使用 Oracle HA 部署 Google Cloud NetApp Volumes

NetApp database solutions

NetApp
June 25, 2026

目录

使用 Oracle HA 部署 Google Cloud NetApp Volumes

为 Google Cloud NetApp Volumes 预置 Google Compute Engine 实例

提供 Google Compute Engine 虚拟机，以在 Google Cloud NetApp Volumes iSCSI 存储上托管 Oracle Database 26ai。此过程包括创建主备用数据库主机和 Fast-Start Failover Observer VM、为 Oracle Net 配置 VPC 防火墙规则、设置主机名解析、准备操作系统以及捕获用于 GCNV 存储配置的 iSCSI 启动程序名称。

步骤 1：创建 VM

在同一区域的不同可用区中创建三个 Google Compute Engine VM，以实现可用区故障隔离。使用 Cloud Console、 `gcloud`、Terraform 或您的标准配置工作流程。

1. 使用下表所示的规格创建三个虚拟机。

   在满足延迟和合规性需求的前提下，优先选择低碳区域以降低总拥有成本并提升可持续性（例如 `us-west1` 与 `us-central1`）：

VM	分区	机器类型	启动磁盘	网络	目的
`oracdb1`	`us-west1-a`	`n4-highmem-8` (sample) 或 `c4-standard-*`	OL 10，50 GB Hyperdisk Balanced（仅限操作系统）	`oracle-vpc` / `oracle-subnet`，gVNIC	主 DB
`oracdb2`	`us-west1-b`	与主体相同	OL 10，50 GB Hyperdisk Balanced（仅限操作系统）	相同	备用数据库
`oradg-obs`	`us-west1-c`	`e2-medium`	OL 10，20 GB Hyperdisk Balanced	相同	FSFO Observer（仅限 Instant Client）

   当延迟或出口（>~200 GiB/月）很重要时，使用 Premium 网络层；在开发/测试中使用 Standard 以降低 TCO。

2. 启用 Shielded VM 功能并验证启动磁盘配置：

   在所有三个虚拟机上启用 **Secure Boot**、**vTPM** 和 **Integrity Monitoring**。

   启动磁盘仅容纳操作系统。 `/u01`、Grid/DB 主目录、暂存和所有 ASM 数据均使用 GCNV iSCSI 卷（请参阅配置 GCNV iSCSI 卷）

   请勿为 `/u01`附加单独的 GCE 数据磁盘。

步骤 2：为 TCP 1521 配置 VPC 防火墙

创建 VPC 防火墙规则，以允许所有三个虚拟机之间的 TCP/1521 进行 Oracle Net redo 传输和 Observer 连接。缺少规则会中断 Data Guard 复制。

1. 创建 VPC 防火墙入口规则，以允许所有三个 VM 内部 IP 之间使用 TCP/1521。使用具有相同允许列表的 VPC 防火墙规则或防火墙策略：

   Cloud Console: VPC 网络 → 防火墙 → 创建规则 `allow-oracle-net-dbhosts` on `oracle-vpc` — 入口，允许，源 = 三个 `/32` IP，TCP 1521。如果需要，镜像出口。

2. 验证每个虚拟机的连接，以确认防火墙规则是否已就位：

   ```
   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
   ```

端口	预期	含义
`22`	已连接	SSH 路径有效
`1521`	连接被拒绝	防火墙打开；Grid 侦听器在 步骤 1：在每个 DB 主机上安装 Oracle Grid Infrastructure (Oracle Restart) 期间启动
要么	超时	修复防火墙或路由

   从所有三个虚拟机向每个对等 IP 运行。

步骤 3：配置主机名、DNS 和 `/etc/hosts`

在所有三个虚拟机上配置主机名和 DNS 解析，以便正向和反向名称解析适用于 Oracle Net、Data Guard Broker 和 Observer。

1. 在所有三台主机上设置主机名并添加 `/etc/hosts`条目。替换 GCE 内部 IP 地址（在 **Compute Engine** → **VM instances** 列表中可见，*Internal IP* 列）：

```
# 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. 验证每个主机的名称解析：

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

步骤 4：仅在数据库主机上准备操作系统

通过安装预安装软件包、创建用户和组、安装 iSCSI 和多路径软件包以及配置 iSCSI 启动程序，在 `oracdb1`和 `oracdb2`上为 Oracle Database 26ai 准备操作系统。观察者设置包含在步骤 4：在 Observer 主机上安装 Oracle Instant Client中。

先决条件： 出站 HTTPS 到 `yum.oracle.com`（私有子网上的 Cloud NAT 或内部镜像）。

1. 安装 Oracle 数据库预安装包，创建 `grid`用户和 ASM 组，并将 `oracle`用户添加到 ASM 组：

```
# 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. 安装 iSCSI、多路径和 JDK 包，然后验证 THP 和时间同步：

```
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. 配置 SELinux、防火墙和 iSCSI 启动程序设置，然后重新启动：

> 安全态势 **(OL 10):** 以下命令将 SELinux 设置为允许模式并禁用 `firewalld`。这仅适用于最小化实验室环境。有关强化的 SELinux 和防火墙配置，请参阅您组织的安全基准。

```
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
```

第 5 步：捕获 iSCSI 启动程序名称 IQN

重新启动后，从每个数据库主机捕获 iSCSI 启动程序名称 (IQN)。您将使用这些 IQN 在 步骤 2：创建主机组 中创建 GCNV 主机组。

1. 从 `oracdb1`捕获 IQN 并进行记录：

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

2. 重复 `oracdb2`并记录其 IQN。每个主机使用一个主机组，以便单个主机的重新启动或 IQN 重新生成不会影响另一个主机的 GCNV iSCSI 卷可见性：

> 克隆的 **VM**： 如果两台主机共享相同的 IQN，请重新生成 `oracdb2`（停止 `iscsi`，清除 `/var/lib/iscsi/nodes/*`，新建 `InitiatorName` `/etc/iscsi/initiatorname.iscsi`，重新启动 `iscsid`）。

下一步是什么?

要为 Oracle 二进制文件和 ASM 磁盘组提供共享存储，请转到 预配 Google Cloud NetApp Volumes iSCSI 池、主机组和卷。

为 Oracle Database 26ai 预配 Google Cloud NetApp Volumes iSCSI 存储

在 Google Compute Engine 上为 Oracle Database 26ai 高可用性配置 Google Cloud NetApp Volumes iSCSI 块存储。此过程包括创建 GCNV Flex Unified 存储池、定义主机组、为每个数据库主机创建 iSCSI 卷、配置 Linux iSCSI 和多路径、对 ASM 后备设备进行分区以及挂载 `/u01`文件系统。

步骤 1：创建 GCNV iSCSI 池

创建两个 Flex Unified 存储池，每个数据库区域一个，为主主机和备用主机提供 iSCSI 卷。每个数据库主机都使用其本地区域池中的卷。

1. 使用 Cloud Console 创建两个存储池。使用下表中的规格，并为每个区域重复创建过程：

池名称	分区	使用者
`oracle-pool-a`	`us-west1-a`	`oracdb1` (主要)
`oracle-pool-b`	`us-west1-b`	`oracdb2`（待机）

NetApp Volumes → **Storage pools** → **Create** 为每个池：

- 服务级别： Flex（非 Premium）
- 类型： 统一
- 区域： 匹配数据库 VM 区域 (`us-west1-a` / `us-west1-b`)
- **PSA**： 连接到 `oracle-vpc`
- *容量：*针对工作负载进行大小调整；当重做、备份或还原超过默认余量时，使用自定义配置的吞吐量/IOPS（每个池高达 5120 MiB/s 或 160K IOPS，每个产品限制）

2. 请等待两个池都达到 `READY`状态后再继续。将池大小调整为与数据库占用空间匹配（步骤 3：创建 GCNV iSCSI 卷中的大小仅为示例）：

> ⓘ 默认模式（本指南）： Flex 统一池使用默认模式 (`--mode=default`）。使用 Cloud Console 或 `gcloud netapp` 创建池和 iSCSI 卷。卷复制、快照和克隆使用 Google Cloud API （步骤 3：GCNV 待机初始化）。

步骤 2：创建主机组

为每个数据库主机创建一个主机组，以便每个虚拟机只能看到自己的卷。主节点和备用主机不得共享 GCNV iSCSI 卷，以维护独立存储。

1. 使用 Cloud Console 为 `oracdb1`创建主机组：

   NetApp 卷 → 主机组 → 创建

   - 名称： `oracdb1-hg`
   - 区域： `us-west1`
   - 类型： iSCSI 启动程序
   - **OS** 类型: Linux
   - 主机： 粘贴来自 `oracdb1` 的 IQN（ `/etc/iscsi/initiatorname.iscsi` 的值）
   - 描述： "Oracle 主主机 oracdb1"
   - 创建

2. 对 `oracdb2`重复此过程，使用名称 `oracdb2-hg`和 `oracdb2` 的 IQN。Observer 主机不需要 GCNV 资源。

步骤 3：创建 GCNV iSCSI 卷

为每个数据库主机创建五个 GCNV iSCSI 卷：一个用于 `/u01`，四个用于 ASM 后备设备。每个主机的卷必须在其本地区域的存储池中使用其对应的主机组创建。

1. 在 `oracle-pool-a` 中为 `oracdb1` 创建五个卷，主机组为 `oracdb1-hg`。请使用下表中的规范：

GCNV iSCSI 卷	大小	用途	多路径别名
`ora_<host>_u01`	100 GiB	`/u01` GCNV iSCSI 卷 — 网格/Oracle 主目录，暂存	`/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` (条纹)	`/dev/mapper/ora_<host>_data_02`
`ora_<host>_arch_01`	100 GiB	ASM `+RECO`	`/dev/mapper/ora_<host>_arch_01`
`ora_<host>_fra_01`	100 GiB	ASM `+FRA`	`/dev/mapper/ora_<host>_fra_01`

   卷名：仅限字母、数字、下划线（无连字符）。

   *最小布局（仅验证）：*每个主机两个 LUN (`*_data`, `*_reco`) with `arch_01p1`→`+RECO` and `arch_01p2`→`+FRA` 对于实验室是可接受的；生产环境每个 步骤 3：创建 GCNV iSCSI 卷 使用五个卷。

2. 使用相同的规范为 `oracdb2`中 `oracle-pool-b`与主机组 `oracdb2-hg`创建五个卷。对于每个池，使用 **NetApp Volumes** → **Volumes** → **Create** — iSCSI，正确的池和主机组，Linux。记录以下信息：

   - iSCSI 门户 IP → `<ISCSI_PORTAL_1>`， `<ISCSI_PORTAL_2>`（主池门户位于 `oracdb1`；备用池门户位于 `oracdb2`——它们可能不同）

- 来自 Cloud Console 的卷序列 — 与主机发现的 WWID 一起使用 步骤 4：为 GCNV iSCSI 卷配置 Linux iSCSI 和多路径

步骤 4：配置 iSCSI 和多路径

在每个数据库主机上配置 iSCSI 和 device-mapper-multipath，以通过两个存储门户 IP 访问 GCNV 卷。在 `oracdb1`上使用主池的门户 IP 运行这些步骤，然后在 `oracdb2`上使用备用池的门户 IP 重复这些步骤。如果主机出口受到限制，请允许 TCP/3260 从每个数据库虚拟机到其 GCNV iSCSI 门户 IP（除了来自步骤 2：VPC 防火墙 — 跨所有三个区域的允许列表 TCP/1521的虚拟机间 TCP/1521）。

1. 发现目标、登录并持久化节点启动：

```
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
```

   重新启动后，在启动 Oracle 之前重新检查：

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

2. 使用默认值和黑名单规则配置 `device-mapper-multipath`：

```
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. 将主机发现的 WWID 别名添加到 `/etc/multipath.conf`（不要猜测— `multipath.conf` *不*展开 shell 变量）。发现 WWID：

```
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
```

将该主机的具体别名附加到 `/etc/multipath.conf`，然后 `sudo systemctl restart multipathd`。

在 `oracdb1`上，追加：

```
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  }
}
```

在 `oracdb2`上，对 `ora_oracdb2_*`别名使用相同的模式，然后：

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

第 5 步：对 ASM 设备进行分区

对四个 ASM 后备设备（不包括 u01）进行分区，每个设备配置一个 GPT 分区供 ASM 使用，然后为 grid 所有权配置 udev 规则。在每个数据库主机上运行这些步骤。

1. 使用 GPT 对四个 ASM 备用设备进行分区，并验证分区：

```
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. 配置 udev 规则以分配 grid 所有权并触发更改：

```
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
```

第6步：格式化和挂载 /u01

使用 XFS 格式化 ora_<host>_u01 GCNV 卷，并使用 `/etc/fstab`中的 UUID 将其永久挂载。 `/u01`文件系统保存 Grid home、Oracle home 和暂存文件。

1. 使用 XFS 格式化多路径设备并捕获其 UUID：

```
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. 将基于 UUID 的挂载条目添加到 `/etc/fstab`并挂载文件系统：

```
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. 为 Grid 和 Oracle 软件创建具有适当所有权的目录结构：

```
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
```

重新启动一次并确认 `/u01` 挂载在 安装 Oracle 软件 之前。

下一步是什么?

要在准备好的主机上安装 Oracle Grid Infrastructure 和 Database 二进制文件，请转到两台主机上的 安装 Oracle Grid Infrastructure 和 Oracle Database 软件。

在 Google Cloud NetApp Volumes 上安装 Oracle Grid Infrastructure 和 Oracle Database 26ai

在每个数据库主机的 Google Cloud NetApp Volumes iSCSI 存储上安装 Oracle Grid Infrastructure with Oracle Restart 和 ASM，然后安装 Oracle Database 26ai 软件。此过程包括暂存 Oracle GoldImages，使用响应文件运行静默安装，在 GCNV 卷上创建 ASM 磁盘组，以及在创建数据库之前使用相同的 Oracle 软件准备主机和备用主机。

步骤 1：在每个数据库主机上安装 Grid Infrastructure

在每个数据库主机上安装 Oracle Grid Infrastructure GoldImage，以启用 Oracle Restart 和 ASM。两台主机都需要各自的 Grid home、ASM 实例和磁盘组；Data Guard 通过 Oracle Net 复制数据，而不是通过共享存储。在 `oracdb1`上完成所有步骤，然后再在 `oracdb2`上重复操作。

1. 将 Oracle GoldImages、Release Update 和 OPatch 二进制文件暂存到 `/u01/stage`：

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

2. 在目标 Grid 主目录中就地解压缩 Grid GoldImage。26ai GoldImage 通过直接解压缩到目标目录来完成安装：

```
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
```

如果 Grid GoldImage 比目标 RU 旧，请在设置过程中使用 `gridSetup.sh -applyRU`流程修补 Grid 主目录，

或使用捆绑了 RU 的 GoldImage。将 Grid 和 Database 主目录保持在相同的预期补丁级别。

3. 在每台主机上构建 `gridSetup`响应文件 `/tmp/grid.rsp`。替换主机名并使用强密码：

```
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. 以静默模式运行 `gridSetup.sh`以复制二进制文件并暂存配置。Expect `Successfully Setup Software with warning(s).`和退出代码 6（警告）或 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. 以 root 身份运行 `orainstRoot.sh`和 `root.sh`。`root.sh`脚本创建 `crsctl`、`srvctl`和 `asmcmd`包装器，并启动 OHAS：

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

6. 运行 `gridSetup.sh -executeConfigTools`以针对响应文件运行配置助理（NETCA、ASMCA、CVU）。这将创建 ASM 实例和 `+DATA`磁盘组。NETCA / ASMCA / CVU 之后的预期 `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. 使用 asmca 创建 +RECO 和 +FRA 磁盘组。单步安装仅创建 +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. 验证 ASM 磁盘组和 Oracle Restart 资源状态：

```
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. 在 oracdb2 上重复上述步骤。 HOST=$(hostname -s) 和 步骤 3 和 4 中的 步骤 7 模式会自动选择该主机的 GCNV iSCSI 设备。

使用相同的 ASM 磁盘组名称 — Data Guard 通过 Oracle Net 而不是存储进行复制。

步骤 2：在每个数据库主机上安装 Oracle 数据库

使用静默纯软件安装方式在每个数据库主机上安装 Oracle Database 26ai 软件主目录，并应用最新的版本更新。在 `oracdb1`上完成所有步骤后，再在 `oracdb2`上重复执行。

1. 将数据库主页、最新的 OPatch 和 RU 修补程序解压缩到各自的目录中。有关 RU 目录布局和 `-applyRU` 路径，请参阅 Oracle 文档：

```
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. 编写安装响应文件，并在应用 RU 的情况下运行静默仅软件安装。在 OL 8/9 上，从 `runInstaller`行中省略 `-applyOneOffs`：

```
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. 运行安装后根脚本：

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

4. 在每个数据库主机上设置 Oracle 环境。在 `ORACLE_SID=orcl` 上使用 `oracdb1`，在 `ORACLE_SID=orcls` 上使用 `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
```

备用数据库在 创建备用数据库 中创建。

下一步是什么?

要为 HA 部署创建生产主实例，请转到 创建 Oracle 主数据库 on `oracdb1`。

在 Google Cloud NetApp Volumes 上创建 Oracle 主数据库

在静默模式下使用 Oracle Database Configuration Assistant 在 Google Cloud NetApp Volumes iSCSI 存储上创建 Oracle 主数据库。此过程包括在 GCNV 支持的 ASM 磁盘组上运行 `dbca`以创建容器数据库和可插拔数据库，配置归档日志目标，以及在启用 Data Guard 后添加基于角色的应用程序服务以实现透明故障转移。

步骤

在 `oracdb1`上使用 `dbca`以静默模式创建 Oracle 容器数据库和可插拔数据库，配置归档日志目标，验证 Oracle Restart 注册，并为透明客户端故障转移添加基于角色的应用程序服务。

1. 在静默模式下运行 `dbca`以在 ASM 磁盘组上创建 CDB 和 PDB：

```
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. 将归档日志指向 `+RECO`并打开并保存可插拔数据库状态。备用数据库在 步骤 2：备用 init.ora、pfile 和 NOMOUNT 中使用匹配的归档日志设置：

```
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. 验证数据库是否正在 Oracle Restart 下运行：

```
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. 创建基于角色的应用程序服务，以便应用程序通过 `orclapp`进行连接，并且在启用 Data Guard 时故障转移是透明的：

```
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
'
```

启用 Data Guard Broker 后， `orclapp` 仅在 PRIMARY 上运行。跨 ASM 磁盘组多路复用控制文件，并根据工作负载调整内存大小。

下一步是什么?

要建立备用保护和故障转移准备，请转到 创建 Oracle 备用数据库 on `oracdb2`。

使用 Google Cloud NetApp Volumes 存储层种子设定创建 Oracle 备用数据库

使用 Google Cloud NetApp Volumes 存储层复制、快照或克隆创建 Oracle 物理备用数据库，与传统的 RMAN 方法相比，这可以加快备用数据库的初始化。此步骤包括配置监听程序、创建备用 pfile、使用 GCNV 复制植入备用卷、最终确定 Oracle 实例以及向 Oracle Restart 注册备用数据库。所有 HA 层级都需要完成这些步骤。对于 **Prod HA (Data Guard + FSFO)** 层级，请在配置 Data Guard Broker、Fast-Start Failover 和 Observer 之前继续执行 Data Guard 最终确定。

步骤 1：配置侦听器和 Data Guard 参数

在两个数据库主机上配置侦听器以支持 Data Guard 连接，包括代理所需的 `_DGMGRL`服务。设置密码文件并在主数据库上配置归档日志参数。

1. 在 `oracdb1`上配置主侦听器并验证环境：

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

2. 将备用侦听器配置 `oracdb2`为包括 `orcls`和 `orcls_DGMGRL`服务：

```
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. 通过两台主机上的 Oracle Restart 重新启动监听器，并验证 `_DGMGRL` 服务是否已注册：

```
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` 必须列出 `<SID>` 和 `<SID>_DGMGRL`。

步骤 2：准备备用 pfile 和 NOMOUNT

通过从主数据库复制密码文件、使用 Data Guard 参数创建最小 init.ora pfile 并以 NOMOUNT 模式启动实例来准备备用数据库实例。

1. 使用 IAP 和 `gcloud compute scp`将主密码文件复制到备用主机：

```
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. 从主数据库查询 `compatible`参数值：

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

3. 在 `oracdb2` 上创建备用 pfile，设置密码文件的所有权，并在 NOMOUNT 模式下启动实例。将上一步中的 `compatible` 值替换为 `<COPY_FROM_PRIMARY>`：

```
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
'
```

备用实例现在处于 NOMOUNT 模式，在 步骤 3：使用 GCNV 初始化备用存储 之前没有数据文件。

步骤 3：使用 GCNV 初始化备用存储

在 `oracle-pool-b`中植入备用卷，将其连接到 `oracdb2`，挂载 ASM 磁盘组，并在 MOUNT 状态下完成备用实例的配置。

使用 GCNV 复制进行生产播种，并使用快照播种进行一次性实验室工作流程。

选择种子路径

根据您的环境和恢复要求选择备用播种方法。

- 推荐用于生产： 使用 复制路径：创建和同步复制 和 复制路径：切换并连接备用卷 中的复制路径。
- 实验室替代方案： 使用 备用路径：来自 Snapshot 的种子。

所有路径在 挂载备用 ASM 磁盘组 和 完成备用实例 处重新汇合。

检查先决条件

在播种备用卷之前，请确认以下先决条件。

- `gcloud netapp` 支持卷复制。
- 两个 **Default-mode** 池位于不同位置 (`oracle-pool-a`, `oracle-pool-b`)。
- 连接到 `oracdb1-hg`的主池上的源卷；通过复制创建的目标卷。
- 从 Cloud Shell 或工作站运行复制，而不是从 DB VM。
- 在 `oracdb2`上，从 第 4 步、第 5 步 和 第 6 步 完成 iSCSI 和 ASM 主机设置。

```
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"
```

- 如有需要，创建备用池：

```
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>
```

创建和同步复制

创建从主卷到备用卷的复制关系，然后等待初始同步完成。

```
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"
```

+

等待，直到 `mirrorState`为 `MIRRORED`，且每个复制的初始同步完成。

切换并连接备用卷

使主卷静默，在最终同步后停止复制，并将目标卷附加到备用主机组。

在主节点上，暂停写入并捕获恢复元数据：

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

允许最后一个复制周期，然后停止复制：

```
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
```

将目标卷附加到 `oracdb2-hg`（复制的 LUN 可能保留源名称）：

```
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"
```

将备用控制文件复制到 `oracdb2`，然后结束主数据库的备份模式：

```
ALTER DATABASE END BACKUP;
```

从 **Snapshot** 创建种子

当不需要持续复制时，请使用此路径进行一次性实验室数据初始化。

对于一次性实验室种子，请创建源快照，并在 `oracle-pool-b`（Cloud Console 或 API）中从该快照创建备用卷。将创建的卷附加到 `oracdb2-hg`，然后继续挂载备用 ASM 磁盘组。

挂载备用 **ASM** 磁盘组

在备用主机上，在数据库恢复前发现连接的存储路径并挂载 ASM 磁盘组。

在 `oracdb2`上，登录到备用池 `iSCSI` 门户并重新扫描多路径设备。如果 `ASM` 磁盘标头与实验室工作流程中的主命名匹配，请使用主样式别名（例如 `ora_oracdb1_data_01`， `ora_oracdb1_arch_01`），设置 `asm_diskstring='/dev/mapper/ora_oracdb1_*p*'`，并确认分区所有权为 `grid:asmadmin`，然后挂载磁盘组：

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

完成备用实例

还原备用控制文件，恢复到捕获的 SCN，转换为物理备用，并启动托管恢复。

```
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;
```

此时，待机状态应为 `PHYSICAL STANDBY`并且 `MOUNTED`已启动托管恢复。

特定层级的后续步骤：

- **Prod HA（无 Data Guard）：** 直接继续 步骤 4：向 Oracle Restart 注册备用数据库。
- **Prod HA (Data Guard + FSFO):** 继续步骤 4：向 Oracle Restart 注册备用数据库，然后继续Data Guard 最终确定步骤。

步骤 4：向 Oracle Restart 注册备用数据库

向 Oracle Restart 注册备用数据库，以便重新启动会自动恢复 ASM 磁盘组、挂载备用数据库并重新启动托管恢复。还将应用程序服务添加到两个数据库资源。

1. 从备用数据库捕获 spfile 位置，并将其注册到 Oracle Restart `oracdb2`。替换查询中的 `<STANDBY_SPFILE_PATH>`（通常在 `+DATA`下）：

```
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. 验证并更新 `oracdb1` 上的主数据库资源，以包括所有 ASM 磁盘组依赖项：

```
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. 将应用服务添加到备用数据库资源 (`orcls` 上 `oracdb2`）。在两侧使用 `role PRIMARY`，以便切换后 `orclapp` 可用：

```
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. 验证以下位置的备用数据库资源 `oracdb2`：

```
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
'
```

下一步是什么?

特定层级：

- **Prod HA（无 Data Guard）：** 要维护基于存储复制的恢复目标，备用初始化已完成，备用数据库已向 Oracle Restart 注册为备份实例。
- **Prod HA (Data Guard + FSFO):** 要启用代理管理的切换和快速启动故障转移，请继续 完成 Data Guard 的备用数据库。

完成 Google Cloud NetApp Volumes 上 Data Guard 的备用数据库

通过创建备用重做日志文件、启用闪回数据库、激活重做传送和验证 Data Guard 状态，最终完成 Google Cloud NetApp Volumes 上 Oracle Data Guard 备用数据库的配置。

特定于层级： 此过程仅适用于 **Prod HA (Data Guard + FSFO)** 层级。

步骤 1：创建备用重做日志文件

在两个数据库主机上创建备用重做日志文件，以支持 Fast-Start Failover。大小必须大于或等于最大的主在线重做日志，且计数应等于（每个线程的在线组数）+ 1。完成 GCNV 播种后，在备用数据库上删除并重新创建备用重做日志，以修复复制的路径。

1. 在主数据库上创建备用重做日志文件(`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. 在 GCNV 播种后，在备用数据库上删除并重新创建备用重做日志文件 (`orcls`）。在 `+DATA/ORCL/…` 下的复制路径会导致 `ORA-19527` / `ORA-16086`，直到重建为止：

```
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;
```

步骤 2：启用闪回并开始恢复

在备用数据库上启用闪回数据库，以支持故障转移后的自动恢复，然后使用实时应用启动托管恢复。必须在启动托管恢复之前启用闪回，因为在 MRP 处于活动状态时无法启用闪回。

1. 关闭备用数据库，在 MOUNT 模式下重新启动，并在以下位置启用闪回数据库 `oracdb2`：

```
# 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. 使用实时应用启动托管恢复：

```
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` 启用实时应用（重做在到达 SRL 时应用）。

步骤 3：启用 redo shipping

通过激活 `LOG_ARCHIVE_DEST_STATE_2` 启用从主库到备库的 redo 传输，该参数在备库初始化步骤的 第 2 步 中被刻意设置为 `DEFER`，以抑制备库创建过程中的 `ORA-12154` 错误。

1. 切换 `LOG_ARCHIVE_DEST_STATE_2`到 `ENABLE`并强制日志切换以启动重做日志传送：

```
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. 验证 redo shipping 是否正常工作：

```
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.
```

如果 `dest_2`显示 `ORA-12154`，则弹跳主节点。在步骤 1：在两个数据库上启用 broker之后，通过 DGMGRL 管理传输。

步骤 4：验证 Data Guard 状态

验证主数据库是否处于 READ WRITE 模式，以及备用数据库是否已通过应用重做日志的托管恢复进行挂载。

1. 验证主数据库角色和打开模式（位于 `oracdb1`）：

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

2. 验证备用数据库角色、打开模式和托管恢复状态（位于 `oracdb2`）：

```
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
```

待机时应为： `PHYSICAL STANDBY | MOUNTED`; `MRP0` with `APPLYING_LOG`。

3. 如果备用节点报告 `MOUNTED`但 apply 未运行，请在 `oracdb2`上重新启动托管恢复：

```
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'
```

下一步是什么?

要激活自动角色管理和故障转移保护，请继续 配置 Oracle Data Guard Broker、快速启动故障转移和 Observer。

在 Google Cloud NetApp Volumes 上为 Oracle Database 26ai 配置 Data Guard Broker 和 Fast-Start Failover

使用专用 Observer 配置 Oracle Data Guard Broker 和 Fast-Start Failover，以在 Google Cloud NetApp Volumes 上启用 Oracle Database 26ai 的自动角色转换。

特定于层级： 此过程仅适用于 **Prod HA**（**Data Guard + FSFO**） 层级。

此过程包括在两个数据库上启用代理，创建 Data Guard 配置，启用具有 MaxAvailability 保护模式的 FSFO，在 Observer 主机上安装 Oracle Instant Client，使用基于钱包的凭据将 Observer 作为 systemd 服务启动，以及测试切换和故障转移。之后 `ENABLE CONFIGURATION`，通过 **DGMGRL**（而非临时 `LOG_ARCHIVE_DEST_*` SQL）管理传输和角色。

步骤 1：启用 Data Guard Broker

在两个数据库主机上启用 Data Guard Broker，并创建在统一管理下链接主数据库和备用数据库的 Broker 配置。

1. 在主数据库主机和备用数据库主机上设置 `dg_broker_start=TRUE`：

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

2. 在主服务器上，使用 OS 身份验证连接到 DGMGRL 并创建 Broker 配置：

> ⓘ 仅在 Observer 主机上，在自动登录钱包存在后使用 `dgmgrl /@orcl`。请勿将密码放在 `dgmgrl`命令行上。

```
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. 验证配置 — 在 步骤 3：配置 FSFO 属性并启用 之前修复任何 `WARNING` 或非 NULL `ERROR`：

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

步骤 2：确认 FSFO 的闪回

确认在两台主机上都启用了 flashback 数据库。FSFO 自动恢复需要闪回功能，这允许前主服务器在故障转移后自动作为备用节点重新加入配置。

1. 确认 `flashback_on`在两个数据库主机上 `YES`：

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

2. 仅在主设备上，如果尚未设置闪回保留：

```
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'
```

步骤 3：配置并启用 FSFO

设置 SYNC 重做传输，配置MaxAvailability保护模式，在每个数据库上定义 FSFO 目标，并启用 Fast-Start Failover。

1. 在两个数据库上将重做传输模式设置为 `SYNC`，并将保护模式提升为 `MaxAvailability`：

```
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. 将 FSFO 目标设置为每个数据库将另一个命名为其故障转移目标，然后配置阈值和自动恢复行为：

```
-- 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. 启用快速启动故障转移并确认配置：

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

步骤 4：在 Observer 上安装 Instant Client

在专用 Observer VM 上安装 Oracle Instant Client (`oradg-obs`），创建专用 `oracle` OS 用户，并配置 Oracle Net 环境，以便 Observer 可以连接到 TCP/1521 上的两个数据库成员。

1. 在 Observer 主机上安装 Oracle Instant Client 软件包 (`oradg-obs`)：

```
# 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. 创建一个专用的 `oracle` OS 用户，该用户将拥有钱包和 systemd 单元：

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

3. 配置 Oracle Net 环境并创建 `tnsnames.ora`，其中包含两个数据库主机的条目：

```
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
```

第5步：将 Observer 作为 systemd 服务运行

使用两个数据库成员的凭据创建一个自动登录钱包，然后将 Observer 配置并启动为 systemd 服务，以便它在重启后仍能正常运行并自动重新连接到配置。

将专用 Data Guard 管理帐户（例如 `SYSDG`）的凭据存储在钱包中，而不是 `SYS`。凭据绝不能出现在 `dgmgrl`命令行上，因为它们对 `ps`和 `journalctl`可见；请始终在 Observer 上使用 `/@<tns_alias>`进行连接。

1. 创建加密钱包并填充两个数据库成员的凭据：

```
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. 生成自动登录钱包 `cwallet.sso`），以便 Observer systemd 服务可以在没有密码提示的情况下启动。如果在运行 `mkstore` 后缺少 `cwallet.sso`，请使用 Instant Client 工具包或数据库主目录中的 `orapki` 来创建它，然后重新添加存储的凭据：

```
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. 创建systemd单元，启用服务，并验证Observer是否已连接：

```
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
```

观察者必须从主节点读取 `CONNECTED`（a `DISCONNECTED` 观察者静默暂停 FSFO）：

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

第6步：测试FSFO

使用 `VALIDATE DATABASE`验证 Data Guard 配置，然后执行计划的切换，并在测试窗口中执行计划外的 VM 重置故障转移，以确认 FSFO 是否端到端正常运行。

1. 测试计划的切换并恢复原始拓扑：

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

2. 在受控测试窗口中使用虚拟机重置来测试计划外故障转移：

   使用虚拟机*重置*（崩溃式测试）；正常的*停止*可能不会触发 FSFO。跟踪 `/var/log/dgmgrl-observer.log` on `oradg-obs`以监控故障转移进度；完成后恢复拓扑。

下一步是什么?

Oracle Data Guard Broker、Fast-Start Failover 和 Observer 配置现已用于此部署。