Oracle Linux Virtualization Manager Database Templates Deployment Guide

Updated
14 March 2023

Contents

About this document

This document shows you how to deploy the Oracle Database Templates for Oracle Linux KVM managed by Oracle Linux Virtualization Manager (OLVM).

Conventions

Template file names look similar to:

```
OLVM_OL7U9_19110DBRAC_KVM-1of2  & 2of2
OLVM_OL8U4_19110DBRAC_KVM-1of2  & 2of2
```

The screenshots in this document are for example purposes and the templates you use might have slightly different names.

Feedback

Feel free to post feedback on the following Oracle forums or contact Oracle Support:

- Oracle Linux Virtualisation Manager Forum
- Oracle RAC

About Oracle Database Templates

Oracle provides templates (pre-configured, pre-built virtual machines) that are fully ready to deploy to a virtualized Oracle Linux Virtualization Manager environment. A typical template includes a guest operating system, database software, and configuration needed for deployment. The Oracle Database Templates for Oracle Linux KVM/OLVM help deploy a virtualized environment in a few short clicks and minutes.

> **IMPORTANT**
> **This guide explains template basics and manual deployment. Please refer to Deploycluster for OLVM documentation for automated deployments!**

An Oracle Database template for Oracle Linux KVM/OLVM consists of single OVA file (inside a zip file) that contains two virtual disks:

- An Oracle Linux 7.9 or Oracle Linux 8.4 operating system
- An Oracle Database version (19c, 21c, etc.)

> **NOTE**
> It should be possible to “Mix-Match” (OS and Oracle disks) versions that are not pre-bundled in same OVA.

Similar database templates are offered for **Oracle Cloud Infrastructure** and **Oracle VM**:

- Oracle Cloud Infrastructure Marketplace
- Oracle VM

Prerequisites

Before deploying an Oracle Database Template for Oracle Linux KVM/OLVM, the Oracle Linux Virtualization Manager must be updated to Release 4.3.6.6-1.0.15 or higher.

Deployment

The following are deployment scenarios with their individual requirements.

- **Single Instance (w/o ASM or RAC)**
  - A single network adapter (vNIC) and no shared disks.
  - The database must reside on a filesystem.
  - No `cloud-init` script is required for default deployment; supply to override defaults.

- **Single Instance/HA Deployment** (Oracle Restart)
  - A single network adapter (vNIC).
  - One disk (by default, more disks are possible) to configure ASM.
  - A simple `cloud-init` script is **required** at initial virtual machine launch.

- **Oracle Real Application Clusters (RAC) Deployment**
  - Each virtual machine requires 2 (or more) network adapters (vNICs).
  - One common shared disk if ASM configured (recommended more disks as per RAID/Storage)
  - For supported Oracle RAC deployment usage of the Deploycluster tool for OLVM is required.

Deployment Overview (Manual, Single Instance)

The following list provides a high-level overview of the deployment steps.

1) Download the desired version of the Database Template to a *staging area* on the KVM Server host
2) From Oracle Linux Virtualization Manager UI, import the staged Database Template.
3) *Optionally* create or identify the (shared) disks to hold the Database.
4) *Optionally* edit the properties for the imported template (OVA file), such as
   - Memory/CPU
   - Additional/Correct virtual vNICs mappings
   - SSH Keys
   - cloud-init script
5) Create a virtual machine using the Database Template.
6) *Optionally* edit the properties for the virtual machine just created, such as
   a. Memory/CPU
   b. Additional/Correct Virtual NICs
   c. SSH Keys
   d. Add (shared) disks (SIHA, RAC)
   e. cloud-init script
7) Launch the virtual machine.

NOTE
For RAC deployments it is required to use the Deploycluster tool for OLVM.

Deploying the Template

Apart from downloading the Database Template, use the Oracle Linux Virtualization Manager User Interface for all other steps.

> **NOTE**
> Fully automated deployment is possible using the Deploycluster tool for OLVM; Refer to tool's documentation for further details.

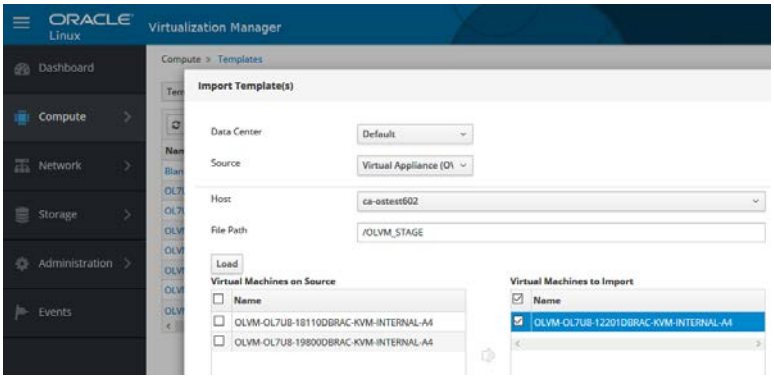
Download the Database Template

Each template consists of a single OVA file (inside a zip file) that holds **two disks**: operating system disk and Oracle disk.

1. Refer to https://www.oracle.com/database/technologies/rac/vm-db-templates.html for download location/versions/instructions
2. Unzip and stage the OVA file on the KVM host from which it will be imported.

Import the Database Template

1. From the Oracle Linux Virtualization Manager UI navigate to **Compute->Templates** and then click **Import**.
2. From the **Import Template(s)** window, do the following:
    a. From the **Source** list, select **OVA**.
    b. From the **Host** list, select the KVM host where you staged the template.
    c. In the **File Path** field, enter directory or full path to the OVA.
    d. Click **Load**.
    e. From the **Virtual Machines on Source** list, select the desired template and click the right arrow to move the template to the **Virtual Machines to Import** list.
    f. Click **Next**.
    g. Click **OK**. The import process starts.



Create or Identify the Disks for the Database

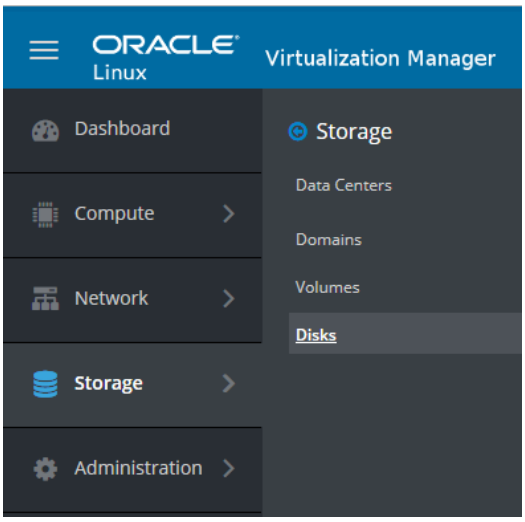
Based on the deployment mode (Single Instance, RAC, clusterware only, etc), use Oracle Linux Virtualization Manager to create or identify the (shared) disks where the database will reside.

- For a **RAC Deployment** refer to Deploycluster tool for OLVM for details and automated deployment options.

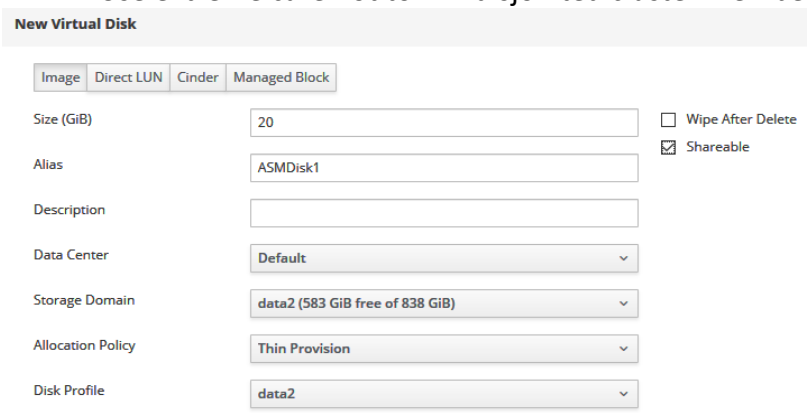
- For a **Single Instance/HA Deployment** (Oracle Restart) **One** (by default, or more) disk is needed to configure ASM.
- For a **Single Instance** No extra disks are needed, hence this step may be skipped. The database must reside on a filesystem.

To create a (shared) disk:

1. From the Oracle Linux Virtualization Manager UI, navigate to **Storage->Disks**- and select **New…**.



2. From the **New Virtual Disk** window, enter the following:
   - Set a **Size (GiB)**.
   - Enter an **Alias** (name).
   - For **Allocation Policy**, select **Thin Provision** for test environments. Higher-end environments should consider using **Preallocated**.
   - Click the **Shareable** box (required if RAC used, or use Deploycluster tool to automate this).
     IMPORTANT
     Use extreme care not to mix disjointed cluster members to avoid corruptions.



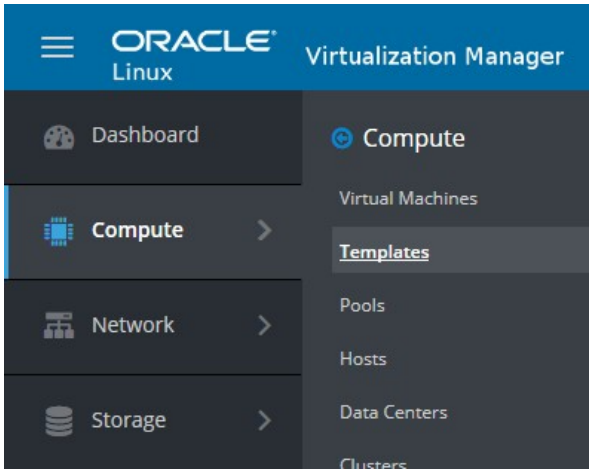
3. Click **OK**.

(*Optional*) Edit the Imported Template

Before creating virtual machines from the newly-imported template, adjust the properties to suit your environment and deployment.

To edit an imported template:

1. From the Oracle Linux Virtualization Manager UI, navigate to **Compute->Templates**-and select **Edit…**.



2. From the **Edit Template (s)** window, locate the template you imported and click **Edit**.
3. Expand the **Show Advanced Options** and review or make changes to the following items:
    - Memory/CPU
    - Add or correct virtual NICs
    - SSH Keys
    - cloud-init script
    - High-Availability => Resume Behavior => Kill
4. Click **OK**.

IMPORTANT

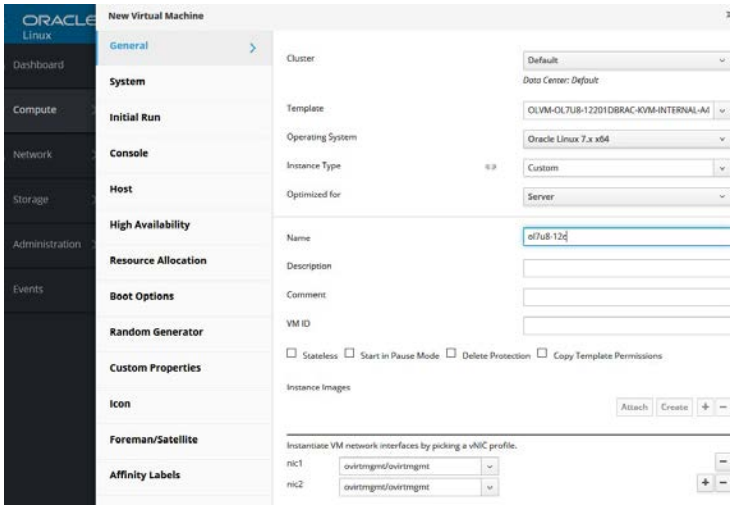
- Single Instance and Single Instance/HA require only one vNIC and optionally support more.
- Oracle RAC deployment requires 2 vNICs.
- The automation allows for a 3rd NIC for the ASM traffic. (See netconfig.txt.)

Create a Virtual Machines Using the Database Template

From this newly imported and optionally edited template, clone as many virtual machines as needed for single instance or for Oracle RAC, use the Deploycluster tool.

To create virtual machines:

1. From the Oracle Linux Virtualization Manager UI, navigate to **Compute->Templates**-and select **New VM.**



2. From the **New Virtual Machine** window, expand the **Show Advanced Options** and review or make changes to the following items:
    - Memory/CPU
    - Add or correct virtual NICs
    - SSH Keys
    - cloud-init script
    - High-Availability => Resume Behavior => Kill
3. Click **OK**.

Edit a Virtual Machine

It is still possible to make changes to the virtual machine before launching. Once it is created, from the Virtual Machines list, do the following:

1. Highlight the virtual machine you just created. It shows as *powered off* in the list.
2. Click **Edit**.
3. From the **Edit Virtual Machine** window, expand the **Show Advanced Options** and review or make changes to the following items:
    - Memory/CPU
    - Add or correct virtual NICs
    - SSH Keys
    - Add (shared) disks (SIHA, RAC)
    - cloud-init script
    - High-Availability => Resume Behavior => Kill
4. Click **OK**.

Launch the Virtual Machine

To launch the newly created virtual machine:

1. From the Oracle Linux Virtualization Manager UI, navigate to **Compute->Virtual Machines**-and select the newly-created virtual machine.
2. Click **Run**.
   The virtual machine is powered on and any optional cloud-init deployment scripts are applied.

3. Once the virtual machine is finished booting, it should obtain an IP address. You can login using SSH and the public key you used when deploying the host.

Using Cloud-init in an Oracle Linux Virtualization Manager Environment

Here are a few things to keep in mind when using cloud-init in Oracle Linux Virtualization Environment:

- To query what script was passed to an already running virtual machine, run the following command from within the virtual machine
  `# cloud-init query userdata`
- If there is already a cloud-init script or text in the field box, leave it and paste your script below. This mean they will run in sequence.
- To customize the database deployment, most of the `params.ini` variable can be passed on the same line as `buildsingle.sh` in the cloud-init script.
- Unlike Oracle Cloud Infrastructure that supports cloud-init with both *bash directive* as well as *cloud-config directive*, the Oracle Linux Virtualization Manager environment supports only *cloud-config directive* in yaml syntax.

It is possible to save the *cloud-init* script on the template (after it is imported) or on the virtual machine created from the template, but before the virtual machine is started.

The following examples use **yaml** syntax.

To auto-configure SIHA/Oracle Restart:

1. Attach an extra ASM disk using “VirtIO-SCSI” (not “VirtIO”) of the desired size to the virtual machine. This means there are now three (3) disks: operating system, Oracle software, ASM.
2. Enter the following *cloud-init* script in the “Initial Run” section under “Custom Script”, in the Oracle Linux Virtualization Manager UI:

```
runcmd:
- mount /u01
- CLONE_SINGLEINSTANCE_HA=yes /u01/racovm/GenerateNetconfig.sh -a
- SIDNAME=siha DBNAME=siha /u01/racovm/buildsingle.sh -s
```

   - Changes to `SIDNAME` or `DBNAME` are optional.
   - The mount `/u01` may be optional as well, however, to remove any doubt and avoid a timing-related issue, always mount `/u01` as first statement.
   - To control the minimum number of ASM disks or their exact names, prepend `ASM_MIN_DISKS=1 ALLDISKS=/dev/sda RACASMDISKSTRING=/dev/sd?` on the `buildsingle.sh` line leaving at least a single space between each argument.

To auto-configure a single instance with a:

- Non-default listener port number
- Non-default SID name
- Non-default DB name
- Pre-loaded sample schema in the created database

Use the following cloud-init script:

```
runcmd:
- mount /u01
- /u01/racovm/GenerateNetconfig.sh -a
- SIDNAME=test DBNAME=test LISTENERPORT=1522 DBCA_SAMPLE_SCHEMA=yes
/u01/racovm/buildsingle.sh -s
```

Notice, all the variable settings must be on the same line as the `buildsingle.sh`.

To avoid the automated database build during initial startup, use the following cloud-init script:

```
runcmd:
- mount /u01
- ls /u01/racovm/
```

Anytime the firstboot logic detects any reference to the `/u01/racovm` folder, it avoids an automated build and allows the cloud-init custom user script to take control. In this case only `ls` is executed, hence the automated database build is skipped. This is useful for manual RAC case, where for some reason Deploycluster cannot be used or debugging.

Create a netconfig.ini File for Deployment

For a single Instance it isn't necessary to create a netconfig.ini file since it is created with the automated deployment. For example: `/u01/racovm/GenerateNetconfig.sh -a`

Here is an example netconfig.ini file for a **Single Instance** deployment.

```
# Sample Single Instance or Single Instance/HA
NODE1=test1
NODE1IP=192.168.1.101
#NODE1PRIV=test1-priv       # Optional
#NODE1PRIVIP=10.10.10.101   # Optional

# Common data
PUBADAP=eth0
PUBMASK=255.255.255.0
PUBGW=192.168.1.1
#PRIVADAP=eth1              # Optional
#PRIVMASK=255.255.255.0     # Optional
DOMAINNAME=localdomain      # May be blank
#DNSIP=    # Starting from 2013 Templates allows multi value

# Single Instance (description in params.ini)
CLONE_SINGLEINSTANCE=yes     # For Single Instance
#CLONE_SINGLEINSTANCE_HA=yes # For Single Instance/HA
```

For Oracle RAC please use the Deploycluster tool for simpler automation, for manual Oracle RAC deployment, use a simple text editor to copy the sample `netconfig-sample64.ini` file to netconfig.ini, and then adjust the names and IPs to suit the environment.

Here is an example netconfig.ini file for a **2-node Oracle RAC cluster** deployment. Make sure there are no duplicate values (IP, names).

```
# Node specific information
NODE1=test30
NODE1VIP=test30-vip
NODE1PRIV=test30-priv
NODE1IP=192.168.1.30
NODE1VIPIP=192.168.1.32
NODE1PRIVIP=10.10.10.30
NODE2=test31
NODE2VIP=test31-vip
NODE2PRIV=test31-priv
NODE2IP=192.168.1.31
NODE2VIPIP=192.168.1.33
NODE2PRIVIP=10.10.10.31
# Common data
PUBADAP=eth0
PUBMASK=255.255.255.0
PUBGW=192.168.1.1
PRIVADAP=eth1
PRIVMASK=255.255.255.0
RACCLUSTERNAME=twonodes30
DOMAINNAME=localdomain     # May be blank
DNSIP=       # Starting from 2013 Templates allows multi value
# RAC specific data
SCANNAME=test30-31-scan
SCANIP=192.168.1.34
```

Example default specifications for a 2-node cluster

- SID ORCL1 and ORCL2
- Database name ORCL
- Grid Infrastructure Home /u01/app/19c/grid
- Oracle RAC Home /u01/app/oracle/product/19c/dbhome_1
- ORACLE_BASE /u01/app/oracle
- Central Inventory /u01/app/oraInventory

Modifying Build Options

By default, the Single Instance or Oracle RAC Cluster build options reside in **/u01/racovm/params.ini** on the virtual machine. You can modify build options by using a properly created cloud-init script that is attached to the template or the deployed virtual machine, or simply use the Deploycluster tool for full automation! Modifying these build options allows full control over things like the database name, SID name, port numbers, etc. You must ensure that the options and values set that are not default in params.ini file match the virtual machines that will use it.

For example, if you set:

```
ALLDISKS="/dev/sda /dev/sdb /dev/sdc /dev/sdd /dev/sde /dev/sdf"
```

The virtual machines must have these 6 device names, sda, sdb, sdc, sdd, sde & sdf. If they don't, then buildcluster fails. To recover from such a failure, either correct params.ini and run buildsingle.sh or

buildcluster.sh manually, or clean all virtual machines as described in the FAQ, adjust their shared disks as needed, and then re-deploy using correct settings.

APPENDIX A – Build Options

Before invoking **/u01/racovm/buildcluster.sh** (or buildsingle.sh for Single Instance) you can edit **/u01/racovm/params.ini** to modify some build options (bottom part of the file). The top part of params.ini should only be modified by advanced users or if instructed to by Oracle Support.

Examples of the options that can be modified are:

```
# Build Database? The BUILD_RAC_DATABASE will build a RAC database and
# BUILD_SI_DATABASE a single instance database (also in a RAC environment)
# Default: yes
BUILD_RAC_DATABASE=yes
#BUILD_SI_DATABASE=yes

# The Database Name
# Default: ORCL
DBNAME=ORCL

#
# The Instance name, may be different than database name. Limited in length of
# 1 to 8 for a RAC DB & 1 to 12 for Single Instance DB of alphanumeric characters.
# Ignored for Policy Managed DB.
# Default: ORCL
SIDNAME=ORCL

# Configures a Single Instance environment, including a database as
# specified in BUILD_SI_DATABASE. In this mode, no Clusterware or ASM will be
# configured, hence all related parameters (e.g. ALLDISKS) are not relevant.
# The database must reside on a filesystem.
# This parameter may be placed in netconfig.ini for simpler deployment.
# Default: no
#CLONE_SINGLEINSTANCE=no

# Configures a Single Instance/HA environment, aka Oracle Restart, including
# a database as specified in BUILD_SI_DATABASE. The database may reside in
# ASM (if RACASMGROUPNAME is defined), or on a filesystem.
# This parameter may be placed in netconfig.ini for simpler deployment.
# Default: no
#CLONE_SINGLEINSTANCE_HA=no

# Local Listener port number (default 1521)
# Default: 1521
LISTENERPORT=1521

# Allows color coding of log messages, errors (red), warning (yellow),
# info (green). By default no colors are used.
# Default: NO
CLONE_LOGWITH_COLORS=no
```

If you do not wish to store the passwords for the root or Oracle user in the configuration file, remove or comment them, and they will be prompted for at the start of the build.

APPENDIX B – Troubleshooting and Frequently Asked Questions

1) Tried a simple POWER UP and deployment worked! However custom deployment kept failing.

The best way to deploy the templates is using the Deploycluster tool for OLVM. It supports fully automated deployments. To debug a failed manual deployment see following question.

2) If the virtual machines pass basic checks and were started successfully and still the deployment is unsuccessful for any reason, where should I look?

Start looking in the virtual machines at the following log files:

/u01/racovm/buildcluster.log or **buildsingle.log**
only on the build node, first (hub node) virtual machine listed

/var/log/clout-init-output.log
Look for template/netconfig related errors.

/var/log/messages
Look for "Oracle DB/RAC Template" messages

For example, a typical failure could be non-empty disks used for ASM. As a safeguard buildcluster will fail if existing ASM data is seen on the disks with messages similar to

```
INFO (node:test3): Specified disk (/dev/sda) in ALLDISKS that will
automatically be partitioned and renamed to (/dev/sda1) appears to be an
active ASM disk: DATA_0000 Failgroup: DATA_0000 in Diskgroup: DATA
```

In this specific case, the corrective action would be to make sure the disks are the correct ones, and then clear them using any of the following methods

 # /u01/racovm/racovm.sh -S cleanlocal (run from first node)
OR
 # /u01/racovm/cleanlocal.sh -X (run from first node)
OR
 # /u01/racovm/diskconfig.sh -X (run from first node)
OR
 # /u01/racovm/racovm.sh -S clean (run from any node; first node must be up)

WARNING
The above commands are **destructive** in that they wipe any data written to the (shared) disks, as well as the 'clean' ones also remove any installed RAC & Grid Infrastructure software from the local or all nodes -- use with caution!

Other common failures include specifying disks in params.ini that do not exist in the VM, or the RACASMDISKSTRING is set to a value which does not discover all disks on all nodes.

Additionally, specifying IP addresses that are already taken or wrong subnet mask for the network may result in failures. Follow the cleanup procedure described in the following question and re-attempt the deployment.

3) Deploy attempt failed, how do I cleanup and start again?

It is typically enough to correct the error (clear data disk from former ASM headers, or change conflicting IP), then simply re-run **buildcluster.sh** or **buildsingle.sh.**

If however a complete clean of the virtual machine is desired, follow the steps below for Manual re-deployment:

```
# cd /u01/racovm/
# ./racovm.sh -S setsshroot,clean
# rm –f netconfig.ini
# cd patches
# cp -a oracle-default-db.service /etc/systemd/system
# restorecon /etc/systemd/system/oracle-default-db.service
# systemctl enable oracle-default-db.service
# cloud-init clean --logs --reboot
```

The last statement reboots into a new deployment as if the template was booted for the first time! This is valid for Single Instance, Manual deployments (not using Deploycluster tool).

It is also possible to remove the –reboot flag and power off the virtual machine. Then, in the Oracle Linux Manager UI, select RUN-ONCE for the virtual machine instead of RUN. The RUN-ONCE boots the virtual machine as if it booted for the first time. Be sure to enable the Initial Run/cloud-init script and, if needed, edit the cloud-init script or the SSH keys.

Cleanup for Automated re-deployment using Deploycluster tool as well as Oracle RAC:

```
# cd /u01/racovm/
# ./racovm.sh -S setsshroot,copykit,clean
# ./doall.sh –sp rm –rf /etc/cloud/cloud-init.disabled
# ./doall.sh cloud-init clean --logs
# ./doall.sh –L last init 0
```

Now, rerun the deploycluster tool (without the –clone* options since the VM will be re-used). For Single Instance cleanup run the same commands without “./doall.sh ..” and it’s flags.

APPENDIX C – References

Oracle Database Templates Homepage:
https://www.oracle.com/database/technologies/rac/vm-db-templates.html
Oracle VM Templates for Oracle Database - Single Instance, Oracle Restart (SIHA) and Oracle RAC (Doc ID 1185244.1)
https://support.oracle.com/knowledge/Oracle%20Cloud/1185244_1.html