

1. Das Datenbank-Home, den neuesten OPatch und den RU-Patch in die jeweiligen Verzeichnisse entpacken. Die Oracle-Dokumentation enthält Informationen zum RU-Verzeichnisaufbau und zum `-applyRU` Pfad.

```
sudo su - oracle
cd /u01/app/oracle/product/26ai/db_1
unzip -q /u01/stage/LINUX.X64_<RELEASE>_db_home.zip
rm -rf OPatch
unzip -q /u01/stage/p6880880_<base>_Linux-x86-64.zip
# latest OPatch
unzip -q /u01/stage/p<RU_PATCH>_<base>_Linux-x86-64.zip -d /u01/stage
# latest 26ai RU
```

2. Die Installationsantwortdatei wird erstellt und die unbeaufsichtigte Softwareinstallation mit angewendetem RU durchgeführt. Unter OL 8/9 ist `-applyOneOffs` in der `runInstaller` Zeile wegzulassen:

```
sudo -u oracle tee /u01/stage/dbinstall.rsp >/dev/null <<'EOF'
oracle.install.option=INSTALL_DB_SWONLY
UNIX_GROUP_NAME=oinstall
INVENTORY_LOCATION=/u01/app/oraInventory
ORACLE_HOME=/u01/app/oracle/product/26ai/db_1
ORACLE_BASE=/u01/app/oracle
oracle.install.db.InstallEdition=EE
oracle.install.db.OSDBA_GROUP=dba
oracle.install.db.OSOPER_GROUP=oper
oracle.install.db.OSBACKUPDBA_GROUP=backupdba
oracle.install.db.OSDGDBA_GROUP=dgdba
oracle.install.db.OSKMDBA_GROUP=kmdba
oracle.install.db.OSRACDBA_GROUP=racdba
oracle.install.db.rootconfig.executeRootScript=false
EOF

sudo -u oracle bash -c '
export CV_ASSUME_DISTID=OEL10    # OEL9 / OEL8.10 if cluvfy requires it
cd /u01/app/oracle/product/26ai/db_1
./runInstaller -applyRU /u01/stage/<RU_PATCH> \
  -applyOneOffs /u01/stage/39292021 \
  -silent -ignorePrereqFailure -responseFile /u01/stage/dbinstall.rsp
'
```

3. Das Root-Skript nach der Installation ausführen:

```
sudo /u01/app/oracle/product/26ai/db_1/root.sh
```

4. Die Oracle-Umgebung wird auf jedem DB-Host festgelegt. Verwendung von `ORACLE_SID=orcl` auf `oracdb1` und `ORACLE_SID=orcls` auf `oracdb2`:

```
sudo -u oracle tee -a /home/oracle/.bash_profile >/dev/null <<'EOF'
export ORACLE_HOME=/u01/app/oracle/product/26ai/db_1
export ORACLE_SID=orcl                          # use 'orcls' on oracdb2
export GRID_HOME=/u01/app/26ai/grid
export PATH=$ORACLE_HOME/bin:$GRID_HOME/bin:$PATH
export TNS_ADMIN=$ORACLE_HOME/network/admin
EOF
```

Die Standby-Datenbank wird in Erstellen Sie die Standby-Datenbank erstellt.

Was kommt als Nächstes?

Um die primäre Produktionsinstanz für Ihre HA-Bereitstellung zu erstellen, gehen Sie zu Die primäre Oracle Datenbank erstellen auf `oracdb1`.

Die primäre Oracle Datenbank auf Google Cloud NetApp Volumes erstellen

Die primäre Oracle Datenbank auf Google Cloud NetApp Volumes iSCSI-Speicher wird mithilfe des Oracle Database Configuration Assistant im Silent-Modus erstellt. Dieses Verfahren umfasst das Ausführen von `dbca` zur Erstellung der Container-Datenbank und der Pluggable Database auf GCNV-gestützten ASM-Festplattengruppen, die Konfiguration der Archivprotokollziele sowie das Hinzufügen eines rollenbasierten Anwendungsdienstes für transparentes Failover nach der Aktivierung von Data Guard.

Schritte

Die Oracle container database und die pluggable database werden auf `oracdb1` unter Verwendung von `dbca` im Silent-Modus erstellt, die Archive-Log-Ziele konfiguriert, die Oracle Restart-Registrierung überprüft und ein rollenbasierter Anwendungsdienst für transparentes Client-Failover hinzugefügt.

1. `dbca`Im Silent-Modus ausführen, um die CDB und PDB auf den ASM-Festplattengruppen zu erstellen:

```
sudo -u oracle bash -c '
export ORACLE_HOME=/u01/app/oracle/product/26ai/db_1
export PATH=$ORACLE_HOME/bin:$PATH

dbca -silent -createDatabase \
  -templateName General_Purpose.dbc \
  -gdbname orcl -sid orcl \
  -characterSet AL32UTF8 -nationalCharacterSet AL16UTF16 \
  -sysPassword "ChangeMe!1" -systemPassword "ChangeMe!1" \
  -emConfiguration NONE \
  -datafileDestination +DATA -storageType ASM \
  -recoveryAreaDestination +FRA -recoveryAreaSize 25000 \
  -enableArchive true -archiveLogMode AUTO \
  -memoryMgmtType AUTO_SGA -totalMemory 4096 \
  -databaseType MULTIPURPOSE \
  -createAsContainerDatabase true -numberOfPDBs 1 \
  -pdbName orclpdb -pdbAdminPassword "ChangeMe!1" \
  -ignorePreReqs
'
```

2. Archivprotokolle werden auf `+RECO` gesetzt, und der Zustand der pluggable database wird geöffnet und gespeichert. Der Standby verwendet entsprechende Archivprotokolleinstellungen in Schritt 2: Standby init.ora, pfile und NOMOUNT:

```
sudo -u oracle bash -c '
export ORACLE_HOME=/u01/app/oracle/product/26ai/db_1
export ORACLE_SID=orcl
$ORACLE_HOME/bin/sqlplus -s / as sysdba <<SQL
ALTER SYSTEM SET log_archive_dest_1='\''LOCATION=+RECO
VALID_FOR=(ALL_LOGFILES,ALL_ROLES) DB_UNIQUE_NAME=orcl'\'' SCOPE=BOTH;
ALTER PLUGGABLE DATABASE ALL OPEN;
ALTER PLUGGABLE DATABASE ALL SAVE STATE;
EXIT
SQL
'
```

3. Es wird überprüft, ob die Datenbank unter Oracle Restart läuft:

```
sudo /u01/app/26ai/grid/bin/srvctl status database -d orcl
# Expected: Database is running

sudo -u oracle sqlplus -s / as sysdba <<<"SELECT name, open_mode,
log_mode FROM v\$database;"
# Expected: ORCL, READ WRITE, ARCHIVELOG
```

4. Ein rollenbasierter Anwendungsdienst sorgt dafür, dass Anwendungen sich über `orclapp` verbinden und ein Failover transparent erfolgt, wenn Data Guard aktiviert ist:

```
sudo -u oracle bash -c '
export GRID_HOME=/u01/app/26ai/grid
export ORACLE_HOME=/u01/app/oracle/product/26ai/db_1
export PATH=$ORACLE_HOME/bin:$GRID_HOME/bin:$PATH

srvctl add service \
  -db orcl \
  -service orclapp \
  -pdb orclpdb \
  -role PRIMARY \
  -policy AUTOMATIC

srvctl start service -db orcl -service orclapp
srvctl status service -db orcl -service orclapp
'
```

Nach der Aktivierung des Data Guard Broker `orclapp` läuft nur auf dem PRIMARY. Steuerdateien über ASM-Festplattengruppen multiplexen und den Arbeitsspeicher auf die Arbeitslast dimensionieren.

Was kommt als Nächstes?

Um Standby-Schutz und Failover-Bereitschaft einzurichten, gehen Sie zu Die Oracle Standby-Datenbank erstellen auf `oracdb2`.

Erstellung der Oracle-Standby-Datenbank mit Google Cloud NetApp Volumes Storage-Layer-Seeding

Die physische Oracle Standby-Datenbank wird mithilfe von Google Cloud NetApp Volumes Storage-Layer-Replizierung, Snapshots oder Klonen erstellt, um die Standby-Initialisierung im Vergleich zu herkömmlichen RMAN-Methoden zu beschleunigen. Dieses Verfahren umfasst die Konfiguration des Listeners, die Erstellung der Standby-pfile, das Seeding der Standby-Volumes mit GCNV-Replizierung, die Finalisierung der Oracle-Instanz und die Registrierung des Standbys bei Oracle Restart. Alle HA-Tiers schließen diese Schritte ab. Für die **Prod HA (Data Guard + FSFO)**-Tier wird mit Data Guard

Abschluss fortgefahren, bevor Data Guard Broker, Fast-Start Failover und der Observer konfiguriert wird.

Schritt 1: Listener und Data Guard Parameter konfigurieren

Der Listener wird auf beiden Datenbankhosts so konfiguriert, dass Data Guard-Verbindungen unterstützt werden, einschließlich des für den Broker erforderlichen `_DGMGRL` Dienstes. Die Kennwortdatei wird eingerichtet und die Archivprotokollparameter auf der primären Datenbank konfiguriert.

1. Den primären Listener konfigurieren und die Umgebung auf `oracdb1` überprüfen:

```
sudo su - oracle
. ~/.bash_profile        # ORACLE_SID=orcl, ORACLE_HOME set
```

2. Den Standby-Listener auf `oracdb2` so konfigurieren, dass die `orcls`- und `orcls_DGMGRL`-Dienste enthalten sind:

```
GRID_HOME=/u01/app/26ai/grid
sudo -u grid tee "$GRID_HOME/network/admin/listener.ora" >/dev/null <<
'EOF'
LISTENER =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = oracdb2.example.internal)(PORT =
1521)))

SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC = (GLOBAL_DBNAME = orcls)        (ORACLE_HOME =
/u01/app/oracle/product/26ai/db_1) (SID_NAME = orcls))
    (SID_DESC = (GLOBAL_DBNAME = orcls_DGMGRL) (ORACLE_HOME =
/u01/app/oracle/product/26ai/db_1) (SID_NAME = orcls)))
EOF
```

3. Der Listener wird auf beiden Hosts über Oracle Restart neu gestartet, und es wird überprüft, ob der `_DGMGRL` Dienst registriert ist:

```
sudo -u grid bash -c '
export GRID_HOME=/u01/app/26ai/grid
export ORACLE_HOME=$GRID_HOME
$GRID_HOME/bin/srvctl stop listener
$GRID_HOME/bin/srvctl start listener
$GRID_HOME/bin/lsnrctl status
'
```

`lsnrctl status`muss `<SID>`auflisten und `<SID>_DGMGRL.

Schritt 2: Standby-pfile vorbereiten und NOMOUNT

Die Standby-Datenbankinstanz wird vorbereitet, indem die Passwortdatei von der primären Datenbank kopiert, eine minimale init.ora-pfile mit Data Guard-Parametern erstellt und die Instanz im NOMOUNT-Modus gestartet wird.

1. Die primäre Passwortdatei wird mithilfe von IAP auf den Standby-Host kopiert und `gcloud compute scp`:

```
PRIMARY_ZONE=us-west1-a        # zone of oracdb1
STANDBY_ZONE=us-west1-b        # zone of oracdb2

gcloud compute scp \
  oracdb1:/u01/app/oracle/product/26ai/db_1/dbs/orapworcl ./orapworcl \
  --zone=$PRIMARY_ZONE --tunnel-through-iap

gcloud compute scp \
  ./orapworcl oracdb2:/u01/app/oracle/product/26ai/db_1/dbs/orapworcls \
  --zone=$STANDBY_ZONE --tunnel-through-iap
```

2. Den Wert des `compatible`-Parameters aus der primären Datenbank abfragen:

```
# On oracdb1
sudo -u oracle sqlplus -s / as sysdba \
  <<<"SELECT value FROM v\$parameter WHERE name='compatible';"
```

3. Die Standby-PFILE auf `oracdb2` erstellen, die Besitzrechte für die Passwortdatei festlegen und die Instanz im NOMOUNT-Modus starten. Der `compatible` Wert aus dem vorherigen Schritt ist durch `<COPY_FROM_PRIMARY>` zu ersetzen:

```
sudo -u oracle mkdir -p /u01/app/oracle/admin/orcls/adump
sudo chown oracle:oinstall
/u01/app/oracle/product/26ai/db_1/dbs/orapworcls
sudo chmod 0600 /u01/app/oracle/product/26ai/db_1/dbs/orapworcls

sudo -u oracle tee /u01/app/oracle/product/26ai/db_1/dbs/initorcls.ora
>/dev/null <<'EOF'
*.db_name='orcl'
*.db_unique_name='orcls'
*.audit_file_dest='/u01/app/oracle/admin/orcls/adump'
*.diagnostic_dest='/u01/app/oracle'
*.compatible='<COPY_FROM_PRIMARY>'
*.sga_target=3072m
```

```
*.pga_aggregate_target=1024m
*.processes=320
*.remote_login_passwordfile='EXCLUSIVE'
*.standby_file_management='AUTO'
*.fal_server='orcl'
*.log_archive_config='DG_CONFIG=(orcl,orcls)'
*.log_archive_dest_1='LOCATION=+RECO VALID_FOR=(ALL_LOGFILES,ALL_ROLES)
DB_UNIQUE_NAME=orcls'
*.log_archive_dest_2='SERVICE=orcl AFFIRM SYNC
VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE) DB_UNIQUE_NAME=orcl'
*.log_archive_dest_state_2='DEFER'
*.log_archive_format='%t_%s_%r.arc'
*.dg_broker_start=TRUE
*.undo_tablespace='UNDOTBS1'
*.open_cursors=300
*.db_create_file_dest='+DATA'
*.db_create_online_log_dest_1='+DATA'
*.db_recovery_file_dest='+FRA'
*.db_recovery_file_dest_size=25000m
EOF

echo "orcls:/u01/app/oracle/product/26ai/db_1:N" | sudo tee -a
/etc/oratab

sudo -u oracle bash -c '
export ORACLE_HOME=/u01/app/oracle/product/26ai/db_1
export ORACLE_SID=orcls
sqlplus / as sysdba <<SQL
STARTUP NOMOUNT
PFILE=/u01/app/oracle/product/26ai/db_1/dbs/initorcls.ora;
EXIT
SQL
'
```

Die Standby-Instanz befindet sich nun im NOMOUNT-Modus und enthält keine Datendateien bis Schritt 3: Standby-Speicher mit GCNV initialisieren.

Schritt 3: Standby-Speicher mit GCNV initialisieren

Die Standby-Volumes werden in `oracle-pool-b` initialisiert, an `oracdb2` angehängt, ASM-Disk-Gruppen eingebunden und die Standby-Instanz im MOUNT-Status finalisiert.

GCNV-Replikation eignet sich für das Seeding in der Produktion, während Snapshot-Seeding für einmalige Labor-Workflows verwendet wird.

Wählen Sie den Startpfad

Die Standby-Seeding-Methode wird entsprechend der Umgebung und den Wiederherstellungsanforderungen ausgewählt.

- **Empfohlen für den Produktiveinsatz:** Der Replikationspfad in Replikationspfad: Replikationen erstellen und synchronisieren und Replikationspfad: Umschaltung und Anbindung von Standby-Volumes wird verwendet.
- **Alternative für Labore:** Alternativer Pfad: Seed aus Snapshot verwenden.

Alle Pfade treffen sich wieder bei Standby-ASM-Diskgruppen einbinden und Die Standby-Instanz finalisieren.

Voraussetzungen prüfen

Die folgenden Voraussetzungen sollten vor der Initialisierung von Standby-Volumes bestätigt werden.

- `gcloud netapp` mit Unterstützung für Volumenreplikation.
- Zwei **Standardmodus**-Pools an verschiedenen Standorten (`oracle-pool-a`, `oracle-pool-b`).
- Quellvolumes im primären Pool sind an `oracdb1-hg`; Zielvolumes werden durch Replikation erstellt.
- Führen Sie die Replikation von der Cloud Shell oder einer Workstation aus – nicht von den DB-VMs.
- Am `oracdb2` vollständige iSCSI- und ASM-Host-Einrichtung von Schritt 4, Schritt 5 und Schritt 6.

```
export PROJECT=<your-gcp-project>
export LOC_A=us-west1-a
export LOC_B=us-west1-b
export DEST_POOL="projects/${PROJECT}/locations/${LOC_B
}/storagePools/oracle-pool-b"
```

- Erstellen eines Standby-Pools bei Bedarf:

```
gcloud netapp storage-pools create oracle-pool-b \
  --project="${PROJECT}" --location="${LOC_B}" \
  --service-level=flex --type=unified --mode=default \
  --capacity=1024 --network=name=<your-vpc>
```

Replikationen erstellen und synchronisieren

Replikationsbeziehungen von primären Volumes zu Standby-Volumes werden erstellt, anschließend wird auf den Abschluss der anfänglichen Synchronisierung gewartet.

```
gcloud netapp volumes replications create repl-oracdb2-data \
  --project="${PROJECT}" --location="${LOC_A}" --volume=oracdb1_data \
  --replication-schedule=EVERY_10_MINUTES \
  --destination-volume-parameters="storage_pool=${DEST_POOL
},volume_id=oracdb2_data,share_name=oracdb2_data"

gcloud netapp volumes replications create repl-oracdb2-reco \
  --project="${PROJECT}" --location="${LOC_A}" --volume=oracdb1_reco \
  --replication-schedule=EVERY_10_MINUTES \
  --destination-volume-parameters="storage_pool=${DEST_POOL
},volume_id=oracdb2_reco,share_name=oracdb2_reco"
```

+

Warten Sie, bis `mirrorState` ist `MIRRORED` und die initiale Synchronisierung für jede Replikation abgeschlossen ist.

Umschalten und Standby-Volumes anschließen

Den primären Host in den Ruhezustand versetzen, die Replikation nach der finalen Synchronisierung stoppen und die Zielvolumes an die Standby-Hostgruppe anhängen.

Auf dem primären System werden Schreibvorgänge angehalten und Wiederherstellungsmetadaten erfasst:

```
ALTER DATABASE BEGIN BACKUP;
SELECT CURRENT_SCN FROM V$DATABASE;
ALTER DATABASE CREATE STANDBY CONTROLFILE AS '/tmp/orcls_stby.ctl';
```

Einen letzten Replikationszyklus zulassen und anschließend die Replikationen stoppen:

```
gcloud netapp volumes replications stop repl-oracdb2-data \
  --project="${PROJECT}" --location="${LOC_A}" --volume=oracdb1_data
--force

gcloud netapp volumes replications stop repl-oracdb2-reco \
  --project="${PROJECT}" --location="${LOC_A}" --volume=oracdb1_reco
--force
```

Zielvolumes an `oracdb2-hg` anhängen (replizierte LUNs behalten möglicherweise die Quellnamen):

```
HG=$(gcloud netapp host-groups describe oracdb2-hg --project="${PROJECT}"
\
  --location=us-west1 --format='value(name)')

gcloud netapp volumes update oracdb2_data --project="${PROJECT}"
--location="${LOC_B}" \
  --block-devices="name=oracdb1_data_lun,host-groups=${HG},os-type=LINUX"
```

Kopieren Sie die Standby-Kontrolldatei nach `oracdb2`, dann den Sicherungsmodus auf dem primären System beenden:

```
ALTER DATABASE END BACKUP;
```

Seed aus Snapshot

Dieser Pfad wird für einmaliges Lab-Seeding verwendet, wenn keine kontinuierliche Replizierung erforderlich ist.

Für eine einmalige Testumgebung einen Quell-Snapshot erstellen und daraus Standby-Volumes in `oracle-pool-b` (Cloud Console oder API) erstellen. Die erstellten Volumes an `oracdb2-hg` anhängen, dann mit Standby-ASM-Diskgruppen einbinden fortfahren.

Standby-ASM-Diskgruppen einbinden

Auf dem Standby-Host werden die angeschlossenen Speicherpfade erkannt und die ASM-Festplattengruppen vor dem Datenbank-Recovery eingebunden.

Auf `oracdb2` bei den iSCSI-Portalen des Standby-Pools anmelden und Multipath-Geräte erneut scannen. Wenn die ASM-Disk-Header der primären Namensgebung in einem Labor-Workflow entsprechen, primäre Aliase verwenden (zum Beispiel `ora_oracdb1_data_01`, `ora_oracdb1_arch_01`), `asm_diskstring='/dev/mapper/ora_oracdb1_*p*'` setzen und bestätigen, dass die Partitionszuordnung `grid:asmadmin` ist, dann die Festplattengruppen einbinden:

```
ALTER DISKGROUP DATA MOUNT FORCE;
ALTER DISKGROUP RECO MOUNT FORCE;
ALTER DISKGROUP FRA MOUNT FORCE;
```

Die Standby-Instanz finalisieren

Die Standby-Kontrolldatei wird wiederhergestellt, eine Wiederherstellung auf das erfasste SCN durchgeführt, in einen physischen Standby konvertiert und die verwaltete Wiederherstellung gestartet.

```
STARTUP NOMOUNT;
RESTORE STANDBY CONTROLFILE FROM '/tmp/orcls_stby.ctl';
ALTER DATABASE MOUNT;
RECOVER DATABASE UNTIL SCN <quiesce_scn>;
ALTER DATABASE CONVERT TO PHYSICAL STANDBY;
SHUTDOWN IMMEDIATE;
STARTUP MOUNT;
ALTER DATABASE RECOVER MANAGED STANDBY DATABASE DISCONNECT FROM SESSION;
```

An diesem Punkt sollte der Standby aktiviert `PHYSICAL STANDBY` und `MOUNTED` mit gestarteter verwalteter Wiederherstellung sein.

Tierspezifische nächste Schritte:

- **Prod HA (ohne Data Guard):** Weiter direkt zu Schritt 4: Standby bei Oracle Restart registrieren.
- **Prod HA (Data Guard + FSFO):** Weiter zu Schritt 4: Standby bei Oracle Restart registrieren, dann weiter zu Data Guard Abschlussschritte.

Schritt 4: Standby bei Oracle Restart registrieren

Die Standby-Datenbank wird bei Oracle Restart registriert, sodass Neustarts automatisch ASM-Festplattengruppen wiederherstellen, die Standby-Datenbank einbinden und die verwaltete Wiederherstellung neu starten. Der Anwendungsdienst wird außerdem zu beiden Datenbank-Ressourcen hinzugefügt.

1. Den Speicherort der spfile aus der Standby-Datenbank erfassen und bei Oracle Restart auf `oracdb2` registrieren. `<STANDBY_SPFILE_PATH>` aus der Abfrage ersetzen (häufig unter `+DATA`):

```
sudo -u oracle bash -c '
export ORACLE_SID=orcls
sqlplus -s / as sysdba <<< "SHOW PARAMETER spfile;"
'

sudo -u oracle bash -c '
export GRID_HOME=/u01/app/26ai/grid
export ORACLE_HOME=/u01/app/oracle/product/26ai/db_1
export PATH=$ORACLE_HOME/bin:$GRID_HOME/bin:$PATH

srvctl add database \
  -db orcls \
  -dbname orcl \
  -oraclehome /u01/app/oracle/product/26ai/db_1 \
  -spfile <STANDBY_SPFILE_PATH> \
  -pwfile /u01/app/oracle/product/26ai/db_1/dbs/orapworcls \
  -role PHYSICAL_STANDBY \
  -startoption MOUNT \
  -stopoption IMMEDIATE \
  -diskgroup DATA,RECO,FRA

srvctl config database -db orcls
srvctl status database -db orcls
'
```

2. Die primäre Datenbankressource auf `oracdb1` wird überprüft und aktualisiert, sodass alle ASM-Festplattengruppen-Abhängigkeiten enthalten sind:

```
sudo -u oracle bash -c '
export GRID_HOME=/u01/app/26ai/grid
export ORACLE_HOME=/u01/app/oracle/product/26ai/db_1
export PATH=$ORACLE_HOME/bin:$GRID_HOME/bin:$PATH
srvctl config database -db orcl
srvctl modify database -db orcl -diskgroup DATA,RECO,FRA
srvctl config database -db orcl
'
```

3. Den Anwendungsdienst zur Standby Datenbankressource (`orcls` auf `oracdb2` hinzufügen. `role PRIMARY` auf beiden Seiten verwenden, sodass `orclapp` nach der Umschaltung verfügbar ist:

```
sudo -u oracle bash -c '
export GRID_HOME=/u01/app/26ai/grid
export ORACLE_HOME=/u01/app/oracle/product/26ai/db_1
export PATH=$ORACLE_HOME/bin:$GRID_HOME/bin:$PATH

srvctl add service \
  -db orcls \
  -service orclapp \
  -pdb orclpdb \
  -role PRIMARY \
  -policy AUTOMATIC

srvctl config service -db orcls -service orclapp
'
```

4. Die Standby-Datenbankressource auf `oracdb2` überprüfen:

```
sudo -u oracle bash -c '
export GRID_HOME=/u01/app/26ai/grid
export ORACLE_HOME=/u01/app/oracle/product/26ai/db_1
export PATH=$ORACLE_HOME/bin:$GRID_HOME/bin:$PATH
srvctl status database -db orcls
'
```

Was kommt als Nächstes?

Tierspezifisch:

- **Prod HA (ohne Data Guard):** Um ein auf Speicherreplikation basierendes Wiederherstellungsziel aufrechtzuerhalten, ist die Standby-Initialisierung abgeschlossen und die Standby-Datenbank wird bei Oracle Restart als Backup-Instanz registriert.
- **Prod HA (Data Guard + FSFO):** Um Broker-verwaltetes Switchover und Fast-Start-Failover zu aktivieren, mit Die Standby-Datenbank für Data Guard finalisieren fortfahren.

Die Standby-Datenbank für Data Guard auf Google Cloud NetApp Volumes finalisieren

Die Standby-Datenbank für Oracle Data Guard auf Google Cloud NetApp Volumes wird finalisiert, indem Standby-Wiederherstellungsprotokolldateien erstellt, Flashback-Datenbank aktiviert, Redo-Shipping aktiviert und der Data Guard-Status überprüft werden.

Tierspezifisch: Dieses Verfahren ist nur für die **Prod HA (Data Guard + FSFO)** Tier erforderlich.

Schritt 1: Erstellen von Standby Wiederherstellungsprotokoll-Logdateien

Standby-Wiederherstellungsprotokolldateien auf beiden Datenbankhosts erstellen, um Fast-Start Failover zu unterstützen. Die Größe muss größer oder gleich dem größten primären Online-Wiederherstellungsprotokoll sein, und die Anzahl sollte (Online-Gruppen pro Thread) + 1 entsprechen. Nach dem GCNV-Seeding die Standby-Wiederherstellungsprotokolle auf dem Standby-Host löschen und neu erstellen, um replizierte Pfade zu korrigieren.

1. Standby-Wiederherstellungsprotokoll-Logdateien auf der primären Datenbank erstellen (`orcl`:

```
ALTER SYSTEM SET db_create_file_dest='+DATA' SCOPE=BOTH;
ALTER DATABASE ADD STANDBY LOGFILE THREAD 1 ('+DATA') SIZE 1024M;
-- repeat (online log groups + 1) times
```

2. Standby-Wiederherstellungsprotokolldateien auf der Standby-Datenbank (`orcls` nach dem GCNV-Seeding löschen und neu erstellen. Replizierte Pfade unter `+DATA/ORCL/…` cause `ORA-19527` / `ORA-16086` bis zum Wiederaufbau:

```
ALTER DATABASE RECOVER MANAGED STANDBY DATABASE CANCEL;
ALTER SYSTEM SET standby_file_management=MANUAL SCOPE=BOTH;
-- DROP STANDBY LOGFILE GROUP for each group# in v$standby_log;
ALTER SYSTEM SET db_create_file_dest='+DATA' SCOPE=BOTH;
ALTER SYSTEM SET standby_file_management=AUTO SCOPE=BOTH;
ALTER DATABASE ADD STANDBY LOGFILE THREAD 1 ('+DATA') SIZE 1024M;
-- repeat (online groups + 1) times; one member per group
ALTER DATABASE RECOVER MANAGED STANDBY DATABASE USING CURRENT LOGFILE
DISCONNECT FROM SESSION;
```

Schritt 2: Flashback aktivieren und Recovery starten

Die Flashback Datenbank auf dem Standby-System wird aktiviert, um die automatische Wiederherstellung nach einem Failover zu unterstützen, anschließend wird die verwaltete Wiederherstellung mit Echtzeit gestartet. Flashback muss vor dem Start der verwalteten Wiederherstellung aktiviert sein, da dies bei aktivem MRP nicht möglich ist.

1. Die Standby-Datenbank herunterfahren, im MOUNT-Modus neu starten und die Flashback-Datenbank auf `oracdb2` aktivieren:

```
# On oracdb2
sudo -u oracle bash -c '
. ~/.bash_profile
export ORACLE_SID=orcls
sqlplus / as sysdba <<SQL
SHUTDOWN IMMEDIATE;
STARTUP MOUNT;
ALTER SYSTEM SET db_flashback_retention_target=1440 SCOPE=BOTH;
ALTER DATABASE FLASHBACK ON;
EXIT
SQL'
```

2. Verwaltete Wiederherstellung mit Echtzeit-Apply starten:

```
sudo -u oracle bash -c '
. ~/.bash_profile
export ORACLE_SID=orcls
sqlplus / as sysdba <<SQL
ALTER DATABASE RECOVER MANAGED STANDBY DATABASE USING CURRENT LOGFILE
DISCONNECT FROM SESSION;
EXIT
SQL'
```

`USING CURRENT LOGFILE`ermöglicht die Echtzeit-Anwendung (Wiederherstellungsprotokoll wird angewendet, sobald es in den SRLs landet).

Schritt 3: Wiederherstellungsprotokoll-Versand aktivieren

Aktivieren des Transports des Wiederherstellungsprotokolls von der Primärdatenbank zur Standby-Datenbank durch Aktivieren von `LOG_ARCHIVE_DEST_STATE_2`, was in Schritt 2 des Standby-Initialisierungsverfahrens bewusst auf `DEFER` gesetzt wurde, um `ORA-12154` Fehler während der Standby-Erstellung zu unterdrücken.

1. Zu `LOG_ARCHIVE_DEST_STATE_2` und `ENABLE` wechseln und einen Log-Switch erzwingen, um die Wiederherstellungsprotokoll-Übertragung zu initiieren:

```
sudo -u oracle bash -c '
. ~/.bash_profile
sqlplus / as sysdba <<SQL
ALTER SYSTEM SET LOG_ARCHIVE_DEST_STATE_2=ENABLE SCOPE=BOTH;
ALTER SYSTEM SWITCH LOGFILE;
ALTER SYSTEM ARCHIVE LOG CURRENT;
EXIT
SQL'
```

2. Überprüfen, ob das Wiederherstellungsprotokoll ordnungsgemäß übertragen wird:

```
sudo -u oracle bash -c '
. ~/.bash_profile
sqlplus / as sysdba <<SQL
SELECT dest_id, status, error FROM v\$archive_dest_status WHERE dest_id
IN (1,2);
EXIT
SQL'
# Expected: dest_id=2, STATUS=VALID, ERROR null.
```

Falls `dest_2` `ORA-12154` angezeigt wird, den Primary neu starten. Nach Schritt 1: Aktivieren Sie den Broker auf beiden Datenbanken den Transport über DGMGRL verwalten.

Schritt 4: Data Guard Status überprüfen

Es sollte sichergestellt sein, dass sich die primäre Datenbank im READ WRITE-Modus befindet und die Standby-Datenbank mit verwaltetem Recovery eingebunden ist, wobei Wiederherstellungsprotokolle angewendet werden.

1. Die primäre Datenbankrolle und der Öffnungsmodus auf `oracdb1` werden überprüft:

```
sudo -u oracle sqlplus -s / as sysdba \
  <<<"SELECT database_role || ' | ' || open_mode FROM v\$database;"
# Expected: PRIMARY | READ WRITE
```

2. Die Rolle der Standby-Datenbank, den Open Mode und den Status des Managed Recovery auf `oracdb2` überprüfen:

```
gcloud compute ssh oracdb2 --tunnel-through-iap --zone=us-west1-b

sudo -u oracle bash <<'BASH'
. ~/.bash_profile
export ORACLE_SID=orcls

sqlplus -s / as sysdba <<'SQL'
SELECT database_role || ' | ' || open_mode
FROM v$database;

SELECT process, status, sequence#
FROM v$managed_standby
WHERE process IN ('MRP0','RFS');

EXIT
SQL
BASH
```

Erwartet im Standby-Modus: `PHYSICAL STANDBY | MOUNTED`; `MRP0` mit `APPLYING_LOG`.

3. Wenn der Standby-Server `MOUNTED` meldet, die Anwendung jedoch nicht läuft, sollte die verwaltete Wiederherstellung auf `oracdb2` neu gestartet werden:

```
sudo -u oracle bash -c '
. ~/.bash_profile
export ORACLE_SID=orcls
sqlplus / as sysdba <<SQL
ALTER DATABASE RECOVER MANAGED STANDBY DATABASE CANCEL;
ALTER DATABASE RECOVER MANAGED STANDBY DATABASE USING CURRENT LOGFILE
DISCONNECT FROM SESSION;
EXIT
SQL'
```

Was kommt als Nächstes?

Um die automatisierte Rollenverwaltung und den Ausfallschutz zu aktivieren, mit Konfigurieren Sie Oracle Data Guard Broker, Fast-Start Failover und den Observer fortfahren.

Konfiguration von Data Guard Broker und Fast-Start-Failover für Oracle Database 26ai auf Google Cloud NetApp Volumes

Oracle Data Guard Broker und Fast-Start Failover mit einem dedizierten Observer

konfigurieren, damit automatische Rollenübergänge für Oracle Database 26ai auf Google Cloud NetApp Volumes möglich sind.

Tierspezifisch: Dieses Verfahren gilt nur für die **Prod HA (Data Guard + FSFO)** Tier.

Dieses Verfahren umfasst die Aktivierung des Brokers auf beiden Datenbanken, die Erstellung der Data Guard Konfiguration, die Aktivierung von FSFO mit MaxAvailability Schutzmodus, die Installation des Oracle Instant Client auf dem Observer-Host, den Start des Observers als systemd-Dienst mit walletbasierten Anmeldeinformationen sowie das Testen von Switchover und Failover. Nach `ENABLE CONFIGURATION` der Einrichtung erfolgt die Verwaltung von Transport und Rollen über **DGMGRL** (nicht per Ad-hoc `LOG_ARCHIVE_DEST_*` SQL).

Schritt 1: Data Guard Broker aktivieren

Der Data Guard Broker wird auf beiden Datenbankhosts aktiviert und die Brokerkonfiguration erstellt, die die primäre und die Standby-Datenbank unter einheitlicher Verwaltung verbindet.

1. Auf den primären und Standby Datenbankhosts `dg_broker_start=TRUE` festlegen:

```
sudo -u oracle bash -c '
. ~/.bash_profile
sqlplus / as sysdba <<SQL
ALTER SYSTEM SET dg_broker_start=TRUE SCOPE=BOTH;
EXIT
SQL'
```

2. Auf dem primären Server erfolgt die Verbindung zu DGMGRL mit Betriebssystem-Authentifizierung, und die Broker-Konfiguration wird erstellt:

> Nur auf dem Observer-Host verwenden Sie `dgmgrl /@orcl` nachdem die Auto-Login-Wallet existiert. Geben Sie keine Passwörter in die `dgmgrl` Befehlszeile ein.

```
sudo -u oracle bash -c '
export ORACLE_HOME=/u01/app/oracle/product/26ai/db_1
export ORACLE_SID=orcl
export PATH=$ORACLE_HOME/bin:$PATH
dgmgrl /
'
```

```
DGMGRL> CREATE CONFIGURATION 'orcl_dg' AS
        PRIMARY DATABASE IS 'orcl' CONNECT IDENTIFIER IS orcl;
DGMGRL> ADD DATABASE 'orcls' AS CONNECT IDENTIFIER IS orcls;
DGMGRL> ENABLE CONFIGURATION;
DGMGRL> SHOW CONFIGURATION;
-- Expect: Configuration Status: SUCCESS, both members SUCCESS.
```

3. Die Konfiguration validieren, alle `WARNING` oder nicht-NULL `ERROR` vor Schritt 3: FSFO-Eigenschaften konfigurieren und aktivieren beheben:

```
DGMGRL> VALIDATE DATABASE 'orcls';
DGMGRL> SHOW CONFIGURATION VERBOSE;
```

Schritt 2: Flashback für FSFO bestätigen

Bestätigt werden muss, dass die Flashback-Datenbank auf beiden Hosts aktiviert ist. Flashback ist für die FSFO Auto-Reinstate-Funktion erforderlich, die es dem ehemaligen primären Host ermöglicht, nach einem Failover automatisch wieder als Standby der Konfiguration beizutreten.

1. Bestätigen `flashback_on` ist `YES`auf beiden Datenbank-Hosts:

```
sudo -u oracle bash -c '
. ~/.bash_profile
sqlplus -s / as sysdba <<<"SELECT flashback_on FROM v\$database;"
'
# Expected on both hosts: YES
```

2. Nur auf dem primären System, falls die Flashback-Aufbewahrung noch nicht festgelegt ist:

```
sudo -u oracle bash -c '
. ~/.bash_profile
export ORACLE_SID=orcl
sqlplus / as sysdba <<SQL
ALTER SYSTEM SET db_flashback_retention_target=1440 SCOPE=BOTH;
EXIT
SQL'
```

Schritt 3: FSFO konfigurieren und aktivieren

SYNC-Redo-Transport einrichten, MaxAvailability Schutzmodus konfigurieren, FSFO-Ziele auf jeder Datenbank definieren und Fast-Start Failover aktivieren.

1. Den Redo-Transportmodus auf `SYNC` für beide Datenbanken festlegen und den Schutzmodus auf `MaxAvailability` erhöhen:

```
DGMGRL> EDIT DATABASE 'orcl'  SET PROPERTY LogXptMode='SYNC';
DGMGRL> EDIT DATABASE 'orcls' SET PROPERTY LogXptMode='SYNC';
DGMGRL> EDIT CONFIGURATION SET PROTECTION MODE AS MaxAvailability;
```

2. Die FSFO-Ziele werden so festgelegt, dass jede Datenbank die andere als Failover-Ziel angibt,

anschließend werden der Schwellenwert und das automatische Wiederherstellungsverhalten konfiguriert:

```
-- Each side names the other
DGMGRL> EDIT DATABASE 'orcl'  SET PROPERTY FastStartFailoverTarget =
'orcls';
DGMGRL> EDIT DATABASE 'orcls' SET PROPERTY FastStartFailoverTarget =
'orcl';

-- 30 s is the default; lower for faster RTO but more sensitive to
network blips
DGMGRL> EDIT CONFIGURATION SET PROPERTY FastStartFailoverThreshold = 30;
DGMGRL> EDIT CONFIGURATION SET PROPERTY FastStartFailoverAutoReinstate =
TRUE;
```

3. Fast-Start-Failover aktivieren und die Konfiguration bestätigen:

```
DGMGRL> ENABLE FAST_START FAILOVER;
DGMGRL> SHOW FAST_START FAILOVER;
-- Expected: Threshold 30 seconds, Target orcls, Observer not yet
registered.
```

Schritt 4: Instant Client auf Observer installieren

Oracle Instant Client auf der dedizierten Observer-VM (`oradg-obs` installieren, einen dedizierten `oracle` OS-Benutzer erstellen und die Oracle Net-Umgebung so konfigurieren, dass der Observer eine Verbindung zu beiden Datenbankmitgliedern über TCP/1521 herstellen kann.

1. Oracle Instant Client Pakete auf dem Observer-Host (`oradg-obs` installieren:

```
# Use -el8 / -el9 if the Observer is on an older OL/RHEL release
sudo dnf install -y oracle-instantclient-release-el10
sudo dnf install -y oracle-instantclient-basic \
                    oracle-instantclient-sqlplus \
                    oracle-instantclient-tools
```

2. Einen dedizierten `oracle`Betriebssystembenutzer erstellen, dem die Wallet und die systemd-Unit gehören:

```
sudo useradd -u 54321 -m oracle
sudo passwd -l oracle
```

3. Die Oracle Net-Umgebung wird konfiguriert und `tnsnames.ora` mit Einträgen für beide Datenbank-Hosts erstellt:

```
sudo mkdir -p /etc/oracle/network/admin
sudo chown -R oracle:oracle /etc/oracle

sudo -u oracle tee /home/oracle/.bash_profile >/dev/null <<'EOF'
export ORACLE_HOME=/usr/lib/oracle/26/client64
export LD_LIBRARY_PATH=$ORACLE_HOME/lib
export PATH=$ORACLE_HOME/bin:$PATH
export TNS_ADMIN=/etc/oracle/network/admin
EOF

# tnsnames.ora — must reach both DB hosts on TCP/1521
sudo tee /etc/oracle/network/admin/tnsnames.ora >/dev/null <<'EOF'
orcl =
  (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = oracdb1)(PORT = 1521))
                (CONNECT_DATA = (SERVER = DEDICATED)(SERVICE_NAME =
orcl)))
orcls =
  (DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = oracdb2)(PORT = 1521))
                (CONNECT_DATA = (SERVER = DEDICATED)(SERVICE_NAME =
orcls)))
EOF
sudo chown oracle:oracle /etc/oracle/network/admin/tnsnames.ora
```

Schritt 5: Observer als systemd Service ausführen

Eine Auto-Login-Wallet mit Anmeldeinformationen für beide Datenbankmitglieder erstellen, anschließend den Observer als systemd-Dienst konfigurieren und starten, sodass er Neustarts übersteht und sich automatisch wieder mit der Konfiguration verbindet.

Anmeldeinformationen für ein dediziertes Data Guard-Administratorkonto (zum Beispiel `SYSDG`) sollten in der Wallet statt in `SYS` gespeichert werden. Anmeldeinformationen dürfen niemals auf einer `dgmgrl` Befehlszeile erscheinen, wo sie für `ps` und `journalctl` sichtbar sind; die Verbindung sollte immer mit `/@<tns_alias>` auf dem Observer hergestellt werden.

1. Die verschlüsselte Wallet wird erstellt und die Anmeldeinformationen für beide Datenbankmitglieder werden hinterlegt:

```
sudo -iu oracle bash <<'BASH'
mkdir -p $TNS_ADMIN/wallet
mkstore -wrl $TNS_ADMIN/wallet -create     # prompts for a wallet
password — store in your secrets manager
mkstore -wrl $TNS_ADMIN/wallet -createCredential orcl  sys ChangeMe!1
mkstore -wrl $TNS_ADMIN/wallet -createCredential orcls sys ChangeMe!1
BASH

sudo tee /etc/oracle/network/admin/sqlnet.ora >/dev/null <<'EOF'
WALLET_LOCATION = (SOURCE = (METHOD = FILE) (METHOD_DATA = (DIRECTORY =
/etc/oracle/network/admin/wallet)))
SQLNET.WALLET_OVERRIDE = TRUE
EOF
sudo chown oracle:oracle /etc/oracle/network/admin/sqlnet.ora
sudo chmod -R 0700 /etc/oracle/network/admin/wallet

sudo -iu oracle ls -l /etc/oracle/network/admin/wallet
# Expected: cwallet.sso and ewallet.p12

sudo -iu oracle bash <<'BASH'
sqlplus -L "/@orcl as sysdba" <<'SQL'
SELECT database_role FROM v$database;
EXIT
SQL
BASH

sudo -iu oracle bash <<'BASH'
sqlplus -L "/@orcls as sysdba" <<'SQL'
SELECT database_role FROM v$database;
EXIT
SQL
BASH

sudo -iu oracle dgmgrl /@orcl  'SHOW CONFIGURATION;'
sudo -iu oracle dgmgrl /@orcls 'SHOW CONFIGURATION;'
```

2. Die Auto-Login-Wallet `cwallet.sso` wird generiert, damit der Observer systemd-Dienst ohne Passwortabfrage starten kann. Wenn `cwallet.sso` nach der Ausführung von `mkstore` fehlt, kann `orapki` aus dem Instant Client Tools-Paket oder einem Datenbank-Home verwendet werden, um sie zu erstellen; anschließend werden die gespeicherten Anmeldeinformationen erneut hinzugefügt.

```
sudo -iu oracle orapki wallet create \
  -wallet /etc/oracle/network/admin/wallet \
  -auto_login
sudo -iu oracle ls -l /etc/oracle/network/admin/wallet
# Expected: cwallet.sso and ewallet.p12
```

3. Die systemd-Unit wird erstellt, der Dienst aktiviert und die Verbindung des Observers überprüft:

```
sudo tee /etc/systemd/system/dgmgrl-observer.service >/dev/null <<'EOF'
[Unit]
Description=Oracle Data Guard Fast-Start Failover Observer
After=network-online.target
Wants=network-online.target

[Service]
Type=simple
User=oracle
Group=oracle
Environment=ORACLE_HOME=/usr/lib/oracle/26/client64
Environment=LD_LIBRARY_PATH=/usr/lib/oracle/26/client64/lib
Environment=TNS_ADMIN=/etc/oracle/network/admin
Environment=PATH=/usr/lib/oracle/26/client64/bin:/usr/bin:/bin
ExecStart=/usr/lib/oracle/26/client64/bin/dgmgrl -silent /@orcl "START
OBSERVER FILE IS '/var/lib/oracle/dgmgrl-observer.dat'"
Restart=always
RestartSec=5

[Install]
WantedBy=multi-user.target
EOF
```

```
sudo install -d -o oracle -g oracle -m 0755 /var/lib/oracle
sudo install -o oracle -g oracle -m 0640 /dev/null /var/log/dgmgrl-
observer.log

sudo tee /etc/logrotate.d/dgmgrl-observer >/dev/null <<'EOF'
/var/log/dgmgrl-observer.log {
    weekly
    rotate 8
    compress delaycompress missingok notifempty
    create 0640 oracle oracle
    copytruncate
}
EOF

sudo systemctl daemon-reload && sudo systemctl enable --now dgmgrl-
observer.service
sudo systemctl status dgmgrl-observer.service
```

Der Observer muss `CONNECTED`vom Primärsystem ablesen (ein `DISCONNECTED`Observer setzt FSFO stillschweigend aus):

```
DGMGRL> SHOW FAST_START FAILOVER;
DGMGRL> SHOW CONFIGURATION;       -- Configuration Status: SUCCESS,
FSFO: ENABLED
```

Schritt 6: FSFO testen

Die Data Guard Konfiguration wird mit `VALIDATE DATABASE` validiert, anschließend erfolgt ein geplanter Switchover und in einem Testfenster ein ungeplanter VM-Reset-Failover, um zu bestätigen, dass FSFO durchgängig funktioniert.

1. Ein geplanter Switchover kann getestet und die ursprüngliche Topologie wiederhergestellt werden:

```
DGMGRL> VALIDATE DATABASE 'orcls';
DGMGRL> SWITCHOVER TO 'orcls';
DGMGRL> SHOW CONFIGURATION;
DGMGRL> SWITCHOVER TO 'orcl';        -- restore topology
```

2. Ein ungeplantes Failover mithilfe eines VM-Resets in einem kontrollierten Testfenster testen:

   Ein VM-Reset (Absturztest) kann verwendet werden; ein normaler Stopp löst möglicherweise keinen FSFO aus. Mit Tail `/var/log/dgmgrl-observer.log` auf `oradg-obs` lässt sich der Failover-Fortschritt überwachen; nach Abschluss kann die Topologie wiederhergestellt werden.

Was kommt als Nächstes?

Die Konfiguration für Oracle Data Guard Broker, Fast-Start Failover und Observer ist nun für diese Bereitstellung eingerichtet.