Oracle Database 12*c*: High Availability New Features

Activity Guide

D79794GC10
Edition 1.0
July 2013
D82518

ORACLE

Authors

Peter Fusek, Anupama Mandya, Mark Fuller, Jim Womack

Technical Contributors and Reviewers

Andrey Gusev, Larry Carpenter, Dominique Jeunot, Harendra Mishra, Janet Stern, Jim Williams, Joel Goodman, John McHugh, Jonathan Creighton, Mark Scardina, Markus Michalewicz, Prasad, Bagal, Raj Kammend, Rick Wessman, Subhransu Basu, Harald van Breederode, Sean Kim, Douglas Williams, Branislav Valny, Allan Graves

This book was published using: Oracle Tutor

Table of Contents

Practices for Lesson 1: Introduction

Chapter 1

Practices for Lesson 1: Overview

Practices Overview

In this practice, you will familiarize yourself with the laboratory environment for this course.

Practice 1-1: Laboratory Introduction

Overview

In this practice, you will familiarize yourself with the laboratory environment for this course.

Tasks

Access to your laboratory environment will be through a graphical display running on your classroom workstation or hosted on a remote machine. Your instructor will provide you with instructions to access your practice environment.

The practice environment for this course is hosted on a server running Oracle Virtual Machine (OVM). In turn, OVM hosts numerous virtual machines (VMs). Each VM is a logically separate server that will be used to run Oracle Database 12*c* software, including Clusterware, Automatic Storage Management (ASM) and Real Application Clusters (RAC).

1. Open a terminal window and execute the command `sudo /usr/sbin/xm list`. You should see output similar to the following example. It shows that your server is hosting five domains. `Domain-0` is the OVM server. The other domains relate to four VMs, which you will use to form a cluster in the following practices.

```
$ sudo /usr/sbin/xm list
Name                        ID   Mem VCPUs      State   Time(s)
Domain-0                     0  1024     2     r-----  54609.2
c01n01                       1  5300     1     -b----     98.0
c01n02                       2  5300     1     -b----     97.2
c01n03                       3  1600     1     -b----     96.9
c01n04                       4  1600     1     -b----     96.1
$
```

   Notice that `c01n01` and `c01n02` contain more memory (5,300 MB) than `c01n03` and `c01n04` (1,600 MB).

2. Using SSH, connect to `c01n01` as the `root` OS user. Enter `oracle` when you are prompted for the password.

   Note that you may see additional messages relating to server identities. Answer `yes` if you are prompted to acknowledge server authenticity.

```
$ ssh root@c01n01
root@c01n01's password: <oracle>
[root@c01n01 ~]#
```

3. Your environment has been pre-configured with a series of shared disk devices that will be used by Oracle Clusterware and ASM. Execute the following command to identify the shared disks.

```
[root@c01n01 ~]# ls -l /dev/xvd* | grep grid
brw-rw---- 1 grid asmadmin 202, 81 Dec  6 06:50 /dev/xvdf1
brw-rw---- 1 grid asmadmin 202, 90 Dec  6 06:50 /dev/xvdf10
brw-rw---- 1 grid asmadmin 202, 91 Dec  6 06:50 /dev/xvdf11
brw-rw---- 1 grid asmadmin 202, 92 Dec  6 06:50 /dev/xvdf12
brw-rw---- 1 grid asmadmin 202, 82 Dec  6 06:50 /dev/xvdf2
```

```
brw-rw---- 1 grid asmadmin 202, 83 Dec  6 06:50 /dev/xvdf3
brw-rw---- 1 grid asmadmin 202, 85 Dec  6 06:50 /dev/xvdf5
brw-rw---- 1 grid asmadmin 202, 86 Dec  6 06:50 /dev/xvdf6
brw-rw---- 1 grid asmadmin 202, 87 Dec  6 06:50 /dev/xvdf7
brw-rw---- 1 grid asmadmin 202, 88 Dec  6 06:50 /dev/xvdf8
brw-rw---- 1 grid asmadmin 202, 89 Dec  6 06:50 /dev/xvdf9
brw-rw---- 1 grid asmadmin 202, 97 Dec  6 06:50 /dev/xvdg1
brw-rw---- 1 grid asmadmin 202, 98 Dec  6 06:50 /dev/xvdg2
brw-rw---- 1 grid asmadmin 202, 99 Dec  6 06:50 /dev/xvdg3
[root@c01n01 ~]#
```

4. Examine `c01n02` to confirm that the shared disk devices are also visible on that node.

```
[root@c01n01 ~]# ssh c01n02 "ls -l /dev/xvd* | grep grid"
brw-rw---- 1 grid asmadmin 202, 81 Dec  6 06:50 /dev/xvdf1
brw-rw---- 1 grid asmadmin 202, 90 Dec  6 06:50 /dev/xvdf10
brw-rw---- 1 grid asmadmin 202, 91 Dec  6 06:50 /dev/xvdf11
brw-rw---- 1 grid asmadmin 202, 92 Dec  6 06:50 /dev/xvdf12
brw-rw---- 1 grid asmadmin 202, 82 Dec  6 06:50 /dev/xvdf2
brw-rw---- 1 grid asmadmin 202, 83 Dec  6 06:50 /dev/xvdf3
brw-rw---- 1 grid asmadmin 202, 85 Dec  6 06:50 /dev/xvdf5
brw-rw---- 1 grid asmadmin 202, 86 Dec  6 06:50 /dev/xvdf6
brw-rw---- 1 grid asmadmin 202, 87 Dec  6 06:50 /dev/xvdf7
brw-rw---- 1 grid asmadmin 202, 88 Dec  6 06:50 /dev/xvdf8
brw-rw---- 1 grid asmadmin 202, 89 Dec  6 06:50 /dev/xvdf9
brw-rw---- 1 grid asmadmin 202, 97 Dec  6 06:50 /dev/xvdg1
brw-rw---- 1 grid asmadmin 202, 98 Dec  6 06:50 /dev/xvdg2
brw-rw---- 1 grid asmadmin 202, 99 Dec  6 06:50 /dev/xvdg3
[root@c01n01 ~]#
```

5. Examine `c01n03` and `c01n04` as well. Notice that the devices are not available on these nodes. You will see how these nodes can participate in the cluster in the following practices.

```
[root@c01n01 ~]# ssh c01n03 "ls -l /dev/xvd* | grep grid"
[root@c01n01 ~]# ssh c01n04 "ls -l /dev/xvd* | grep grid"
[root@c01n01 ~]#
```

So far, you have examined the cluster nodes and identified the two main differences (the amount of system memory and the availability of shared storage) between them.

6. The nodes have all been preconfigured with the requirements for installing Oracle Database 12*c*. This includes the OS user accounts required for a role-separated installation. Using `id`, examine the `grid` user account and take note of the OS groups that are associated with it.

```
[root@c01n01 ~]$ id grid
uid=54322(grid) gid=54321(oinstall)
groups=54321(oinstall),54323(asmadmin),54324(asmdba),54325(asmop
er)
[root@c01n01 ~]$
```

7. Using `id`, examine the `oracle` user account and identify the OS groups associated with it.

```
[root@c01n01 ~]$ id oracle
uid=54321(oracle) gid=54321(oinstall)
groups=54321(oinstall),54322(dba),54324(asmdba),54326(oper)
[root@c01n01 ~]$
```

   The `root`, `grid`, and `oracle` OS user accounts have all been preconfigured to enable passwordless SSH between the cluster nodes. For the `grid` and `oracle` users, this is required to install Oracle Database 12*c*. For the `root` user, this configuration simplifies some of the practice tasks (as you have already seen in steps 4 and 5 above).

8. Using `su`, assume the identity of the `oracle` user and confirm the configuration of passwordless SSH for the `oracle` OS user.

```
[root@c01n01 ~]# su - oracle
[oracle@c01n01 ~]$ ssh c01n02
[oracle@c01n02 ~]$
```

9. Exit your terminal session.

```
[oracle@c01n02 ~]$ exit
logout
Connection to c01n02 closed.
[oracle@c01n01 ~]$ exit
logout
[root@c01n01 ~]# exit
logout
Connection to c01n01 closed.
$
```

   Note: Each of the following practice exercises will instruct you to commence one or more terminal sessions by using different user accounts. Ensure that you always start a new terminal session when instructed to do so, and exit all of your terminal sessions at the end of each practice.

   Congratulations! You have now completed the laboratory introduction.

Practices for Lesson 2: Flex Clusters

Chapter 2

Practices for Lesson 2: Overview

Practices Overview

In these practices, you will:

- Configure a new Flex Cluster with Flex ASM and RAC
- Configure highly available application resources on Flex Cluster Leaf Nodes

Practice 2-1: Configuring a New Flex Cluster with Flex ASM and RAC

Overview

In this practice, you will install and configure a new Flex Cluster with Flex ASM and a RAC database.

Tasks

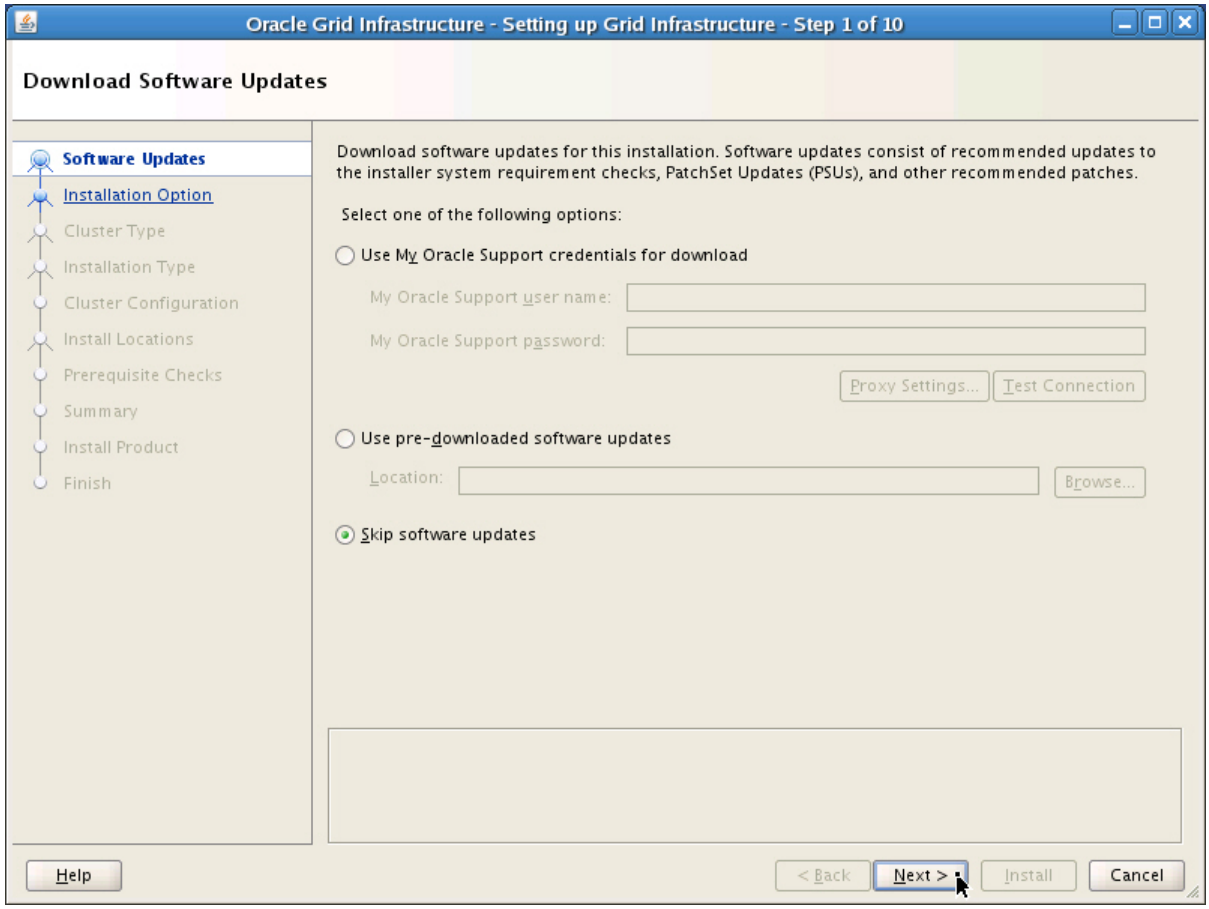
1. Establish a terminal session connected to `c01n01` by using the `grid` OS user. Ensure that you specify the `-X` option for `ssh`.

```
$ ssh -X grid@c01n01
grid@c01n01's password: <oracle>
[grid@c01n01 ~]$
```

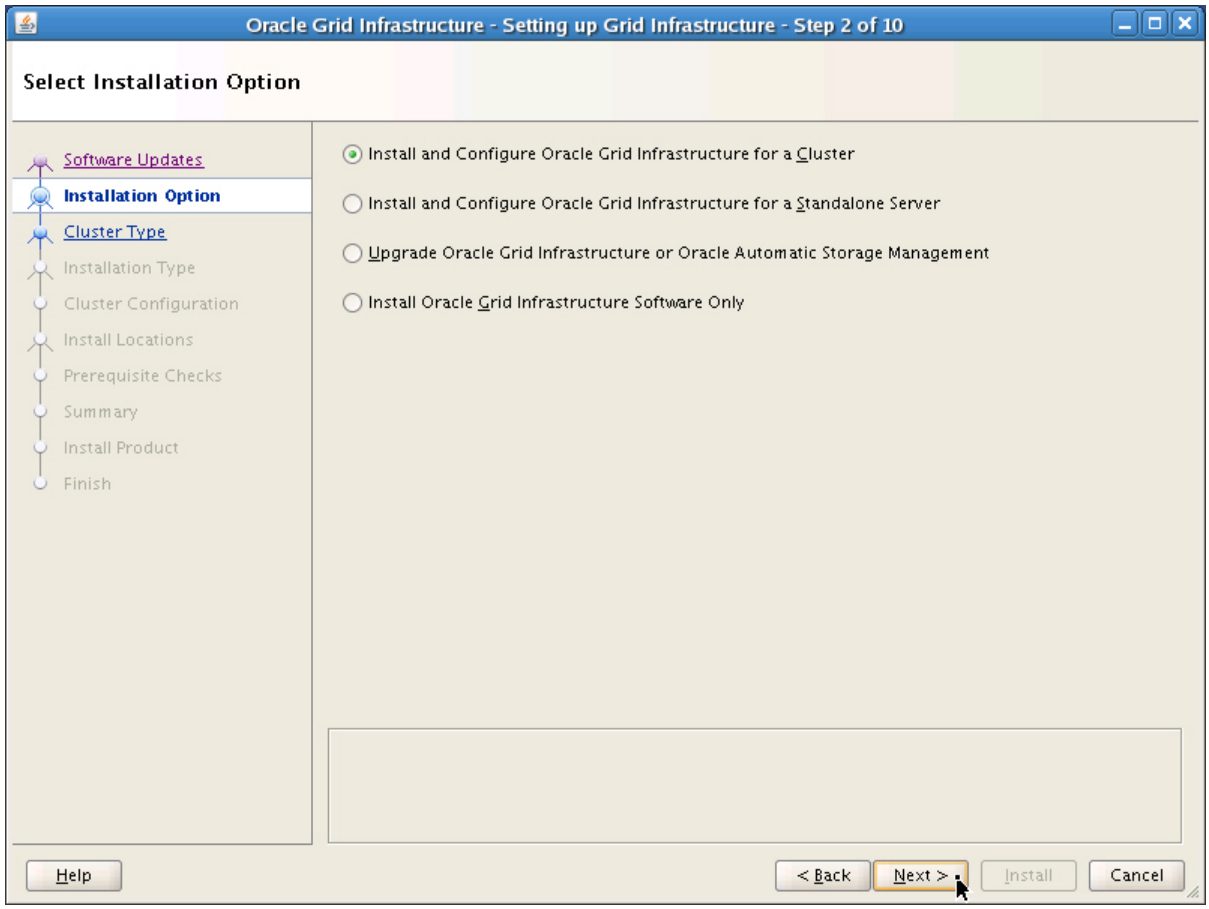
2. Start the Oracle Clusterware release 12.1 installer.

```
[grid@c01n01 ~]$ /stage/12.1/clusterware/runInstaller
Starting Oracle Universal Installer...
```

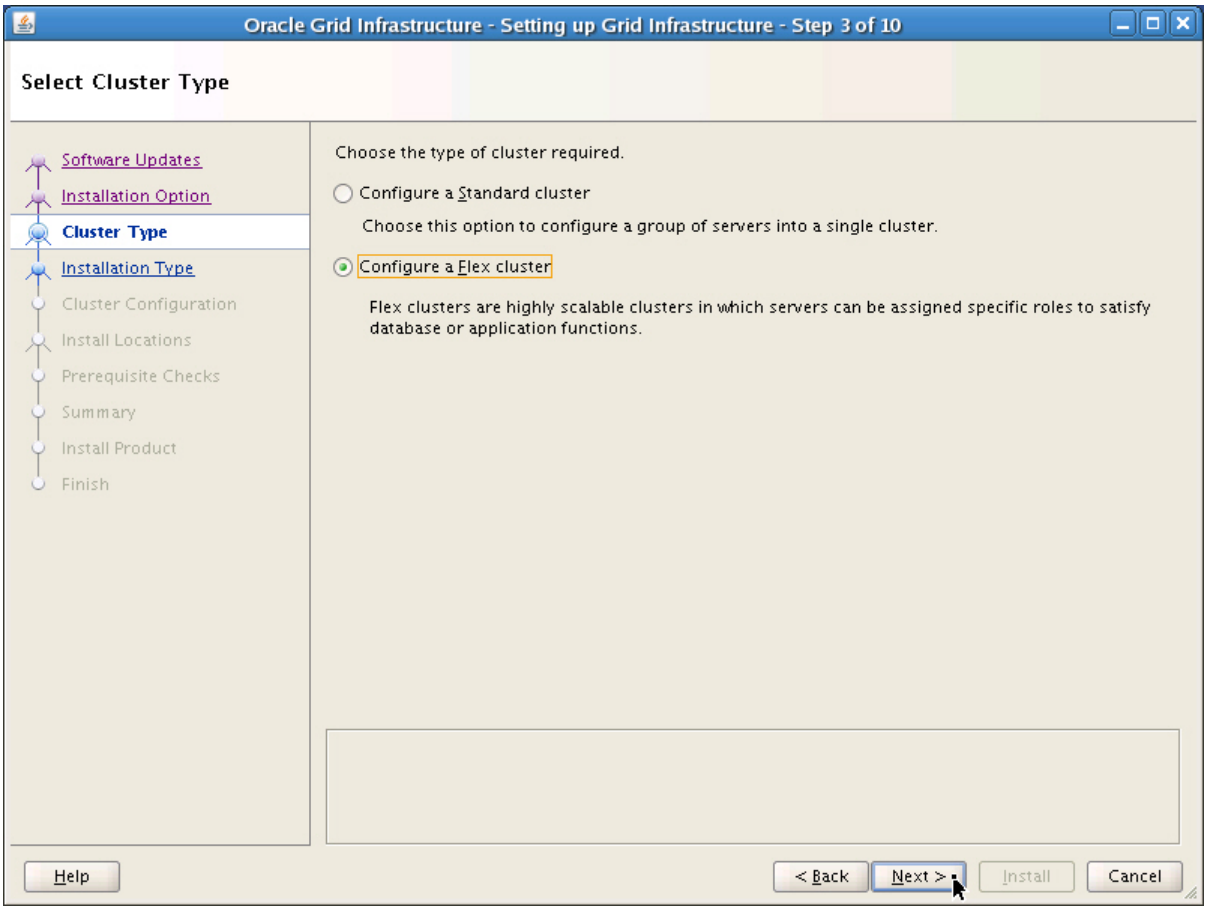
3. On the Download Software Updates screen, click Next to accept the default selection (Skip software updates).



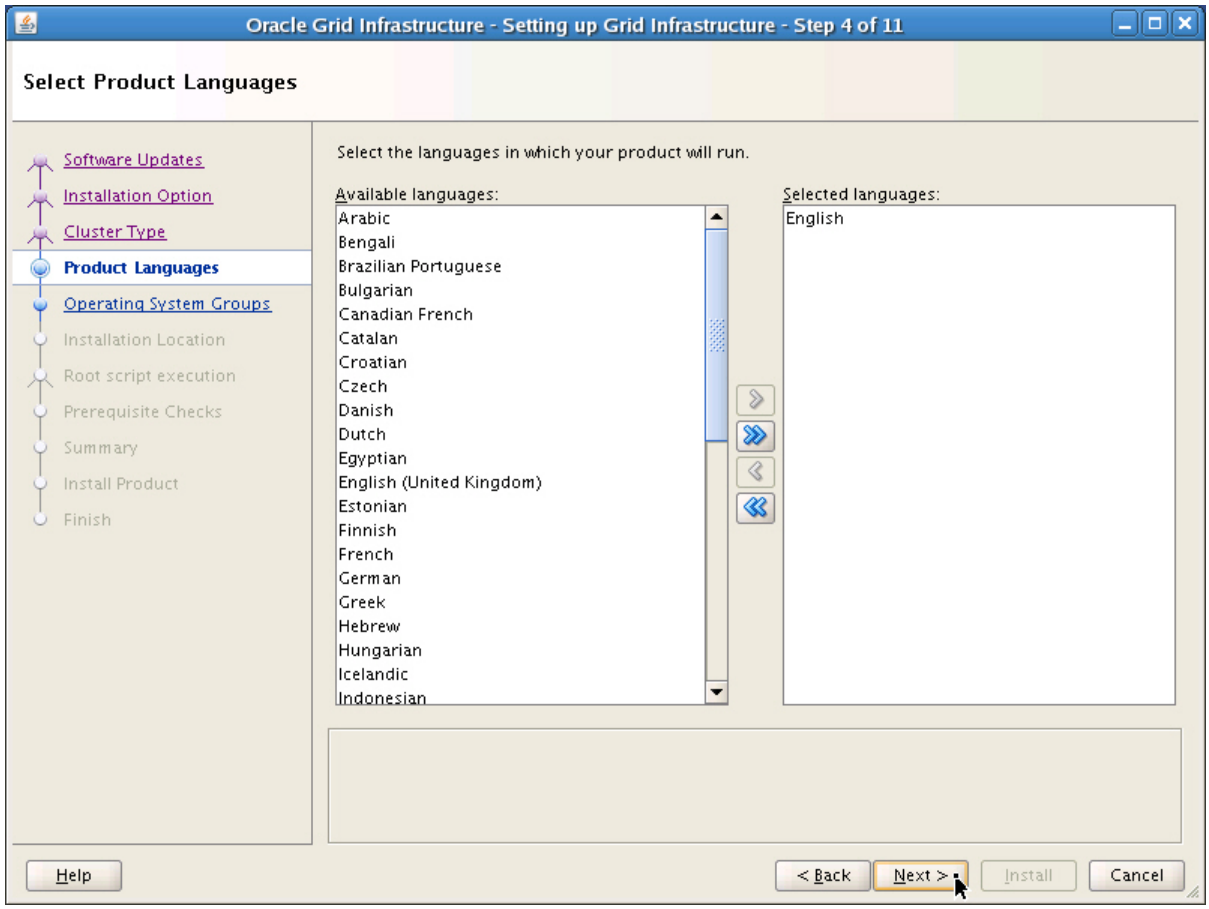
4. On the Select Installation Option screen, click Next to accept the default selection ("Install and Configure Oracle Grid Infrastructure for a Cluster").



5. On the Select Cluster Type screen, select “Configure a Flex cluster” and click Next.

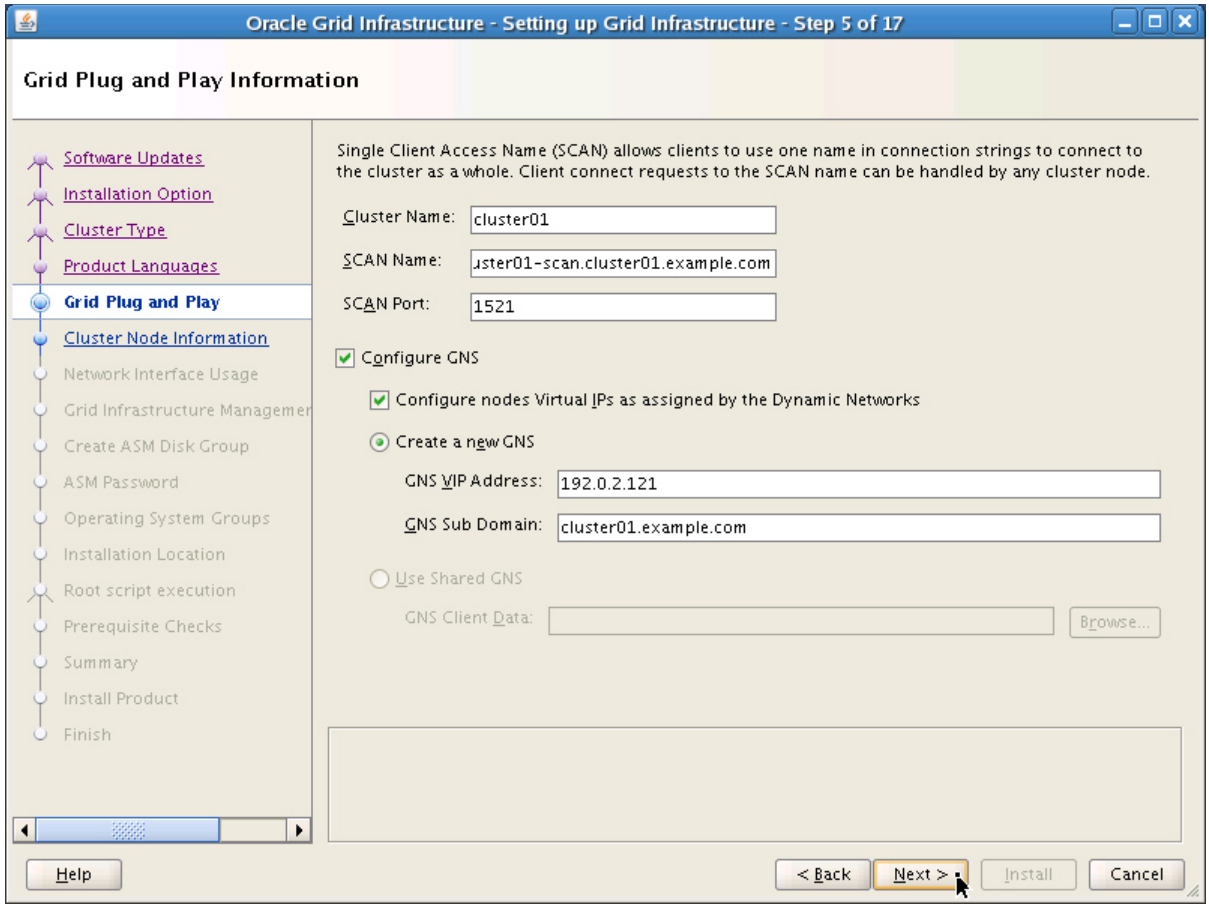


6. On the Select Product Languages screen, click Next to accept the default selection (English).

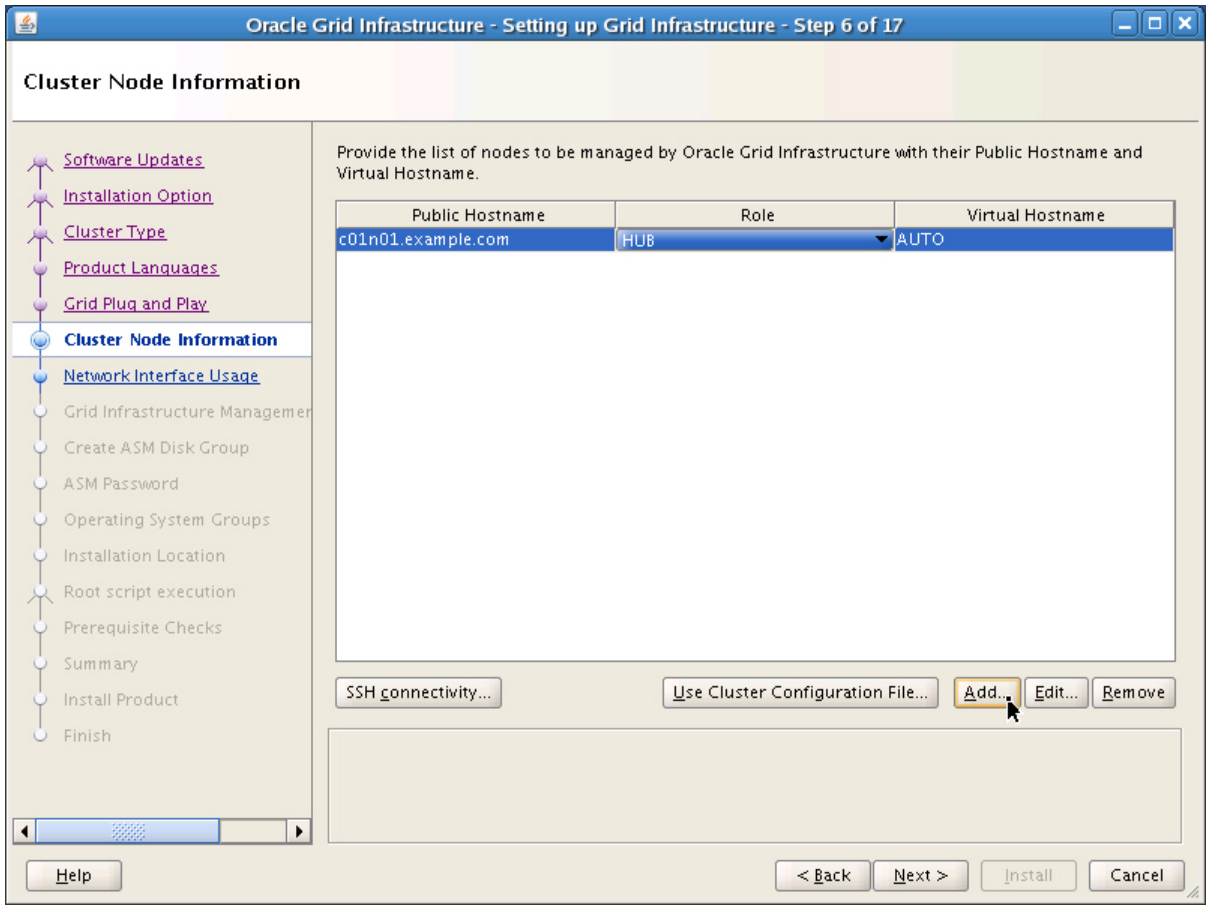


7. Use the “Grid Plug and Play Information” screen to configure the following settings:
   - Cluster Name: `cluster01`
   - GNS VIP Address: `192.0.2.121`
   - GNS Sub Domain: `cluster01.example.com`

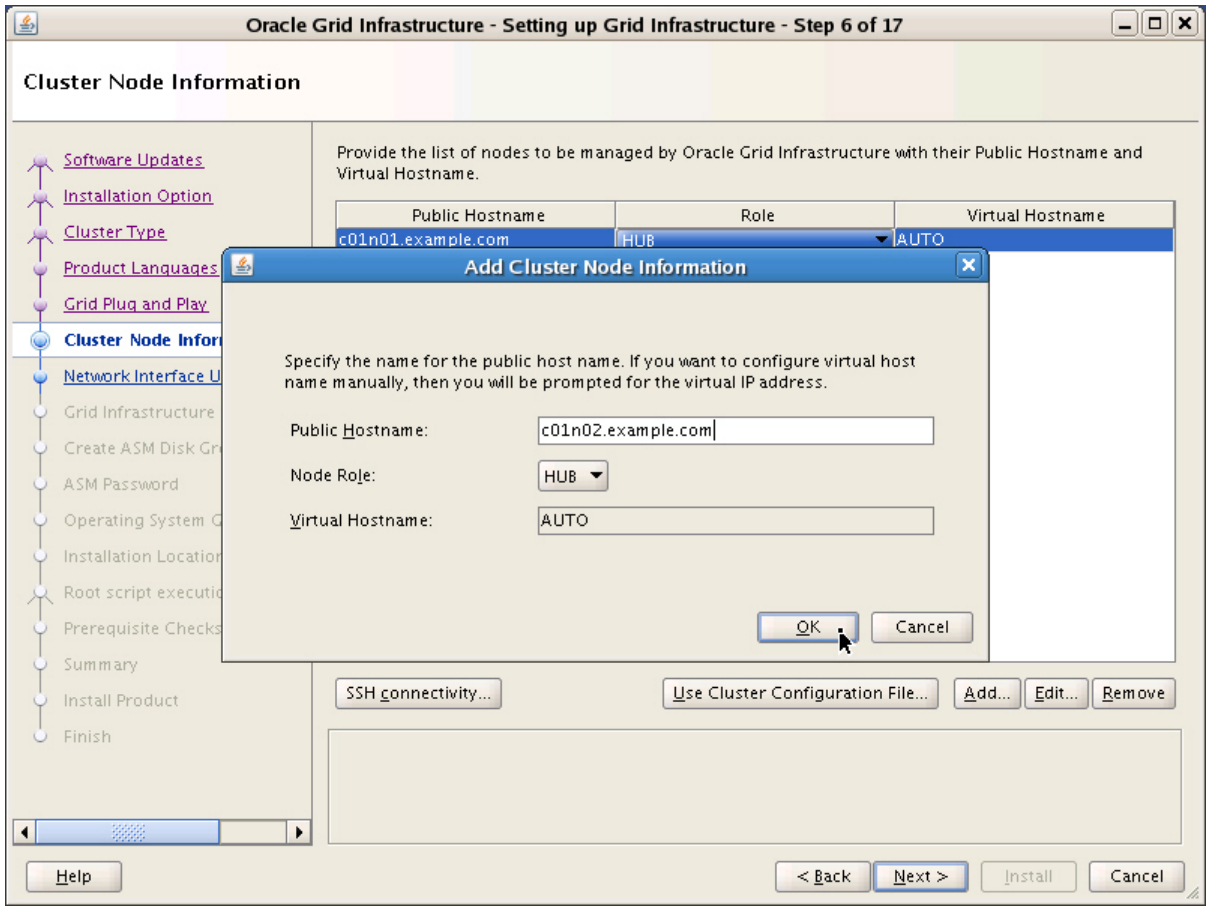
   Once your screen matches the following screenshot, click Next to continue.



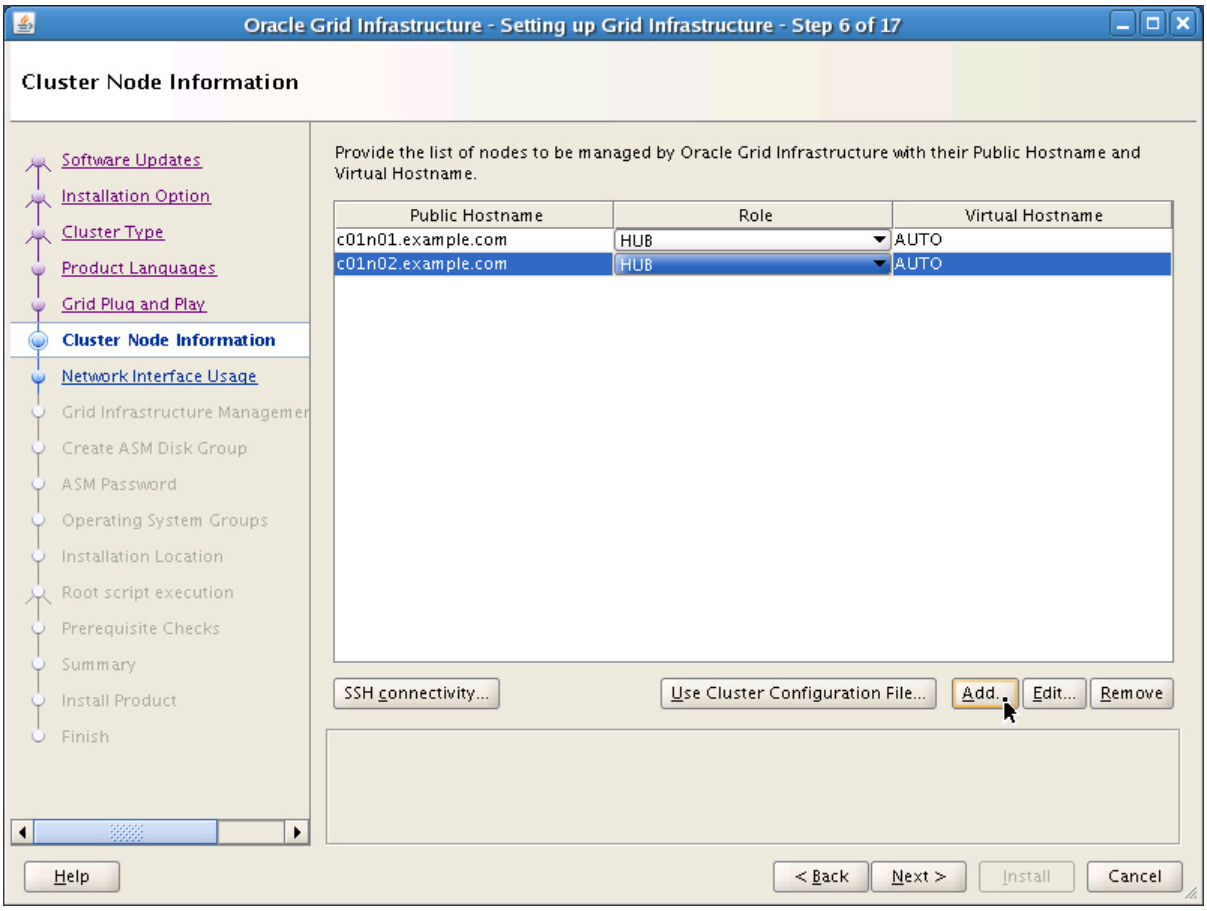
8. On the Cluster Node Information screen, click Add to begin the process of specifying additional cluster nodes.



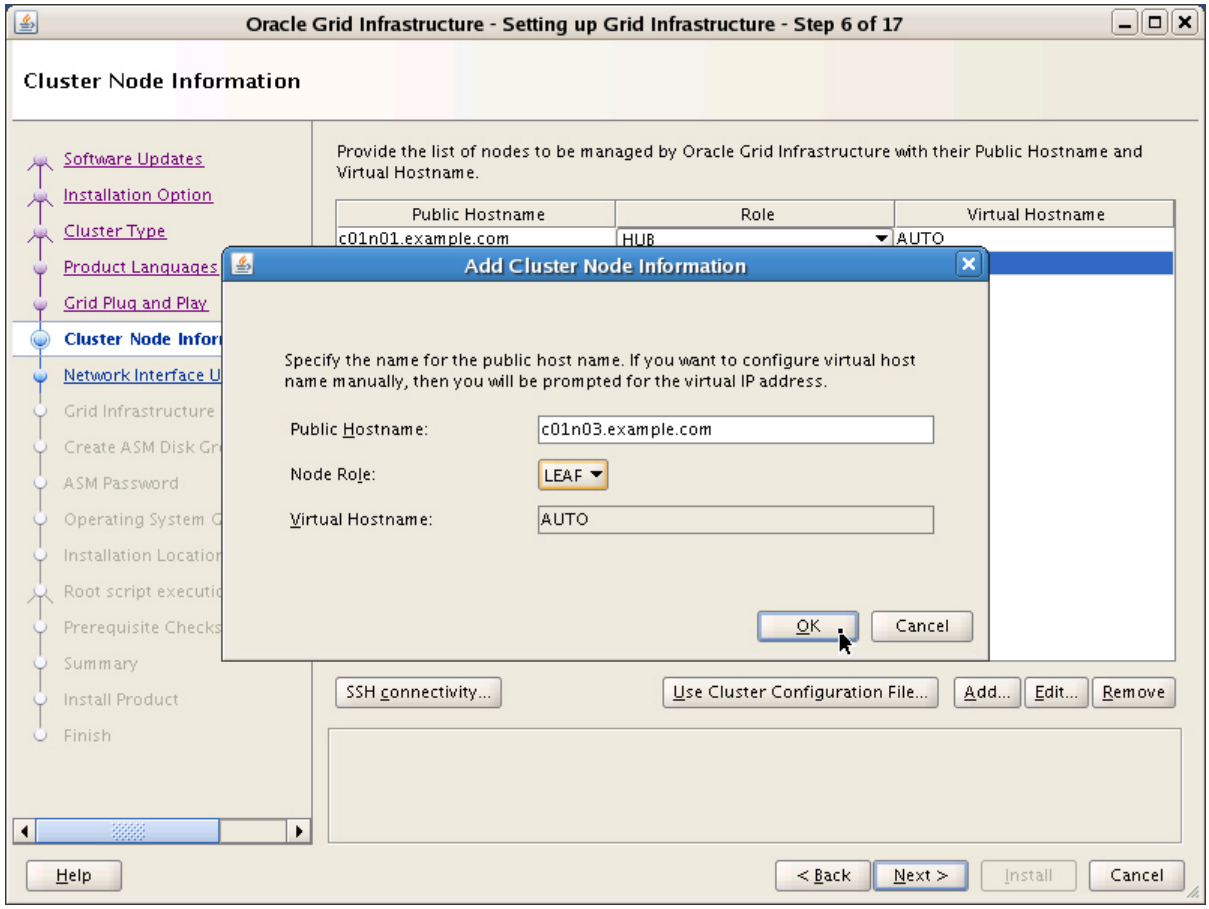
9. Use the Add Cluster Node Information dialog box to add `c01n02.example.com` as a `HUB` node, and click OK.



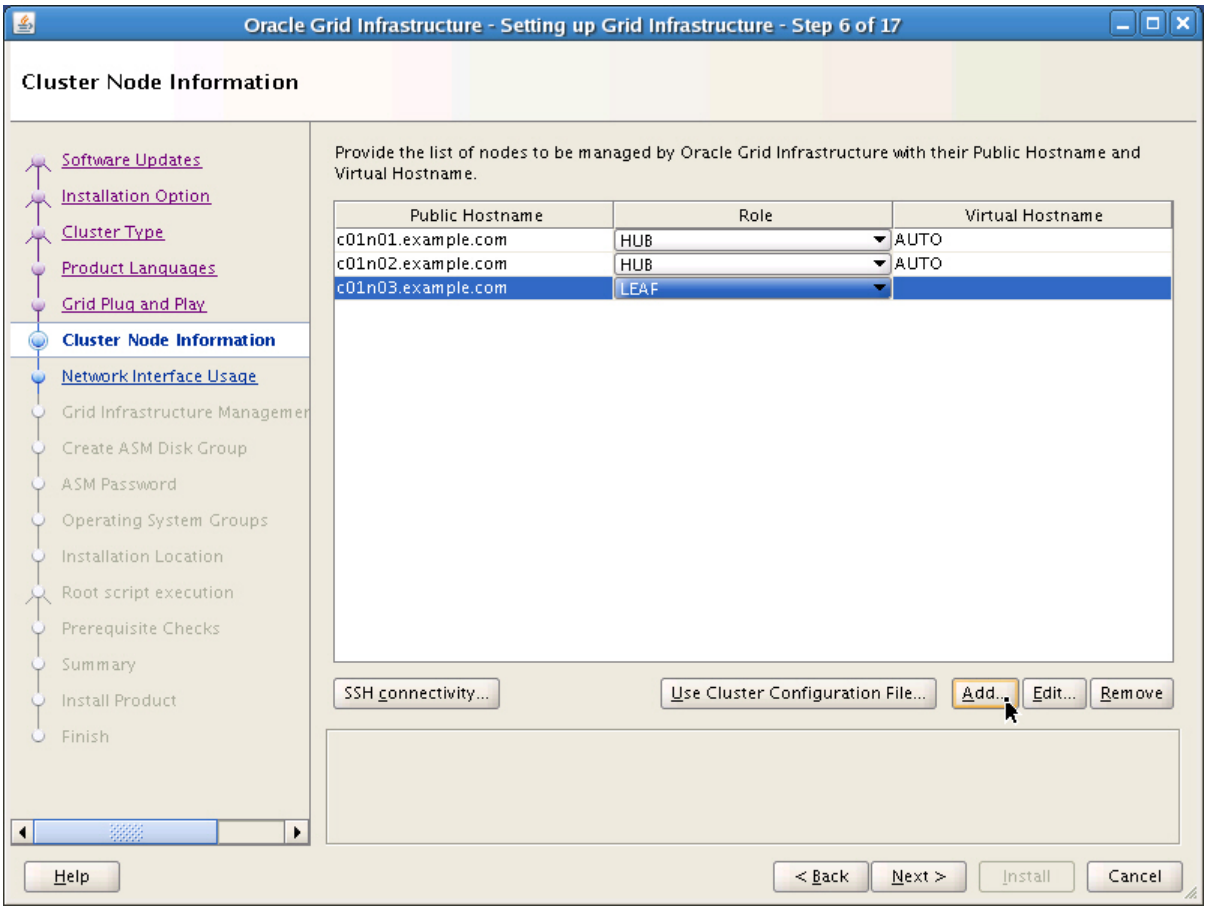
10. Back in the Cluster Node Information screen, click Add to add another node.



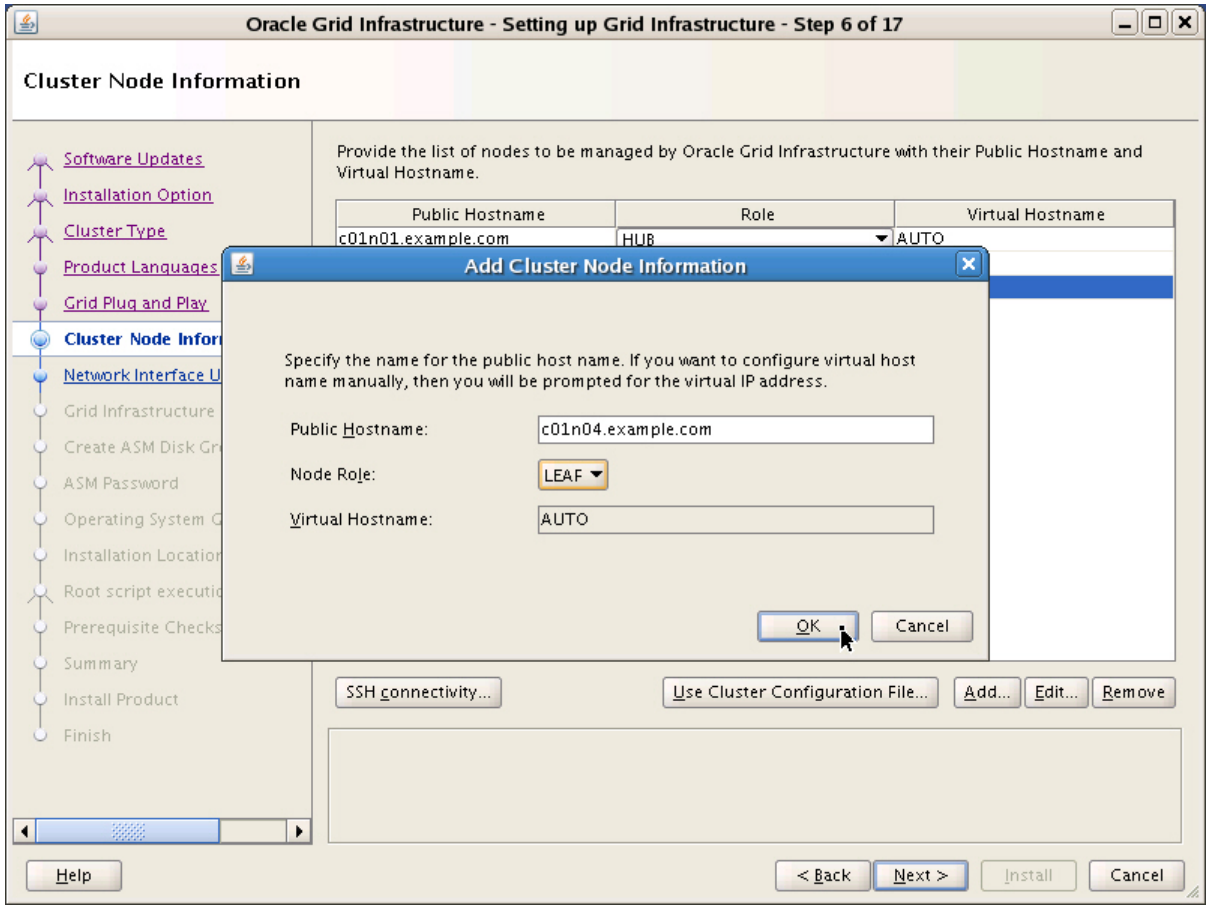
11. Use the Add Cluster Node Information dialog box to add `c01n03.example.com` as a `LEAF` node, and click OK.



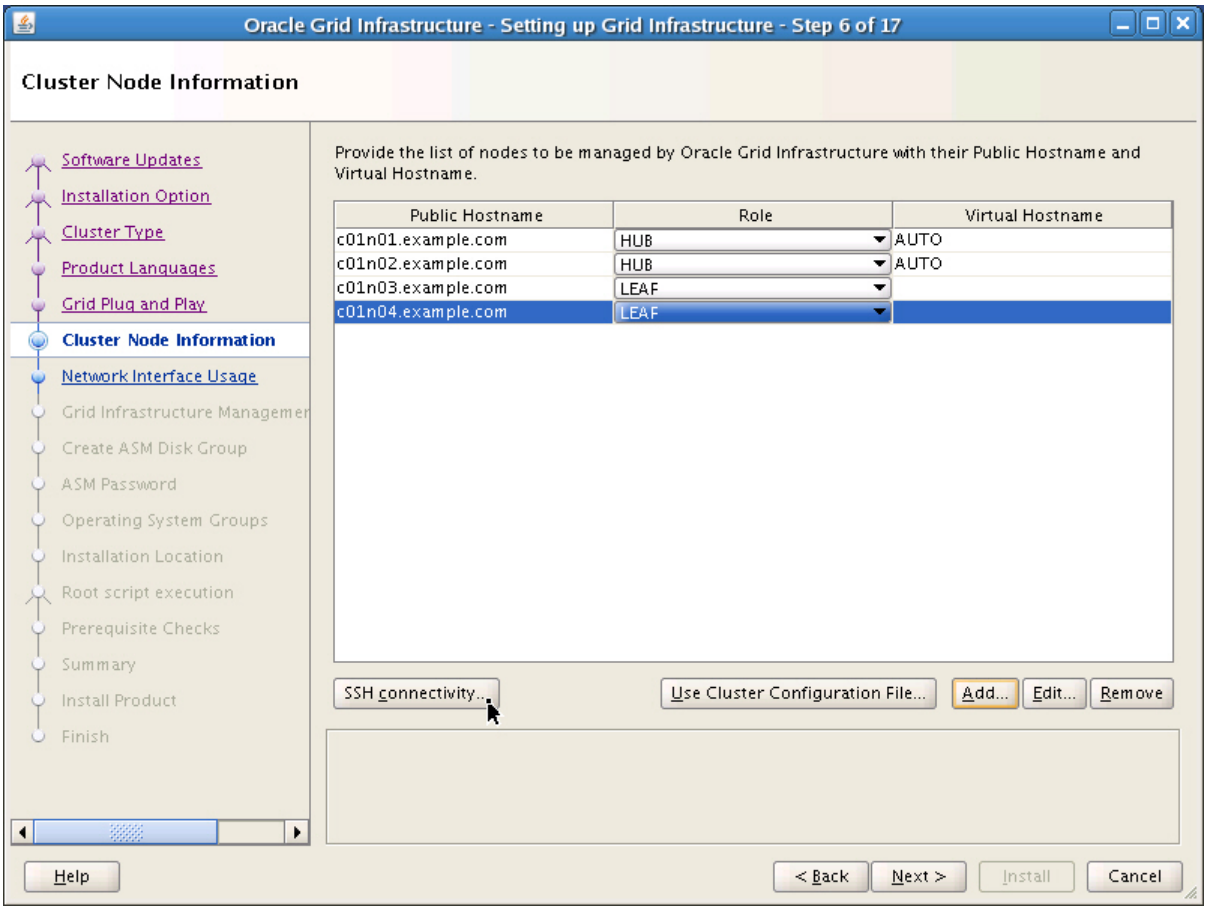
12. Back in the Cluster Node Information screen, click Add to add another node.



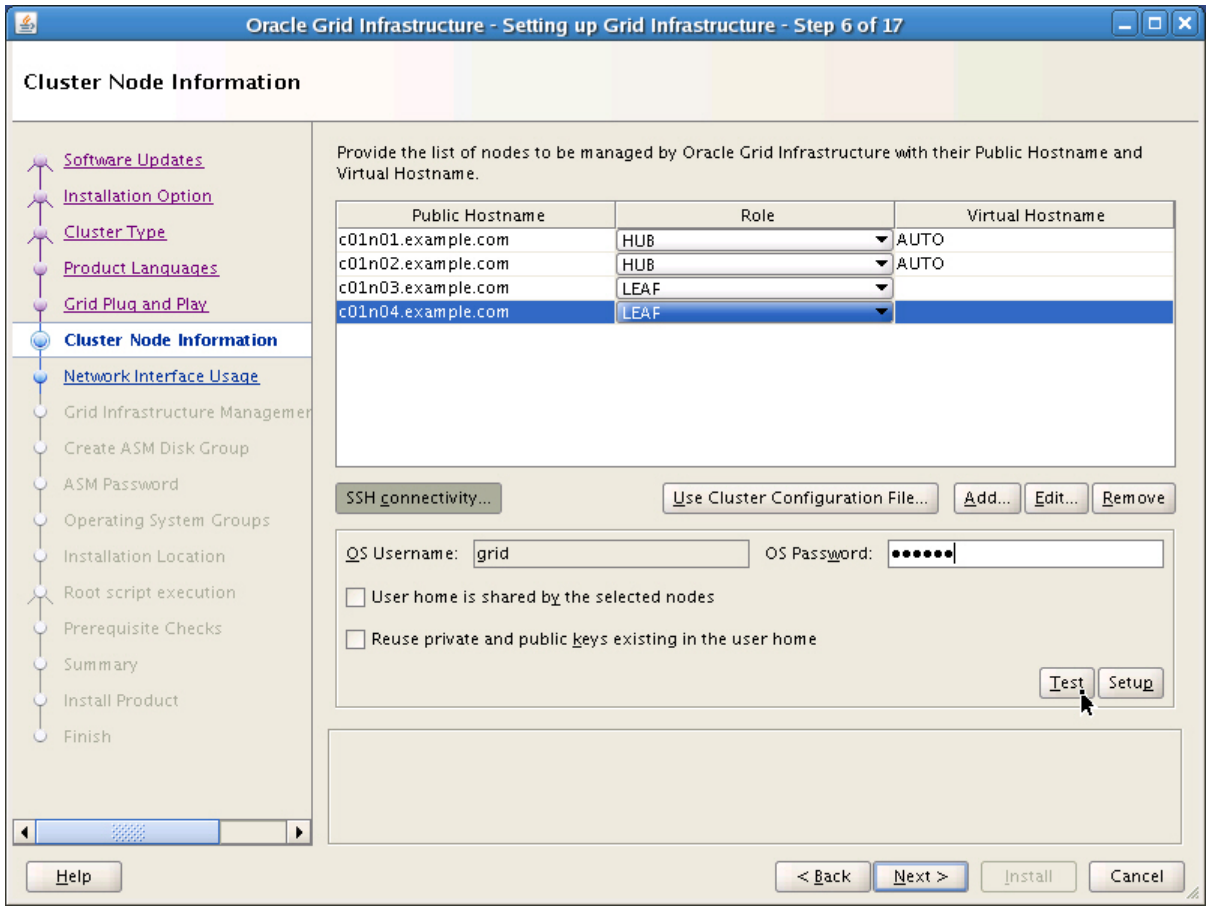
13. Use the Add Cluster Node Information dialog box to add `c01n04.example.com` as a `LEAF` node, and click OK.



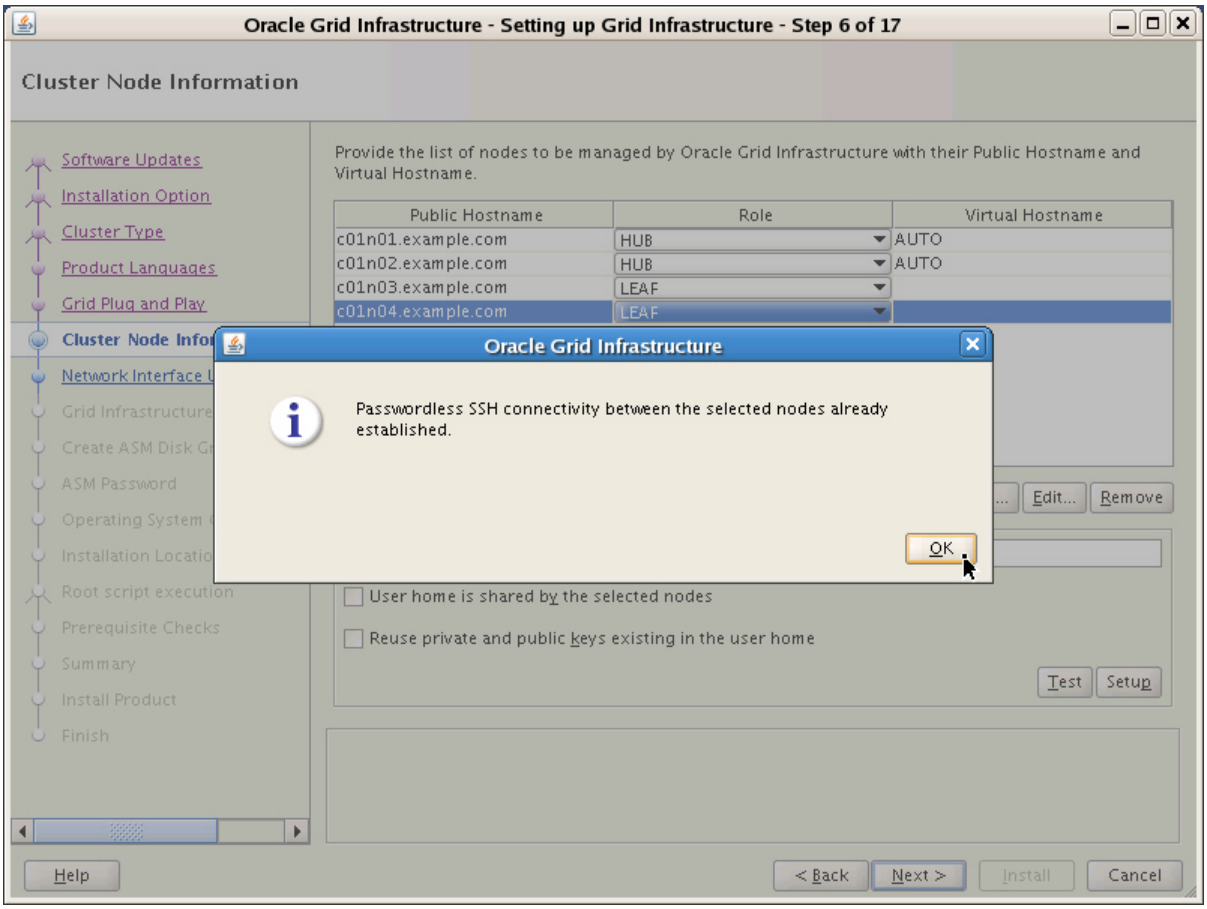
14. At this point, your OUI screen should list four cluster nodes. Click SSH Connectivity.



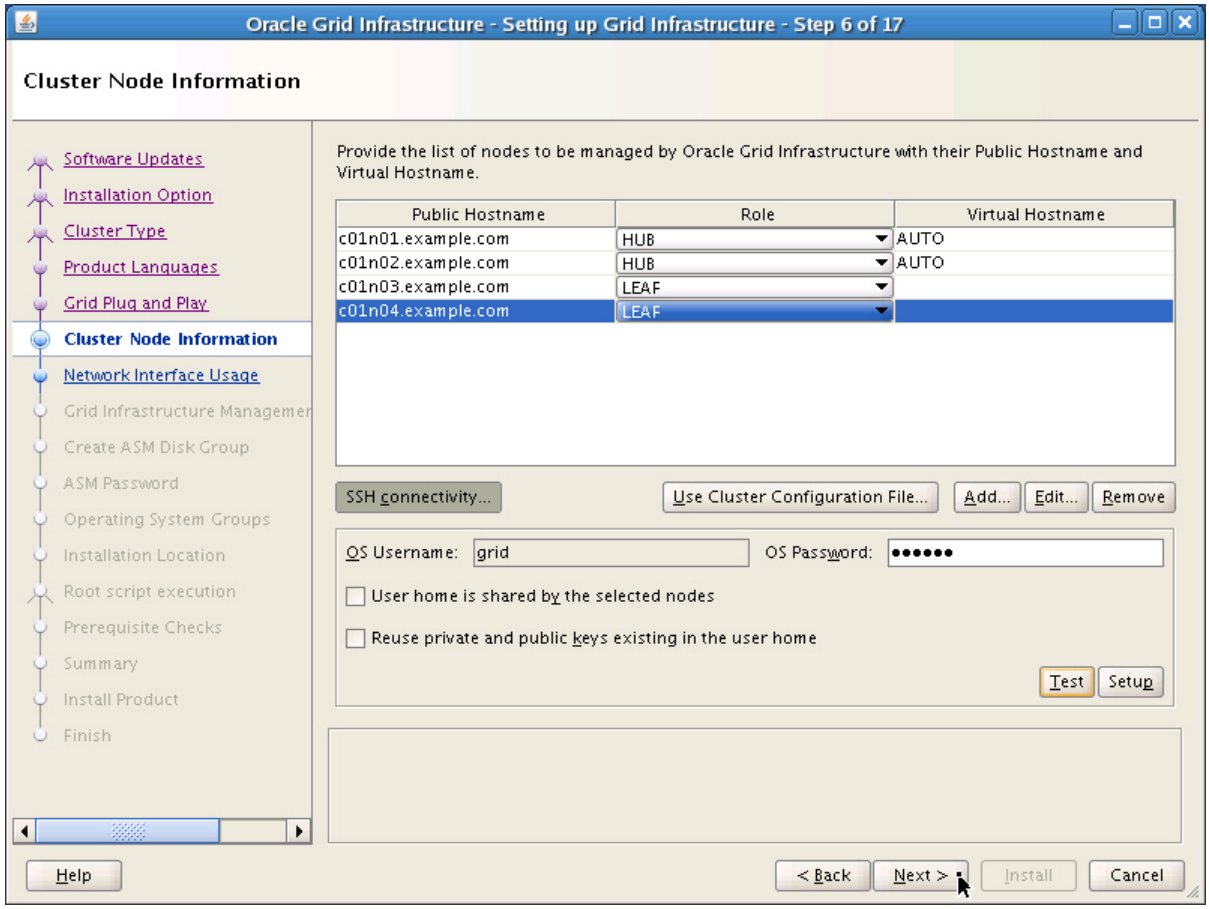
15. Enter `oracle` into the OS Password field and click Test to confirm that the required SSH connectivity is configured across the cluster.



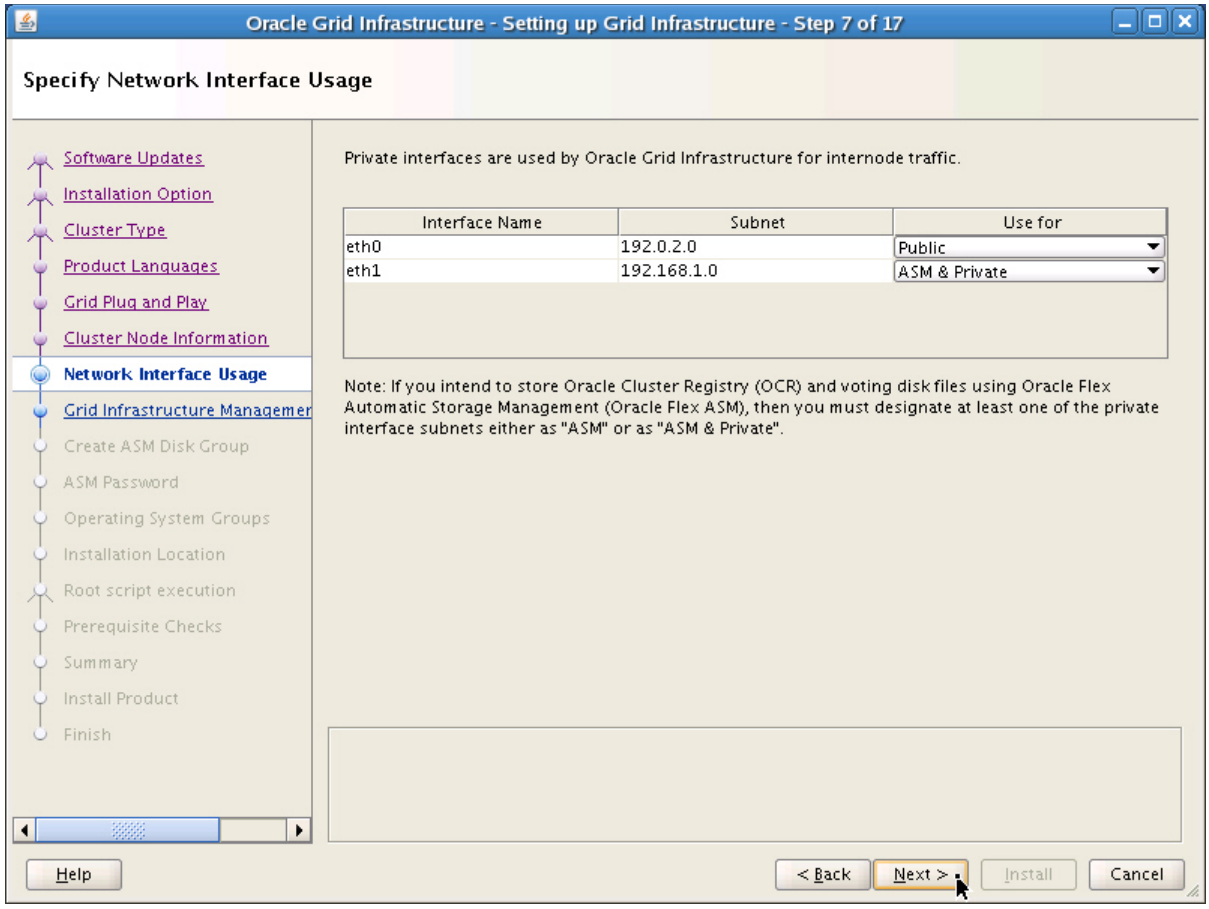
16. Your laboratory environment is preconfigured with the required SSH connectivity so you will next see a dialog box confirming this. Click OK to continue.



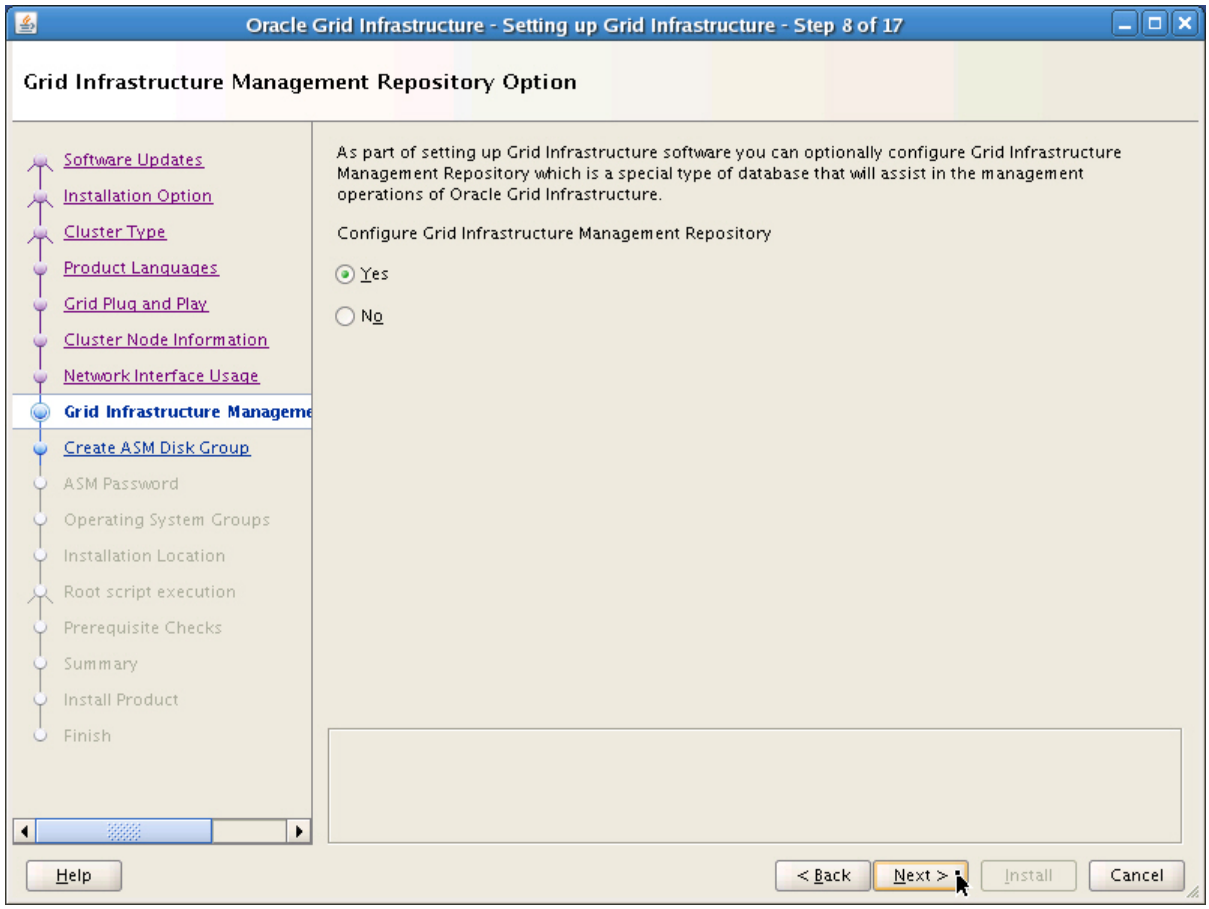
17. If the required SSH connectivity was not present, you could now click Setup to perform the required configuration. However, because the laboratory environment is already configured correctly, click Next to continue.



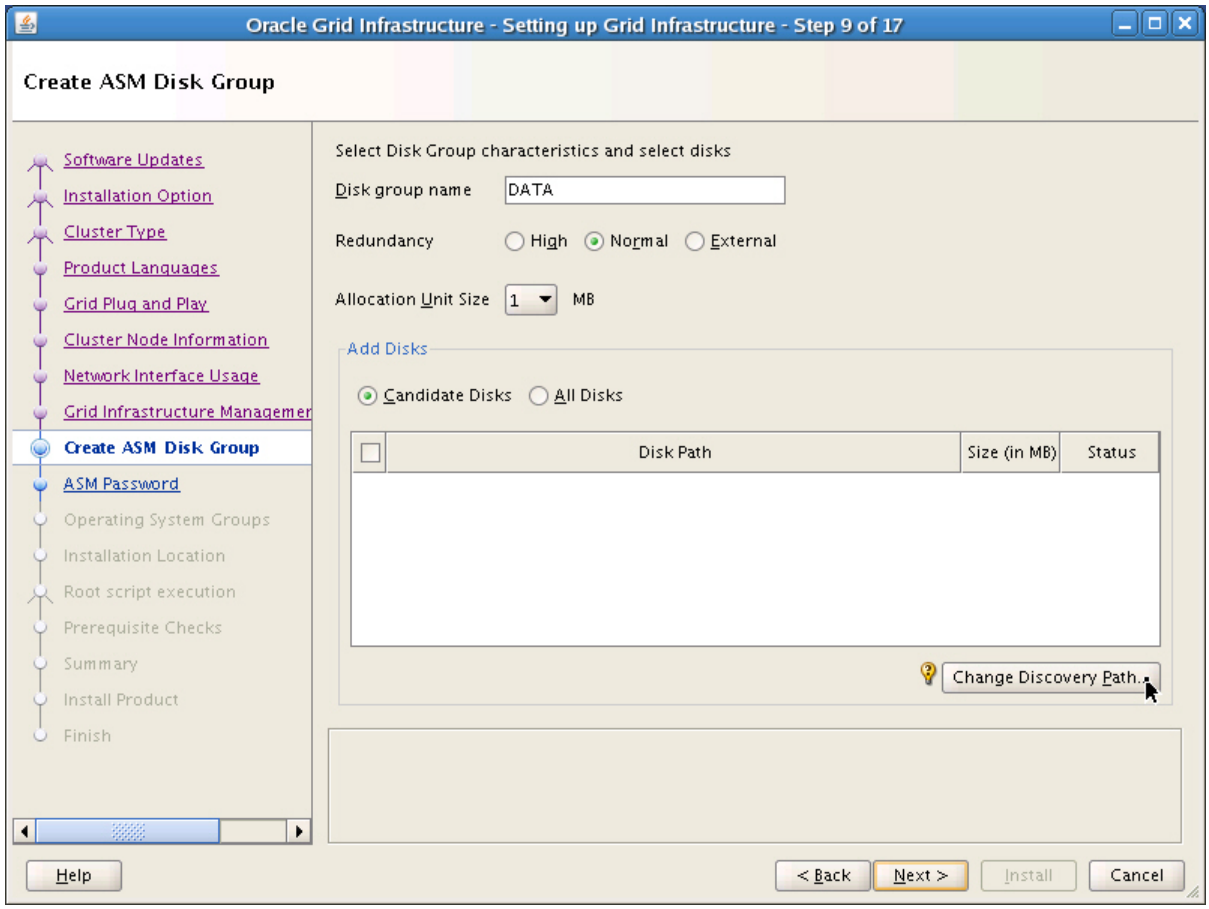
18. On the Specify Network Interface Usage screen, ensure that network interface `eth0` is designated as the Public network and that network interface `eth1` is designated as the ASM & Private network. Click Next to continue.



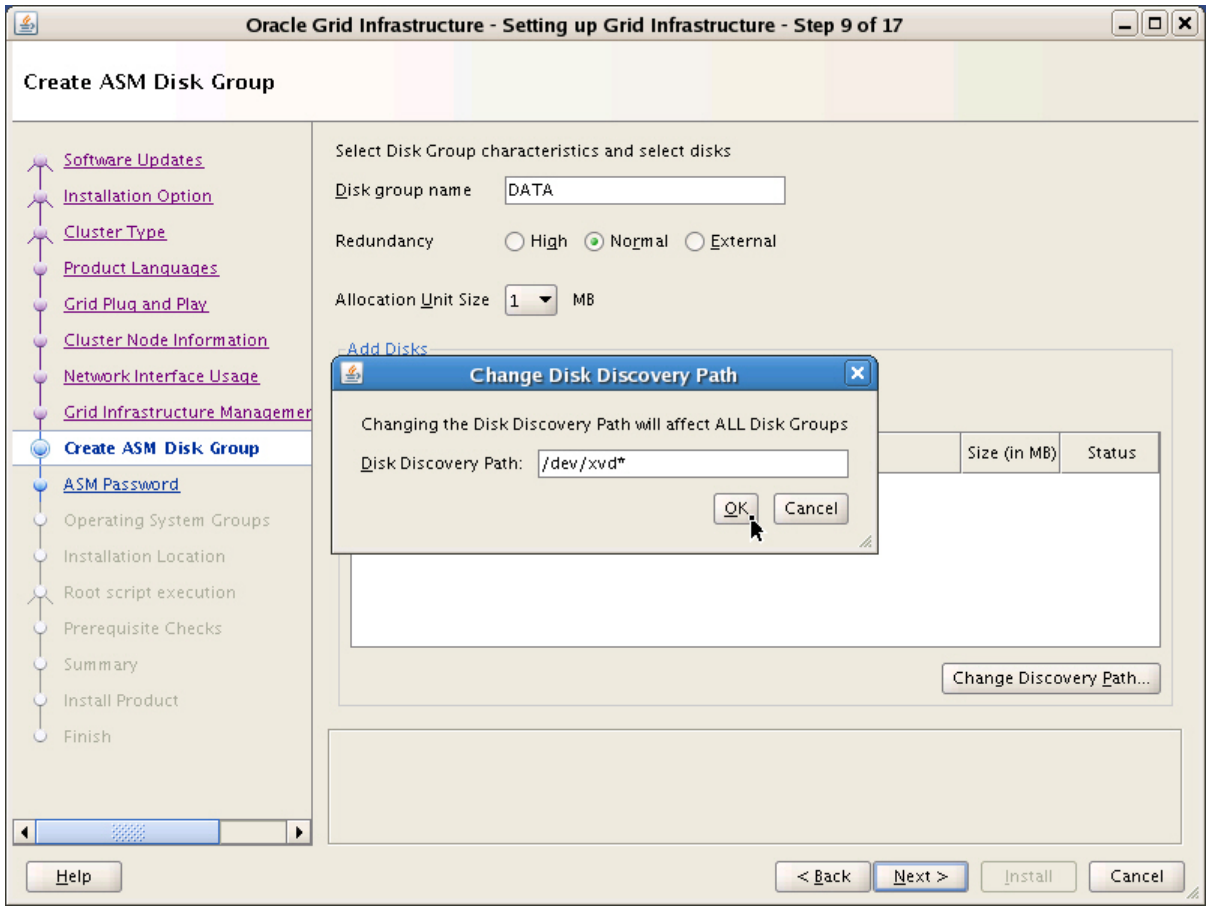
19. On the Grid Infrastructure Management Repository Option screen, select Yes and click Next to continue.



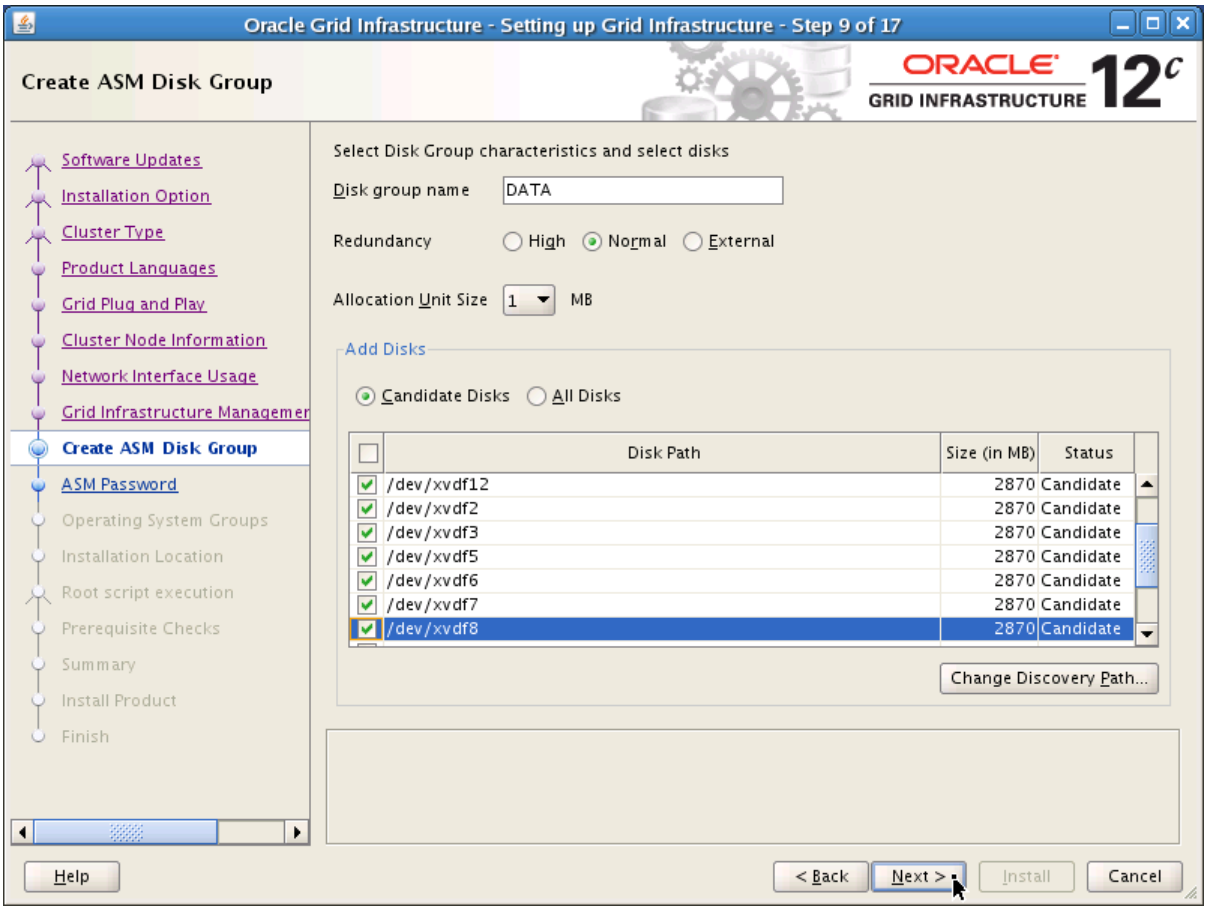
20. On the Create ASM Disk Group screen, click Change Discovery Path to commence the process of specifying the ASM disks.



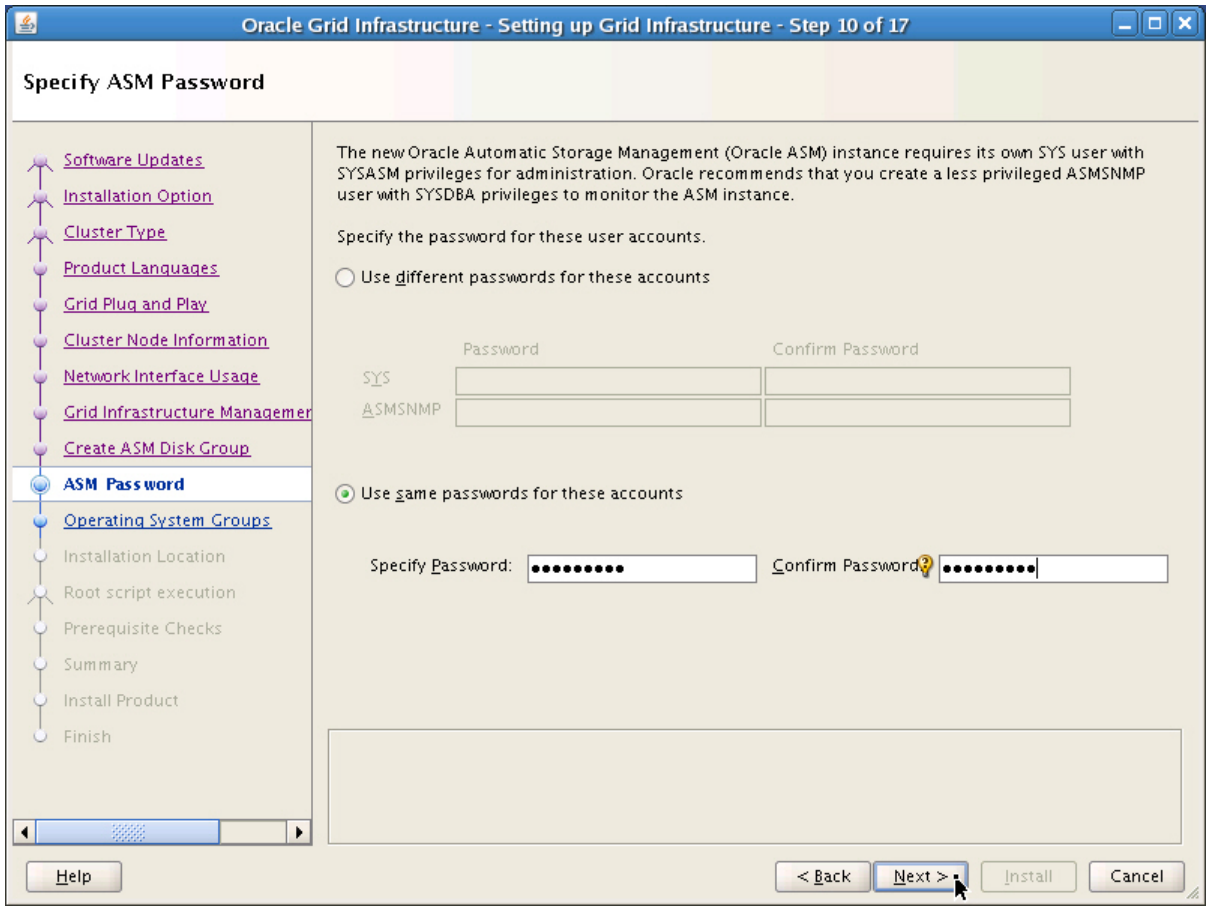
21. In the Change Disk Discovery Path dialog box, set the Disk Discovery Path to `/dev/xvd*` and click OK.



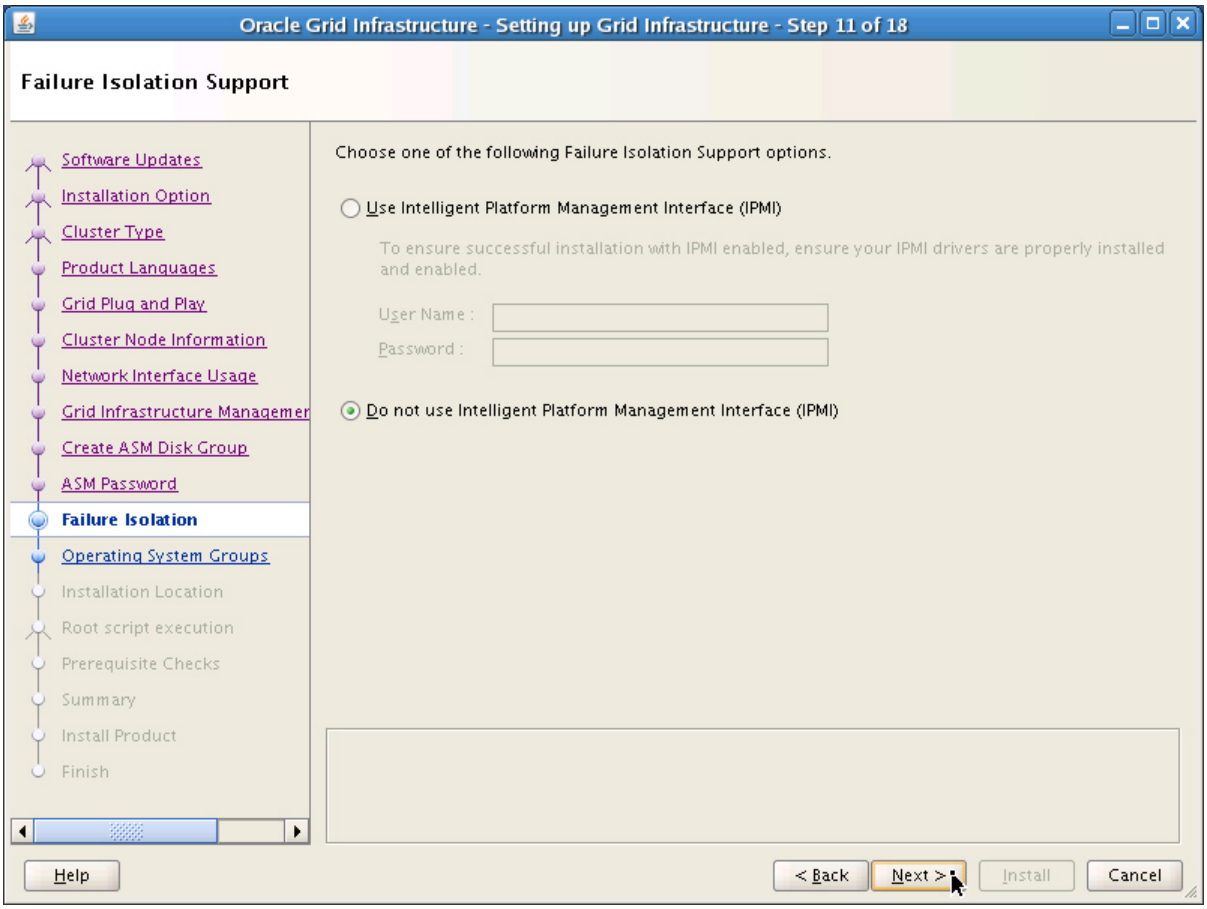
22. Back on the Create ASM Disk Group screen, select the first 10 candidate disks (`/dev/xvdf1`, `/dev/xvdf10`, `/dev/xvdf11`, `/dev/xvdf12`, `/dev/xvdf2`, `/dev/xvdf3`, `/dev/xvdf5`, `/dev/xvdf6`, `/dev/xvdf7`, and `/dev/xvdf8`) and click Next to continue.



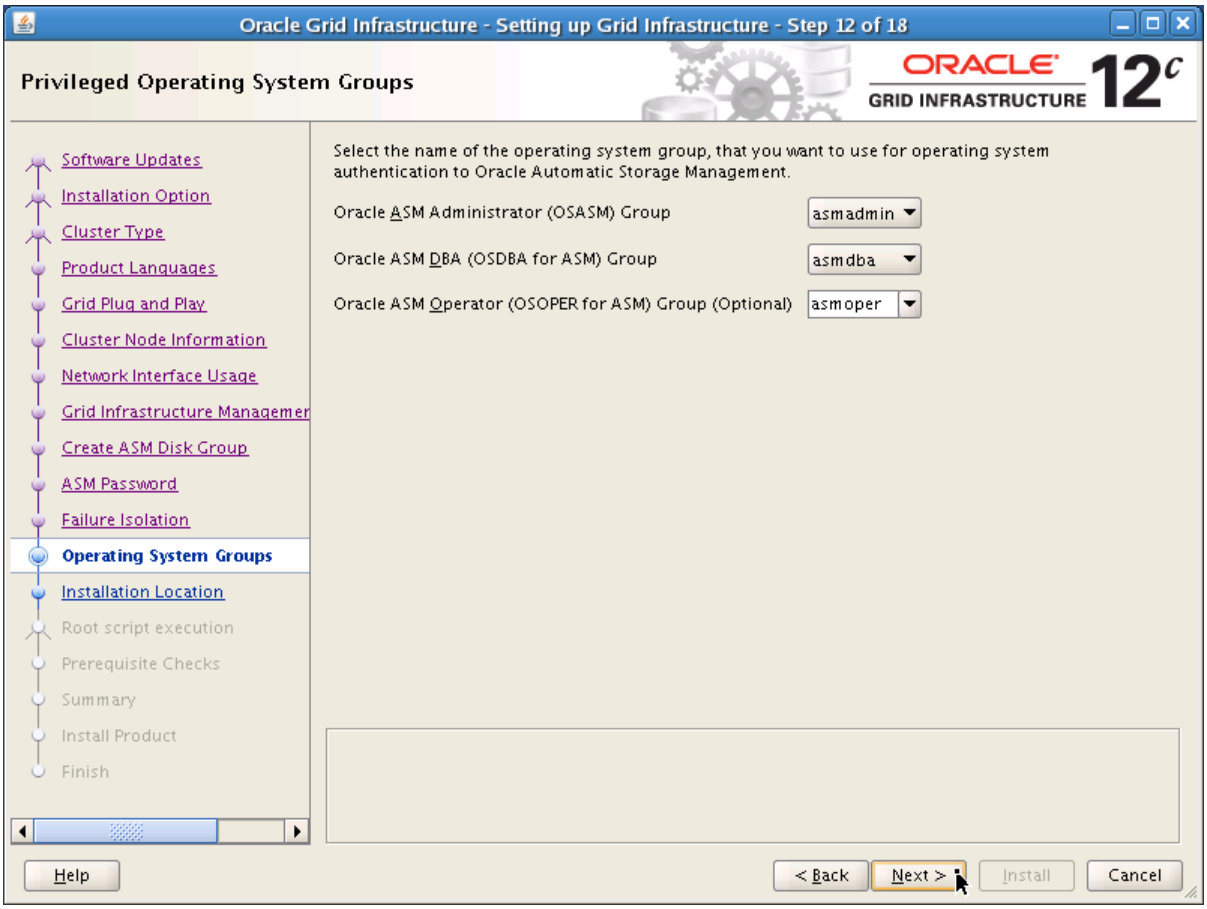
23. On the Specify ASM Password screen, select “Use same passwords for these accounts” and enter `oracle_4U` as the password. Then click Next to continue.



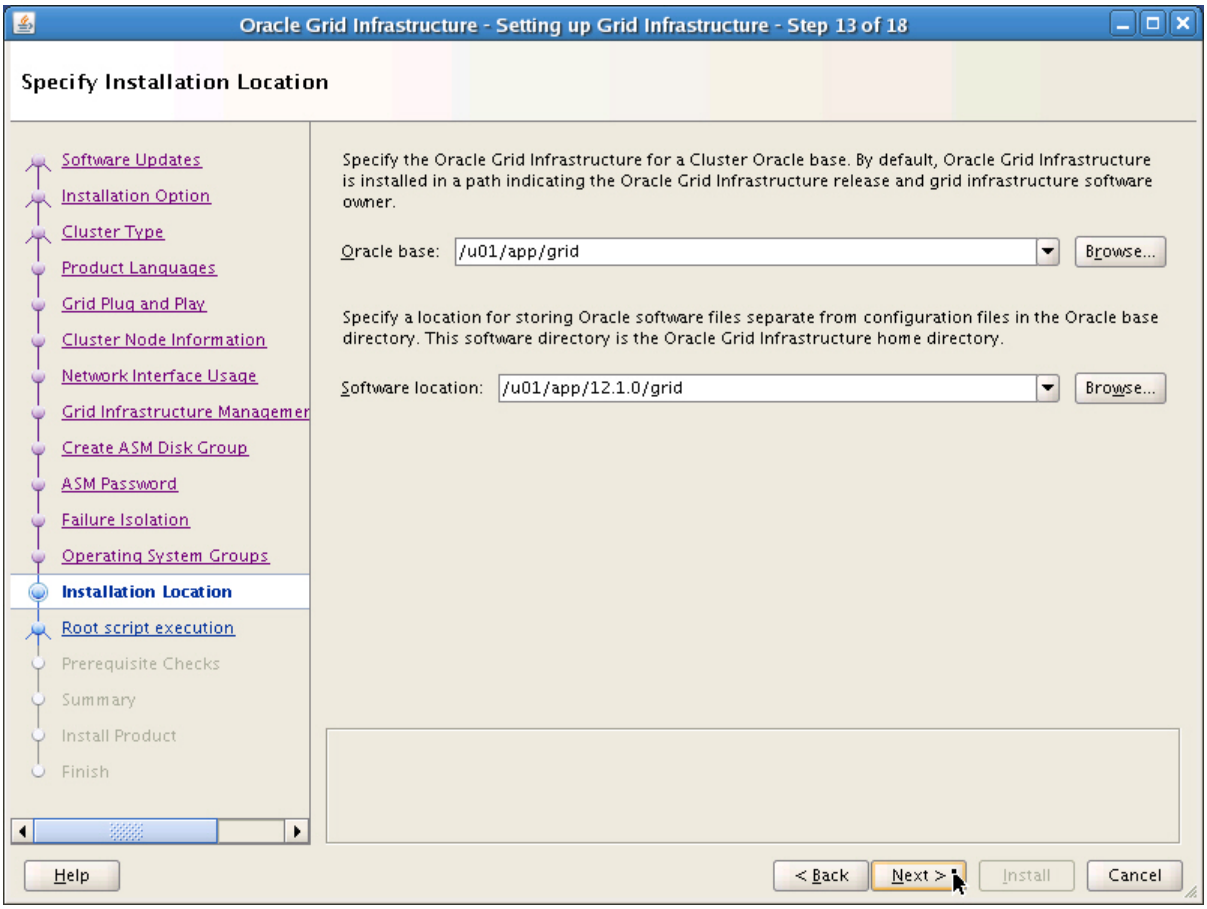
24. On the Failure Isolation Support screen, click Next to accept the default setting (Do not use IPMI).



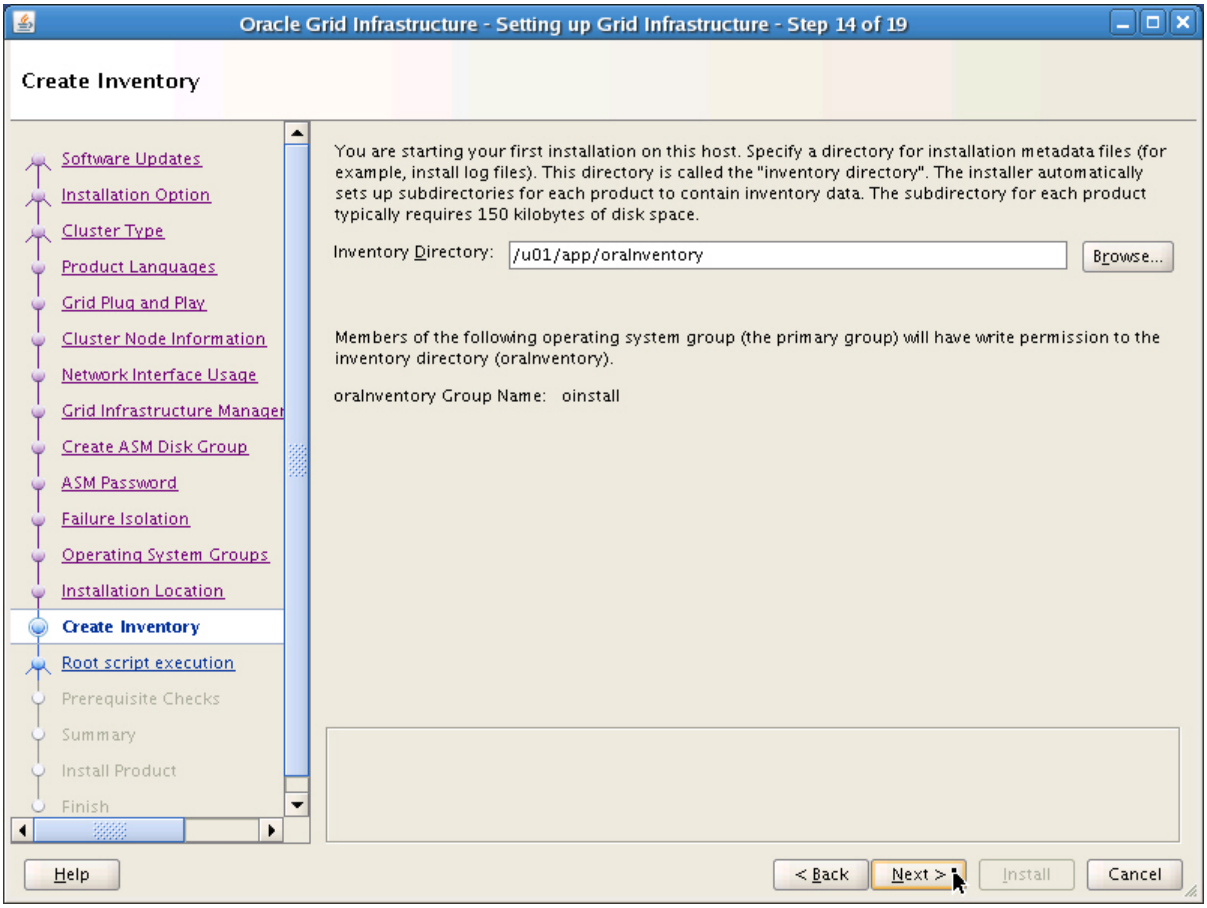
25. On the Privileged Operating System Groups screen, click Next to accept the default privileged operating system groups.



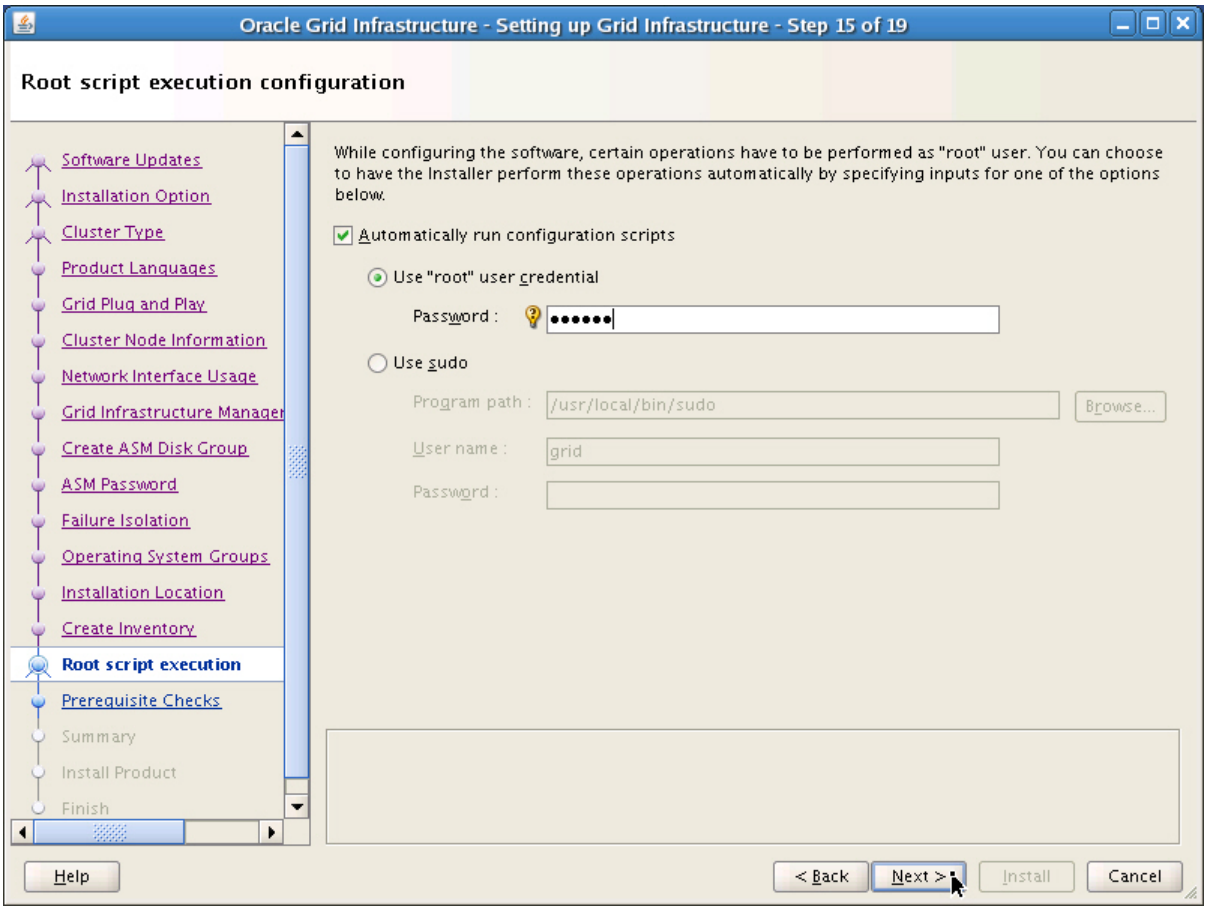
26. On the Specify Installation Location screen, click Next to accept the default installation location.



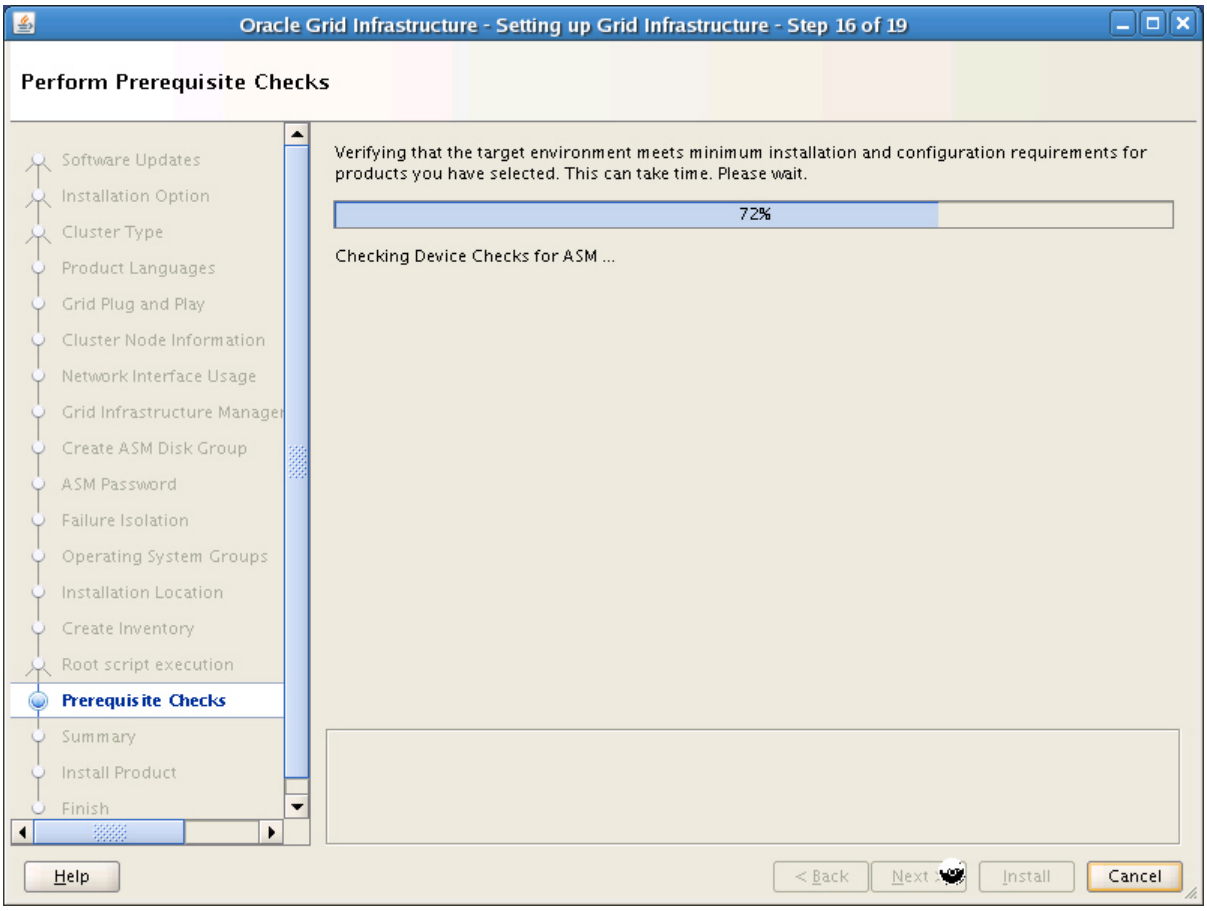
27. On the Create Inventory screen, click Next to accept the default installation inventory location.



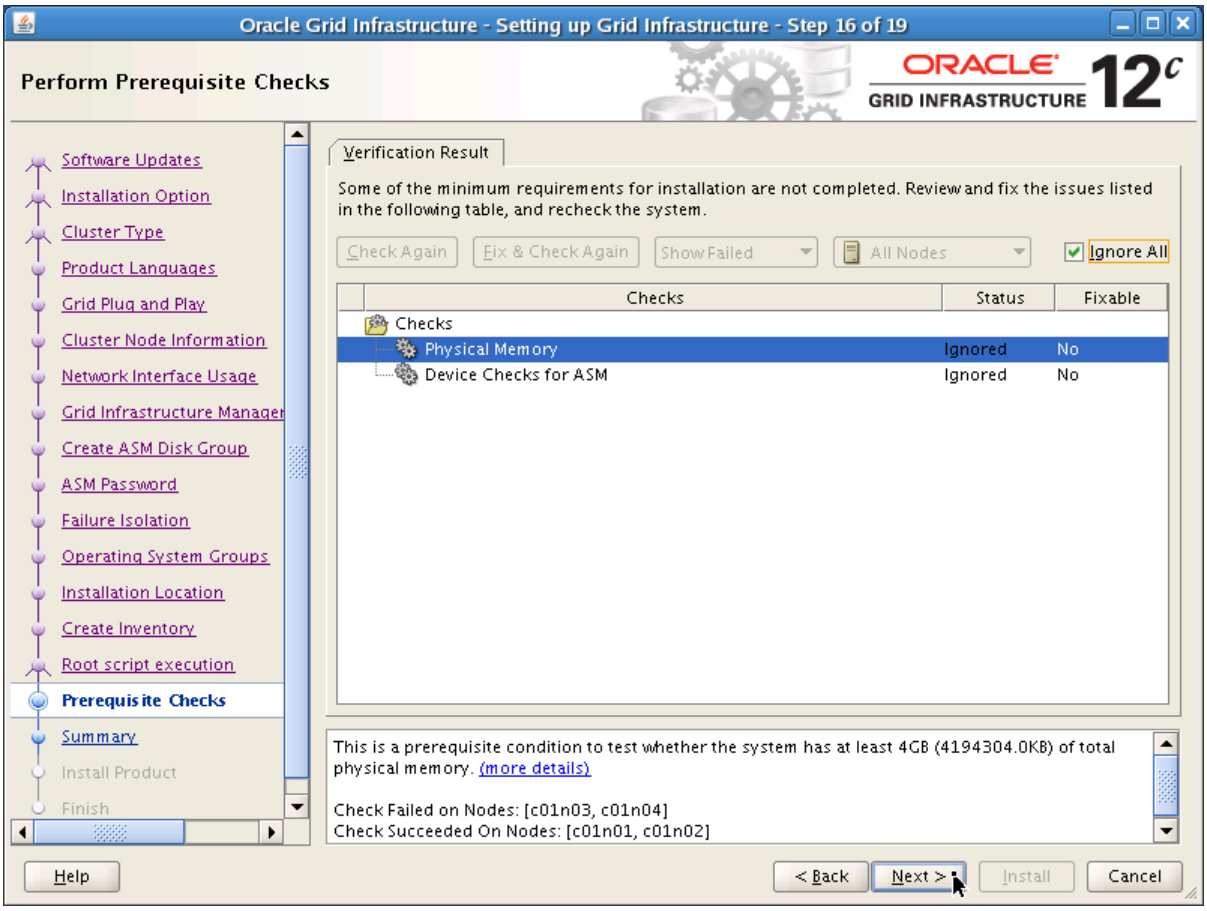
28. On the “Root script execution configuration” screen, check “Automatically run configuration scripts” and select “Use ‘root’ user credential.” Enter `oracle` as the password and click Next to proceed.



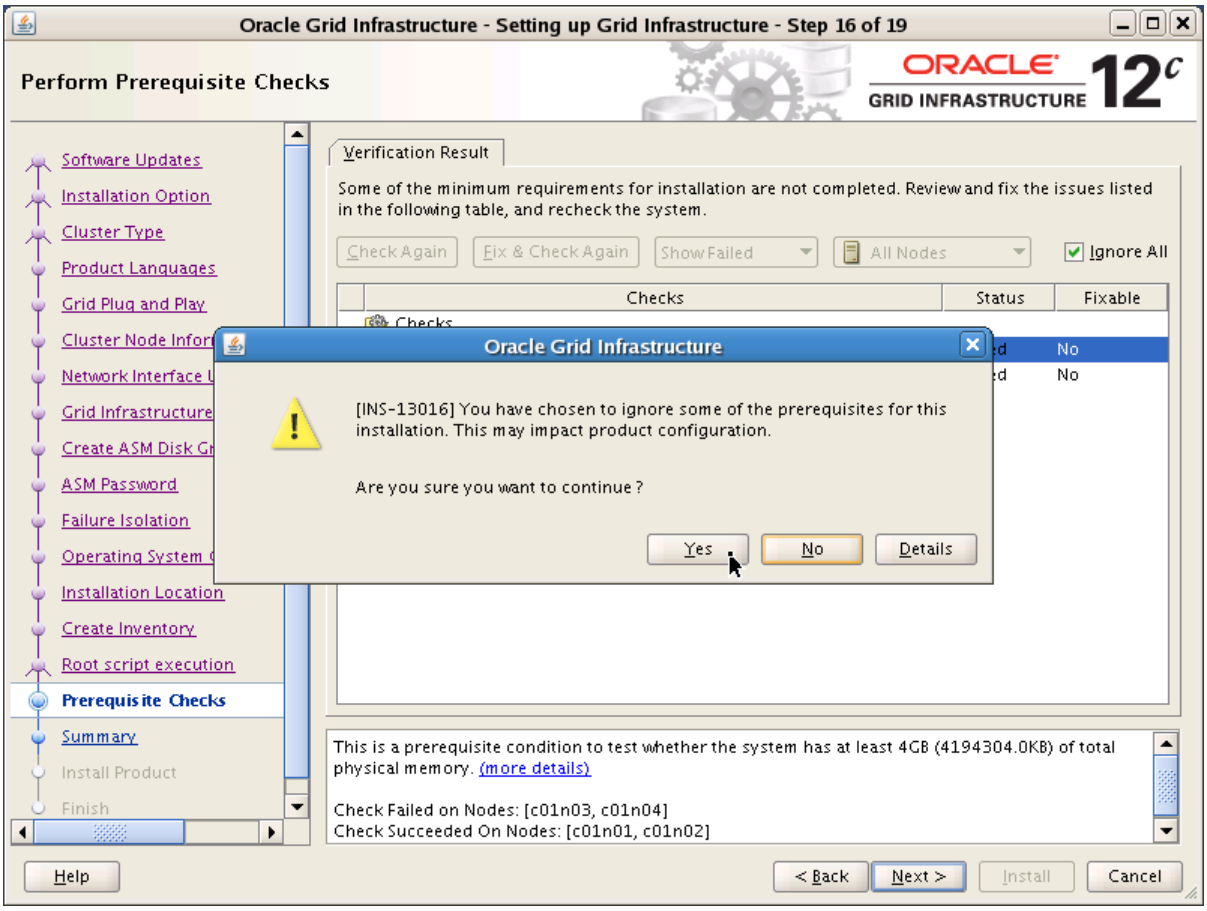
29. Wait while a series of prerequisite checks are performed.



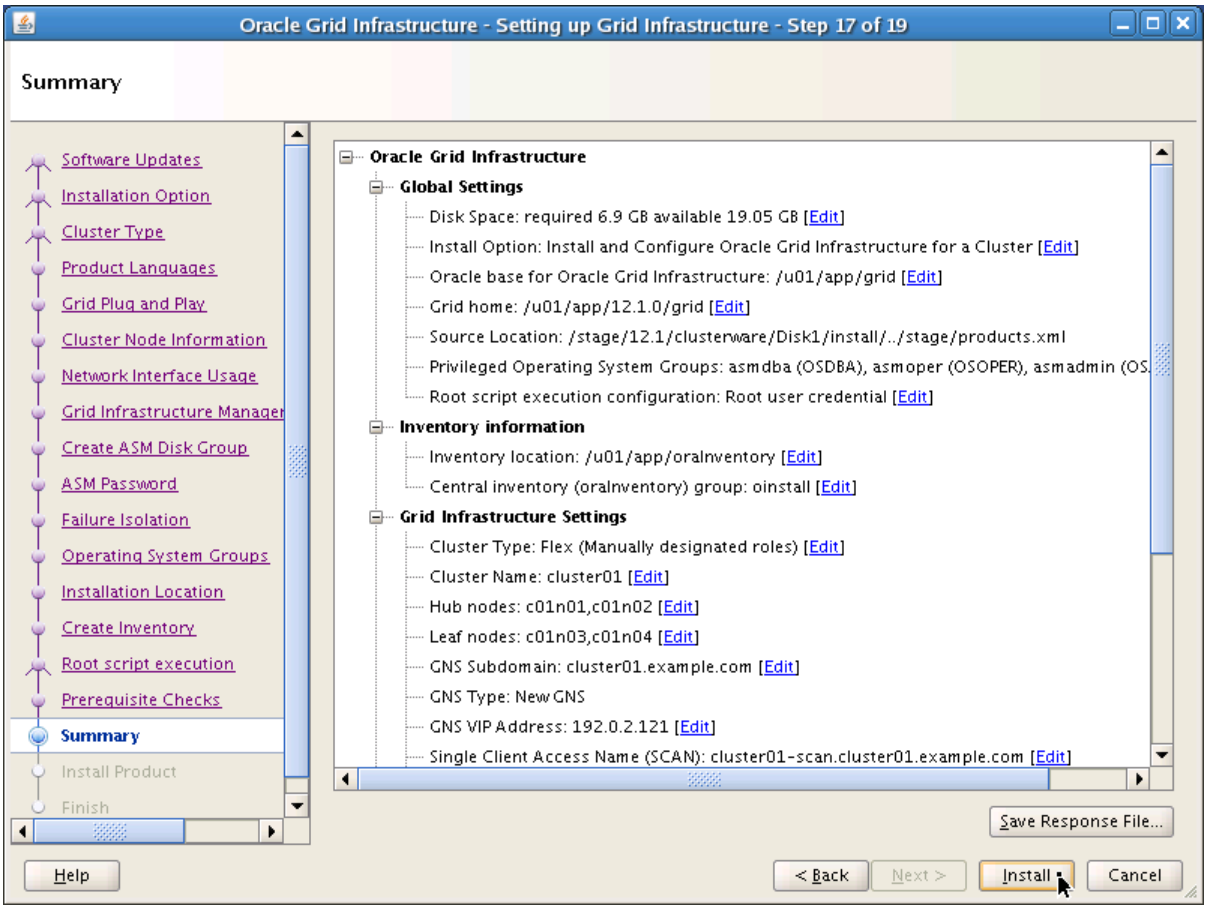
30. Due to the constraints of this laboratory environment, you may see a series of warnings resulting from the prerequisite checks. If the warnings relate to “Physical Memory” and “Device Checks for ASM” (as illustrated in the following screenshot) you may safely ignore them. Check “Ignore all” and then click Next to proceed.



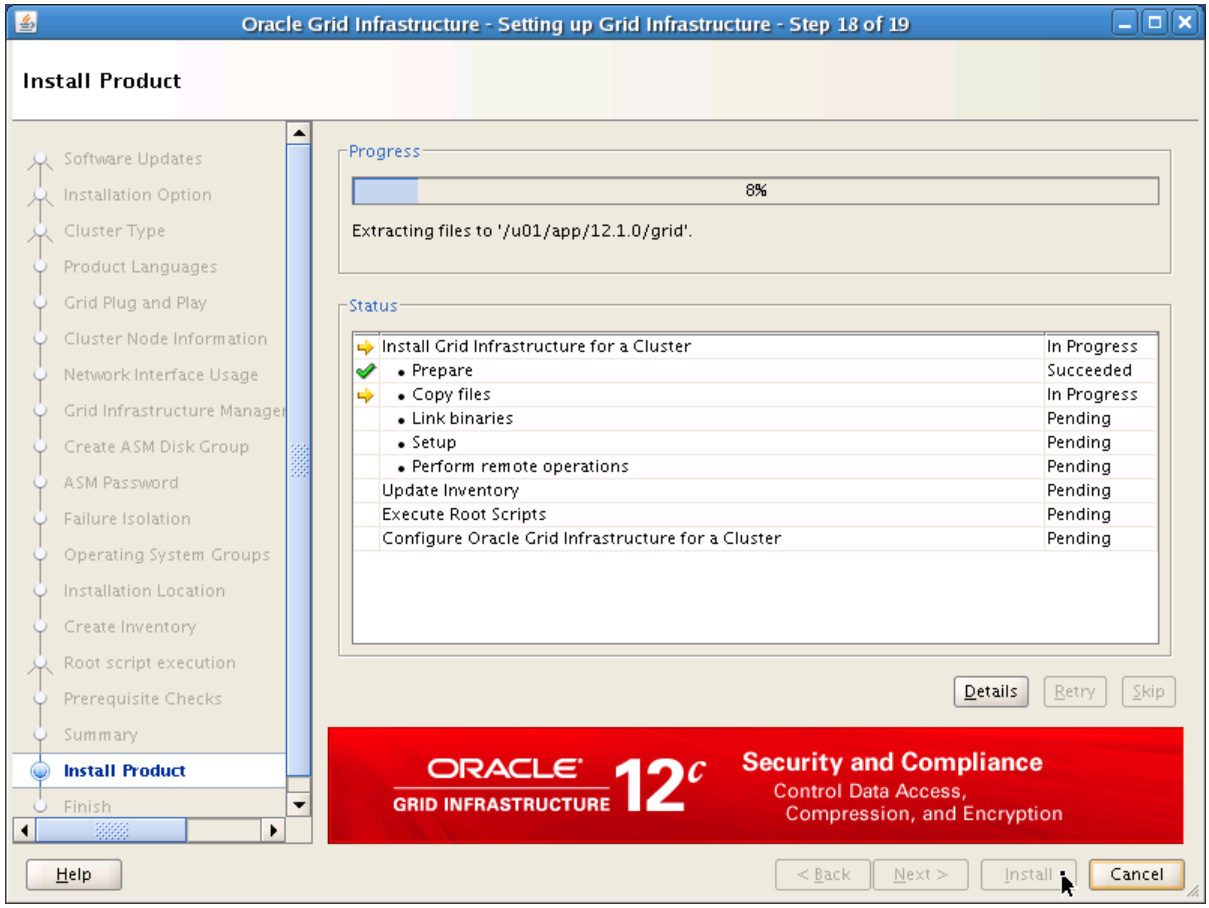
31. In the confirmation dialog box, click Yes to continue.



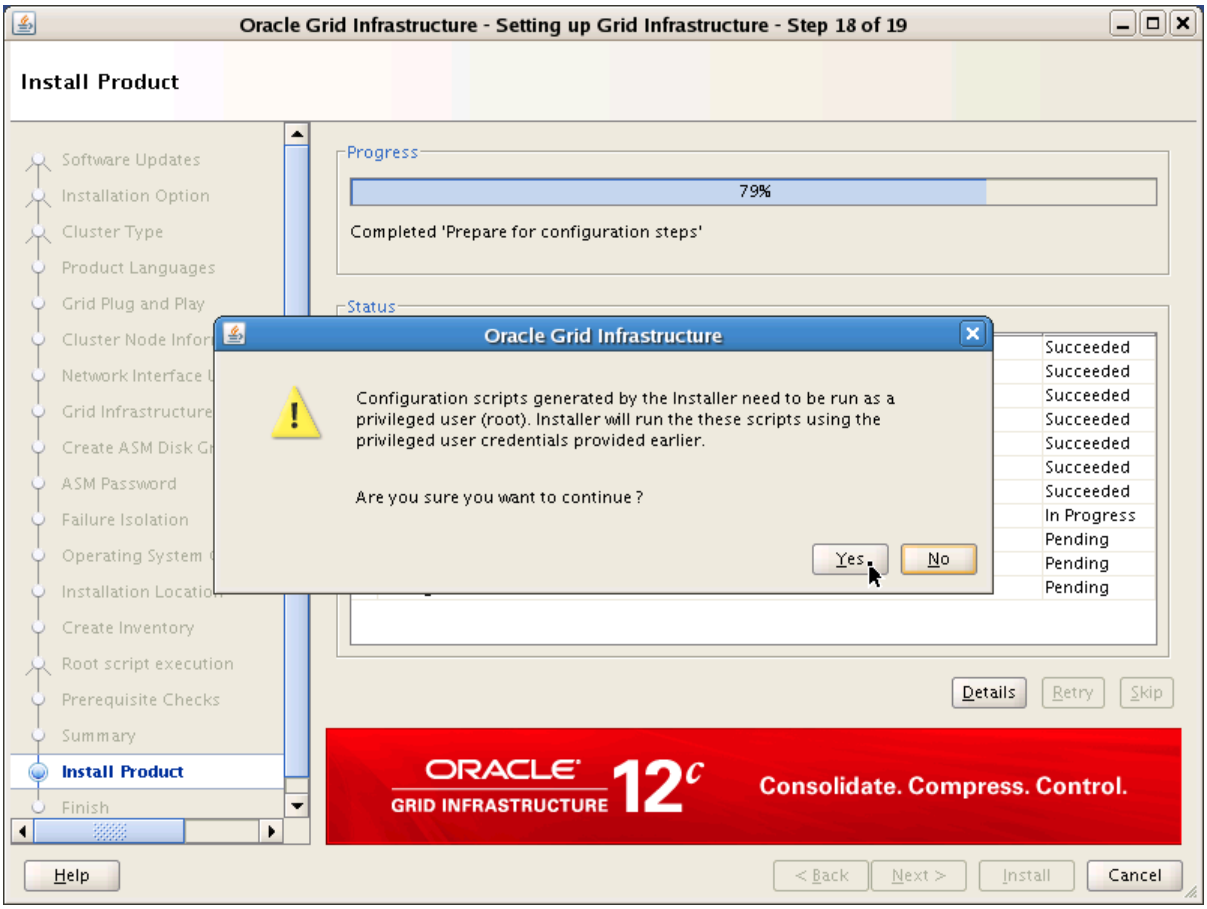
32. Examine the Summary screen. When you are ready, click Install to commence the installation.



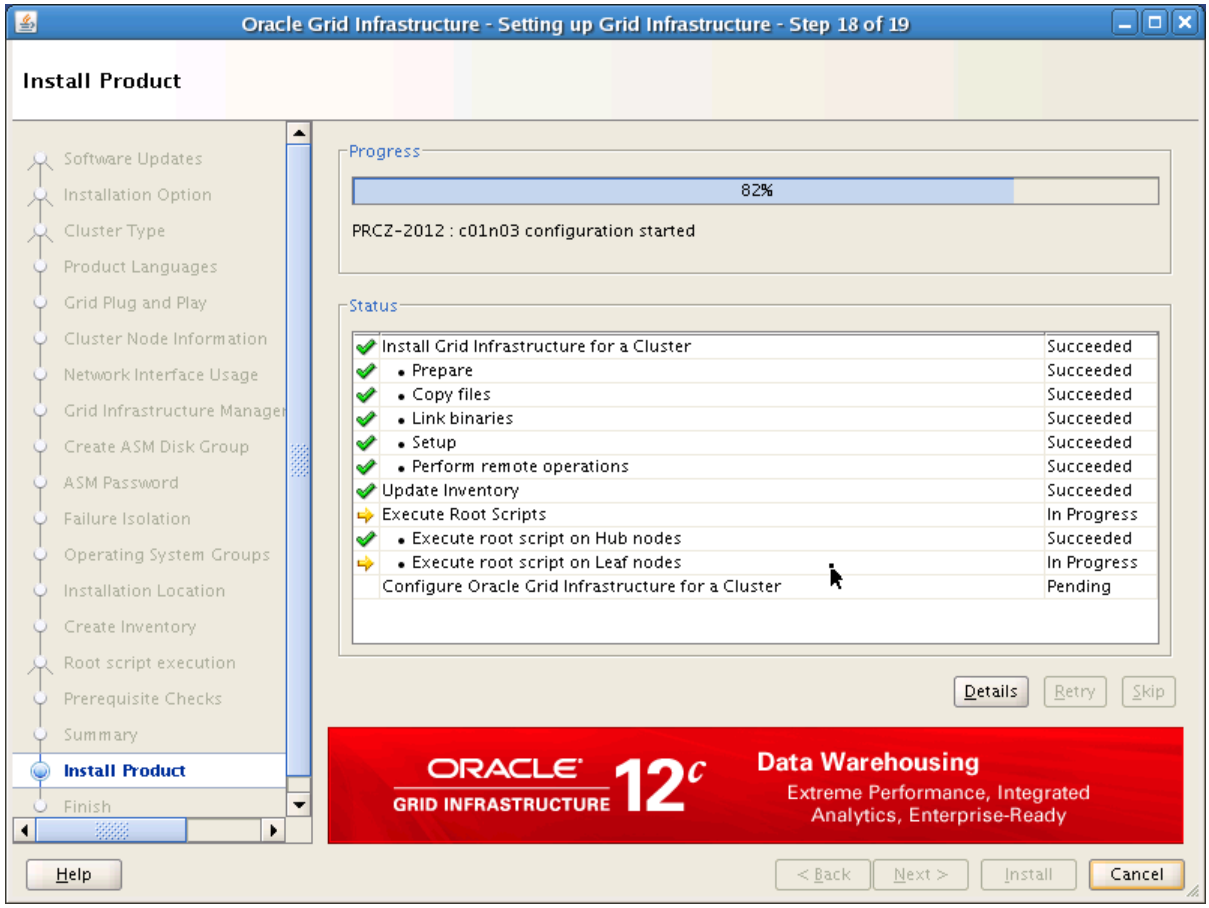
Oracle Grid Infrastructure release 12.1 now installs on the cluster. The Install Product screen shows the progress of the installation.



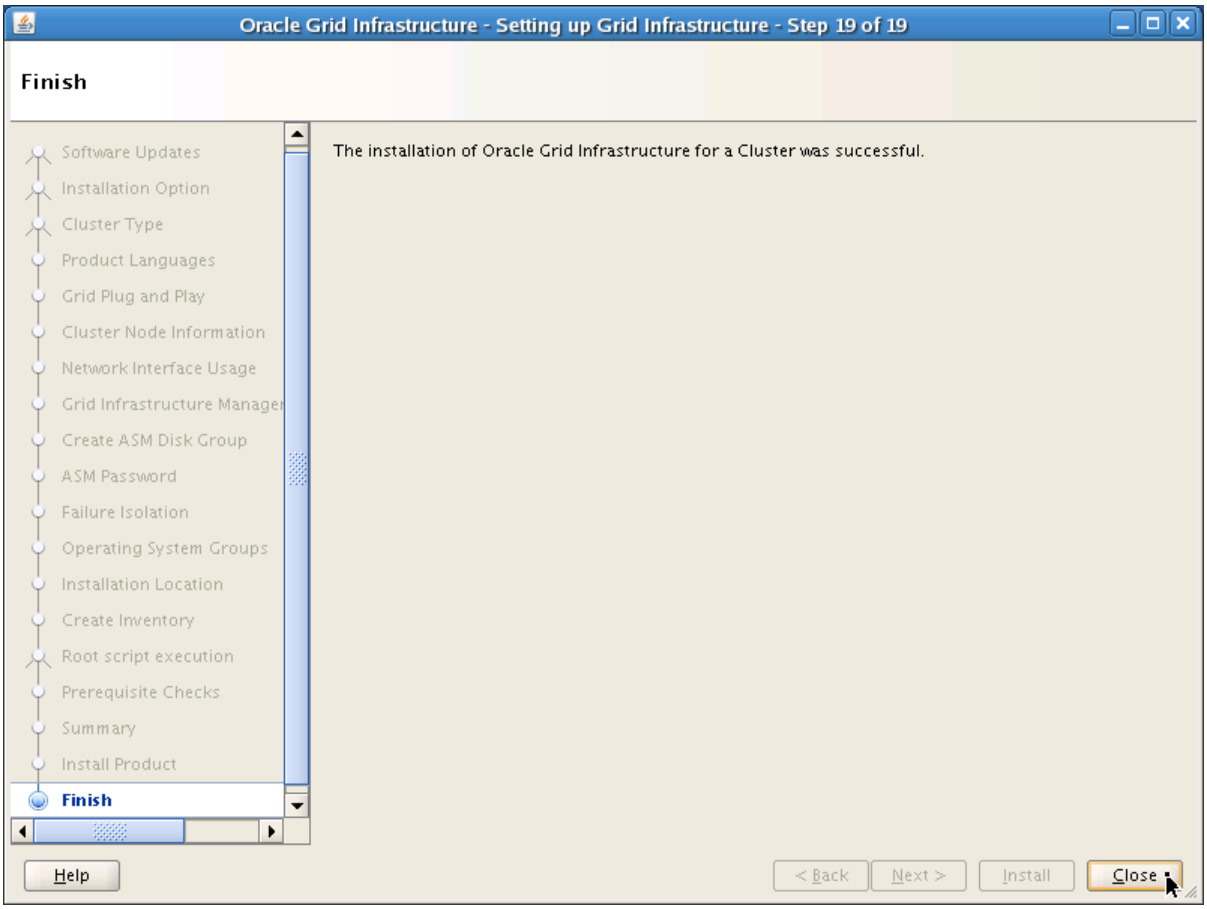
33. Oracle Universal Installer pauses prior to executing the root configuration scripts. Click Yes to proceed.



Oracle Universal Installer now automatically executes the root configuration scripts, and you can follow the progress on the Install Product screen.



34. After configuration completes, you will see the following screen. Click Close to close Oracle Universal Installer.



35. Back in your terminal session, configure the environment by using the `oraenv` script. Enter `+ASM1` when you are prompted for an ORACLE_SID value.

```
[grid@c01n01 ~]$ . oraenv
ORACLE_SID = [grid] ? +ASM1
The Oracle base has been set to /u01/app/grid
[grid@c01n01 ~]$
```

36. Now, check the status of the cluster. Ensure that all the listed services are online on all the cluster nodes.

```
[grid@c01n01 ~]$ crsctl check cluster -all
**************************************************************
c01n01:
CRS-4537: Cluster Ready Services is online
CRS-4529: Cluster Synchronization Services is online
CRS-4533: Event Manager is online
**************************************************************
c01n02:
CRS-4537: Cluster Ready Services is online
CRS-4529: Cluster Synchronization Services is online
CRS-4533: Event Manager is online
**************************************************************
c01n03:
CRS-4537: Cluster Ready Services is online
CRS-4529: Cluster Synchronization Services is online
CRS-4533: Event Manager is online
**************************************************************
c01n04:
CRS-4537: Cluster Ready Services is online
CRS-4529: Cluster Synchronization Services is online
CRS-4533: Event Manager is online
**************************************************************
[grid@c01n01 ~]$
```

37. List the Clusterware resources. Ensure that all the Clusterware resources are running as shown in the following output. Notice the following new or altered resources in release 12.1:
    - Flex ASM listeners: `ora.ASMNET1LSNR_ASM.lsnr`
    - Leaf listeners: `ora.LISTENER_LEAF.lsnr`
    - Flex ASM ADVM Proxy instances: `ora.proxy_advm`
    - Flex ASM instances: `ora.asm`
    - Grid Infrastructure Management Repository listener: `ora.MGMTLSNR`
    - Grid Infrastructure Management Repository instance: `ora.mgmtdb`

```
[grid@c01n01 ~]$ crsctl status resource -t
--------------------------------------------------------------------------------
Name           Target  State        Server                   State details
--------------------------------------------------------------------------------
Local Resources
--------------------------------------------------------------------------------
ora.ASMNET1LSNR_ASM.lsnr
               ONLINE  ONLINE       c01n01                   STABLE
```

```
               ONLINE  ONLINE     c01n02         STABLE
ora.DATA.dg
               ONLINE  ONLINE     c01n01         STABLE
               ONLINE  ONLINE     c01n02         STABLE
ora.LISTENER.lsnr
               ONLINE  ONLINE     c01n01         STABLE
               ONLINE  ONLINE     c01n02         STABLE
ora.LISTENER_LEAF.lsnr
               OFFLINE OFFLINE    c01n03         STABLE
               OFFLINE OFFLINE    c01n04         STABLE
ora.net1.network
               ONLINE  ONLINE     c01n01         STABLE
               ONLINE  ONLINE     c01n02         STABLE
ora.ons
               ONLINE  ONLINE     c01n01         STABLE
               ONLINE  ONLINE     c01n02         STABLE
ora.proxy_advm
               ONLINE  ONLINE     c01n01         STABLE
               ONLINE  ONLINE     c01n02         STABLE
--------------------------------------------------------------------
Cluster Resources
--------------------------------------------------------------------
ora.LISTENER_SCAN1.lsnr
      1        ONLINE  ONLINE     c01n02         STABLE
ora.LISTENER_SCAN2.lsnr
      1        ONLINE  ONLINE     c01n01         STABLE
ora.LISTENER_SCAN3.lsnr
      1        ONLINE  ONLINE     c01n01         STABLE
ora.MGMTLSNR
      1        ONLINE  ONLINE     c01n01         169.254.62.237
                                                 192.168.1.11,
                                                 STABLE
ora.asm
      1        ONLINE  ONLINE     c01n01         STABLE
      2        ONLINE  ONLINE     c01n02         STABLE
      3        OFFLINE OFFLINE                   STABLE
ora.c01n01.vip
      1        ONLINE  ONLINE     c01n01         STABLE
ora.c01n02.vip
      1        ONLINE  ONLINE     c01n02         STABLE
ora.cvu
      1        ONLINE  ONLINE     c01n01         STABLE
ora.gns
```

```
      1        ONLINE  ONLINE      c01n01          STABLE
ora.gns.vip
      1        ONLINE  ONLINE      c01n01          STABLE
ora.mgmtdb
      1        ONLINE  ONLINE      c01n01          Open,STABLE
ora.oc4j
      1        ONLINE  ONLINE      c01n01          STABLE
ora.scan1.vip
      1        ONLINE  ONLINE      c01n02          STABLE
ora.scan2.vip
      1        ONLINE  ONLINE      c01n01          STABLE
ora.scan3.vip
      1        ONLINE  ONLINE      c01n01          STABLE
--------------------------------------------------------------------------------
[grid@c01n01 ~]$
```

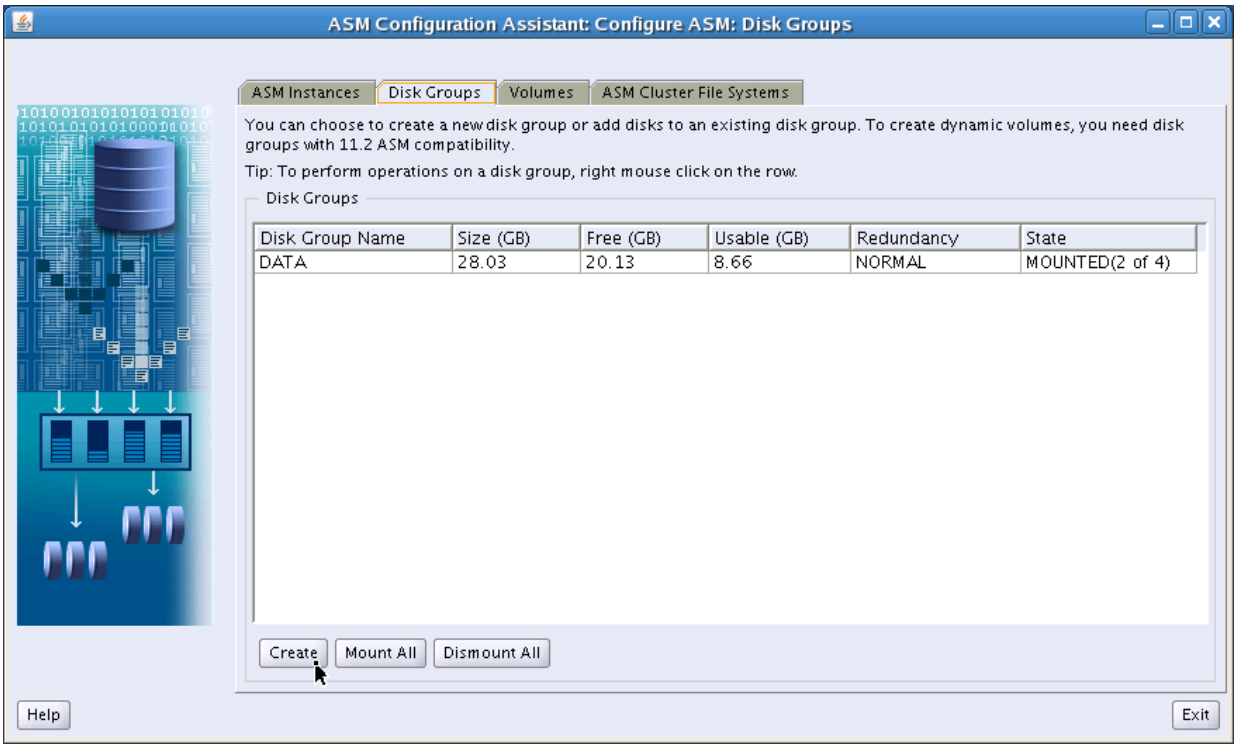
38. Stop the Grid Infrastructure Management Repository and associated listener. This will effectively disable the collection of system metrics by Cluster Health Monitor and prevent the use of other features such as QoS Management. Normally you would not perform this action on your cluster; however, the processing limitations of this laboratory environment require it to speed up the upcoming tasks.

```
[grid@c01n01 ~]$ srvctl stop mgmtdb
[grid@c01n01 ~]$ srvctl stop mgmtlsnr
[grid@c01n01 ~]$
```

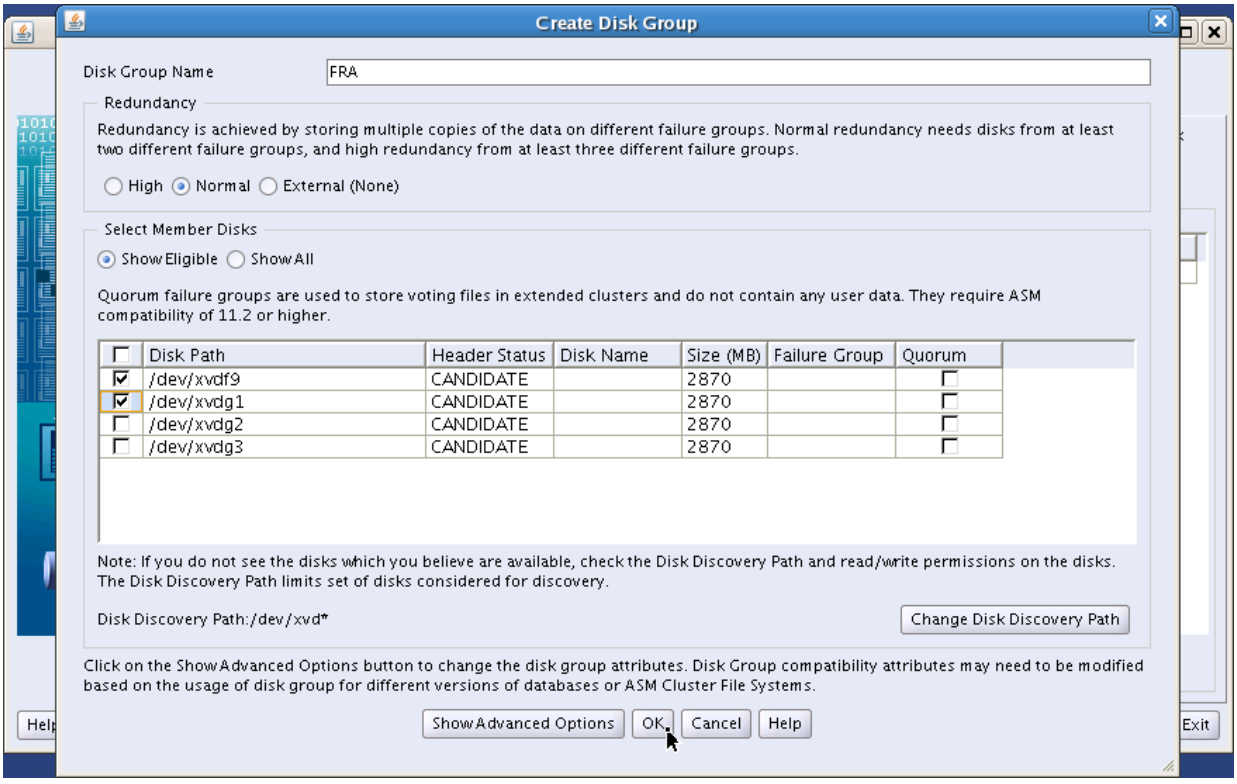
39. At this point, you have configured a Flex Cluster with Flex ASM. Later you will create a RAC database on the cluster. In preparation for this, you will now create another ASM disk group to house the Fast Recovery Area (FRA). Start the ASM Configuration Assistant (`asmca`).

```
[grid@c01n01 ~]$ asmca
```

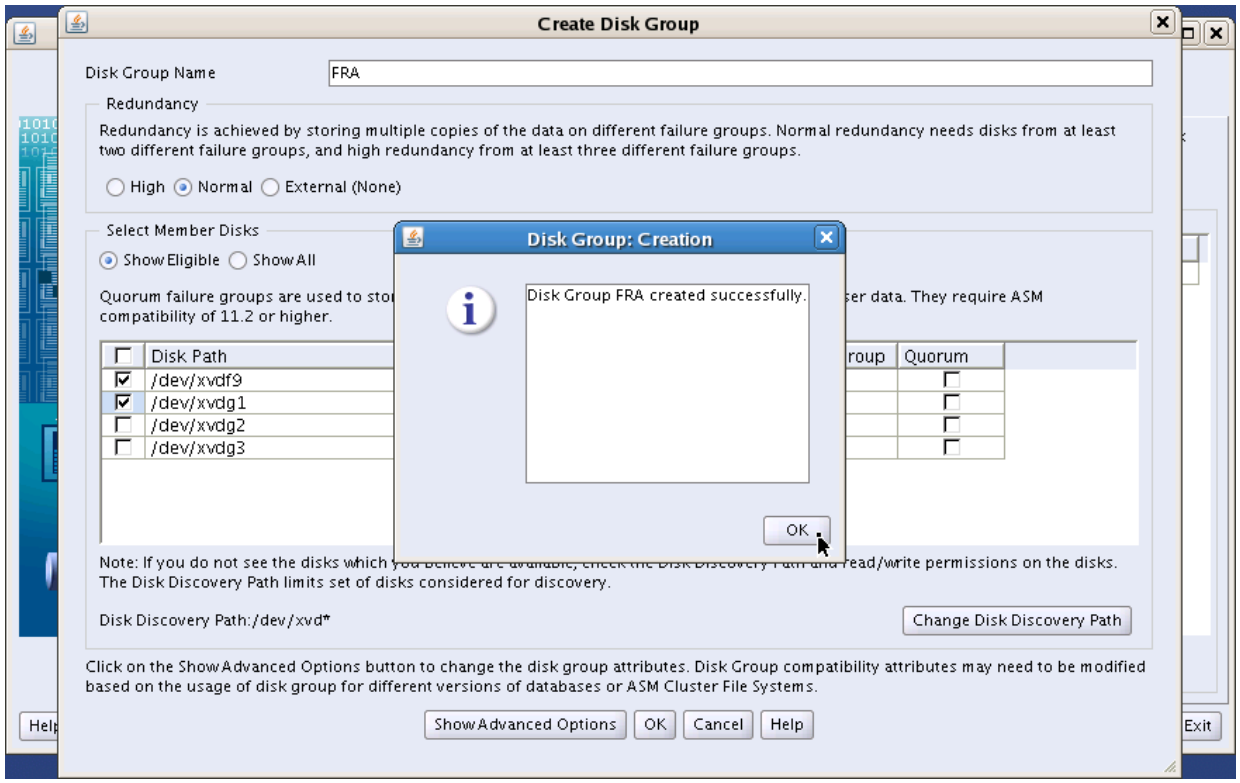
40. After the ASM Configuration Assistant appears, click Create.



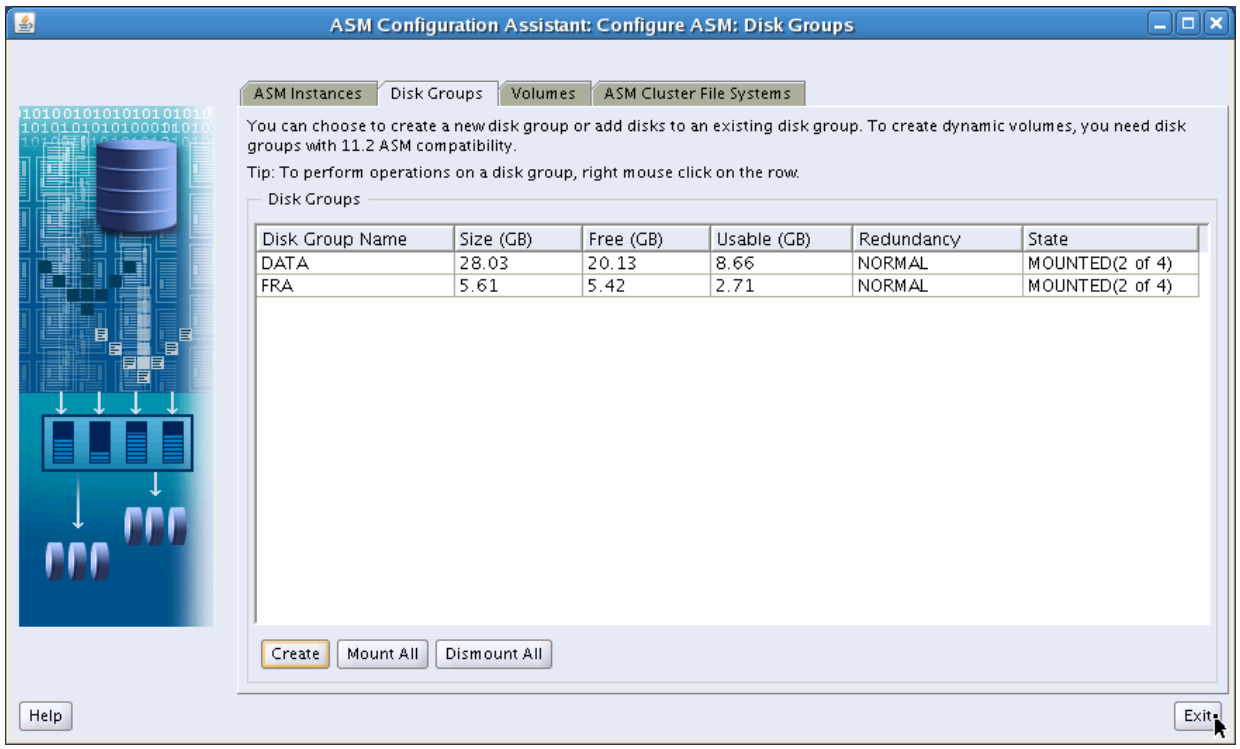
41. In the Create Disk Group window, enter `FRA` as the disk group name and select first two candidate disks (`/dev/xvdf9`, and `/dev/xvdg1`). Then click OK to create the disk group.



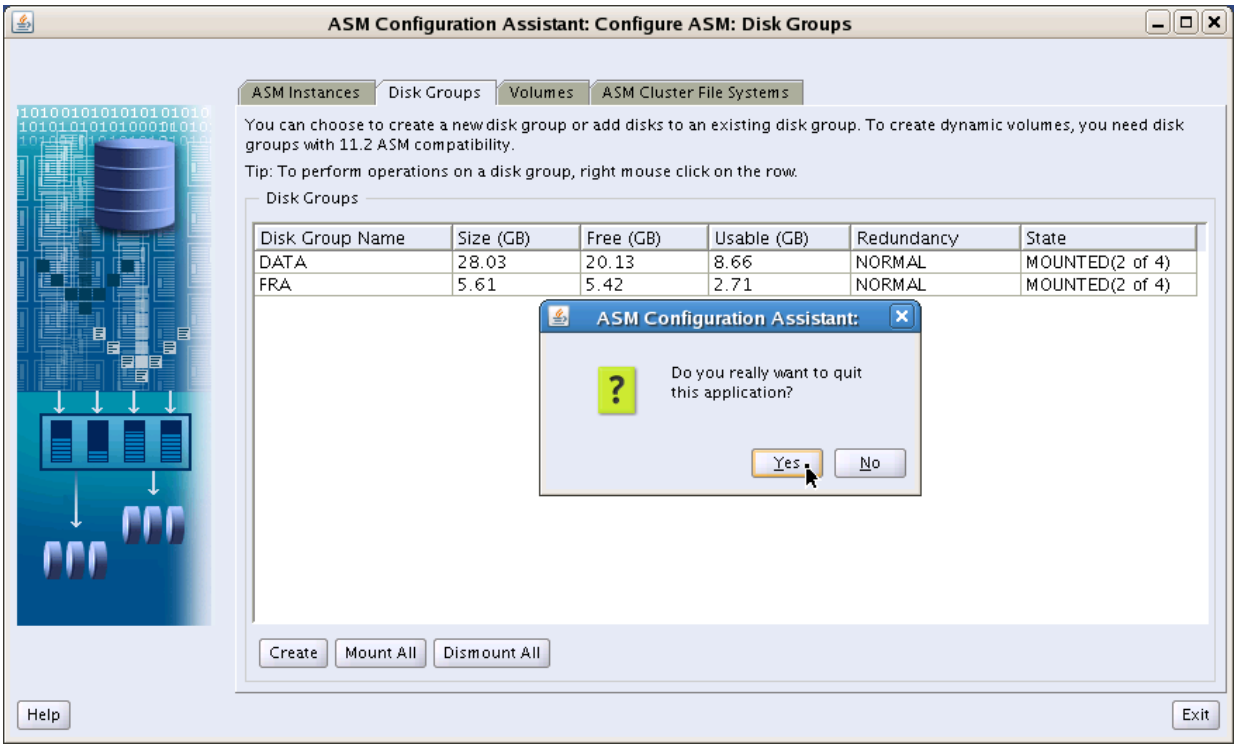
42. After the disk group creation process completes, you will see a dialog box indicating that the disk group has been created. Click OK to proceed.



43. Click Exit to quit the ASM Configuration Assistant.



44. Click Yes to confirm that you want to quit the ASM Configuration Assistant.



45. Next, you will install the Oracle Real Application Clusters (RAC) software. Establish a new terminal session connected to `c01n01` by using the `root` OS user.

```
$ ssh root@c01n01
root@c01n01's password: <oracle>
[root@c01n01 ~]#
```

46. Create a new base directory for the Oracle Database software installation and set the required ownership settings for the new directory.

```
[root@c01n01 ~]# mkdir -p /u01/app/oracle
[root@c01n01 ~]# chown oracle:oinstall /u01/app/oracle
[root@c01n01 ~]#
```

47. Create the base directory on the other cluster nodes.

```
[root@c01n01 ~]# ssh c01n02 "mkdir -p /u01/app/oracle; chown
oracle:oinstall /u01/app/oracle"
[root@c01n01 ~]# ssh c01n03 "mkdir -p /u01/app/oracle; chown
oracle:oinstall /u01/app/oracle"
[root@c01n01 ~]# ssh c01n04 "mkdir -p /u01/app/oracle; chown
oracle:oinstall /u01/app/oracle"
[root@c01n01 ~]#
```

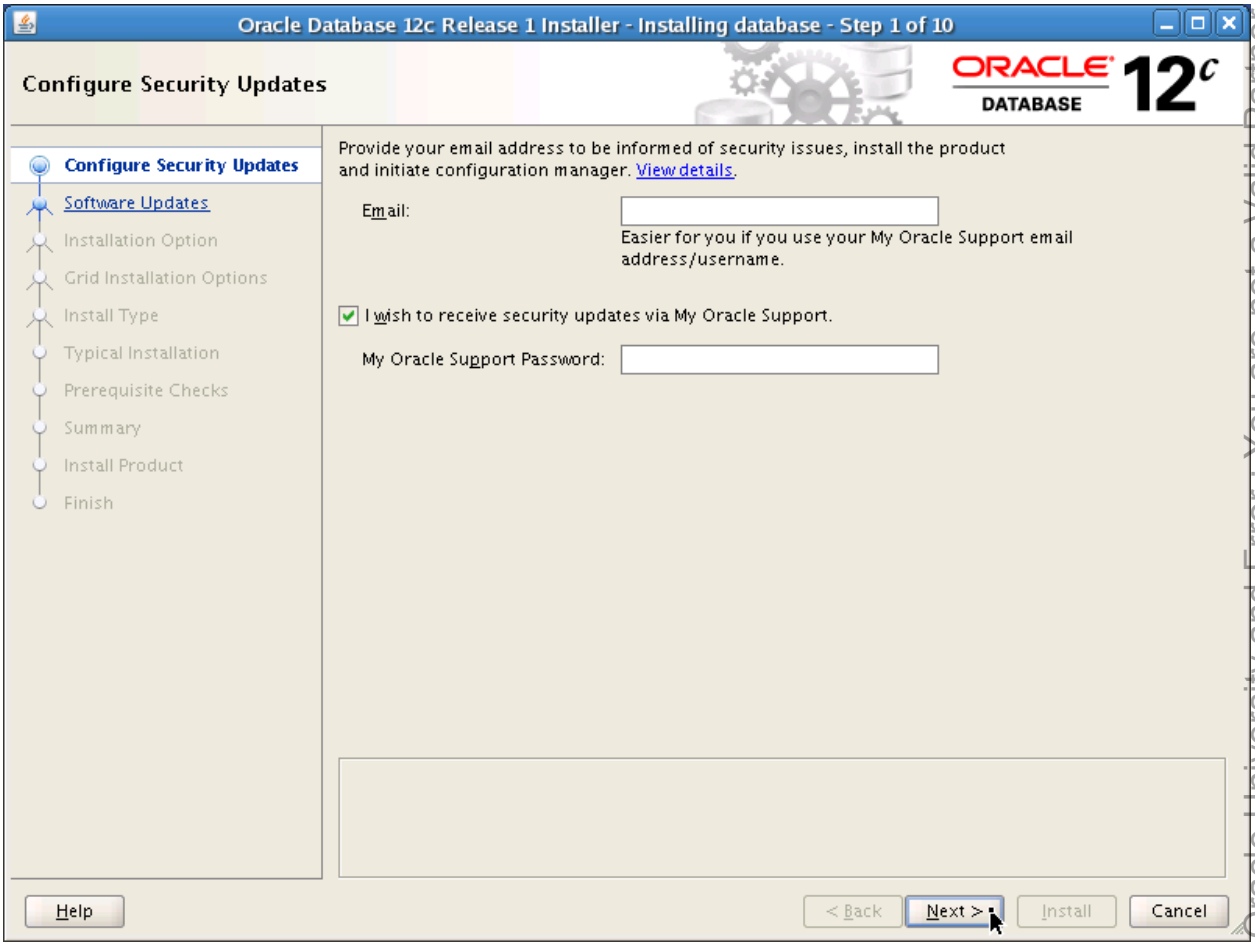
48. Establish another new terminal session connected to `c01n01` by using the `oracle` OS user. Ensure that you specify the `-X` option for `ssh`.

```
$ ssh -X oracle@c01n01
oracle@c01n01's password: <oracle>
[oracle@c01n01 ~]$
```

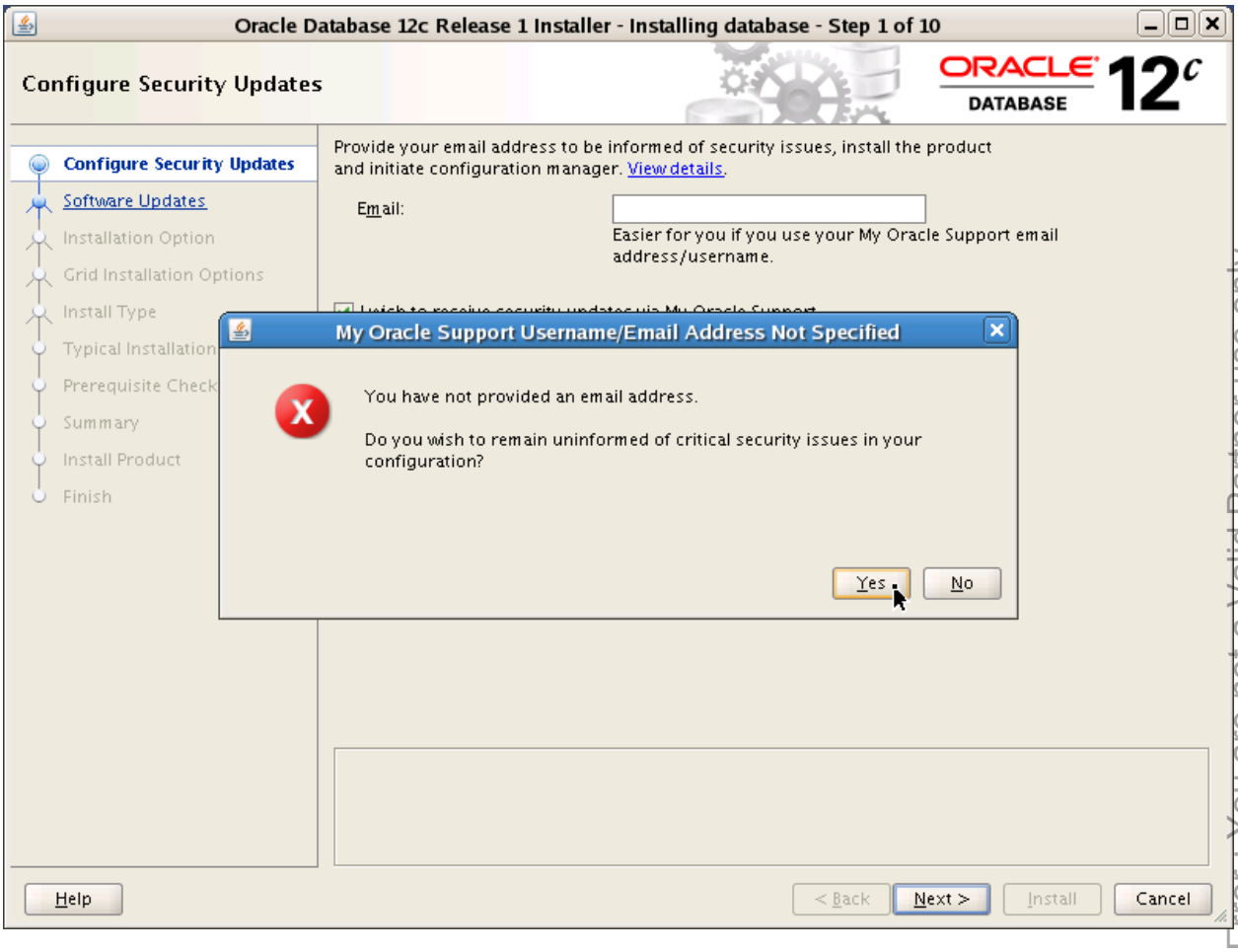
49. Start the Oracle Database release 12.1 installer.

```
[oracle@c01n01 ~]$ /stage/12.1/database/runInstaller
Starting Oracle Universal Installer...
```

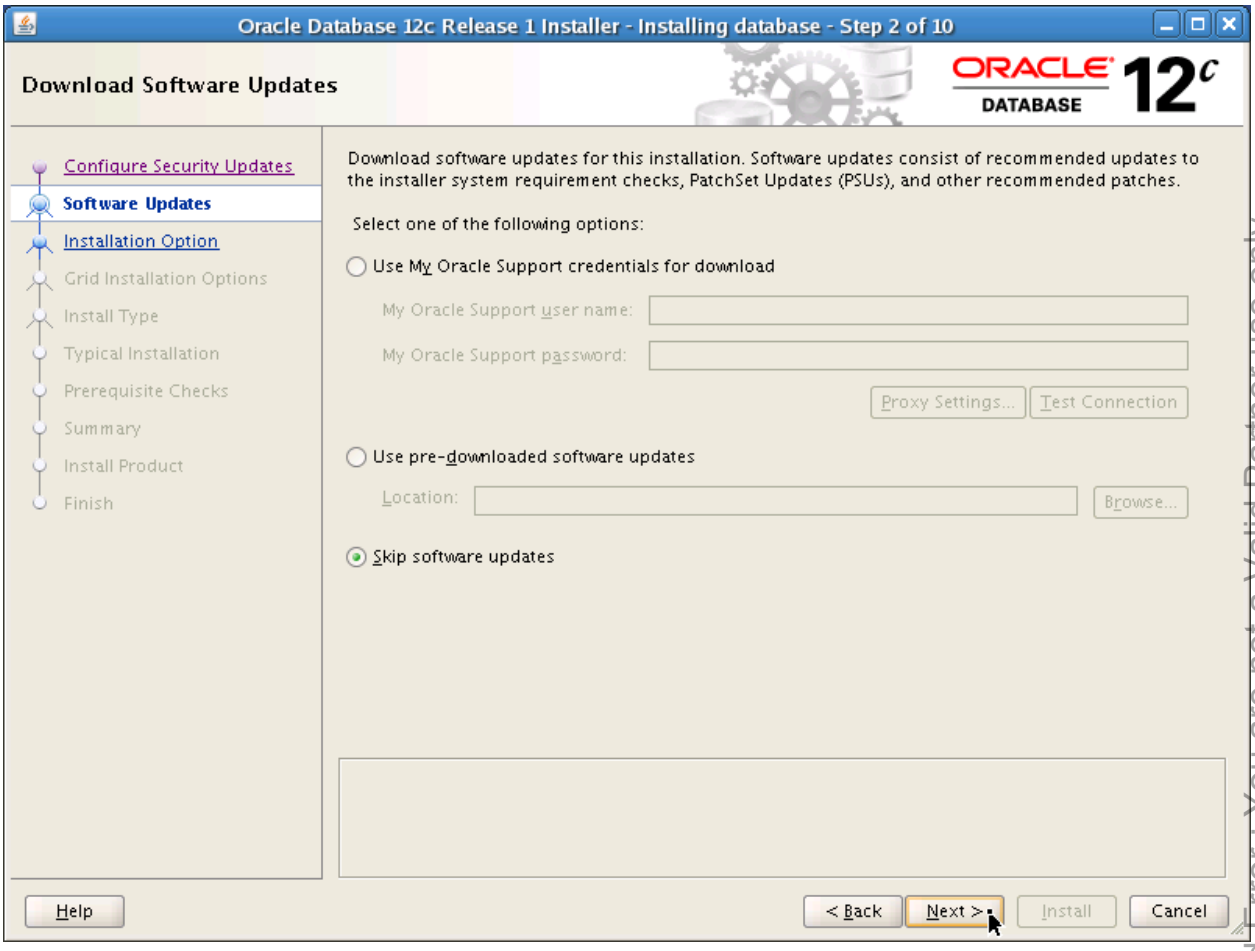
50. On the Configure Security Updates screen, click Next.



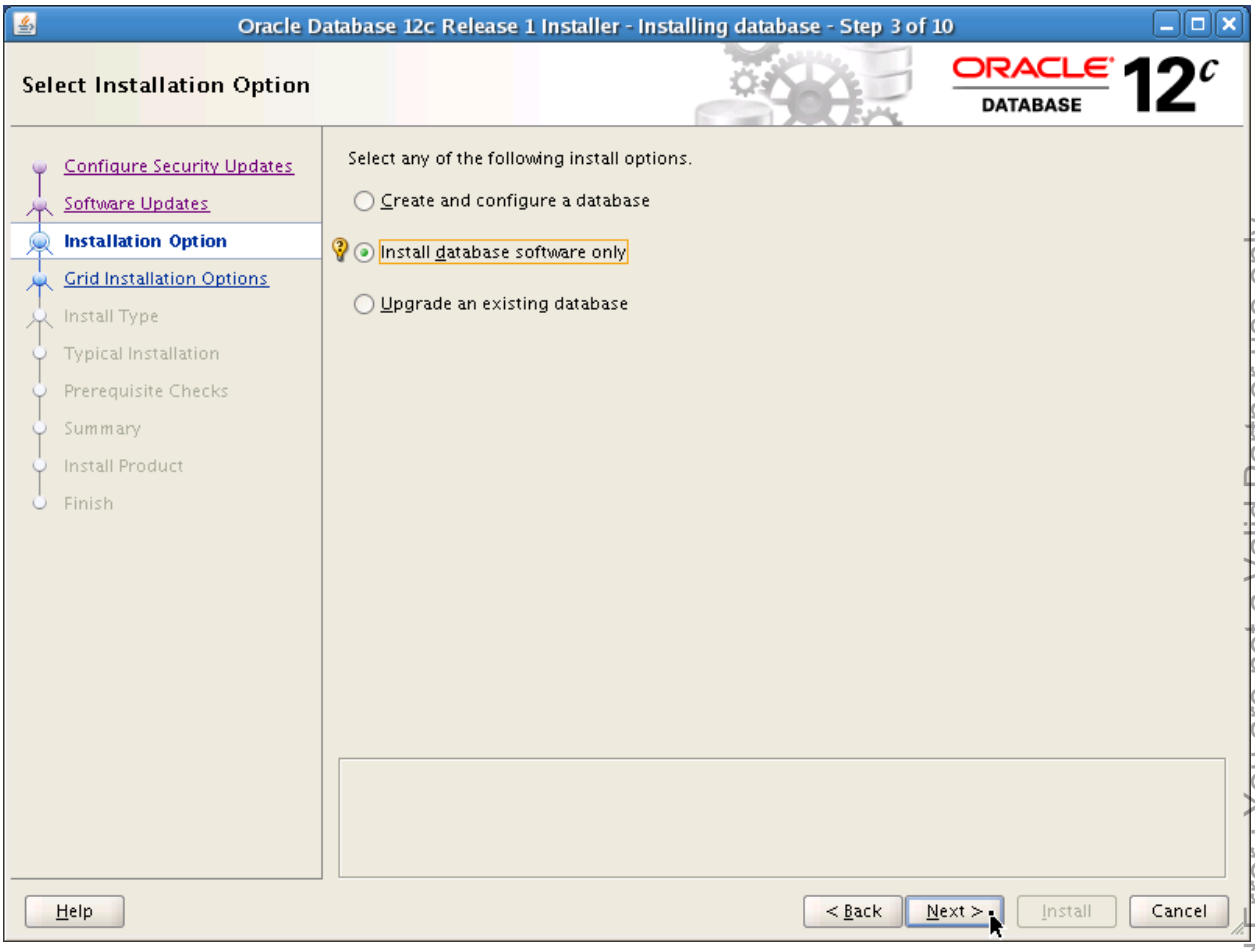
51. Click Yes to confirm that you will not configure security updates for this installation.



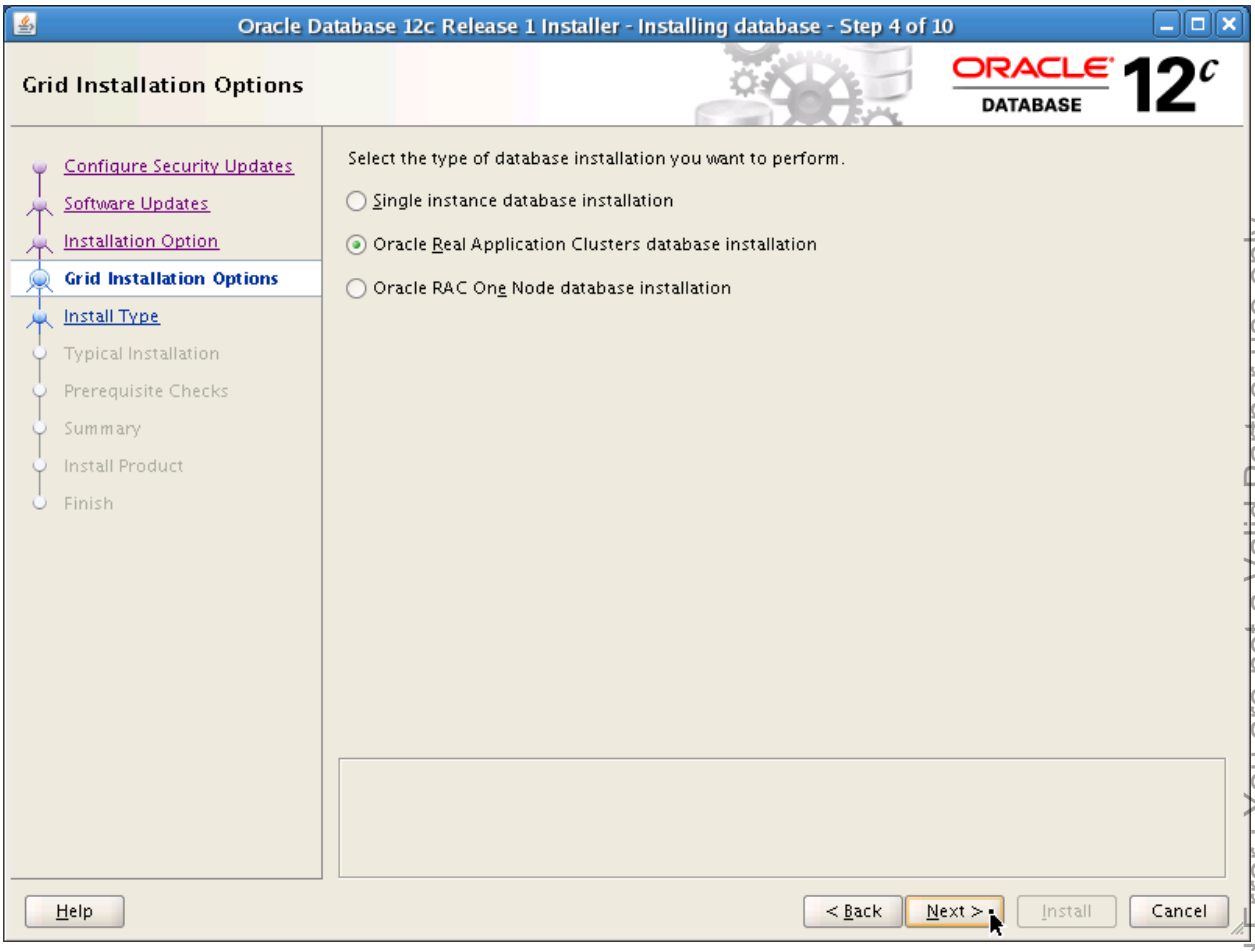
52. On the Download Software Updates screen, click Next to accept the default selection (Skip software updates).



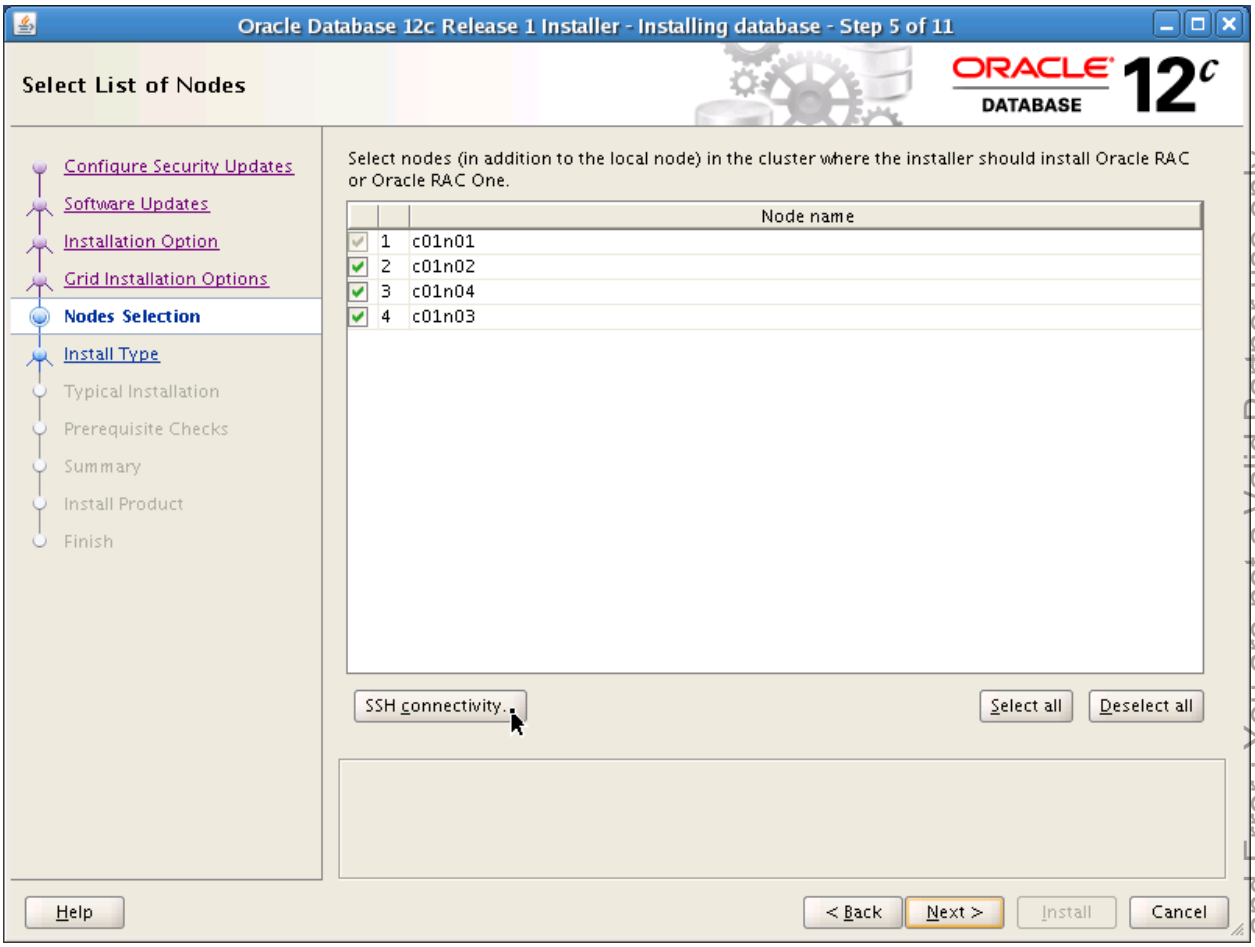
53. On the Select Installation Option screen, select “Install database software only” and click Next.



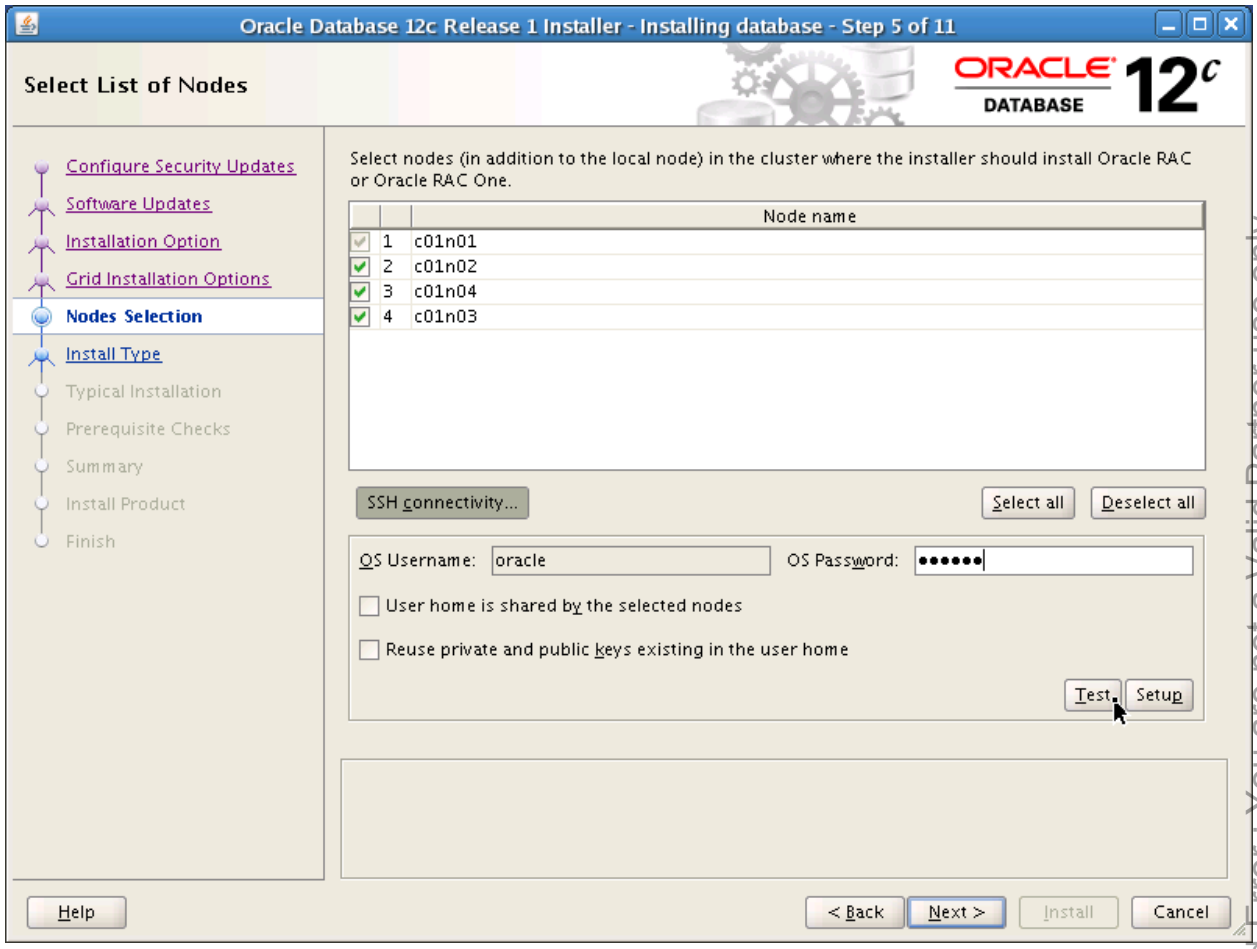
54. On the Gird Installation Options screen, click Next to accept the default selection ("Oracle Real Application Clusters database installation").



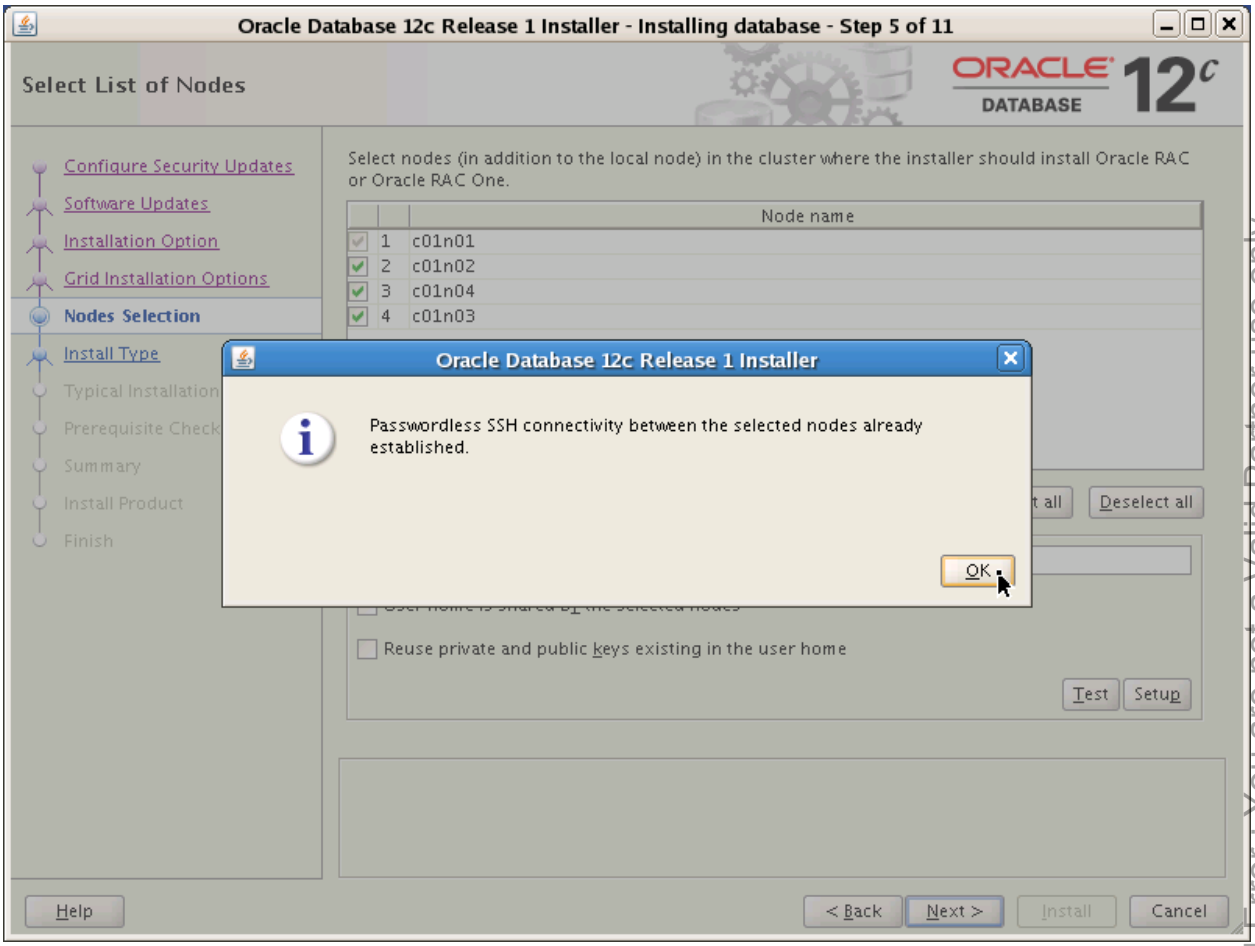
55. On the “Select List of Nodes” screen, ensure that all the cluster nodes are selected and click SSH Connectivity. Note that Oracle recommends that you install the Oracle Database software on all the cluster nodes, even Leaf Nodes. This simplifies things if you ever want to convert a Leaf Node to a Hub Node and run database instances on it.



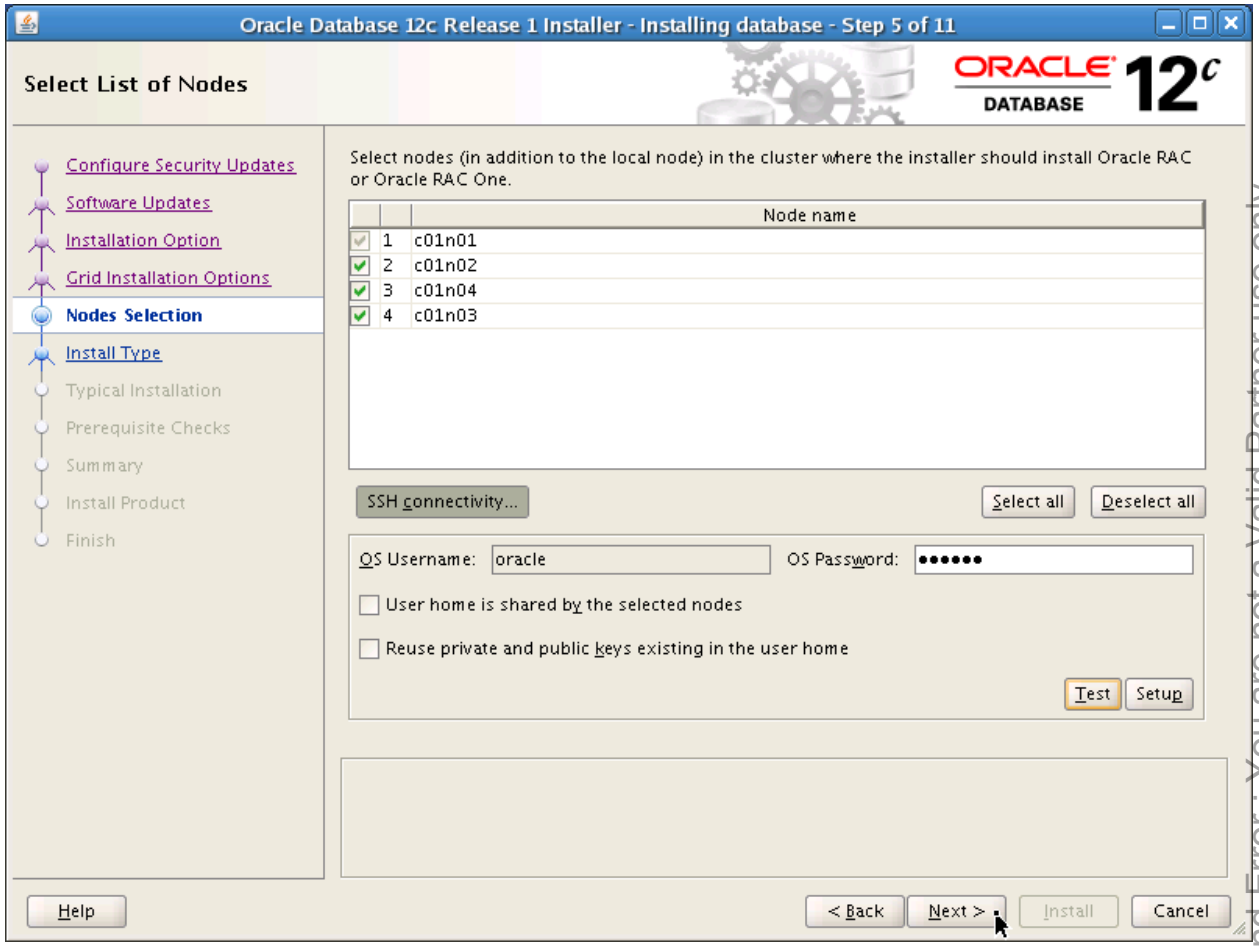
56. Enter `oracle` into the OS Password field and click Test to confirm that the required SSH connectivity is configured across the cluster.



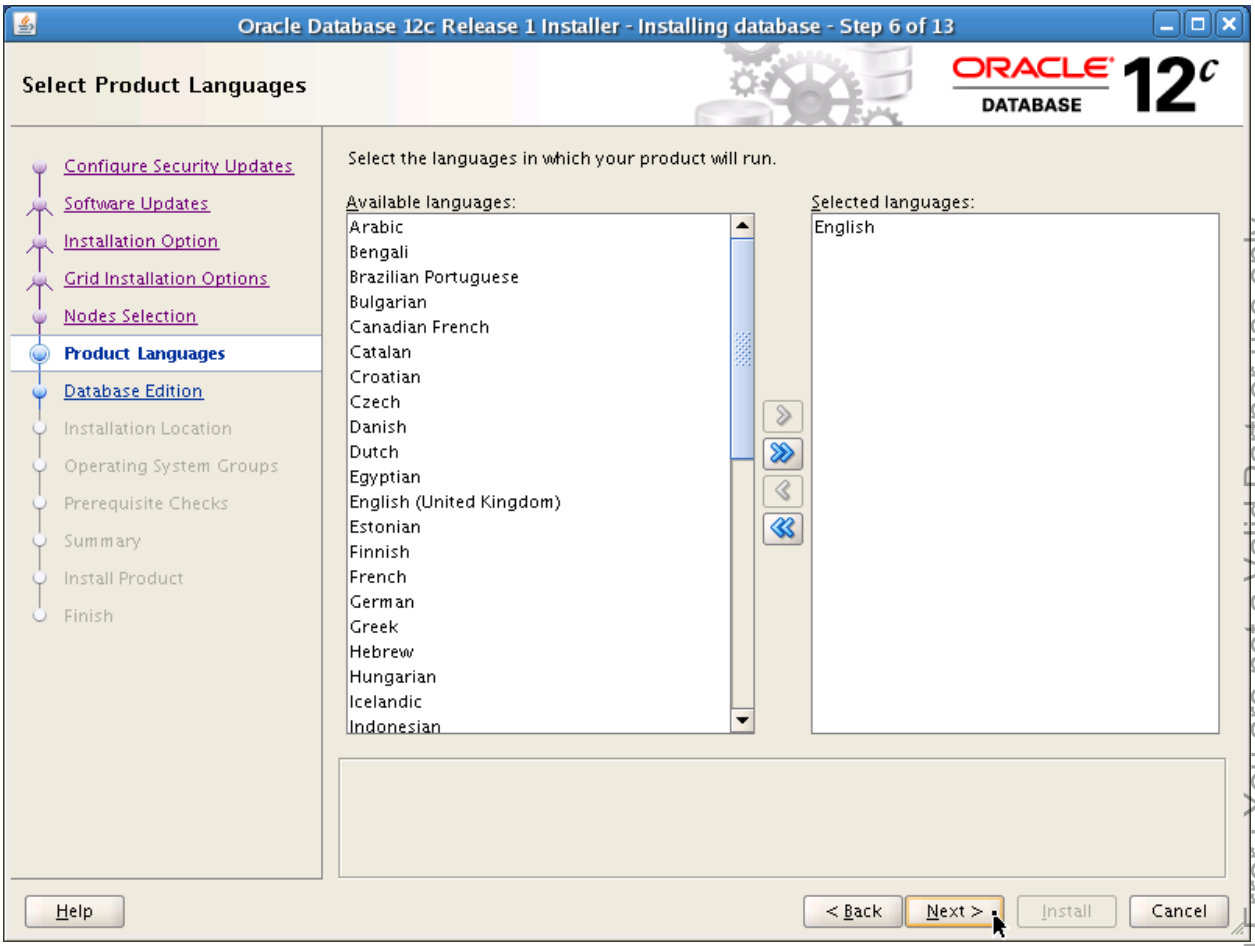
57. Because your laboratory environment is preconfigured with the required SSH connectivity, you will see a dialog box confirming this. Click OK to continue.



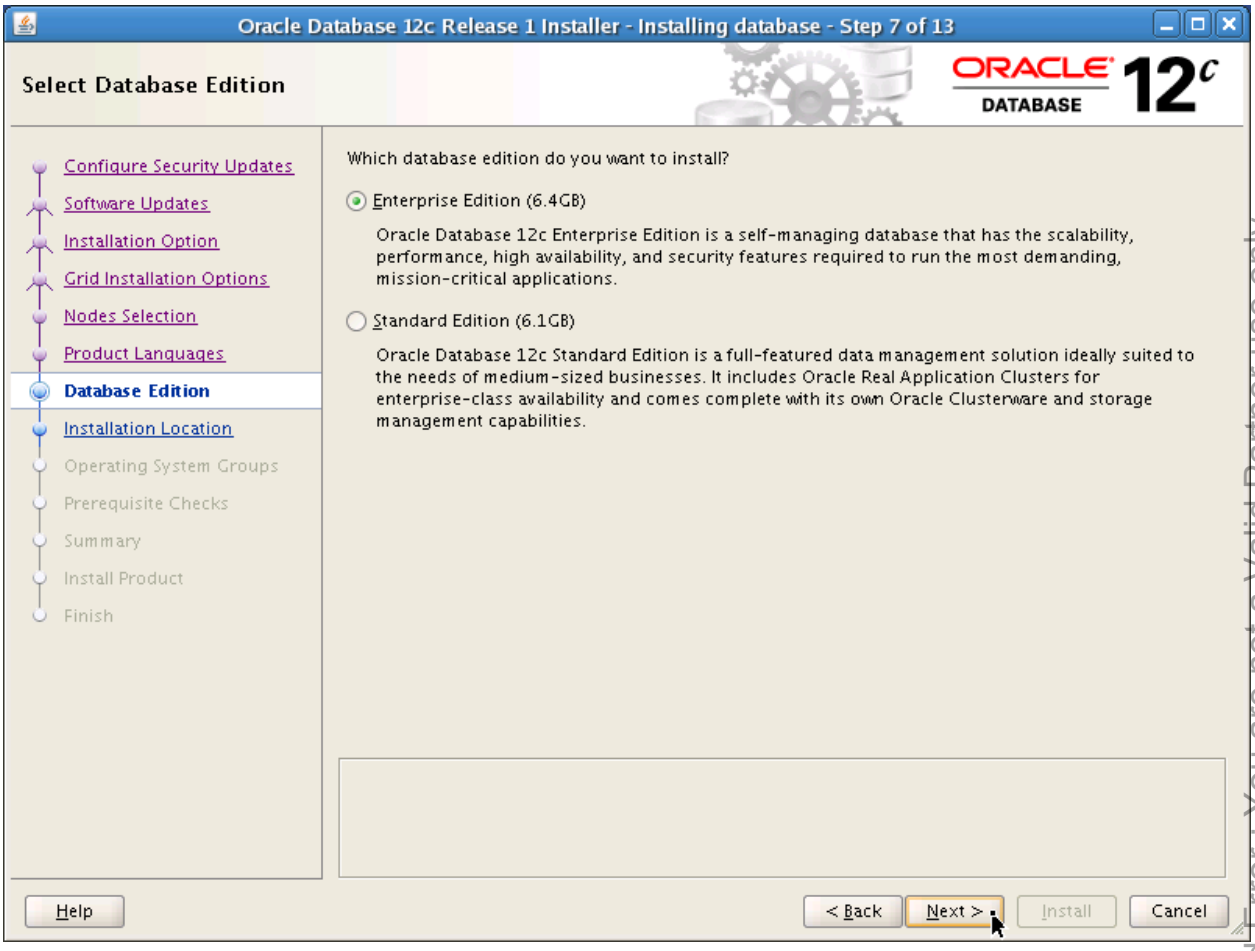
58. If the required SSH connectivity were not present, you could now click Setup to perform the required configuration. However, because the laboratory environment is already configured correctly, click Next to continue.



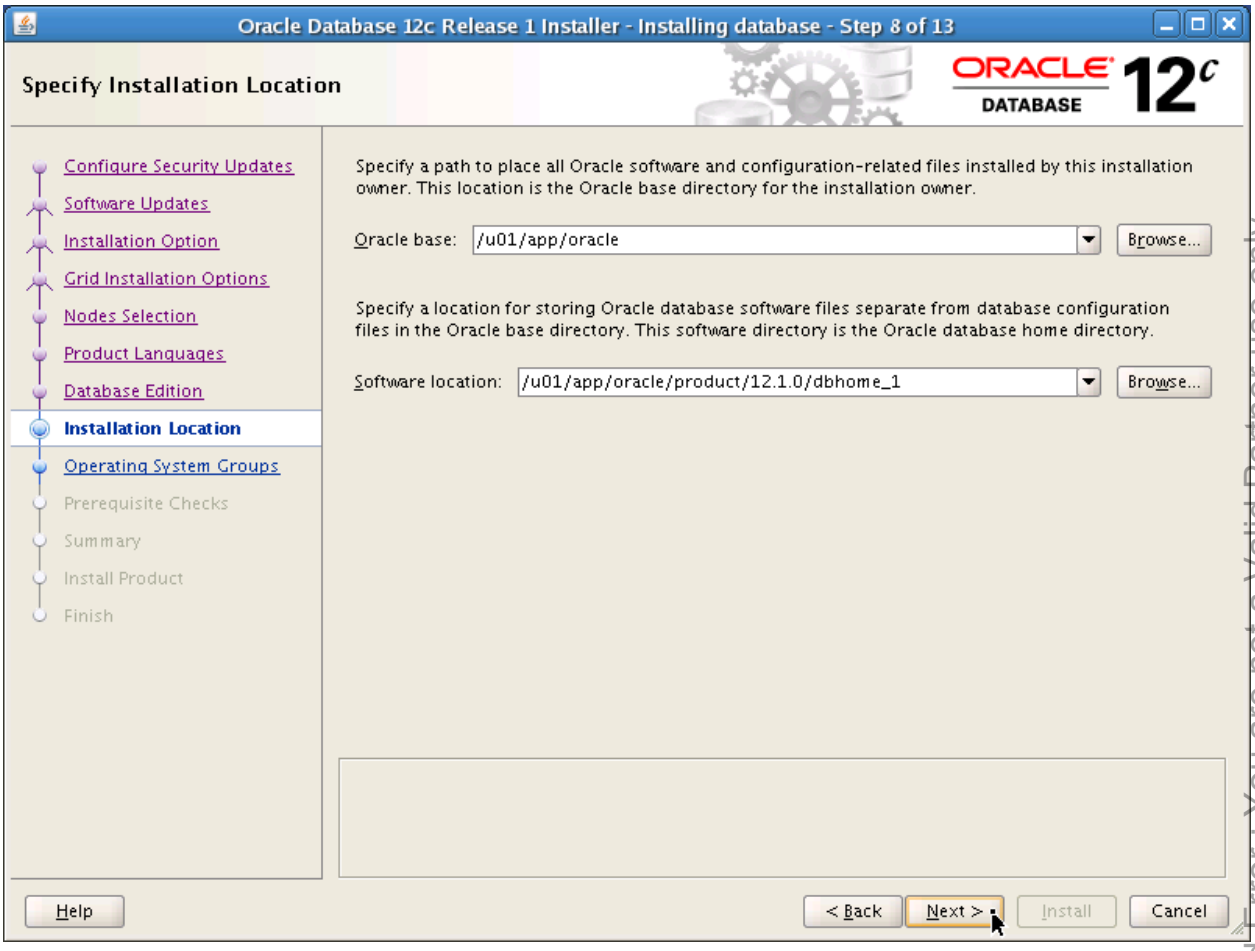
59. On the Select Product Languages screen, click Next to accept the default selection (English).



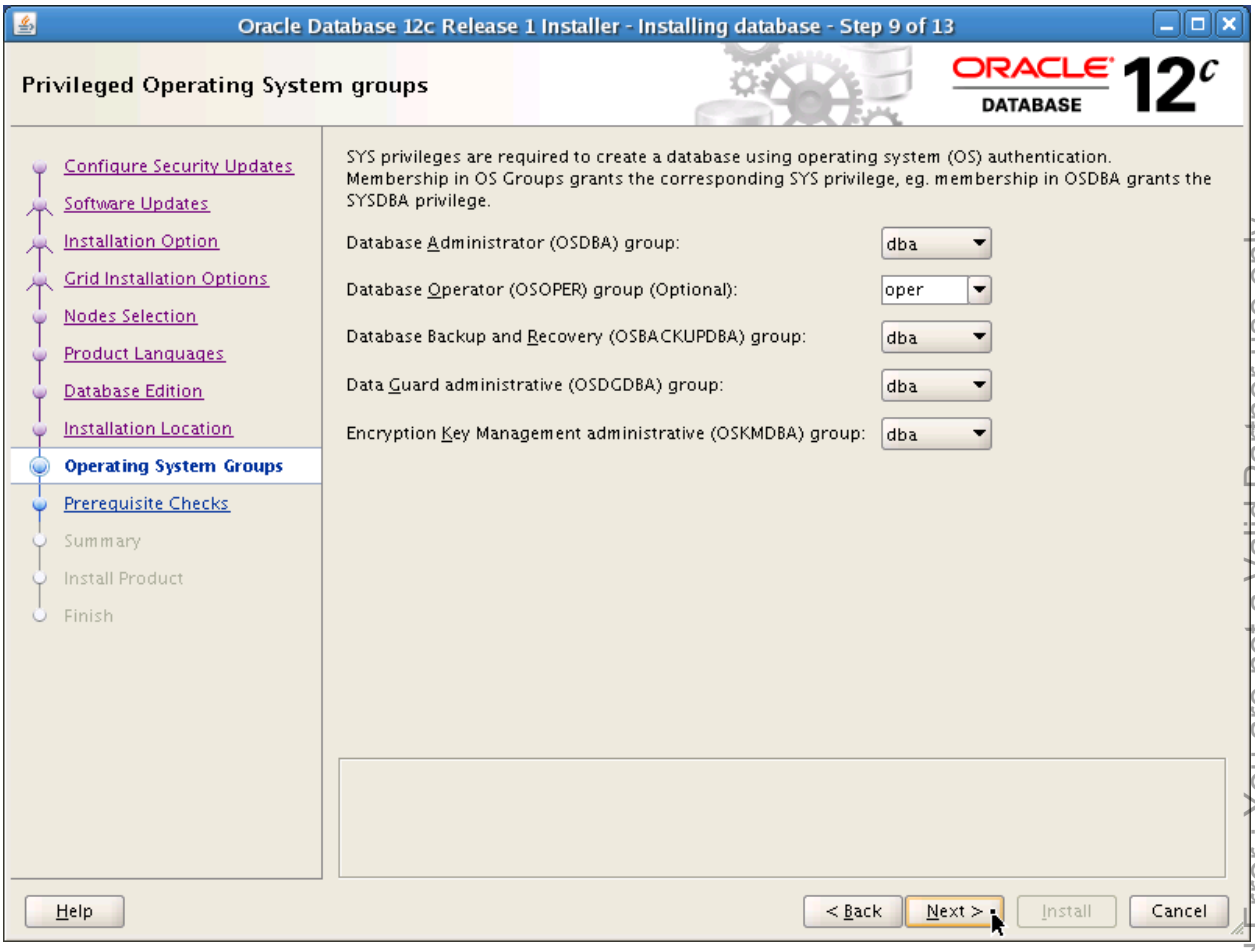
60. On the Select Database Edition screen, click Next to accept the default selection (Enterprise Edition).



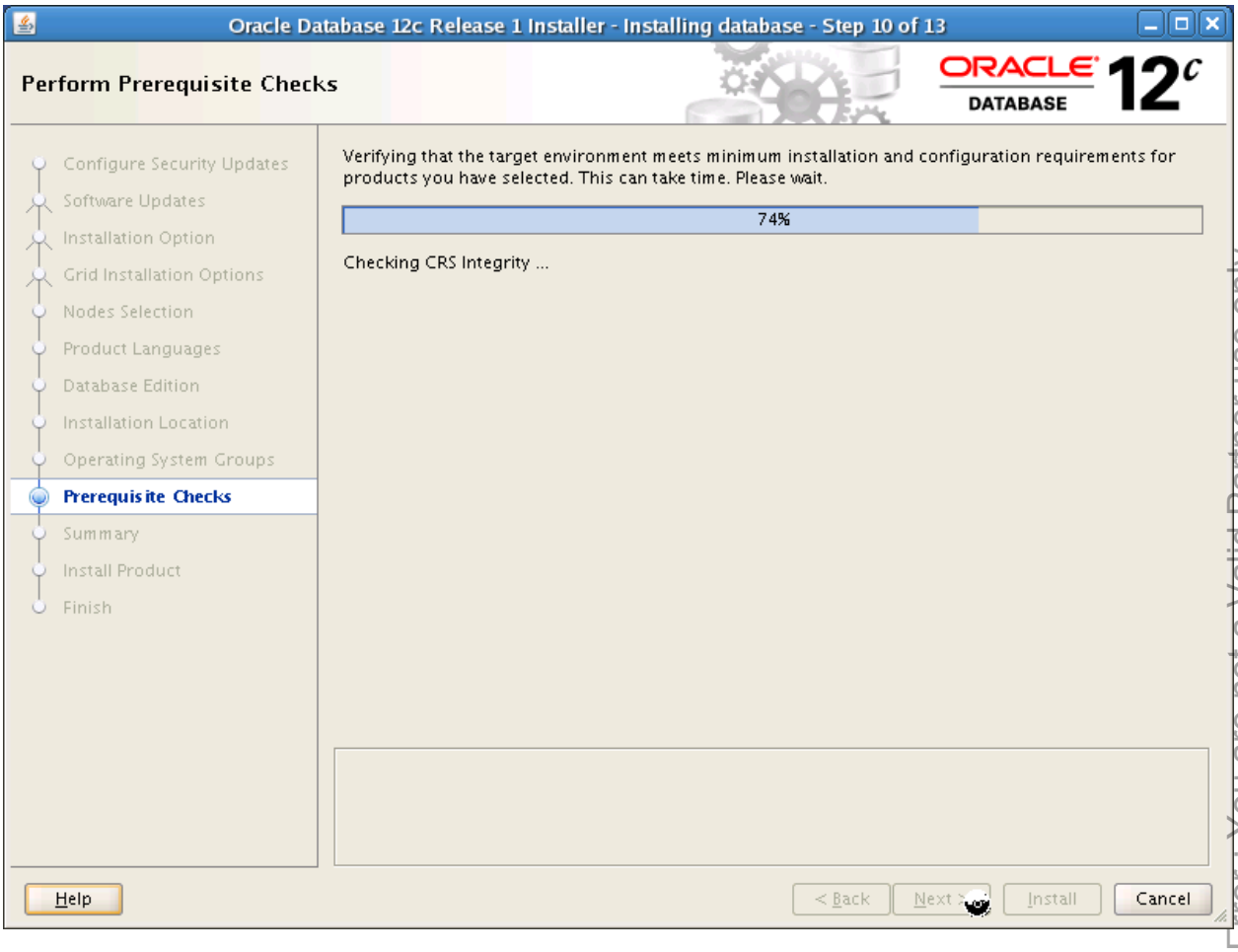
61. On the Specify Installation Location screen, click Next to accept the default installation location.



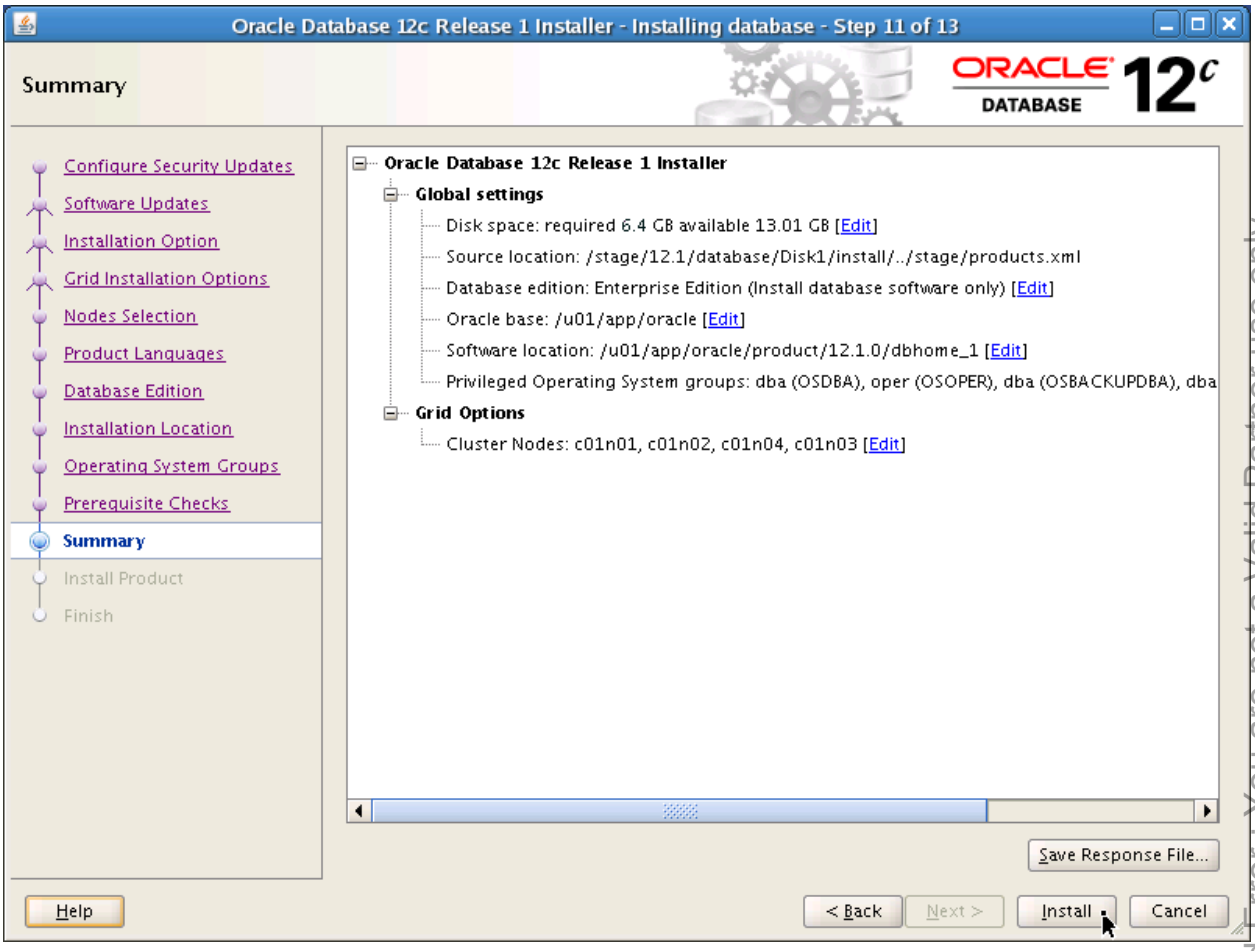
62. On the “Privileged Operating System groups” screen, click Next to accept the default settings.



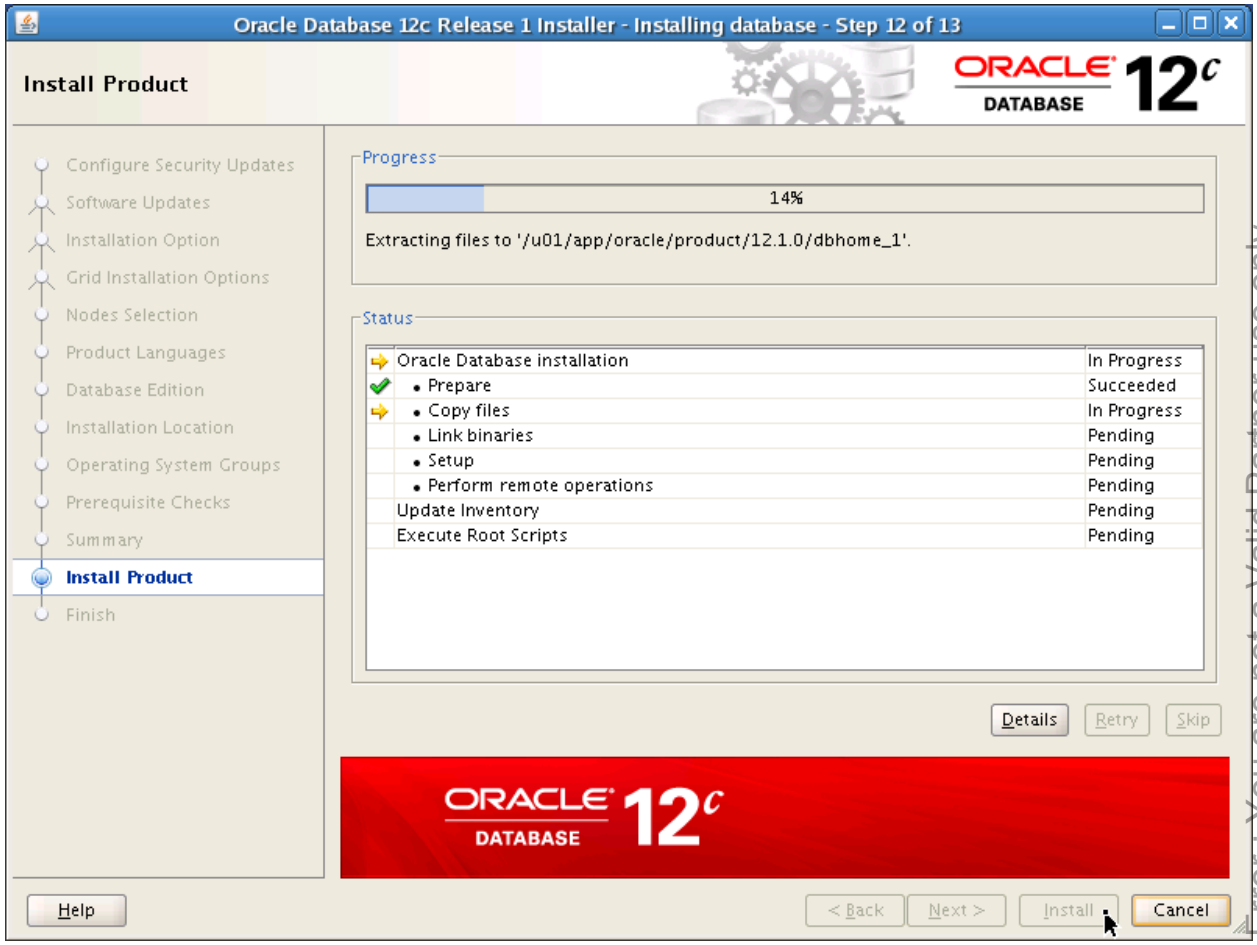
63. Wait while a series of prerequisite checks is performed.



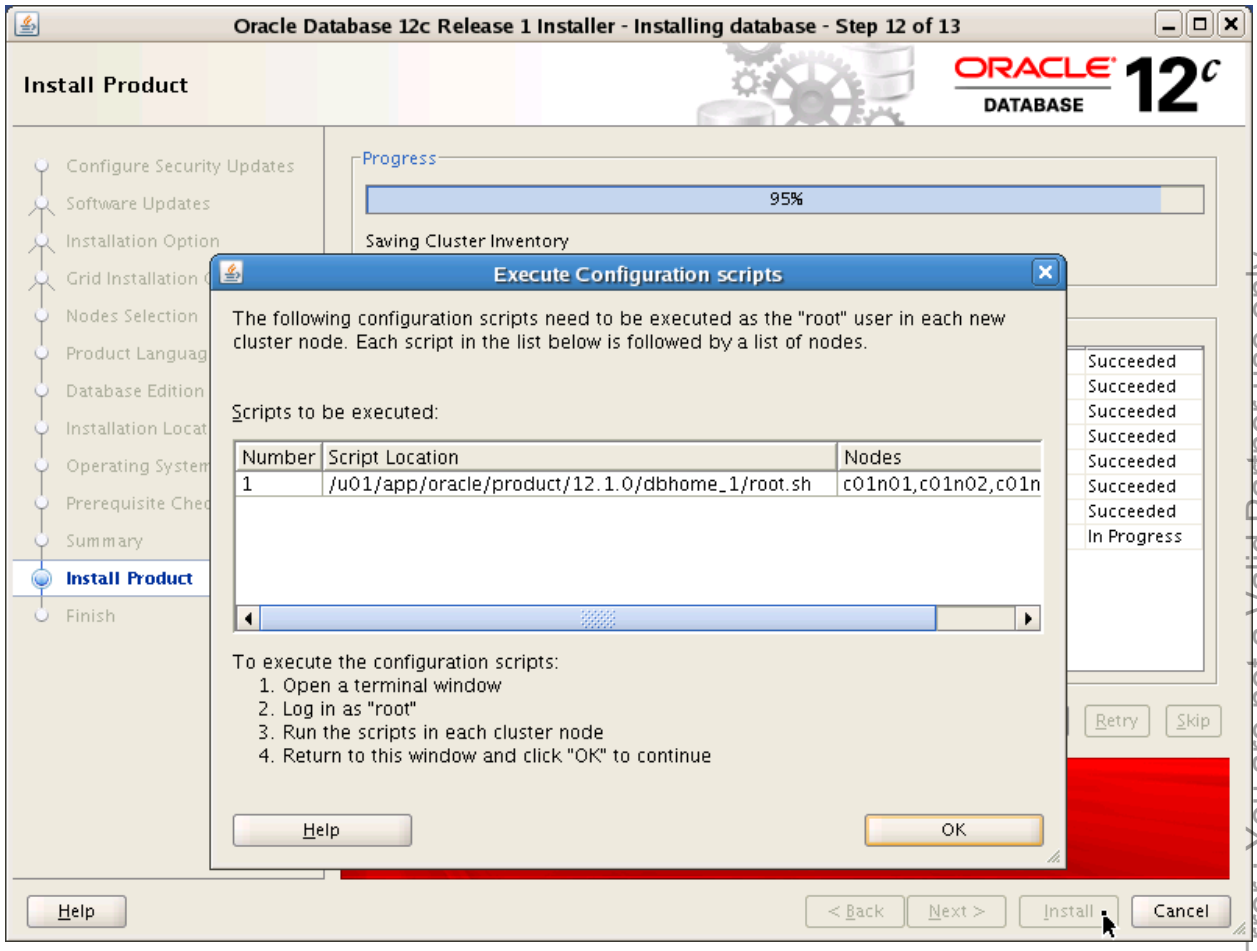
64. Examine the Summary screen. When you are ready, click Install to commence the installation.



Oracle Database release 12.1 software now installs on the cluster. The Install Product screen displays the progress of the installation.



65. Wait until you see the “Execute Configuration scripts” dialog box.



66. Back in your `root` terminal session on `c01n01`, execute the configuration script. Press the Enter key when you are prompted for the local bin directory location.

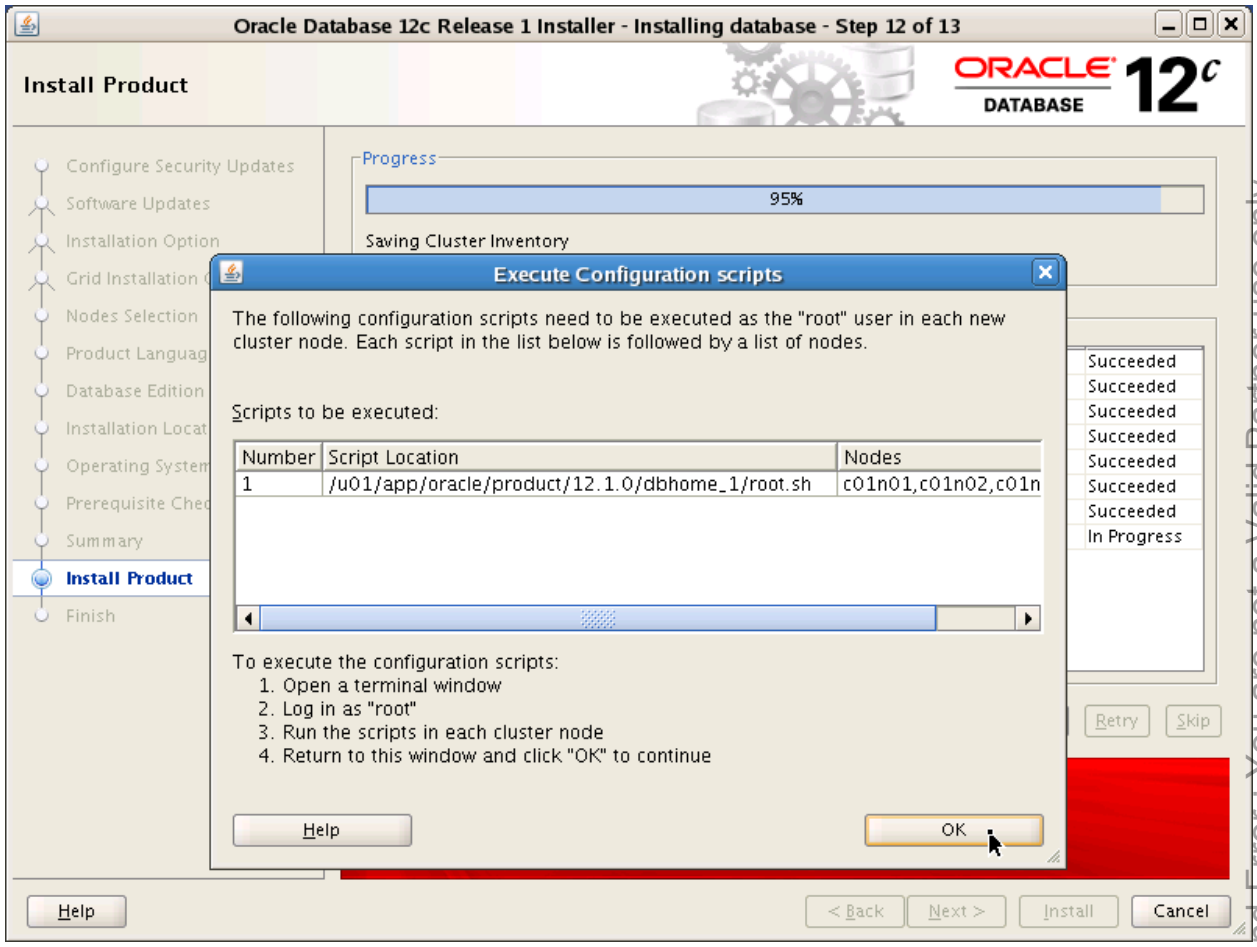
```
[root@c01n01 ~]# /u01/app/oracle/product/12.1.0/dbhome_1/root.sh
Performing root user operation for Oracle 12c

The following environment variables are set as:
    ORACLE_OWNER= oracle
    ORACLE_HOME=  /u01/app/oracle/product/12.1.0/dbhome_1

Enter the full pathname of the local bin directory: [/usr/local/bin]:
The contents of "dbhome" have not changed. No need to overwrite.
The contents of "oraenv" have not changed. No need to overwrite.
The contents of "coraenv" have not changed. No need to overwrite.

Entries will be added to the /etc/oratab file as needed by
Database Configuration Assistant when a database is created
Finished running generic part of root script.
Now product-specific root actions will be performed.
[root@c01n01 ~]#
```

67. Execute the configuration script on all of the other cluster nodes.

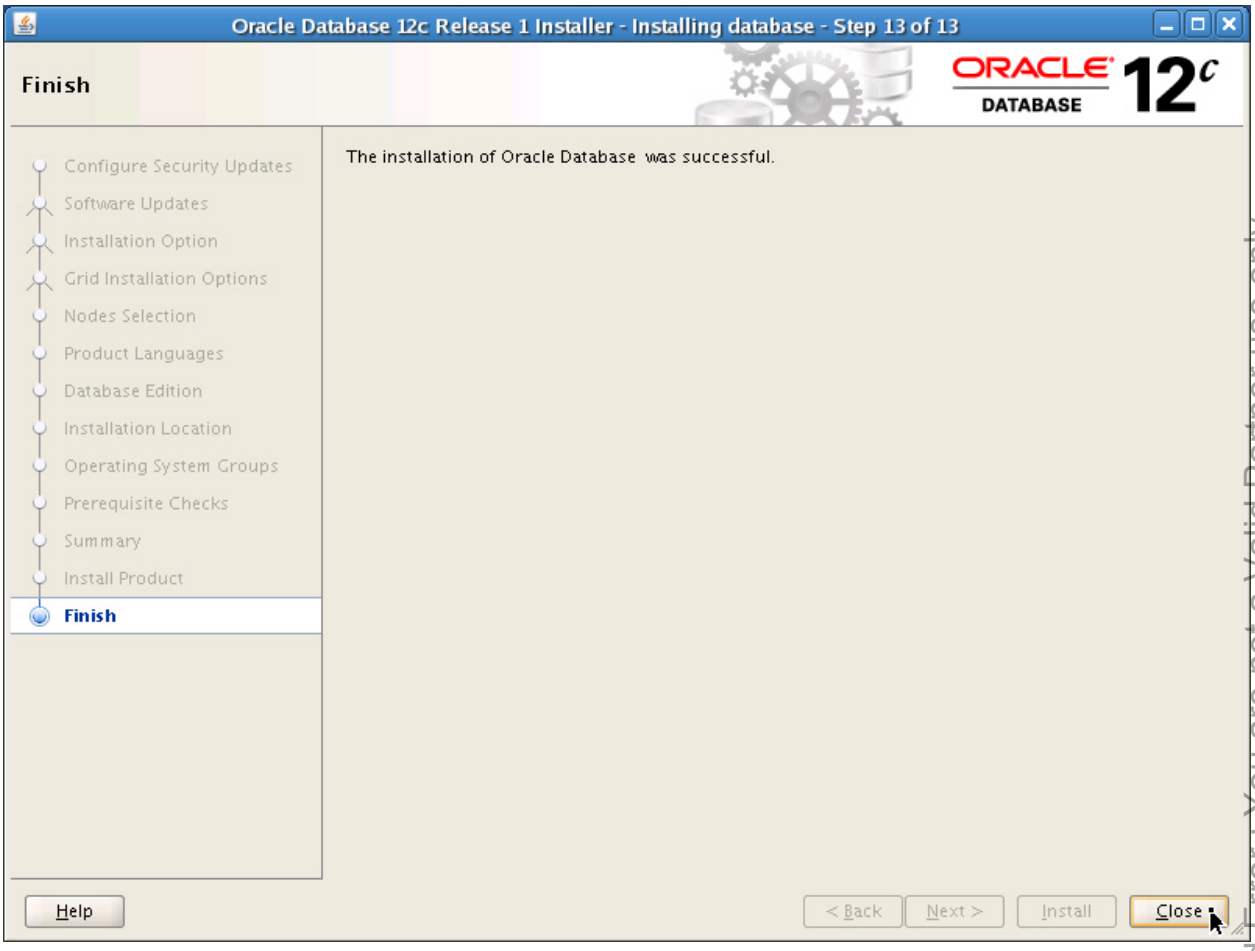
```
[root@c01n01 ~]# ssh c01n02 /u01/app/oracle/product/12.1.0/dbhome_1/root.sh
Performing root user operation for Oracle 12c

The following environment variables are set as:
    ORACLE_OWNER= oracle
    ORACLE_HOME=  /u01/app/oracle/product/12.1.0/dbhome_1

Enter the full pathname of the local bin directory: [/usr/local/bin]:
The contents of "dbhome" have not changed. No need to overwrite.
The contents of "oraenv" have not changed. No need to overwrite.
The contents of "coraenv" have not changed. No need to overwrite.

Entries will be added to the /etc/oratab file as needed by
Database Configuration Assistant when a database is created
Finished running generic part of root script.
Now product-specific root actions will be performed.
[root@c01n01 ~]# ssh c01n03 /u01/app/oracle/product/12.1.0/dbhome_1/root.sh
Performing root user operation for Oracle 12c

The following environment variables are set as:
    ORACLE_OWNER= oracle
    ORACLE_HOME=  /u01/app/oracle/product/12.1.0/dbhome_1

Enter the full pathname of the local bin directory: [/usr/local/bin]:
The contents of "dbhome" have not changed. No need to overwrite.
The contents of "oraenv" have not changed. No need to overwrite.
The contents of "coraenv" have not changed. No need to overwrite.

Entries will be added to the /etc/oratab file as needed by
Database Configuration Assistant when a database is created
Finished running generic part of root script.
Now product-specific root actions will be performed.
[root@c01n01 ~]# ssh c01n04 /u01/app/oracle/product/12.1.0/dbhome_1/root.sh
Performing root user operation for Oracle 12c

The following environment variables are set as:
    ORACLE_OWNER= oracle
    ORACLE_HOME=  /u01/app/oracle/product/12.1.0/dbhome_1

Enter the full pathname of the local bin directory: [/usr/local/bin]:
The contents of "dbhome" have not changed. No need to overwrite.
The contents of "oraenv" have not changed. No need to overwrite.
The contents of "coraenv" have not changed. No need to overwrite.

Entries will be added to the /etc/oratab file as needed by
Database Configuration Assistant when a database is created
Finished running generic part of root script.
Now product-specific root actions will be performed.
```

```
[root@c01n01 ~]#
```

68. After you have executed the required configuration script on all of the cluster nodes, return to your Oracle Universal Installer session and click OK to proceed.



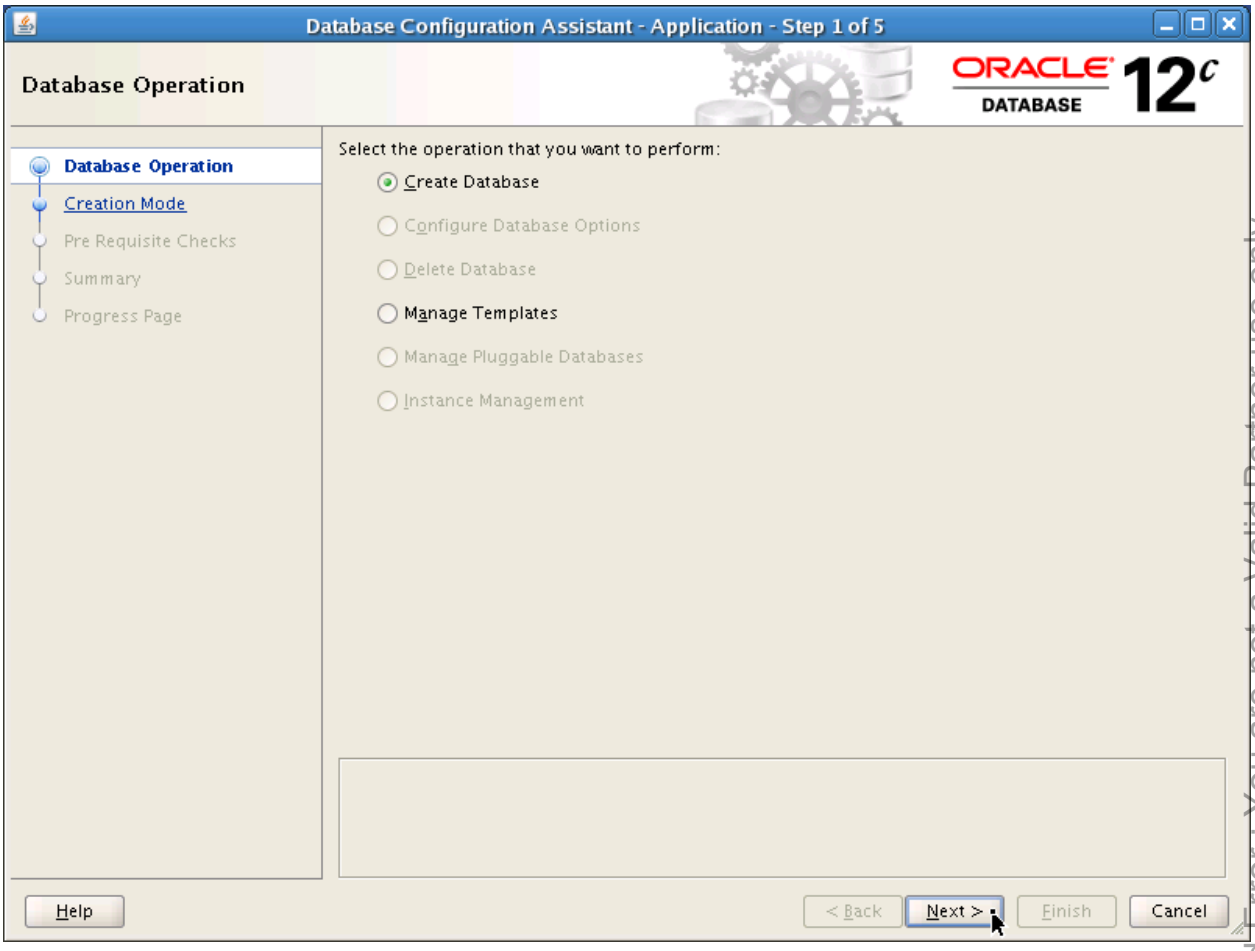
69. After configuration completes, you will see the following screen. Click Close to close Oracle Universal Installer.



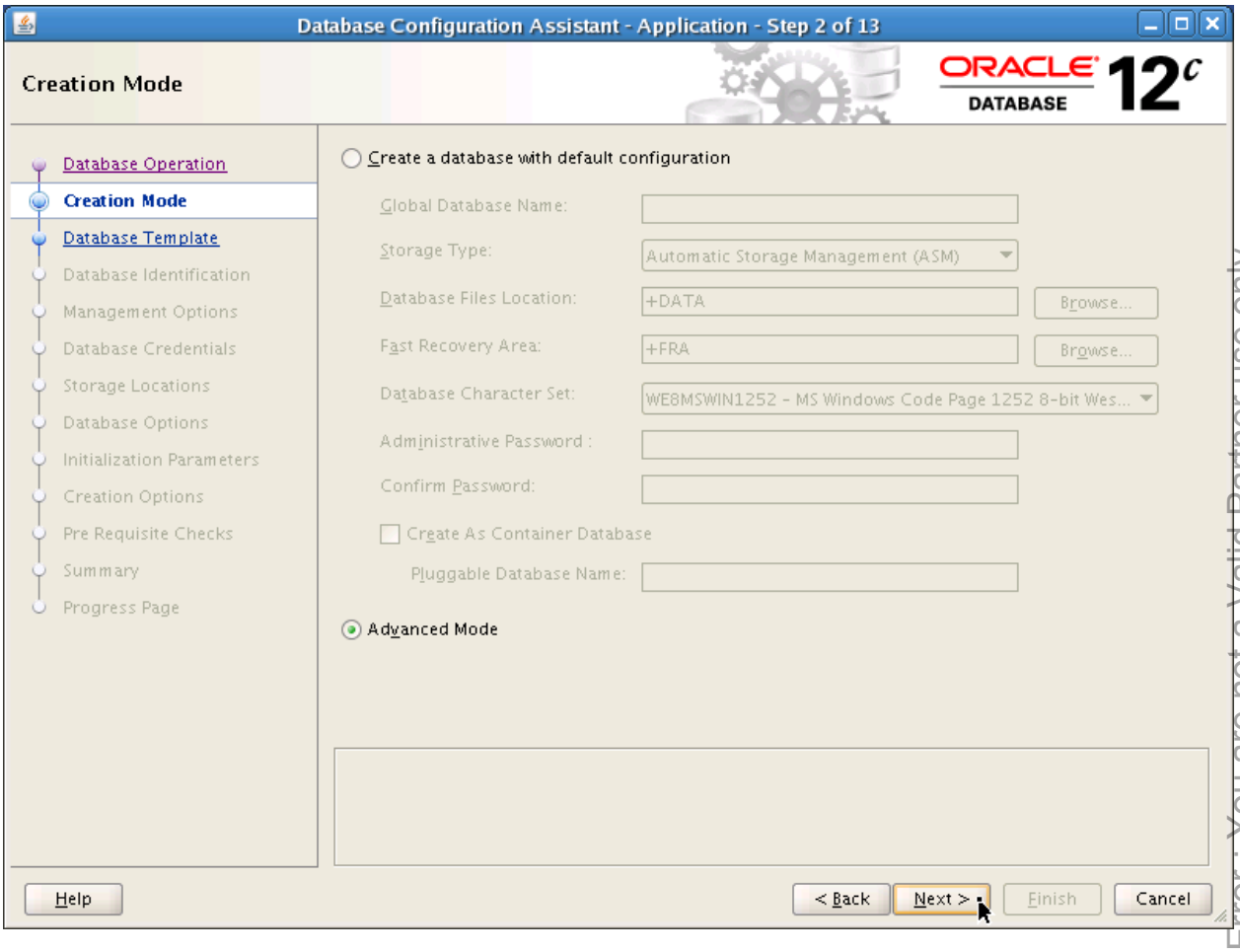
70. Now, you will create a RAC database. Start the Database Configuration Assistant (`dbca`).

```
[oracle@c01n01 ~]$ /u01/app/oracle/product/12.1.0/dbhome_1/bin/dbca
```

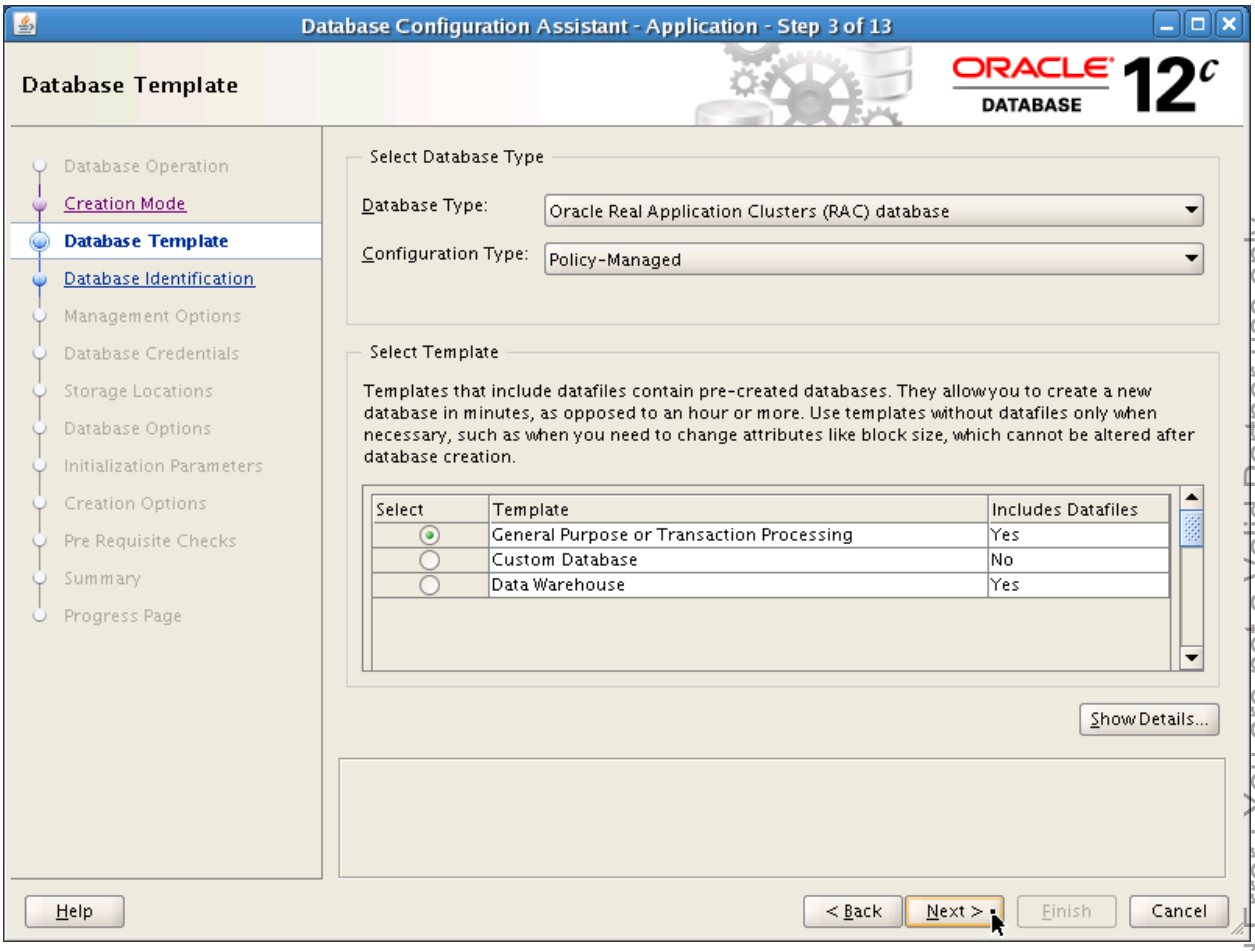
71. On the Database Operation screen, click Next to accept the default selection (Create Database).



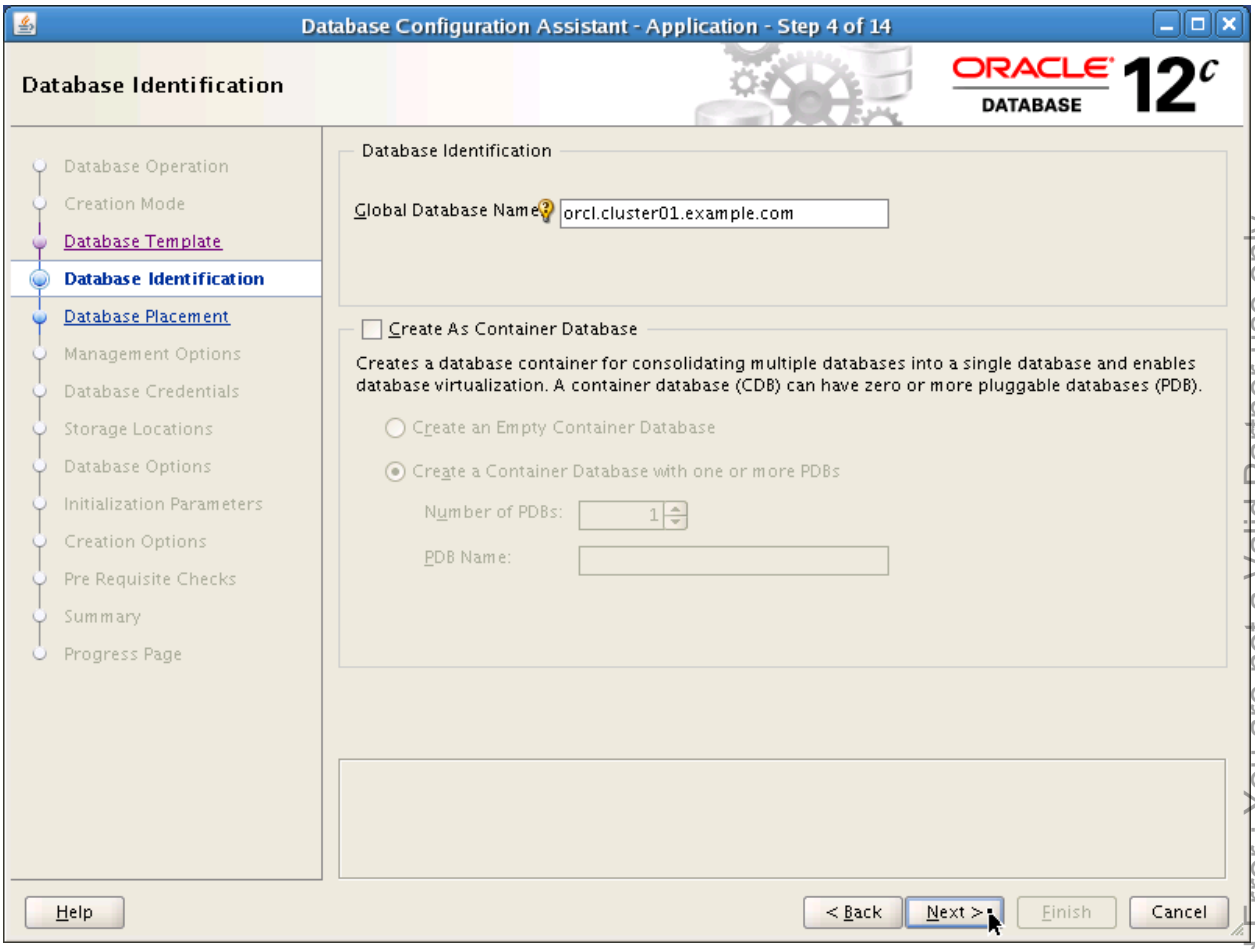
72. On the Creation Mode screen, click Next to accept the default selection (Advanced Mode).



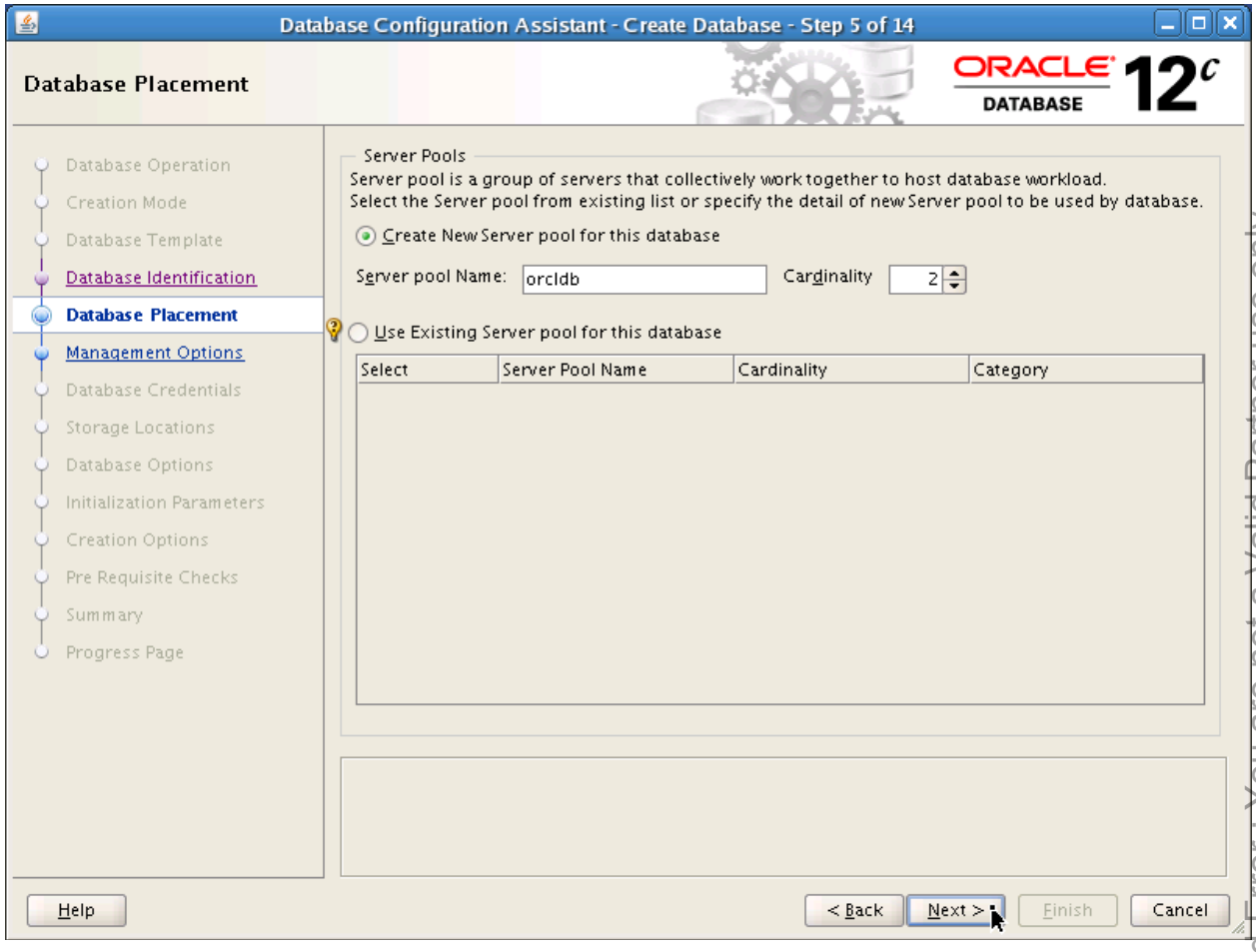
73. On the Database Template screen, click Next to accept the default settings for a Policy-Managed RAC database by using the General Purpose template.



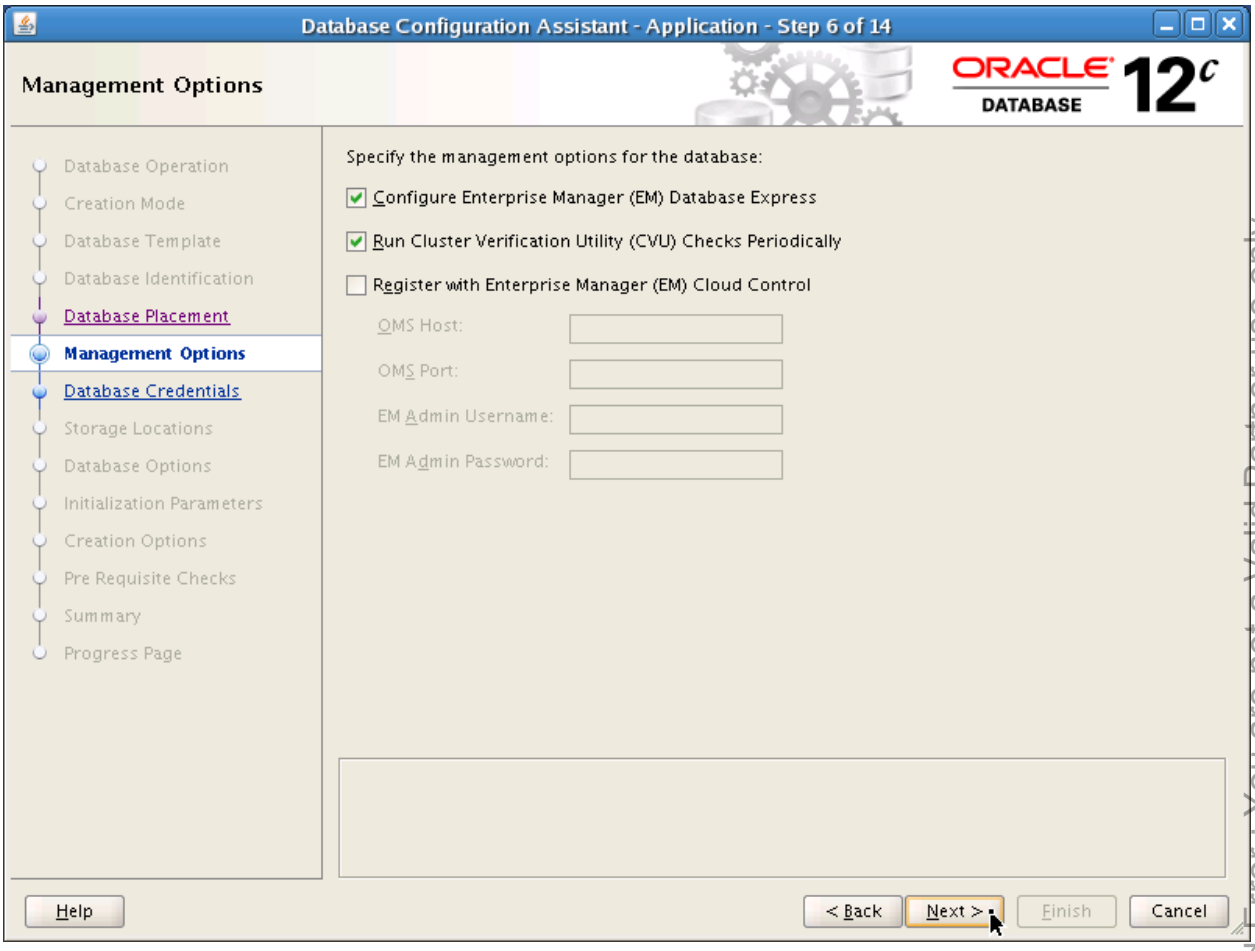
74. On the Database Identification screen, specify `orcl.cluster01.example.com` as the Global Database Name and click Next.



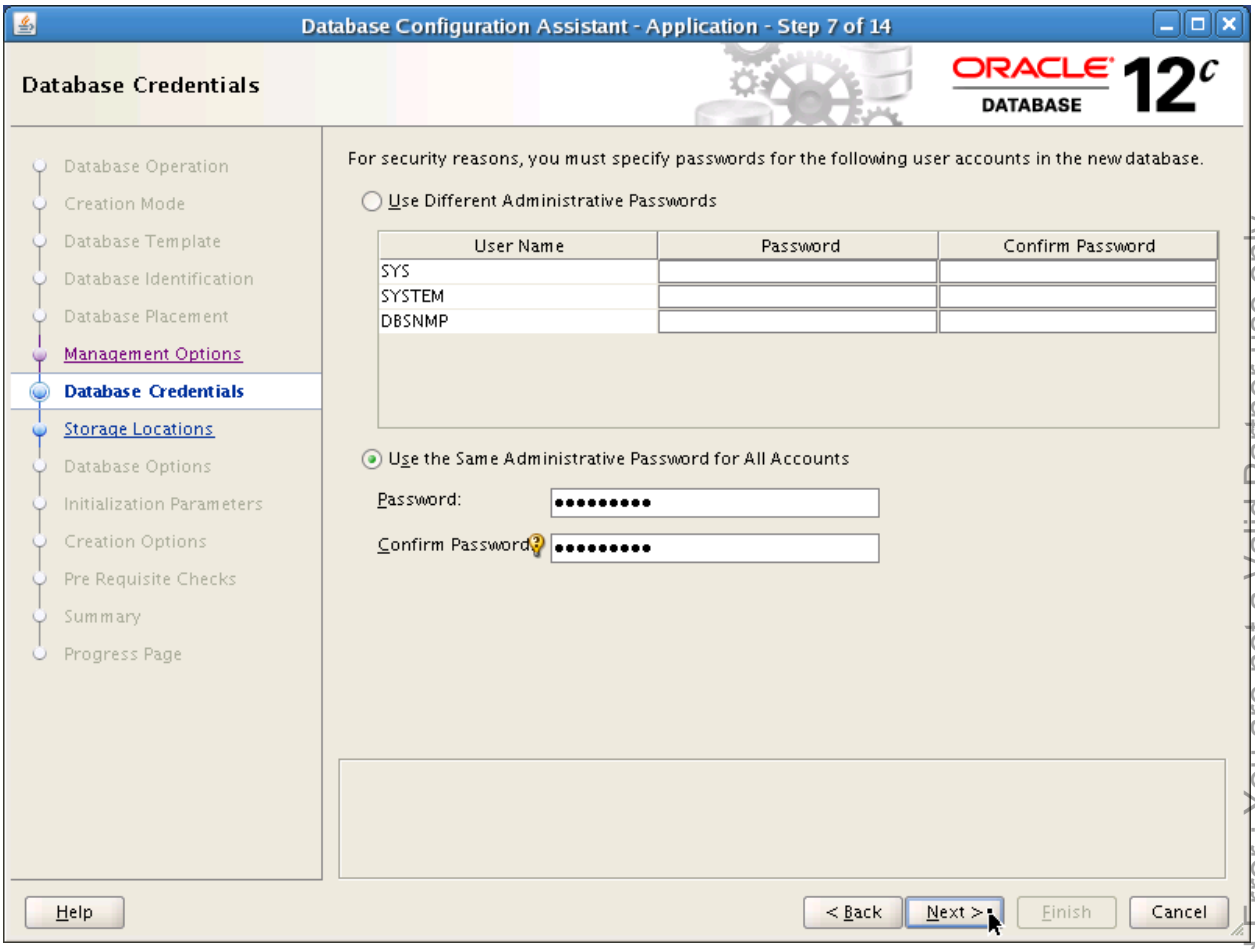
75. On the Database Placement screen, specify `orcldb` for “Server pool Name” and set its cardinality to `2`. Click Next to proceed.



76. On the Management Options screen, click Next to accept the default selections (Configure EM Database Express and Run CVU Checks Periodically).

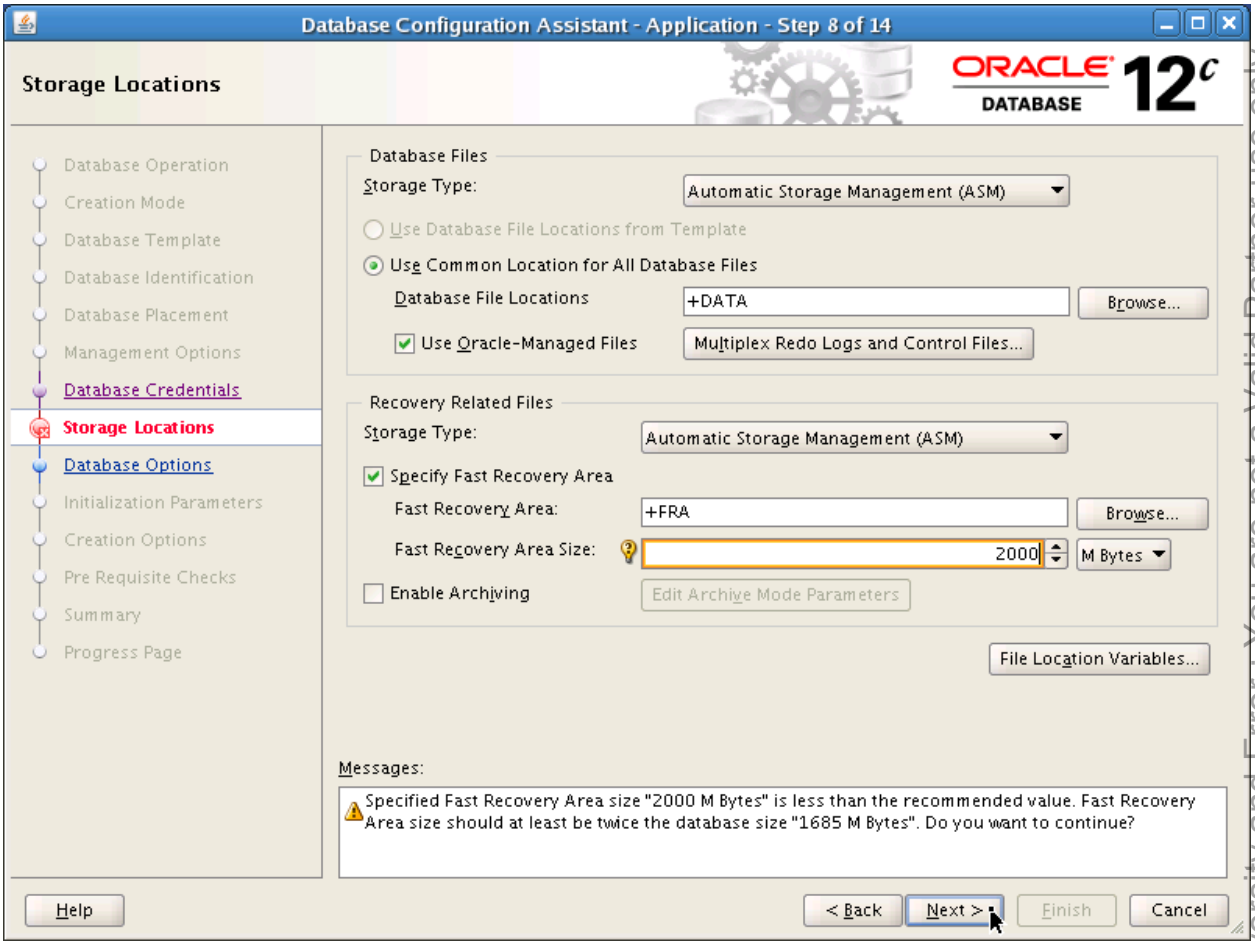


77. On the Database Credentials screen, select “Use the Same Administrative Password for All Accounts” and enter `oracle_4U` as the password. Then click Next to continue.

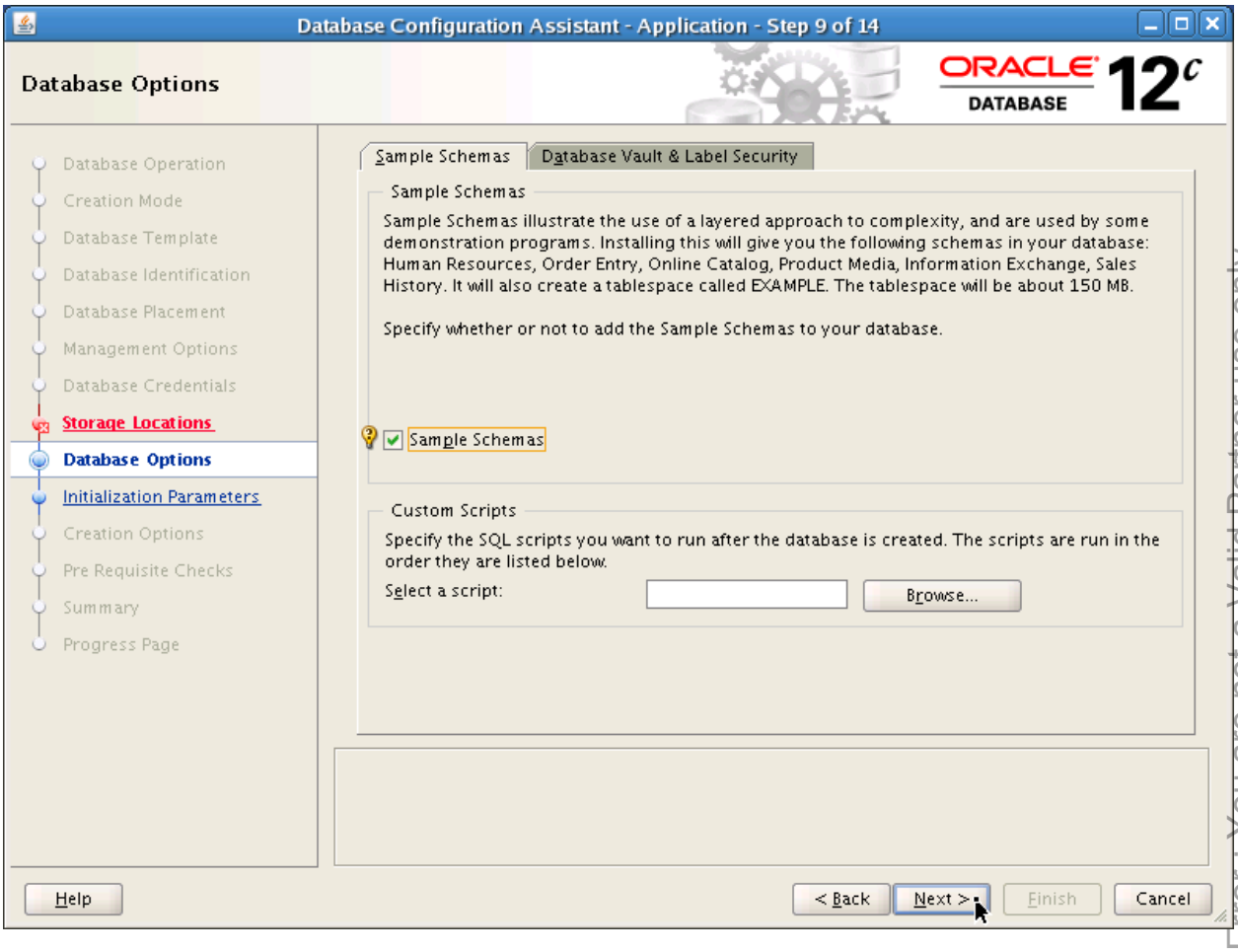


78. On the Storage Locations screen, make the following adjustments:
    - Fast Recovery Area: `+FRA`
    - Fast Recovery Area Size: `2000`

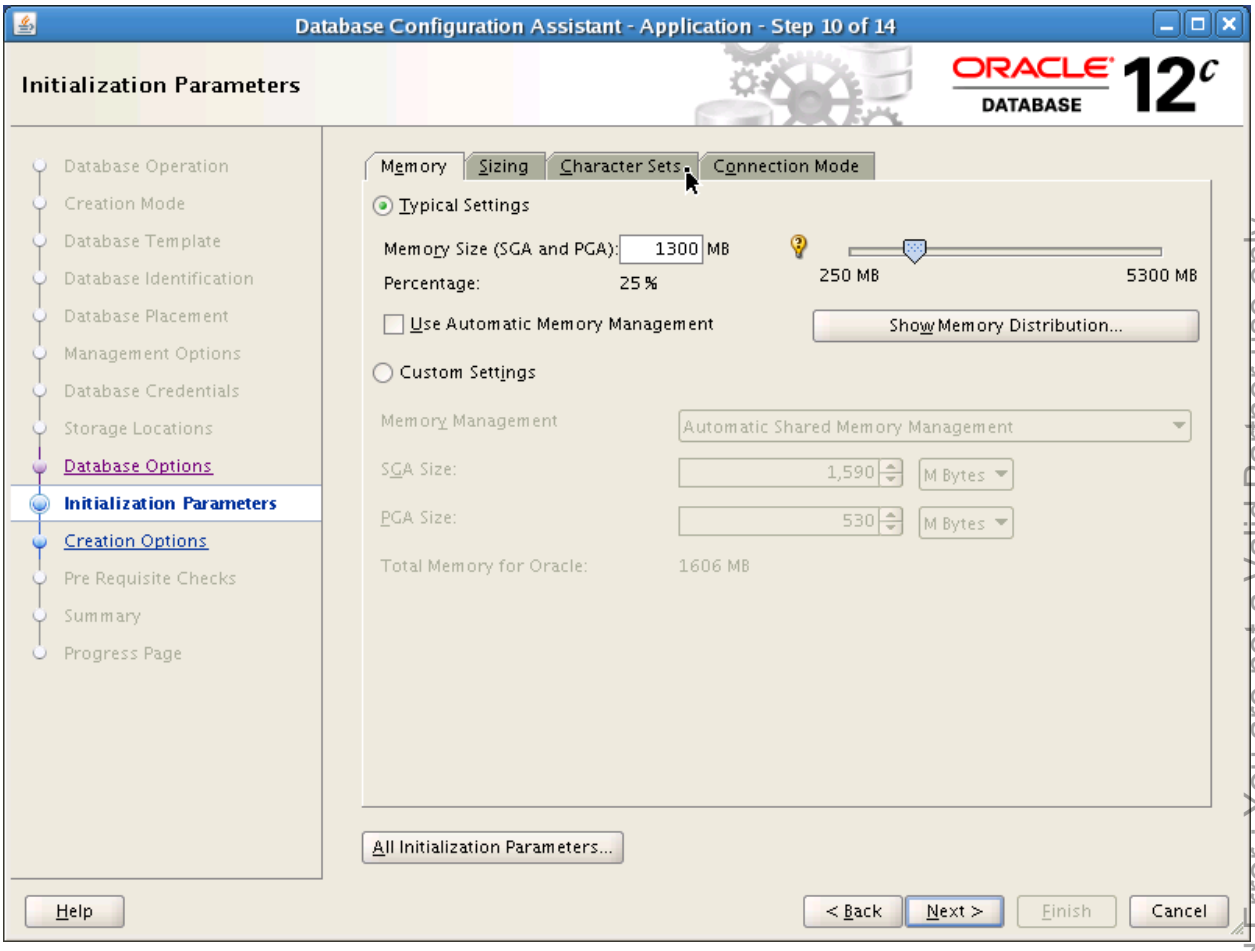
    Space constraints in the laboratory environment require that you ignore the warning regarding the size of the Fast Recovery Area. Once your screen matches the following screenshot, click Next to continue.



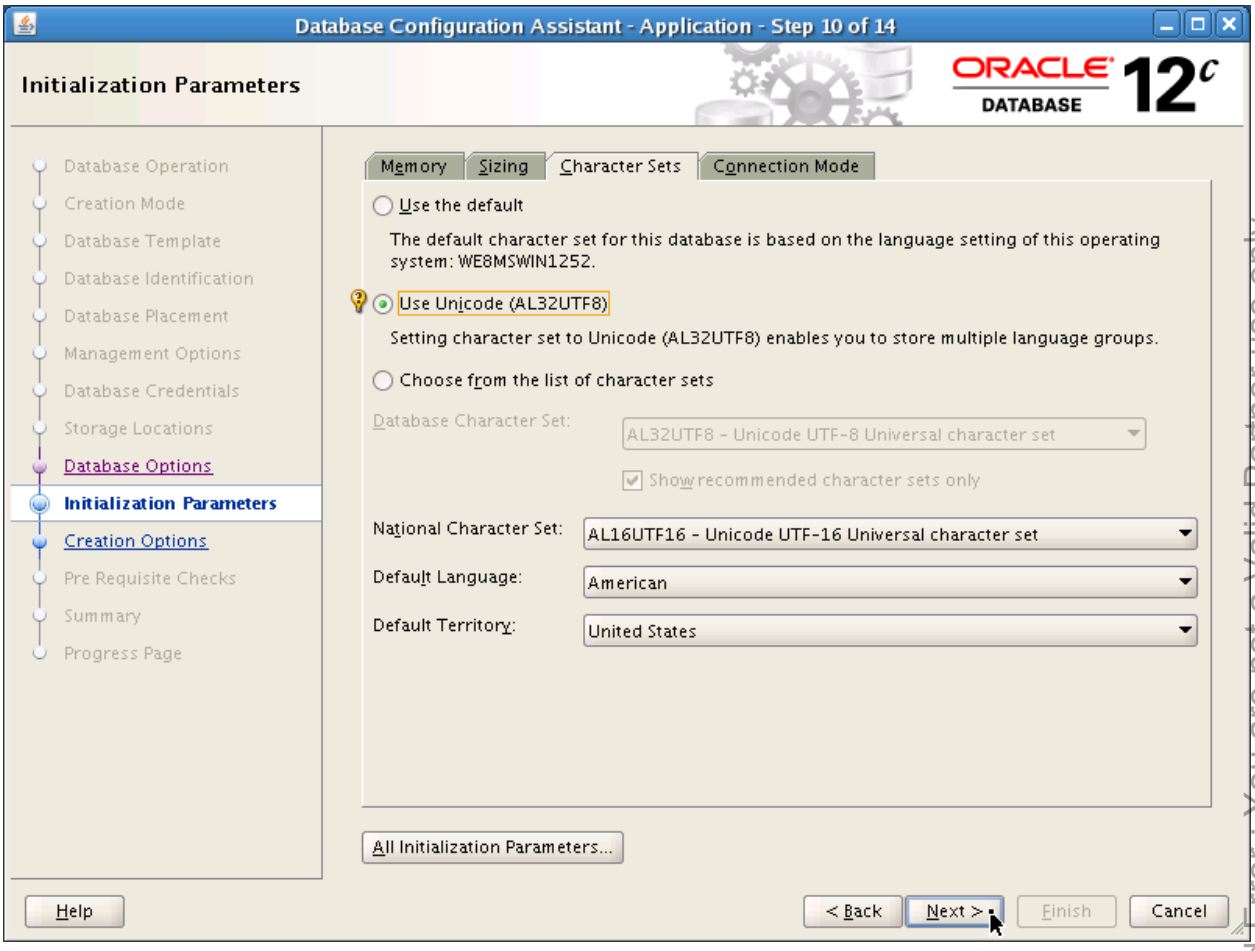
79. On the Database Options screen, select Sample Schemas and click Next.



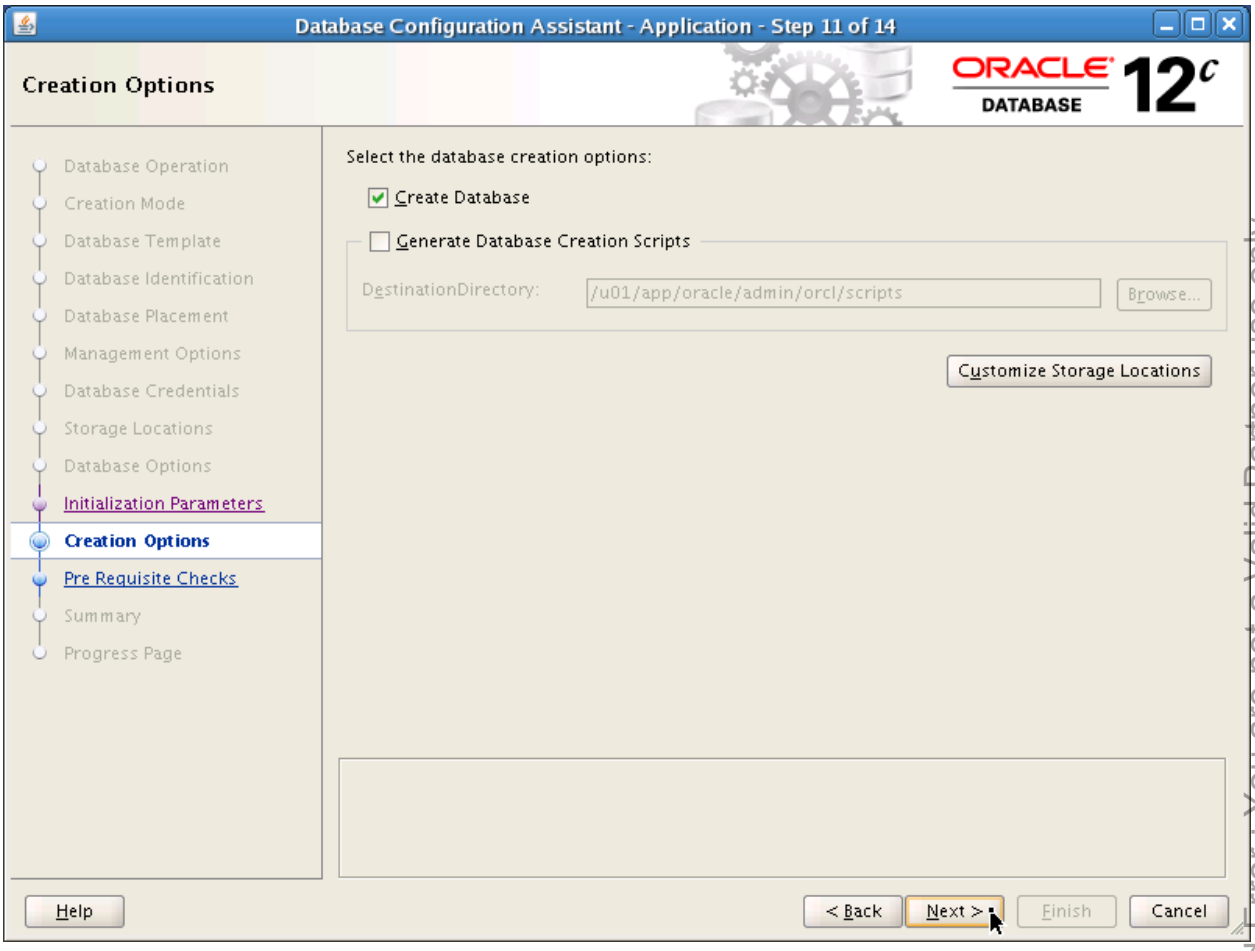
80. On the Memory tab, in the Initialization Parameters screen, set Memory Size (SGA and PGA) to 1300 and click the Character Sets tab.



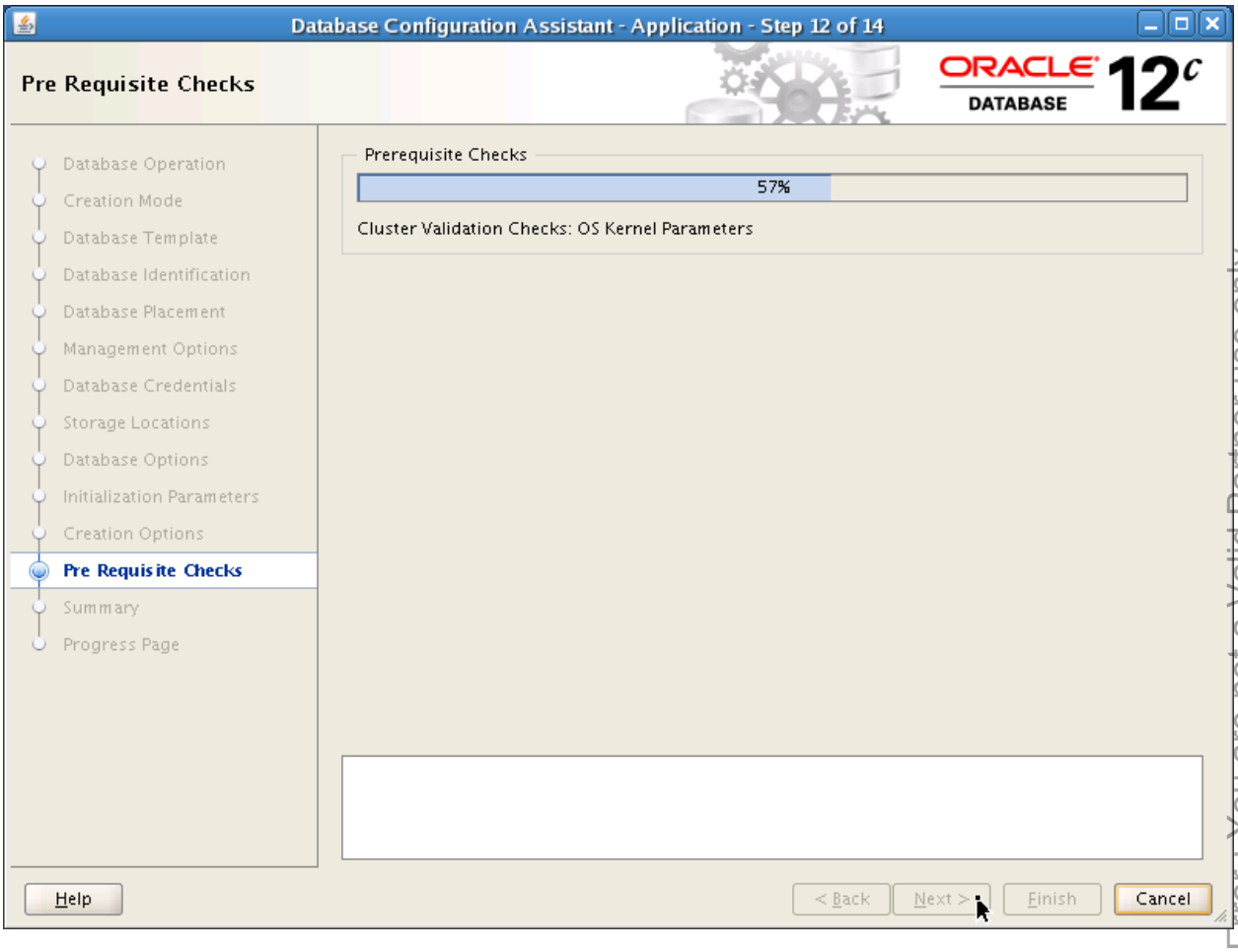
81. On the Character Sets tab, in the Initialization Parameters screen, select Use Unicode (AL32UTF8) and click Next.



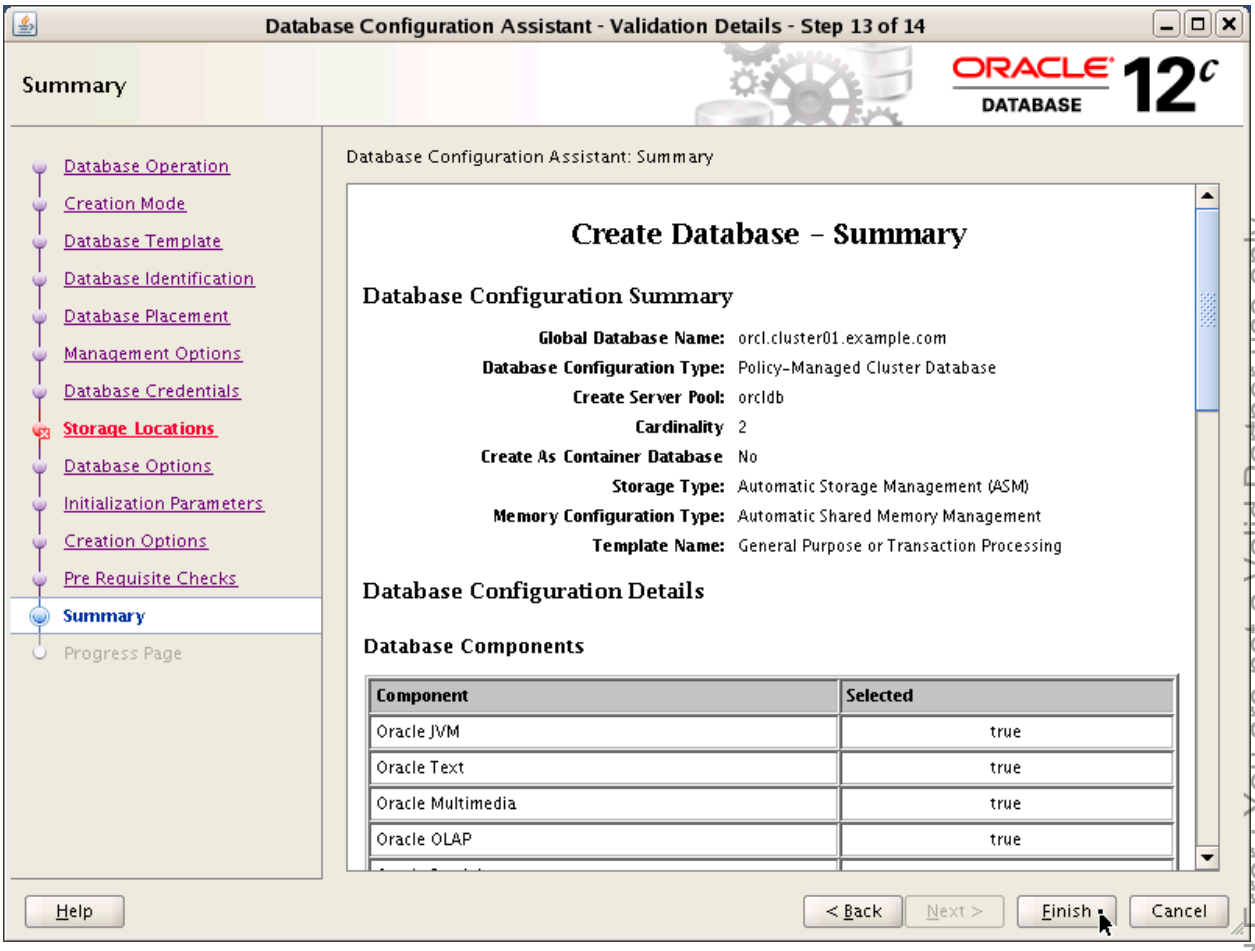
82. On the Creation Options screen, click Next to accept the default selection (Create Database).



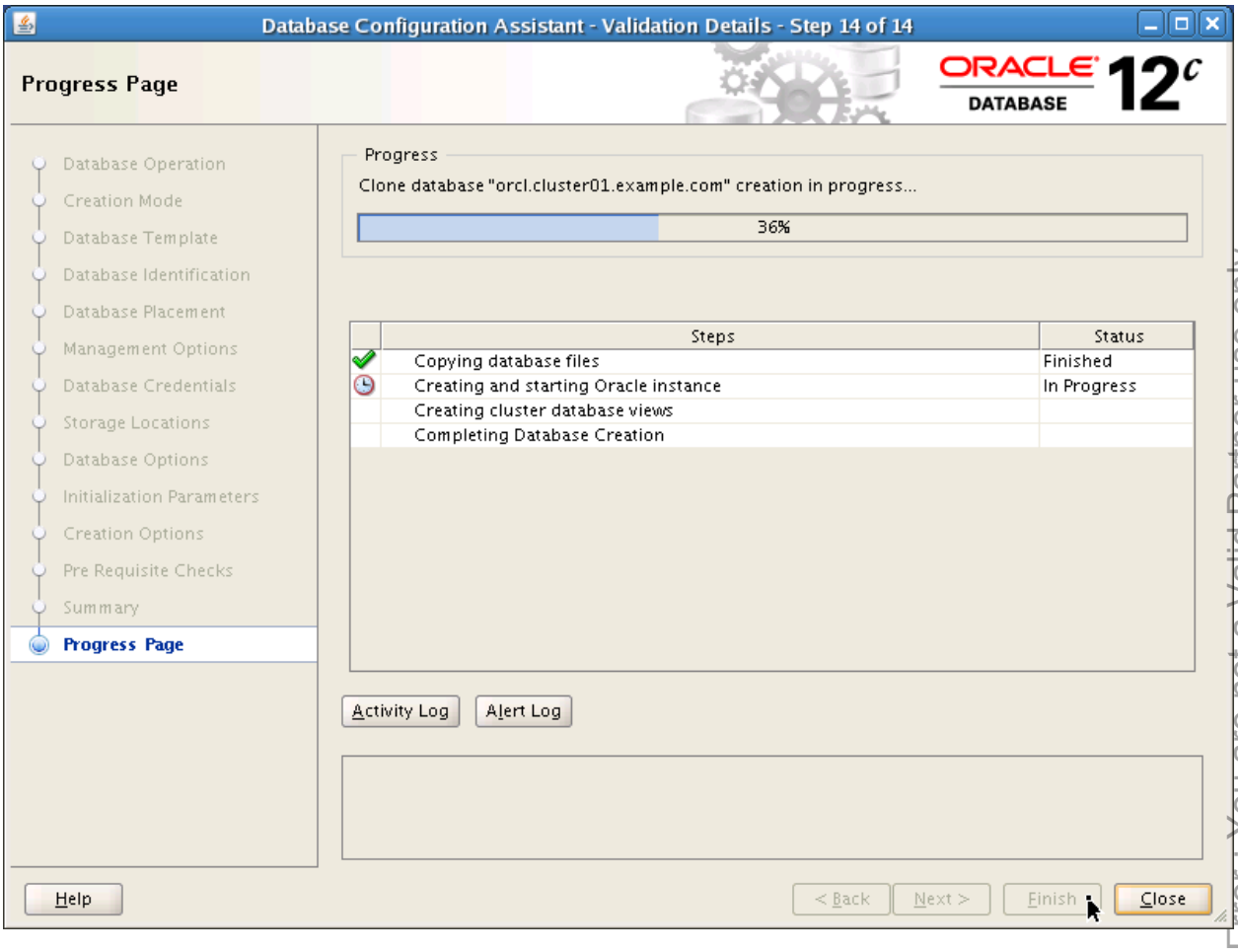
83. Wait while a series of prerequisite checks are performed.



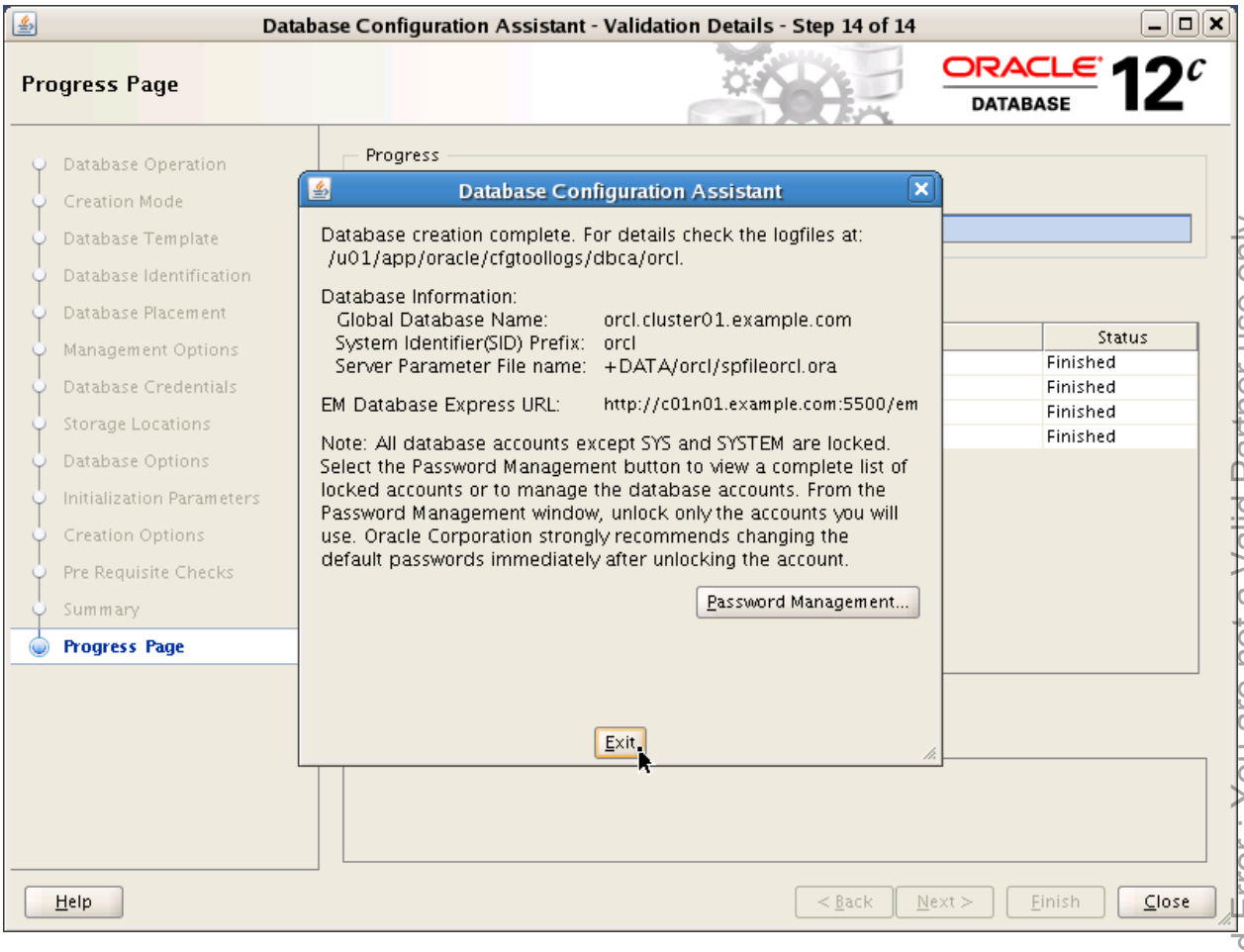
84. Examine the Summary screen. When you are ready, click Finish to commence the database creation process.



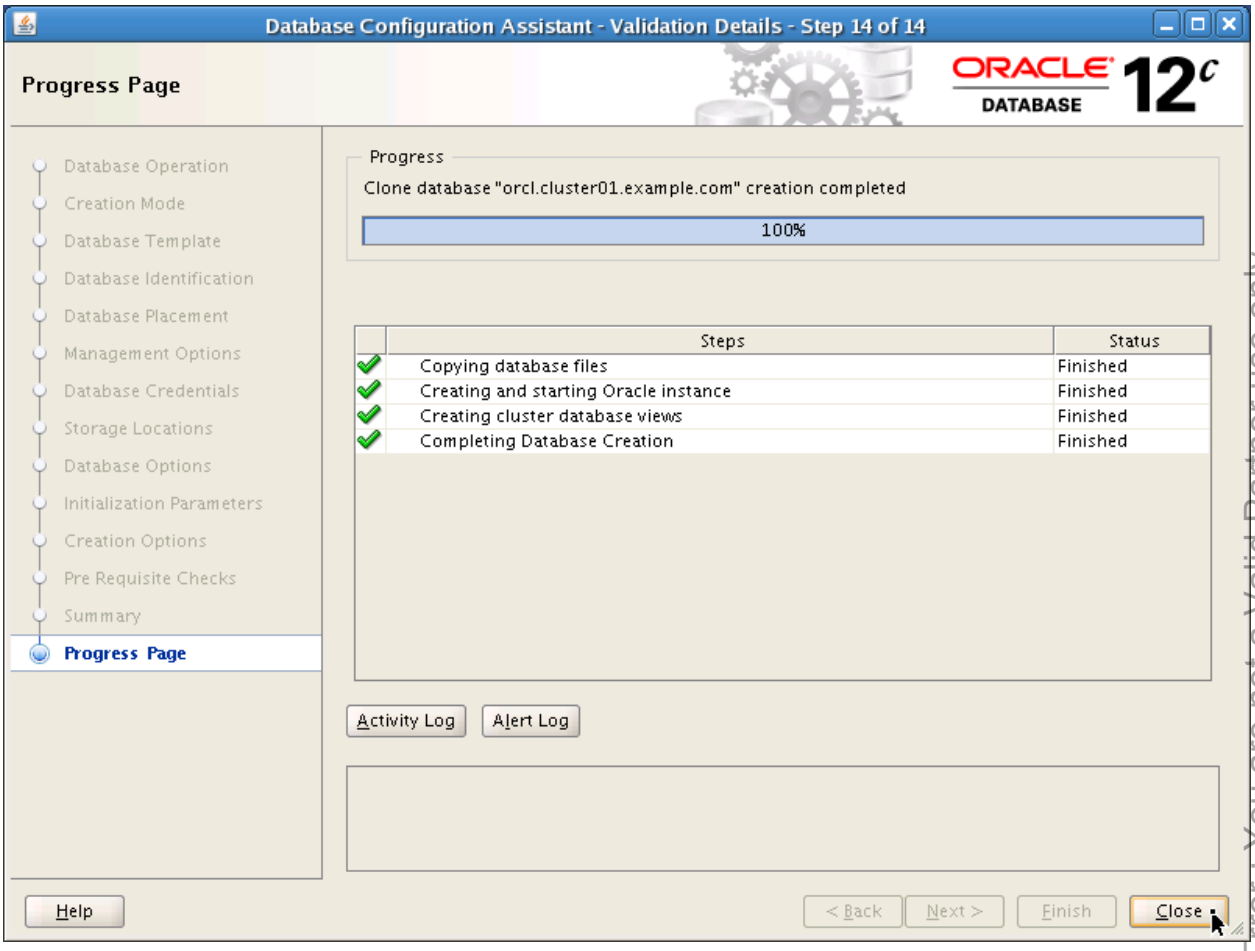
85. Follow the database creation process on the Progress Page.



86. Examine the dialog box that indicates that the database creation process is completed. When you are ready, click Exit continue.



87. Click Close to quit the Database Configuration Assistant.



88. Back in the `oracle` user terminal, configure the environment by using the `oraenv` script. Enter `orcl` when you are prompted for an ORACLE_SID value.

```
[oracle@c01n01 ~]$ . oraenv
ORACLE_SID = [oracle] ? orcl
The Oracle base has been set to /u01/app/oracle
[oracle@c01n01 ~]$
```

89. View the status of the newly created RAC database. Note which database instance is running on the different cluster nodes.

```
[oracle@c01n01 ~]$ srvctl status database -d orcl
Instance orcl_1 is running on node c01n02
Instance orcl_2 is running on node c01n01
[oracle@c01n01 ~]$
```

Congratulations! You have successfully configured an Oracle Database 12*c* Flex Cluster with Flex ASM and a RAC database.

Practice 2-2: Configuring Highly Available Application Resources on Flex Cluster Leaf Nodes

Overview

In this practice, you will create a series of highly available application resources running on one of the Flex Cluster Leaf Nodes.

Assumptions

This practice relies on the configuration established in Practice 2-1.

Tasks

1. Establish a terminal session connected to `c01n01` by using the `grid` OS user.

```
$ ssh grid@c01n01
grid@c01n01's password: <oracle>
[grid@c01n01 ~]$
```

2. Configure the environment by using the `oraenv` script. Enter `+ASM1` when you are prompted for an ORACLE_SID value.

```
[grid@c01n01 ~]$ . oraenv
ORACLE_SID = [grid] ? +ASM1
The Oracle base has been set to /u01/app/grid
[grid@c01n01 ~]$
```

3. Examine the cluster to identify the role for each node in the cluster.

```
[grid@c01n01 ~]$ crsctl get node role status -all
Node 'c01n01' active role is 'hub'
Node 'c01n02' active role is 'hub'
Node 'c01n04' active role is 'leaf'
Node 'c01n03' active role is 'leaf'
[grid@c01n01 ~]$
```

4. Examine the cluster to identify the currently defined server pools and server allocations. Notice that currently the Hub Nodes are allocated to the `orcldb` server pool, which contains the RAC database that you created in the last practice. In addition, both Leaf Nodes are currently in the `Free` pool.

```
[grid@c01n01 ~]$ crsctl status serverpool
NAME=Free
ACTIVE_SERVERS=c01n03 c01n04

NAME=Generic
ACTIVE_SERVERS=

NAME=ora.orcldb
ACTIVE_SERVERS=c01n01 c01n02

[grid@c01n01 ~]$
```

5. Examine the cluster to identify the currently defined server categories. You will learn more about server categories later in the course, but for now simply note the existence of the built-in category `ora.leaf.category`. All the Leaf Nodes in the cluster are implicitly associated with this category.

```
[grid@c01n01 ~]$ crsctl status category
NAME=ora.hub.category
ACL=owner:root:rwx,pgrp:root:r-x,other::r--
ACTIVE_CSS_ROLE=hub
EXPRESSION=

NAME=ora.leaf.category
ACL=owner:root:rwx,pgrp:root:r-x,other::r--
ACTIVE_CSS_ROLE=leaf
EXPRESSION=

[grid@c01n01 ~]$
```

6. Create a new server pool to house a series of highly available application resources on one of the Flex Cluster Leaf Nodes.

```
[grid@c01n01 ~]$ srvctl add srvpool -serverpool my_app_pool -min
1 -max 1 -category "ora.leaf.category"
[grid@c01n01 ~]$
```

7. Reexamine the server pools. Notice that one of the leaf nodes has been allocated to the newly created server pool (`my_app_pool`).

```
[grid@c01n01 ~]$ crsctl status serverpool
NAME=Free
ACTIVE_SERVERS=c01n04

NAME=Generic
ACTIVE_SERVERS=

NAME=ora.my_app_pool
ACTIVE_SERVERS=c01n03

NAME=ora.orcldb
ACTIVE_SERVERS=c01n01 c01n02

[grid@c01n01 ~]$
```

8. Navigate to the directory that contains the scripts for this practice.

```
[grid@c01n01 ~]$ cd /stage/labs/2_2
[grid@c01n01 2_2]$
```

9. Examine the provided action script (`action.scr`). Soon you will use this script to create a series of highly available application resources on a Flex Cluster Leaf Node. The application resources will all perform the following set of simple tasks.
   - When the application resource is started, it creates an empty file at `/tmp/<resource_name>` on the node running the application resource.
   - When the application resource is stopped or cleaned, the file at `/tmp/<resource_name>` is deleted.
   - When the application resource is checked, a test is performed to check the existence of the file at `/tmp/<resource_name>`.
   - In addition, regardless of the action performed, log entries are written to the CRSD agent log file, which can be found at `/u01/app/12.1.0/grid/log/<hostname>/agent/crsd/scriptagent_grid/scriptagent_grid.log`.

```
[grid@c01n01 2_2]$ more action.scr
#!/bin/sh
TOUCH=/bin/touch
RM=/bin/rm
PATH_NAME=/tmp/$_CRS_NAME

#
# These messages go into the CRSD agent log file.
echo " *******   `date` ********** "
echo "Action script '$_CRS_ACTION_SCRIPT' for
resource[$_CRS_NAME] called for action $1"
#

case "$1" in
  'start')
     echo "START entry point has been called.."
     echo "Creating the file: $PATH_NAME"
     $TOUCH $PATH_NAME
     exit 0
     ;;

  'stop')
     echo "STOP entry point has been called.."
     echo "Deleting the file: $PATH_NAME"
     $RM $PATH_NAME
     exit 0
     ;;
```

```
  'check')
    echo "CHECK entry point has been called.."
    if [ -e $PATH_NAME ]; then
        echo "Check -- SUCCESS"
        exit 0
    else
        echo "Check -- FAILED"
        exit 1
    fi
    ;;

  'clean')
     echo "CLEAN entry point has been called.."
     echo "Deleting the file: $PATH_NAME"
     $RM -f $PATH_NAME
     exit 0
     ;;

esac

[grid@c01n01 2_2]$
```

10. Examine the `create_res.sh` file. This file contains the commands that you will next use to create three application resources named `my_dep_res1`, `my_dep_res2`, and `my_resource`. Notice that all three resources are associated with the `my_app_pool` server pool. Notice also that `my_resource` has a mandatory (hard) dependency on `my_dep_res1` and an optional (soft) dependency on `my_dep_res2`. Finally, notice that `my_resource` also depends on the RAC database (`ora.orcl.db`) that you created in the previous practice. This illustrates how you can unify the management of databases and applications within a Flex Cluster.

```
[grid@c01n01 2_2]$ cat create_res.sh
crsctl add resource my_dep_res1 -type cluster_resource -attr
"ACTION_SCRIPT=/stage/labs/2_2/action.scr,PLACEMENT=restricted,S
ERVER_POOLS=ora.my_app_pool"


crsctl add resource my_dep_res2 -type cluster_resource -attr
"ACTION_SCRIPT=/stage/labs/2_2/action.scr,PLACEMENT=restricted,S
ERVER_POOLS=ora.my_app_pool"


crsctl add resource my_resource -type cluster_resource -attr
"ACTION_SCRIPT=/stage/labs/2_2/action.scr,PLACEMENT=restricted,S
ERVER_POOLS=ora.my_app_pool,START_DEPENDENCIES='hard(my_dep_res1
, global:uniform:ora.orcl.db) pullup:always(my_dep_res1,
global:ora.orcl.db)
```

```
weak(my_dep_res2)',STOP_DEPENDENCIES='hard(my_dep_res1,
global:ora.orcl.db)'"

[grid@c01n01 2_2]$
```

11. Execute `create_res.sh` to create the application resources.

```
[grid@c01n01 2_2]$ ./create_res.sh
[grid@c01n01 2_2]$
```

12. Examine the newly created cluster resources. Note that currently the resources exist but have not been started.

```
[grid@c01n01 2_2]$ crsctl status resource -t -w "NAME co my"
--------------------------------------------------------------------------------
Name           Target  State      Server        State details
--------------------------------------------------------------------------------
Cluster Resources
--------------------------------------------------------------------------------
my_dep_res1
      1        OFFLINE OFFLINE                  STABLE
my_dep_res2
      1        OFFLINE OFFLINE                  STABLE
my_resource
      1        OFFLINE OFFLINE                  STABLE
--------------------------------------------------------------------------------
[grid@c01n01 2_2]$
```

13. Establish another terminal session connected to `c01n01` as the `oracle` OS user.

```
$ ssh oracle@c01n01
Password: <oracle>
[oracle@c01n01 ~]$
```

14. Configure the environment by using the `oraenv` script. Enter `orcl` when you are prompted for an ORACLE_SID value.

```
[oracle@c01n01 ~]$ . oraenv
ORACLE_SID = [oracle] ? orcl
The Oracle base has been set to /u01/app/oracle
[oracle@c01n01 ~]$
```

15. So that you can exercise the dependency between `my_resource` and your RAC database, stop the `orcl` database.

```
[oracle@c01n01 ~]$ srvctl stop database -d orcl
[oracle@c01n01 ~]$
```

16. Back in your grid terminal session, start the `my_resource` application resource. Notice that `my_dep_res1` and `my_dep_res2` are started automatically to fulfill the dependency definitions and that all three resources are started on the server associated with the `my_app_pool` server pool. Note also that the `orcl` database is also started as defined in the dependency definitions for the `my_resource` application resource. This illustrates how a Flex Cluster can be used to manage a RAC database and associated application resources.

```
[grid@c01n01 2_2]$ crsctl start resource my_resource
CRS-2672: Attempting to start 'my_dep_res1' on 'c01n03'
CRS-2672: Attempting to start 'my_dep_res2' on 'c01n03'
CRS-2672: Attempting to start 'ora.orcl.db' on 'c01n02'
CRS-2672: Attempting to start 'ora.orcl.db' on 'c01n01'
CRS-2676: Start of 'my_dep_res1' on 'c01n03' succeeded
CRS-2676: Start of 'my_dep_res2' on 'c01n03' succeeded
CRS-2676: Start of 'ora.orcl.db' on 'c01n01' succeeded
CRS-2676: Start of 'ora.orcl.db' on 'c01n02' succeeded
CRS-2672: Attempting to start 'my_resource' on 'c01n03'
CRS-2676: Start of 'my_resource' on 'c01n03' succeeded
[grid@c01n01 2_2]$
```

17. Reexamine the application resources to confirm their status.

```
[grid@c01n01 2_2]$ crsctl status resource -t -w "NAME co my"
--------------------------------------------------------------------------------
Name           Target  State        Server          State details
--------------------------------------------------------------------------------
Cluster Resources
--------------------------------------------------------------------------------
my_dep_res1
      1        ONLINE  ONLINE       c01n03          STABLE
my_dep_res2
      1        ONLINE  ONLINE       c01n03          STABLE
my_resource
      1        ONLINE  ONLINE       c01n03          STABLE
--------------------------------------------------------------------------------
[grid@c01n01 2_2]$
```

18. Check the files under `/tmp` on the node running the application resources. You should expect to see a series of empty files, each bearing the name of one of the application resources.

```
[grid@c01n01 2_2]$ ssh c01n03 ls -l /tmp/my*
-rw-r--r-- 1 grid oinstall 0 Nov 26 02:39 /tmp/my_dep_res1
-rw-r--r-- 1 grid oinstall 0 Nov 26 02:39 /tmp/my_dep_res2
-rw-r--r-- 1 grid oinstall 0 Nov 26 02:39 /tmp/my_resource
[grid@c01n01 2_2]$
```

19. Using the `oracle` terminal session, confirm also that your RAC database (`orcl`) started as part of starting the `my_resource` application resource.

```
[oracle@c01n01 ~]$ srvctl status database -d orcl
Instance orcl_1 is running on node c01n02
Instance orcl_2 is running on node c01n01
[oracle@c01n01 ~]$
```

20. Now, shut down Oracle Clusterware on the node hosting the application resources.

```
[grid@c01n01 2_2]$ su -c "crsctl stop cluster -n c01n03"
Password: <oracle>
CRS-2673: Attempting to stop 'ora.crsd' on 'c01n03'
CRS-2790: Starting shutdown of Cluster Ready Services-managed
resources on 'c01n03'
CRS-2673: Attempting to stop 'my_dep_res2' on 'c01n03'
CRS-2673: Attempting to stop 'my_resource' on 'c01n03'
CRS-2677: Stop of 'my_dep_res2' on 'c01n03' succeeded
CRS-2677: Stop of 'my_resource' on 'c01n03' succeeded
CRS-2673: Attempting to stop 'my_dep_res1' on 'c01n03'
CRS-2677: Stop of 'my_dep_res1' on 'c01n03' succeeded
CRS-2792: Shutdown of Cluster Ready Services-managed resources
on 'c01n03' has completed
CRS-2677: Stop of 'ora.crsd' on 'c01n03' succeeded
CRS-2673: Attempting to stop 'ora.cluster_interconnect.haip' on
'c01n03'
CRS-2673: Attempting to stop 'ora.ctssd' on 'c01n03'
CRS-2673: Attempting to stop 'ora.evmd' on 'c01n03'
CRS-2673: Attempting to stop 'ora.storage' on 'c01n03'
CRS-2677: Stop of 'ora.cluster_interconnect.haip' on 'c01n03'
succeeded
CRS-2677: Stop of 'ora.storage' on 'c01n03' succeeded
CRS-2677: Stop of 'ora.ctssd' on 'c01n03' succeeded
CRS-2677: Stop of 'ora.evmd' on 'c01n03' succeeded
CRS-2673: Attempting to stop 'ora.cssd' on 'c01n03'
CRS-2677: Stop of 'ora.cssd' on 'c01n03' succeeded
[grid@c01n01 2_2]$
```

21. Examine the status of the server pools. You should see that the previously unallocated Leaf Node has been moved to the `my_app_pool` server pool.

```
[grid@c01n01 2_2]$ crsctl status serverpool
NAME=Free
ACTIVE_SERVERS=

NAME=Generic
ACTIVE_SERVERS=
```

```
NAME=ora.my_app_pool
ACTIVE_SERVERS=c01n04

NAME=ora.orcldb
ACTIVE_SERVERS=c01n01 c01n02

[grid@c01n01 2_2]$
```

22. Reexamine the status of the application resources. Notice that they have been failed-over to the surviving Leaf Node.

```
[grid@c01n01 2_2]$ crsctl status resource -t -w "NAME co my"
--------------------------------------------------------------------
Name           Target  State      Server        State details
--------------------------------------------------------------------
Cluster Resources
--------------------------------------------------------------------
my_dep_res1
      1        ONLINE  ONLINE     c01n04        STABLE
my_dep_res2
      1        ONLINE  ONLINE     c01n04        STABLE
my_resource
      1        ONLINE  ONLINE     c01n04        STABLE
--------------------------------------------------------------------
[grid@c01n01 2_2]$
```

23. Reexamine the contents of the `/tmp` directory on both Leaf Nodes. Notice that the files you saw in step 18 no longer exist and that there is a newer set of files on the other Leaf Node.

```
[grid@c01n01 2_2]$ ssh c01n03 ls -l /tmp/my*
ls: cannot access /tmp/my*: No such file or directory
[grid@c01n01 2_2]$ ssh c01n04 ls -l /tmp/my*
-rw-r--r-- 1 grid oinstall 0 Nov 26 02:52 /tmp/my_dep_res1
-rw-r--r-- 1 grid oinstall 0 Nov 26 02:52 /tmp/my_dep_res2
-rw-r--r-- 1 grid oinstall 0 Nov 26 02:52 /tmp/my_resource
[grid@c01n01 2_2]$
```

24. Restart Oracle Clusterware on the inactive Leaf Node.

```
[grid@c01n01 2_2]$ su -c "crsctl start cluster -n c01n03"
Password: <oracle>
CRS-2672: Attempting to start 'ora.evmd' on 'c01n03'
CRS-2672: Attempting to start 'ora.cssdmonitor' on 'c01n03'
CRS-2676: Start of 'ora.cssdmonitor' on 'c01n03' succeeded
CRS-2672: Attempting to start 'ora.cssd' on 'c01n03'
CRS-2672: Attempting to start 'ora.diskmon' on 'c01n03'
CRS-2676: Start of 'ora.diskmon' on 'c01n03' succeeded
CRS-2676: Start of 'ora.evmd' on 'c01n03' succeeded
```

```
CRS-2676: Start of 'ora.cssd' on 'c01n03' succeeded
CRS-2672: Attempting to start 'ora.storage' on 'c01n03'
CRS-2672: Attempting to start 'ora.ctssd' on 'c01n03'
CRS-2672: Attempting to start 'ora.cluster_interconnect.haip' on
'c01n03'
CRS-2676: Start of 'ora.storage' on 'c01n03' succeeded
CRS-2676: Start of 'ora.cluster_interconnect.haip' on 'c01n03'
succeeded
CRS-2676: Start of 'ora.ctssd' on 'c01n03' succeeded
CRS-2672: Attempting to start 'ora.crsd' on 'c01n03'
CRS-2676: Start of 'ora.crsd' on 'c01n03' succeeded
[grid@c01n01 2_2]$
```

Congratulations! You have successfully configured highly available application resources on Flex Cluster Leaf Nodes.

Practices for Lesson 3: Policy-Based Cluster Management

Chapter 3

Practices for Lesson 3: Overview

Practices Overview

In this practice, you will configure and use policy-based cluster management.

Practice 3-1: Configuring and Using Policy-Based Cluster Management

Overview

In this practice, you will configure server categories and the policy set. You will examine the effect of various changes to verify the dynamic nature of policy-based cluster management. Finally, you will see how easy it is to activate policies.

In this practice, you will configure policy-based cluster management on the Flex Cluster that you created in the previous practice. However, note that categorization and policy-based cluster management have no dependency on Flex Clusters and that you can apply all the principles of this practice to a standard cluster.

Assumptions

This practice relies on the configuration performed in Practice 2-2.

Tasks

1. Establish a terminal session connected to `c01n01` by using the `grid` OS user.

```
$ ssh grid@c01n01
grid@c01n01's password: <oracle>
[grid@c01n01 ~]$
```

2. Configure the environment by using the `oraenv` script. Enter `+ASM1` when you are prompted for an ORACLE_SID value.

```
[grid@c01n01 ~]$ . oraenv
ORACLE_SID = [grid] ? +ASM1
The Oracle base has been set to /u01/app/grid
[grid@c01n01 ~]$
```

3. Examine the extended server attributes associated with `c01n01` and `c01n02`. Notice the amount of physical memory associated with each server (5,142 MB). Notice also that the attributes identify these nodes as Hub Nodes in the cluster.

```
[grid@c01n01 ~]$ crsctl status server c01n01 -f
NAME=c01n01
MEMORY_SIZE=5142
CPU_COUNT=1
CPU_CLOCK_RATE=2922
CPU_HYPERTHREADING=0
CPU_EQUIVALENCY=1000
DEPLOYMENT=other
CONFIGURED_CSS_ROLE=hub
RESOURCE_USE_ENABLED=1
SERVER_LABEL=
PHYSICAL_HOSTNAME=
STATE=ONLINE
ACTIVE_POOLS=ora.orcldb
```

```
STATE_DETAILS=
ACTIVE_CSS_ROLE=hub

[grid@c01n01 ~]$ crsctl status server c01n02 -f
NAME=c01n02
MEMORY_SIZE=5142
CPU_COUNT=1
CPU_CLOCK_RATE=2922
CPU_HYPERTHREADING=0
CPU_EQUIVALENCY=1000
DEPLOYMENT=other
CONFIGURED_CSS_ROLE=hub
RESOURCE_USE_ENABLED=1
SERVER_LABEL=
PHYSICAL_HOSTNAME=
STATE=ONLINE
ACTIVE_POOLS=ora.orcldb
STATE_DETAILS=
ACTIVE_CSS_ROLE=hub

[grid@c01n01 ~]$
```

4. Examine the extended server attributes associated with `c01n03` and `c01n04`. Notice that these nodes contain less physical memory on each node (1,558 MB) and that they are identified as Leaf Nodes in this Flex Cluster.

```
[grid@c01n01 ~]$ crsctl status server c01n03 -f
NAME=c01n03
MEMORY_SIZE=1558
CPU_COUNT=1
CPU_CLOCK_RATE=2922
CPU_HYPERTHREADING=0
CPU_EQUIVALENCY=1000
DEPLOYMENT=other
CONFIGURED_CSS_ROLE=leaf
RESOURCE_USE_ENABLED=1
SERVER_LABEL=
PHYSICAL_HOSTNAME=
STATE=ONLINE
ACTIVE_POOLS=Free
STATE_DETAILS=
ACTIVE_CSS_ROLE=leaf

[grid@c01n01 ~]$ crsctl status server c01n04 -f
```

```
NAME=c01n04
MEMORY_SIZE=1558
CPU_COUNT=1
CPU_CLOCK_RATE=2922
CPU_HYPERTHREADING=0
CPU_EQUIVALENCY=1000
DEPLOYMENT=other
CONFIGURED_CSS_ROLE=leaf
RESOURCE_USE_ENABLED=1
SERVER_LABEL=
PHYSICAL_HOSTNAME=
STATE=ONLINE
ACTIVE_POOLS=ora.my_app_pool
STATE_DETAILS=
ACTIVE_CSS_ROLE=leaf

[grid@c01n01 ~]$
```

5. Examine the built-in category definitions. These are implicitly used to categorize the cluster nodes as Hub Nodes or Leaf Nodes based on the `ACTIVE_CSS_ROLE` setting. Note that these categories also exist in a standard cluster; however, in a standard cluster all the nodes are designated as Hub Nodes.

```
[grid@c01n01 ~]$ crsctl status category
NAME=ora.hub.category
ACL=owner:root:rwx,pgrp:root:r-x,other::r--
ACTIVE_CSS_ROLE=hub
EXPRESSION=

NAME=ora.leaf.category
ACL=owner:root:rwx,pgrp:root:r-x,other::r--
ACTIVE_CSS_ROLE=leaf
EXPRESSION=

[grid@c01n01 ~]$
```

6. Examine the servers that are associated with the built-in categories. Confirm that the server-to-category mappings are as expected.

```
[grid@c01n01 ~]$ crsctl status server -category ora.hub.category
NAME=c01n01
STATE=ONLINE

NAME=c01n02
STATE=ONLINE
```

```
[grid@c01n01 ~]$ crsctl status server -category
ora.leaf.category
NAME=c01n03
STATE=ONLINE

NAME=c01n04
STATE=ONLINE

[grid@c01n01 ~]$
```

7. Create a user-defined category that includes the servers that contain more than 4,000 MB of system memory.

```
[grid@c01n01 ~]$ crsctl add category big -attr
"EXPRESSION='(MEMORY_SIZE > 4000)'"
[grid@c01n01 ~]$
```

8. Examine the category definition that you created in the previous step. Notice that the category definition includes `ACTIVE_CSS_ROLE=hub` by default. You could modify the `ACTIVE_CSS_ROLE` attribute if you wanted to categorize Leaf Nodes.

```
[grid@c01n01 ~]$ crsctl status category big
NAME=big
ACL=owner:grid:rwx,pgrp:oinstall:rwx,other::r--
ACTIVE_CSS_ROLE=hub
EXPRESSION=(MEMORY_SIZE > 4000)

[grid@c01n01 ~]$
```

9. Examine the servers associated with the `big` category. Confirm that `c01n01` and `c01n02` are associated with the `big` category.

```
[grid@c01n01 ~]$ crsctl status server -category big
NAME=c01n01
STATE=ONLINE

NAME=c01n02
STATE=ONLINE

[grid@c01n01 ~]$
```

10. Examine the categories associated with the `c01n01` server. Notice that a server can be associated with multiple categories.

```
[grid@c01n01 ~]$ crsctl status category -server c01n01
NAME=big
ACL=owner:grid:rwx,pgrp:oinstall:rwx,other::r--
ACTIVE_CSS_ROLE=hub
EXPRESSION=(MEMORY_SIZE > 4000)

NAME=ora.hub.category
ACL=owner:root:rwx,pgrp:root:r-x,other::r--
ACTIVE_CSS_ROLE=hub
EXPRESSION=

[grid@c01n01 ~]$
```

At this point, you have configured a category and examined category definitions along with server-to-category associations. Next, you will make use of the category definitions as you configure and exercise policy-based cluster management.

11. Examine the policy set. At this point, because no policy set configuration has been performed, the only policy listed is the `Current` policy. The `Current` policy is a special built-in policy that contains all of the currently active server pool definitions.

```
[grid@c01n01 ~]$ crsctl status policyset
ACL=owner:grid:rwx,pgrp:oinstall:rwx,other::r--
LAST_ACTIVATED_POLICY=
SERVER_POOL_NAMES=Free
POLICY
  NAME=Current
  DESCRIPTION=This policy is built-in and managed automatically
to reflect current configuration
  SERVERPOOL
    NAME=Free
    IMPORTANCE=0
    MAX_SIZE=-1
    MIN_SIZE=0
    SERVER_CATEGORY=
    SERVER_NAMES=
  SERVERPOOL
    NAME=Generic
    IMPORTANCE=0
    MAX_SIZE=-1
    MIN_SIZE=0
    SERVER_CATEGORY=
    SERVER_NAMES=
  SERVERPOOL
```

```
    NAME=ora.my_app_pool
    IMPORTANCE=0
    MAX_SIZE=1
    MIN_SIZE=1
    SERVER_CATEGORY=ora.leaf.category
    SERVER_NAMES=
  SERVERPOOL
    NAME=ora.orcldb
    IMPORTANCE=0
    MAX_SIZE=2
    MIN_SIZE=0
    SERVER_CATEGORY=ora.hub.category
    SERVER_NAMES=
[grid@c01n01 ~]$
```

12. Examine the policy set definition provided in the file located at `/stage/labs/3_1/policyset.txt`. The policy set definition contains two policies. The `day` policy enables the `orcldb` server pool to use two Hub Nodes while not providing any allocation for the `bigpool` server pool. The `night` policy still prioritizes the `orcldb` server pool; however, it also provides an allocation for the `bigpool` server pool. Notice that the `bigpool` server pool references the `big` server category that you defined earlier in this practice. Notice also that the policy set does not list the `my_app_pool` server pool in the `SERVER_POOL_NAMES` attribute. This demonstrates how it is possible for you to define server pools that are outside policy set management.

```
[grid@c01n01 ~]$ more /stage/labs/3_1/policyset.txt
SERVER_POOL_NAMES=Free bigpool ora.orcldb
POLICY
  NAME=day
  DESCRIPTION=The day policy
  SERVERPOOL
    NAME=ora.orcldb
    IMPORTANCE=10
    MAX_SIZE=2
    MIN_SIZE=1
    SERVER_CATEGORY=ora.hub.category
  SERVERPOOL
    NAME=bigpool
    IMPORTANCE=0
    MAX_SIZE=0
    MIN_SIZE=0
    SERVER_CATEGORY=big
POLICY
  NAME=night
  DESCRIPTION=The night policy
```

```
  SERVERPOOL
    NAME=ora.orcldb
    IMPORTANCE=10
    MAX_SIZE=2
    MIN_SIZE=1
    SERVER_CATEGORY=ora.hub.category
  SERVERPOOL
    NAME=bigpool
    IMPORTANCE=5
    MAX_SIZE=1
    MIN_SIZE=1
    SERVER_CATEGORY=big

[grid@c01n01 ~]$
```

13. Modify the policy set to load the configuration file.

```
[grid@c01n01 ~]$ crsctl modify policyset -file
/stage/labs/3_1/policyset.txt
[grid@c01n01 ~]$
```

14. Reexamine the policy set to confirm that the configuration file was loaded in the previous step. Notice that the `LAST_ACTIVATED_POLICY` attribute is not set and that the `Current` policy is unchanged from what you observed earlier. This is because no policies have been activated yet.

```
[grid@c01n01 ~]$ crsctl status policyset
ACL=owner:grid:rwx,pgrp:oinstall:rwx,other::r--
LAST_ACTIVATED_POLICY=
SERVER_POOL_NAMES=Free bigpool ora.orcldb
POLICY
  NAME=Current
  DESCRIPTION=This policy is built-in and managed automatically
to reflect current configuration
  SERVERPOOL
    NAME=Free
    IMPORTANCE=0
    MAX_SIZE=-1
    MIN_SIZE=0
    SERVER_CATEGORY=
    SERVER_NAMES=
  SERVERPOOL
    NAME=Generic
    IMPORTANCE=0
    MAX_SIZE=-1
    MIN_SIZE=0
    SERVER_CATEGORY=
```

```
    SERVER_NAMES=
  SERVERPOOL
    NAME=ora.my_app_pool
    IMPORTANCE=0
    MAX_SIZE=1
    MIN_SIZE=1
    SERVER_CATEGORY=ora.leaf.category
    SERVER_NAMES=
  SERVERPOOL
    NAME=ora.orcldb
    IMPORTANCE=0
    MAX_SIZE=2
    MIN_SIZE=0
    SERVER_CATEGORY=ora.hub.category
    SERVER_NAMES=
POLICY
  NAME=day
  DESCRIPTION=The day policy
  SERVERPOOL
    NAME=Free
    IMPORTANCE=0
    MAX_SIZE=-1
    MIN_SIZE=0
    SERVER_CATEGORY=
    SERVER_NAMES=
  SERVERPOOL
    NAME=bigpool
    IMPORTANCE=0
    MAX_SIZE=0
    MIN_SIZE=0
    SERVER_CATEGORY=big
    SERVER_NAMES=
  SERVERPOOL
    NAME=ora.orcldb
    IMPORTANCE=10
    MAX_SIZE=2
    MIN_SIZE=1
    SERVER_CATEGORY=ora.hub.category
    SERVER_NAMES=
POLICY
  NAME=night
  DESCRIPTION=The night policy
```

```
  SERVERPOOL
    NAME=Free
    IMPORTANCE=0
    MAX_SIZE=-1
    MIN_SIZE=0
    SERVER_CATEGORY=
    SERVER_NAMES=
  SERVERPOOL
    NAME=bigpool
    IMPORTANCE=5
    MAX_SIZE=1
    MIN_SIZE=1
    SERVER_CATEGORY=big
    SERVER_NAMES=
  SERVERPOOL
    NAME=ora.orcldb
    IMPORTANCE=10
    MAX_SIZE=2
    MIN_SIZE=1
    SERVER_CATEGORY=ora.hub.category
    SERVER_NAMES=
[grid@c01n01 ~]$
```

15. Activate the `day` policy.

```
[grid@c01n01 ~]$ crsctl modify policyset -attr
"LAST_ACTIVATED_POLICY='day'"
[grid@c01n01 ~]$
```

16. Reexamine the policy set. Confirm that `LAST_ACTIVATED_POLICY=day` and that the `Current` policy settings reflect the `day` policy.

```
[grid@c01n01 ~]$ crsctl status policyset
LAST_ACTIVATED_POLICY=day
SERVER_POOL_NAMES=Free bigpool ora.orcldb
POLICY
  NAME=Current
  DESCRIPTION=This policy is built-in and managed automatically
to reflect current configuration
  SERVERPOOL
    NAME=Free
    IMPORTANCE=0
    MAX_SIZE=-1
    MIN_SIZE=0
    SERVER_CATEGORY=
    SERVER_NAMES=
```

```
  SERVERPOOL
    NAME=Generic
    IMPORTANCE=0
    MAX_SIZE=-1
    MIN_SIZE=0
    SERVER_CATEGORY=
    SERVER_NAMES=
  SERVERPOOL
    NAME=bigpool
    IMPORTANCE=0
    MAX_SIZE=0
    MIN_SIZE=0
    SERVER_CATEGORY=big
    SERVER_NAMES=
  SERVERPOOL
    NAME=ora.my_app_pool
    IMPORTANCE=0
    MAX_SIZE=1
    MIN_SIZE=1
    SERVER_CATEGORY=ora.leaf.category
    SERVER_NAMES=
  SERVERPOOL
    NAME=ora.orcldb
    IMPORTANCE=10
    MAX_SIZE=2
    MIN_SIZE=1
    SERVER_CATEGORY=ora.hub.category
    SERVER_NAMES=
POLICY
  NAME=day
  DESCRIPTION=The day policy
  SERVERPOOL
    NAME=Free
    IMPORTANCE=0
    MAX_SIZE=-1
    MIN_SIZE=0
    SERVER_CATEGORY=
    SERVER_NAMES=
  SERVERPOOL
    NAME=bigpool
    IMPORTANCE=0
    MAX_SIZE=0
```

```
    MIN_SIZE=0
    SERVER_CATEGORY=big
    SERVER_NAMES=
  SERVERPOOL
    NAME=ora.orcldb
    IMPORTANCE=10
    MAX_SIZE=2
    MIN_SIZE=1
    SERVER_CATEGORY=ora.hub.category
    SERVER_NAMES=
POLICY
  NAME=night
  DESCRIPTION=The night policy
  SERVERPOOL
    NAME=Free
    IMPORTANCE=0
    MAX_SIZE=-1
    MIN_SIZE=0
    SERVER_CATEGORY=
    SERVER_NAMES=
  SERVERPOOL
    NAME=bigpool
    IMPORTANCE=5
    MAX_SIZE=1
    MIN_SIZE=1
    SERVER_CATEGORY=big
    SERVER_NAMES=
  SERVERPOOL
    NAME=ora.orcldb
    IMPORTANCE=10
    MAX_SIZE=2
    MIN_SIZE=1
    SERVER_CATEGORY=ora.hub.category
    SERVER_NAMES=
[grid@c01n01 ~]$
```

17. Examine the server pool allocations. Confirm that both Hub Nodes (`c01n01` and `c01n02`) are still allocated to the `orcldb` server pool in line with the `day` policy.

```
[grid@c01n01 ~]$ crsctl status serverpool
NAME=Free
ACTIVE_SERVERS=c01n03

NAME=Generic
ACTIVE_SERVERS=

NAME=bigpool
ACTIVE_SERVERS=

NAME=ora.my_app_pool
ACTIVE_SERVERS=c01n04

NAME=ora.orcldb
ACTIVE_SERVERS=c01n01 c01n02

[grid@c01n01 ~]$
```

18. Activate the `night` policy.

```
[grid@c01n01 ~]$ crsctl modify policyset -attr
"LAST_ACTIVATED_POLICY='night'"
[grid@c01n01 ~]$
```

One of the key benefits of policy-based cluster management is that administrators can alter the cluster to reflect a different policy simply by activating the policy. In such cases, Oracle Clusterware performs the required server reallocations and also automatically starts and stops resources as required.

19. Examine the server pool allocations. Confirm that the servers allocated to each pool are consistent with the `night` policy.

```
[grid@c01n01 ~]$ crsctl status serverpool
NAME=Free
ACTIVE_SERVERS=c01n03

NAME=Generic
ACTIVE_SERVERS=

NAME=bigpool
ACTIVE_SERVERS=c01n02

NAME=ora.my_app_pool
ACTIVE_SERVERS=c01n04

NAME=ora.orcldb
ACTIVE_SERVERS=c01n01

[grid@c01n01 ~]$
```

20. One of the side effects of activating the night policy is that the RAC database (`orcl`) that previously ran on both `c01n01` and `c01n02` now only runs on one node. Confirm that only one instance of the `orcl` database is running. This demonstrates how Oracle Clusterware automatically starts and stops required resources when a policy change is made.

```
[grid@c01n01 ~]$ srvctl status database -d orcl
Instance orcl_2 is running on node c01n01
[grid@c01n01 ~]$
```

21. To illustrate the dynamic nature of policy-based cluster management, modify the `big` category so that no servers can be associated with it. Notice that the category change immediately causes a server re-allocation, which in turn causes an instance of the `orcl` database to start up.

```
[grid@c01n01 ~]$ crsctl modify category big -attr
"EXPRESSION='(MEMORY_SIZE > 8000)'"
CRS-2672: Attempting to start 'ora.orcl.db' on 'c01n02'
CRS-2676: Start of 'ora.orcl.db' on 'c01n02' succeeded
[grid@c01n01 ~]$
```

22. Reexamine the server pool allocations. Notice that no servers are associated with the `bigpool` server pool because of the change to the `big` category, which in turn results in two servers being allocated to the `orcldb` server pool.

```
[grid@c01n01 ~]$ crsctl status serverpool
NAME=Free
ACTIVE_SERVERS=c01n03

NAME=Generic
ACTIVE_SERVERS=

NAME=bigpool
ACTIVE_SERVERS=

NAME=ora.my_app_pool
ACTIVE_SERVERS=c01n04

NAME=ora.orcldb
ACTIVE_SERVERS=c01n01 c01n02

[grid@c01n01 ~]$
```

Congratulations! You have now configured and used the new policy-based cluster management capabilities in Oracle Clusterware release 12.1.

Practices for Lesson 4: What-If Command Evaluation

Chapter 4

Practices for Lesson 4: Overview

Practices Overview

In this practice, you will exercise what-if command evaluation.

Practice 4-1: Using What-If Command Evaluation

Overview

In this practice, you perform what-if command evaluation by using the different commands for various different resources. You will also examine the different output formatting options available with what-if command evaluation.

Assumptions

This practice relies on the configuration performed in Practice 2-2.

Tasks

1. Establish a terminal session connected to `c01n01` by using the `grid` OS user.

```
$ ssh grid@c01n01
Password: <oracle>
[grid@c01n01 ~]$
```

2. Configure the environment by using the `oraenv` script. Enter `+ASM1` when you are prompted for an ORACLE_SID value.

```
[grid@c01n01 ~]$ . oraenv
ORACLE_SID = [grid] ? +ASM1
The Oracle base has been set to /u01/app/grid
[grid@c01n01 ~]$
```

3. Check the status of the application resources that you created in Practice 2-2. If the application resources are not online, start them by executing `crsctl start my_resource`.

```
[grid@c01n01 ~]$ crsctl status resource -t -w "NAME co my"
--------------------------------------------------------------------------------
Name           Target  State        Server                   State details
--------------------------------------------------------------------------------
Cluster Resources
--------------------------------------------------------------------------------
my_dep_res1
      1        ONLINE  ONLINE       c01n04                   STABLE
my_dep_res2
      1        ONLINE  ONLINE       c01n04                   STABLE
my_resource
      1        ONLINE  ONLINE       c01n04                   STABLE
--------------------------------------------------------------------------------
[grid@c01n01 ~]$
```

4. Use what-if command evaluation to analyze the effect of forcibly stopping the resource `my_dep_res1`. Notice that forcibly stopping `my_dep_res1` also causes `my_resource` to stop. This is consistent with the mandatory (hard) dependency that exists between these resources.

```
[grid@c01n01 ~]$ crsctl eval stop resource my_dep_res1 -f

Stage Group 1:
--------------------------------------------------------------------
Stage Number    Required        Action
--------------------------------------------------------------------

     1              Y           Resource 'my_resource' (1/1)
                                will be in state
                                [OFFLINE]

     2              Y           Resource 'my_dep_res1' (1/1)
                                will be in state
                                [OFFLINE]

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

[grid@c01n01 ~]$
```

5. Use what-if command evaluation to analyze the effect of forcibly stopping the resource `my_dep_res2`. Notice that forcibly stopping `my_dep_res2` does not have any effect on `my_resource`. This is consistent with the optional (soft) dependency that exists between these resources.

```
[grid@c01n01 ~]$ crsctl eval stop resource my_dep_res2 -f

Stage Group 1:
--------------------------------------------------------------------
Stage Number    Required        Action
--------------------------------------------------------------------

     1              Y           Resource 'my_dep_res2' (1/1)
                                will be in state
                                [OFFLINE]

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

[grid@c01n01 ~]$
```

6. Stop the `my_dep_res1` and `my_dep_res2` resources. Confirm that each command behaves as indicated by the previous what-if command evaluations.

```
[grid@c01n01 ~]$ crsctl stop resource my_dep_res1 -f
CRS-2673: Attempting to stop 'my_resource' on 'c01n04'
CRS-2677: Stop of 'my_resource' on 'c01n04' succeeded
CRS-2673: Attempting to stop 'my_dep_res1' on 'c01n04'
CRS-2677: Stop of 'my_dep_res1' on 'c01n04' succeeded
[grid@c01n01 ~]$ crsctl stop resource my_dep_res2 -f
CRS-2673: Attempting to stop 'my_dep_res2' on 'c01n04'
CRS-2677: Stop of 'my_dep_res2' on 'c01n04' succeeded
[grid@c01n01 ~]$
```

7. Use what-if command evaluation to analyze the effect of restarting `my_resource`. Notice that starting `my_resource` would also cause the dependent resources `my_dep_res1` and `my_dep_res2` to start. Note also that starting `my_dep_res1` is listed as a required (mandatory) action and starting `my_dep_res2` is listed as not required (optional). This matches the mandatory and optional dependency definitions that exist between the resources.

```
[grid@c01n01 ~]$ crsctl eval start resource my_resource

Stage Group 1:
--------------------------------------------------------------------
Stage Number    Required        Action
--------------------------------------------------------------------

     1              Y           Resource 'my_dep_res1' (1/1)
                                will be in state
                                [ONLINE] on server [c01n04]
                    N           Resource 'my_dep_res2' (1/1)
                                will be in state
                                [ONLINE|INTERMEDIATE] on server
                                [c01n04]

     2              Y           Resource 'my_resource' (1/1)
                                will be in state
                                [ONLINE|INTERMEDIATE] on server
                                [c01n04]

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

[grid@c01n01 ~]$
```

8. Now use what-if command evaluation to analyze the effect of removing the server `c01n02` from the cluster. Notice how the output is divided into stages. In this example, the actions in stage one all relate to the immediate effects of removing the server. This includes removing `c01n02` from its current server pool and various resource stopping because the server is no longer available. In stage two, you can see how the VIP resources associated with `c01n02` are migrated to another server (`c01n01`). Finally, in stage three, a SCAN listener resource is also migrated. The migration of the SCAN listener resource occurs last and appears in a separate stage because that resource depends on one of the migrated VIP resources.

```
[grid@c01n01 ~]$ crsctl eval delete server c01n02 -f

Stage Group 1:
--------------------------------------------------------------------
Stage Number    Required        Action
--------------------------------------------------------------------

     1              Y           Server 'c01n02' will be removed
                                from pools [ora.orcldb]
                    Y           Resource 'ora.ASMNET1LSNR_ASM.
                                lsnr' (c01n02)
                                will be in state [OFFLINE]
                    Y           Resource 'ora.DATA.dg' (c01n02)
                                will be in state [OFFLINE]
                    Y           Resource 'ora.FRA.dg' (c01n02)
                                will be in state [OFFLINE]
                    Y           Resource 'ora.LISTENER.lsnr'
                                (c01n02) will be in
                                state [OFFLINE]
                    Y           Resource 'ora.LISTENER_SCAN1.
                                lsnr' (1/1) will be
                                in state [OFFLINE]
                    Y           Resource 'ora.asm' (2/1) will be
                                in state [OFFLINE]
                    Y           Resource 'ora.c01n02.vip' (1/1)
                                will be in state [OFFLINE]
                    Y           Resource 'ora.net1.network'
                                (c01n02) will be in
                                state [OFFLINE]
                    Y           Resource 'ora.ons' (c01n02) will
                                be in state [OFFLINE]
                    Y           Resource 'ora.orcl.db' (1/1)
                                will be in state [OFFLINE]
                    Y           Resource 'ora.proxy_advm'
                                (c01n02) will be in
                                state [OFFLINE]
```

```
                     Y           Resource 'ora.scan1.vip' (1/1)
                                 will be in state [OFFLINE]

     2               Y           Resource 'ora.c01n02.vip' (1/1)
                                 will be in state
                                 [ONLINE|INTERMEDIATE] on server
                                 [c01n01]
                     Y           Resource 'ora.scan1.vip' (1/1)
                                 will be in state
                                 [ONLINE] on server [c01n01]

     3               Y           Resource 'ora.LISTENER_SCAN1.
                                 lsnr' (1/1) will be
                                 in state [ONLINE|INTERMEDIATE]
                                 on server [c01n01]

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

[grid@c01n01 ~]$
```

9. You can also modify the output from the `crsctl eval` command by using the `-admin` option along with a series of additional modifiers. Reexecute the previous what-if command evaluation, but this time add the `-admin -a` options. See how this version of the command provides a concise summary of the servers allocated to each server pool following the removal of `c01n02`.

```
[grid@c01n01 ~]$ crsctl eval delete server c01n02 -f -admin -a


NAME = Free
ACTIVE_SERVERS  = c01n03

NAME = Generic
ACTIVE_SERVERS  =

NAME = bigpool
ACTIVE_SERVERS  =

NAME = ora.my_app_pool
ACTIVE_SERVERS  = c01n04

NAME = ora.orcldb
ACTIVE_SERVERS  = c01n01

[grid@c01n01 ~]$
```

10. Modify the previous command to just show the server pool changes following the removal of `c01n02`.

```
[grid@c01n01 ~]$ crsctl eval delete server c01n02 -f -admin -x


NAME = ora.orcldb
Old Active Servers = c01n01 c01n02
New Active Servers = c01n01

[grid@c01n01 ~]$
```

11. Again modify the previous command to show a complete list of cluster resources along with an indication of how the resources are affected by the removal of `c01n02`.

```
[grid@c01n01 ~]$ crsctl eval delete server c01n02 -f -admin -l
resources -a

--------------------------------------------------------------------
Name           Target  State        Server             Effect
--------------------------------------------------------------------
Local Resources
--------------------------------------------------------------------
ora.ASMNET1LSNR_ASM.lsnr
               ONLINE  ONLINE       c01n01
               ONLINE  OFFLINE      c01n02             Stopped
ora.DATA.dg
               ONLINE  ONLINE       c01n01
               ONLINE  OFFLINE      c01n02             Stopped
ora.FRA.dg
               ONLINE  ONLINE       c01n01
               ONLINE  OFFLINE      c01n02             Stopped
ora.LISTENER.lsnr
               ONLINE  ONLINE       c01n01
               ONLINE  OFFLINE      c01n02             Stopped
ora.LISTENER_LEAF.lsnr
               OFFLINE OFFLINE      c01n03
               OFFLINE OFFLINE      c01n04
ora.net1.network
               ONLINE  ONLINE       c01n01
               ONLINE  OFFLINE      c01n02             Stopped
ora.ons
               ONLINE  ONLINE       c01n01
               ONLINE  OFFLINE      c01n02             Stopped
ora.proxy_advm
```

```
               ONLINE  ONLINE       c01n01
               ONLINE  OFFLINE      c01n02           Stopped
--------------------------------------------------------------------------------
Cluster Resources
--------------------------------------------------------------------------------
my_dep_res1
      1        OFFLINE OFFLINE
my_dep_res2
      1        OFFLINE OFFLINE
my_resource
      1        OFFLINE OFFLINE
ora.LISTENER_SCAN1.lsnr
      1        ONLINE  ONLINE       c01n01           Relocated
ora.LISTENER_SCAN2.lsnr
      1        ONLINE  ONLINE       c01n01
ora.LISTENER_SCAN3.lsnr
      1        ONLINE  ONLINE       c01n01
ora.MGMTLSNR
      1        OFFLINE OFFLINE
ora.asm
      1        ONLINE  ONLINE       c01n01
      2        ONLINE  OFFLINE                       Stopped
      3        OFFLINE OFFLINE
ora.c01n01.vip
      1        ONLINE  ONLINE       c01n01
ora.c01n02.vip
      1        ONLINE  ONLINE       c01n01           Relocated
ora.cvu
      1        ONLINE  ONLINE       c01n01
ora.gns
      1        ONLINE  ONLINE       c01n01
ora.gns.vip
      1        ONLINE  ONLINE       c01n01
ora.mgmtdb
      1        OFFLINE OFFLINE
ora.oc4j
      1        ONLINE  ONLINE       c01n01
ora.orcl.db
      1        ONLINE  OFFLINE                       Stopped
      2        ONLINE  ONLINE       c01n01
ora.scan1.vip
      1        ONLINE  ONLINE       c01n01           Relocated
```

```
ora.scan2.vip
      1        ONLINE  ONLINE       c01n01
ora.scan3.vip
      1        ONLINE  ONLINE       c01n01
--------------------------------------------------------------------
[grid@c01n01 ~]$
```

12. Establish another terminal session connected to `c01n01` as the `oracle` OS user.

```
$ ssh oracle@c01n01
Password: <oracle>
[oracle@c01n01 ~]$
```

13. Configure the environment by using the `oraenv` script. Enter `orcl` when you are prompted for an ORACLE_SID value.

```
[oracle@c01n01 ~]$ . oraenv
ORACLE_SID = [oracle] ? orcl
The Oracle base has been set to /u01/app/oracle
[oracle@c01n01 ~]$
```

14. In preparation for the next part of this practice, stop the `orcl` database.

```
[oracle@c01n01 ~]$ srvctl stop database -d orcl
[oracle@c01n01 ~]$
```

15. Back in the grid terminal session, stop the `FRA` diskgroup resource, and forcibly stop the SCAN VIPs.

```
[grid@c01n01 ~]$ crsctl stop resource ora.FRA.dg
CRS-2673: Attempting to stop 'ora.FRA.dg' on 'c01n02'
CRS-2673: Attempting to stop 'ora.FRA.dg' on 'c01n01'
CRS-2677: Stop of 'ora.FRA.dg' on 'c01n02' succeeded
CRS-2677: Stop of 'ora.FRA.dg' on 'c01n01' succeeded
[grid@c01n01 ~]$ crsctl stop resource ora.scan1.vip -f
CRS-2673: Attempting to stop 'ora.LISTENER_SCAN1.lsnr' on
'c01n02'
CRS-2677: Stop of 'ora.LISTENER_SCAN1.lsnr' on 'c01n02'
succeeded
CRS-2673: Attempting to stop 'ora.scan1.vip' on 'c01n02'
CRS-2677: Stop of 'ora.scan1.vip' on 'c01n02' succeeded
[grid@c01n01 ~]$ crsctl stop resource ora.scan2.vip -f
CRS-2673: Attempting to stop 'ora.LISTENER_SCAN2.lsnr' on
'c01n01'
CRS-2677: Stop of 'ora.LISTENER_SCAN2.lsnr' on 'c01n01'
succeeded
CRS-2673: Attempting to stop 'ora.scan2.vip' on 'c01n01'
CRS-2677: Stop of 'ora.scan2.vip' on 'c01n01' succeeded
[grid@c01n01 ~]$ crsctl stop resource ora.scan3.vip -f
```

```
CRS-2673: Attempting to stop 'ora.LISTENER_SCAN3.lsnr' on
'c01n01'
CRS-2677: Stop of 'ora.LISTENER_SCAN3.lsnr' on 'c01n01'
succeeded
CRS-2673: Attempting to stop 'ora.scan3.vip' on 'c01n01'
CRS-2677: Stop of 'ora.scan3.vip' on 'c01n01' succeeded
[grid@c01n01 ~]$
```

16. So far, you have used what-if command evaluation in conjunction with the `crsctl` command. Typically, the `crsctl` command is available to cluster administrators but not to database administrators. However, database administrators can perform what-if command evaluation by using the `srvctl` command with the `-eval` option. In the `oracle` terminal session, use `srvctl ... -eval` to analyze the effect of starting the `orcl` database. Notice that the command output does not contain as much detail as that provided by `crsctl eval`. However, it does clearly indicate the effects associated with restarting the database in the current environment.

```
[oracle@c01n01 ~]$ srvctl start database -d orcl -eval
Resource ora.FRA.dg will be started on nodes c01n01,c01n02
VIP scan1 will be started on node c01n02
VIP scan2 will be started on node c01n01
VIP scan3 will be started on node c01n01
Listener LISTENER_SCAN1 will be started on node c01n02
Listener LISTENER_SCAN2 will be started on node c01n01
Listener LISTENER_SCAN3 will be started on node c01n01
Database orcl will be started on nodes c01n02,c01n01
[oracle@c01n01 ~]$
```

17. Restart the `orcl` database.

```
[oracle@c01n01 ~]$ srvctl start database -d orcl
[oracle@c01n01 ~]$
```

18. Back in the `grid` terminal session, confirm that the required resources all started as predicted by the what-if command evaluation in step 14.

```
[grid@c01n01 ~]$ srvctl status diskgroup -g FRA
Disk Group FRA is running on c01n02,c01n01
[grid@c01n01 ~]$ srvctl status scan_listener
SCAN Listener LISTENER_SCAN1 is enabled
SCAN listener LISTENER_SCAN1 is running on node c01n02
SCAN Listener LISTENER_SCAN2 is enabled
SCAN listener LISTENER_SCAN2 is running on node c01n01
SCAN Listener LISTENER_SCAN3 is enabled
SCAN listener LISTENER_SCAN3 is running on node c01n01
[grid@c01n01 ~]$ srvctl status database -d orcl
Instance orcl_1 is running on node c01n02
Instance orcl_2 is running on node c01n01
[grid@c01n01 ~]$
```

19. What-if command evaluation by using `srvctl ... -eval` is not limited to analyzing the effects of starting and stopping databases. Use the following command to check the effect of permitting a maximum of one server in the `orcldb` server pool.

```
[grid@c01n01 ~]$ srvctl modify srvpool -serverpool orcldb -max 1
-eval
Database orcl will be stopped on node c01n01
Server c01n01 will be moved from pool orcldb to pool Free
[grid@c01n01 ~]$
```

Congratulations! You have now seen various examples of how to use what-if command evaluation.

Practices for Lesson 5: Other Clusterware New Features

Chapter 5

Practices for Lesson 5

Practices Overview

There are no practices for this lesson.

Practices for Lesson 6: Flex ASM

Chapter 6

Practices for Lesson 6: Overview

Practices Overview

In this practice, you will crash an ASM instance and examine how the client database transparently fails over to another Flex ASM instance.

Practice 6-1: Client Database Failover with Flex ASM

Overview

In this practice, you will crash an ASM instance and examine how the client database transparently fails over to another Flex ASM instance.

Assumptions

This practice relies on the configuration performed in Practice 2-1.

Tasks

1. Establish a terminal session connected to `c01n01` by using the `grid` OS user.

```
$ ssh grid@c01n01
grid@c01n01's password: <oracle>
[grid@c01n01 ~]$
```

2. Configure the environment by using the `oraenv` script. Enter `+ASM1` when you are prompted for an ORACLE_SID value.

```
[grid@c01n01 ~]$ . oraenv
ORACLE_SID = [grid] ? +ASM1
The Oracle base has been set to /u01/app/grid
[grid@c01n01 ~]$
```

3. Change to the directory that contains the scripts associated with this practice.

```
[grid@c01n01 ~]$ cd /stage/labs/6_1
[grid@c01n01 6_1]$
```

4. Examine the contents of `asm_clients.sql`. This query shows the clients connected to all of the Flex ASM instances in the cluster.

```
[grid@c01n01 6_1]$ cat asm_clients.sql
col client_instance_name format a20

select distinct i.instance_name asm_instance_name,
c.instance_name client_instance_name, c.db_name, c.status
from gv$instance i, gv$asm_client c
where i.inst_id=c.inst_id;

exit
[grid@c01n01 6_1]$
```

5. Examine the clients connected to each of the Flex ASM instances. In the following example, `orcl_1` is connected to `+ASM2`, and `orcl_2` is connected to `+ASM1`. Take note of the connection mappings in your environment, because they may differ. In particular, take note of the database instance connected to `+ASM2`.

```
[grid@c01n01 6_1]$ sqlplus / as sysasm @asm_clients

SQL*Plus: Release 12.1.0.1.0 Production...

ASM_INSTANCE_NAM CLIENT_INSTANCE_NAME DB_NAME  STATUS
---------------- -------------------- -------- ------------
+ASM1            +APX1                +APX     CONNECTED
+ASM1            +ASM1                +ASM     CONNECTED
+ASM1            orcl_2               orcl     CONNECTED
+ASM2            +APX2                +APX     CONNECTED
+ASM2            +ASM2                +ASM     CONNECTED
+ASM2            orcl_1               orcl     CONNECTED

6 rows selected.

Disconnected from Oracle Database 12c...
[grid@c01n01 6_1]$
```

6. Use the following command to take note of which server is running the database instance connected to `+ASM2`. In this case, the database instance is `orcl_1` running on `c01n02`; however, this may vary in your environment.

```
[grid@c01n01 6_1]$ srvctl status database -d orcl
Instance orcl_1 is running on node c01n02
Instance orcl_2 is running on node c01n01
[grid@c01n01 6_1]$
```

7. Establish another terminal session, using the `oracle` OS user, which is connected to the server that you identified in the previous step.

   Note that you may see additional messages relating to server identities. Answer `yes` if you are prompted to acknowledge server authenticity.

```
$ ssh oracle@c01n02
oracle@c01n02's password: <oracle>
[oracle@c01n02 ~]$
```

8. Configure the environment using the `oraenv` script. Enter `orcl` when you are prompted for an ORACLE_SID value.

```
[oracle@c01n02 ~]$ . oraenv
ORACLE_SID = [oracle] ? orcl
The Oracle base has been set to /u01/app/oracle
[oracle@c01n02 ~]
```

9. Set the ORACLE_SID environment variable to reference the database instance connected to +ASM2.

```
[oracle@c01n02 ~]$ export ORACLE_SID=orcl_1
[oracle@c01n02 ~]$
```

10. Change to the directory that contains the scripts associated with this practice.

```
[oracle@c01n02 ~]$ cd /stage/labs/6_1
[oracle@c01n02 6_1]$
```

11. Connect to the RAC database instance as shown below. Confirm that you are connected to the database instance that is a client of +ASM2.

```
[oracle@c01n02 6_1]$ sqlplus system/oracle_4U

SQL*Plus: Release 12.1.0.1.0 Production...

SQL> select instance_name from v$instance;

INSTANCE_NAME
----------------
orcl_1

SQL>
```

12. Start a workload using the script provided. The script executes a series of transactions that flush the buffer cache, query data, and update data. After you have started the workload, leave it running in the background and proceed to the next step.

```
SQL> @workload

System altered.


SYSTIMESTAMP
---------------------------------------------------------------------
28-NOV-12 03.22.41.102216 AM +00:00


  COUNT(*) AVG(AMOUNT_SOLD)
---------- ----------------
    799745       127.856296


99999 rows updated.


Commit complete.
```

13. Back in your `grid` terminal session, abort the Flex ASM instance on `c01n02` (`+ASM2`).

```
[grid@c01n01 6_1]$ srvctl stop asm -node c01n02 -stopoption
ABORT -force
[grid@c01n01 6_1]$
```

14. Confirm that no ASM instance is running on `c01n02`.

```
[grid@c01n01 6_1]$ crsctl status resource ora.asm -t
--------------------------------------------------------------------
Name           Target  State        Server       State details
--------------------------------------------------------------------
Cluster Resources
--------------------------------------------------------------------
ora.asm
      1        ONLINE  ONLINE       c01n01       STABLE
      2        OFFLINE OFFLINE                   STABLE
      3        OFFLINE OFFLINE                   STABLE
--------------------------------------------------------------------
[grid@c01n01 6_1]$
```

15. Back in your `oracle` terminal session, confirm that the workload is still running. This demonstrates how Flex ASM improves availability by transparently failing over client database instances if a Flex ASM instance fails.

```
SYSTIMESTAMP
---------------------------------------------------------------
28-NOV-12 03.23.48.651444 AM +00:00


  COUNT(*) AVG(AMOUNT_SOLD)
---------- ----------------
    140332       298.433246


99999 rows updated.


Commit complete.


System altered.


SYSTIMESTAMP
---------------------------------------------------------------
```

16. Return to your `grid` terminal session and reexamine the Flex ASM client connections. Notice that both `orcl` database instances are now connected to the remaining Flex ASM instance.

```
[grid@c01n01 6_1]$ sqlplus / as sysasm @asm_clients

SQL*Plus: Release 12.1.0.1.0 Production...

ASM_INSTANCE_NAM CLIENT_INSTANCE_NAME DB_NAME  STATUS
---------------- -------------------- -------- ------------
+ASM1            +APX1                +APX     CONNECTED
+ASM1            +APX2                +APX     CONNECTED
+ASM1            +ASM1                +ASM     CONNECTED
+ASM1            orcl_1               orcl     CONNECTED
+ASM1            orcl_2               orcl     CONNECTED

Disconnected from Oracle Database 12c...
[grid@c01n01 6_1]$
```

17. Return to your `oracle` terminal session. If the workload is still running, press Ctrl + C to stop the workload. Exit the database session after stopping the workload.

```
  COUNT(*) AVG(AMOUNT_SOLD)
---------- ----------------
    886003       138.977861


99999 rows updated.


Commit complete.

^Calter system flush buffer_cache
*
ERROR at line 1:
ORA-01013: user requested cancel of current operation



SQL> exit
Disconnected from Oracle Database 12c...
[oracle@c01n02 6_1]$
```

18. Return to your `grid` terminal session and restart the Flex ASM instance that you stopped earlier in this practice.

```
[grid@c01n01 6_1]$ srvctl start asm -node c01n02
[grid@c01n01 6_1]$
```

19. Confirm that Flex ASM is again running on `c01n01` and `c01n02`.

```
[grid@c01n01 6_1]$ srvctl status asm
ASM is running on c01n02,c01n01
[grid@c01n01 6_1]$
```

20. Re-examine the Flex ASM client connections. Notice that both `orcl` database instances are still connected to one Flex ASM instance. This is because Flex ASM does not redistribute clients when an ASM instance is added.

```
[grid@c01n01 6_1]$ sqlplus / as sysasm @asm_clients

SQL*Plus: Release 12.1.0.1.0 Production...

ASM_INSTANCE_NAM CLIENT_INSTANCE_NAME DB_NAME  STATUS
---------------- -------------------- -------- ------------
+ASM1            +APX1                +APX     CONNECTED
+ASM1            +APX2                +APX     CONNECTED
+ASM1            +ASM1                +ASM     CONNECTED
+ASM1            orcl_1               orcl     CONNECTED
+ASM1            orcl_2               orcl     CONNECTED

Disconnected from Oracle Database 12c...
[grid@c01n01 6_1]$
```

21. Connect to ASM as an ASM administrator.

```
[grid@c01n01 6_1]$ sqlplus / as sysasm

SQL*Plus: Release 12.1.0.1.0 Production...

SQL>
```

22. Select one of the `orcl` database instances and relocate it. This statement closes the connection between the database instance and Flex ASM instance, triggering a reconnection to another Flex ASM instance.

```
SQL> alter system relocate client 'orcl_1:orcl';

System altered.

SQL>
```

23. Relocate the ADVM proxy instance +APX2.

```
SQL> alter system relocate client '+APX2:+APX';

System altered.

SQL>
```

24. Re-examine the Flex ASM client connections. Confirm that the output is consistent with the relocations performed in the last two steps.

```
SQL> @asm_clients

ASM_INSTANCE_NAM CLIENT_INSTANCE_NAME DB_NAME  STATUS
---------------- -------------------- -------- ------------
+ASM1            +APX1                +APX     CONNECTED
+ASM1            +ASM1                +ASM     CONNECTED
+ASM1            orcl_2               orcl     CONNECTED
+ASM2            +APX2                +APX     CONNECTED
+ASM2            orcl_1               orcl     CONNECTED

Disconnected from Oracle Database 12c...
[grid@c01n01 6_1]$
```

Congratulations! You have exercised client database fail over with Flex ASM.

Practices for Lesson 7: Other ASM New Features

Chapter 7

Practices for Lesson 7: Overview

Practices Overview

In these practices, you will:

- Examine a more efficient method for disk replacement
- Use and manage ASM-based password files

Practice 7-1: More Efficient Disk Replacement

Overview

In this practice, you will exercise a new command that enables an ASM disk to be efficiently replaced by another ASM disk. You will also see how the new disk replacement capability compares with the capability available in previous versions.

Assumptions

This practice relies on the configuration established in Practice 2-1.

Tasks

1. Establish a terminal session connected to `c01n01` by using the `grid` OS user.

```
$ ssh grid@c01n01
Password: <oracle>
[grid@c01n01 ~]$
```

2. Configure the environment by using the `oraenv` script. Enter `+ASM1` when you are prompted for an ORACLE_SID value.

```
[grid@c01n01 ~]$ . oraenv
ORACLE_SID = [grid] ? +ASM1
The Oracle base has been set to /u01/app/grid
[grid@c01n01 ~]$
```

3. Use the ASM command utility (ASMCMD) to examine the disks in the `DATA` disk group. Notice that each disk in the `DATA` disk group contains a similar amount of free space (approximately 1,600 MB).

```
[grid@c01n01 ~]$ asmcmd lsdsk -k -G data
Total_MB  Free_MB  OS_MB  Name       Failgroup  Failgroup_Type
Library  Label  UDID  Product  Redund   Path
    2870     1588   2870  DATA_0000  DATA_0000  REGULAR
System                          UNKNOWN  /dev/xvdf1
    2870     1594   2870  DATA_0001  DATA_0001  REGULAR
System                          UNKNOWN  /dev/xvdf10
    2870     1620   2870  DATA_0002  DATA_0002  REGULAR
System                          UNKNOWN  /dev/xvdf11
    2870     1597   2870  DATA_0003  DATA_0003  REGULAR
System                          UNKNOWN  /dev/xvdf12
    2870     1618   2870  DATA_0004  DATA_0004  REGULAR
System                          UNKNOWN  /dev/xvdf2
    2870     1633   2870  DATA_0005  DATA_0005  REGULAR
System                          UNKNOWN  /dev/xvdf3
    2870     1632   2870  DATA_0006  DATA_0006  REGULAR
System                          UNKNOWN  /dev/xvdf5
    2870     1626   2870  DATA_0007  DATA_0007  REGULAR
System                          UNKNOWN  /dev/xvdf6
    2870     1639   2870  DATA_0008  DATA_0008  REGULAR
System                          UNKNOWN  /dev/xvdf7
```

```
    2870     1627   2870  DATA_0009  DATA_0009  REGULAR
System                          UNKNOWN  /dev/xvdf8
[grid@c01n01 ~]$
```

4. Use the ASM command utility (ASMCMD) to examine the free (candidate) disks available to ASM. Notice that both candidate disks are the same size (2,870 MB) as the disks already allocated to the DATA disk group.

```
[grid@c01n01 ~]$ asmcmd lsdsk --candidate -k
Total_MB  Free_MB  OS_MB  Name       Failgroup  Failgroup_Type
Library  Label  UDID  Product  Redund   Path
       0         0   2870                        REGULAR
System                          UNKNOWN  /dev/xvdg2
       0         0   2870                        REGULAR
System                          UNKNOWN  /dev/xvdg3
[grid@c01n01 ~]$
```

In the first part of this practice, you will replace an ASM disk by dropping it and then adding another disk to the disk group. This was the only method available in previous versions.

5. Connect to ASM as an ASM administrator.

```
[grid@c01n01 ~]$ sqlplus / as sysasm

SQL*Plus: Release 12.1.0.1.0 Production...

SQL>
```

6. Configure the SQL*Plus session to display timing information.

```
SQL> set timing on
SQL>
```

7. Execute the following command to drop one of the disks in the DATA disk group and add another disk into the DATA disk group. Note the amount of time taken to complete the operation (2 minutes and 39 seconds in the example below).

```
SQL> alter diskgroup data drop disk data_0000 add disk
'/dev/xvdg3' rebalance wait;

Diskgroup altered.

Elapsed: 00:02:39.60
SQL>
```

8. Exit your SQL*Plus session.

```
SQL> exit
Disconnected from Oracle Database 12c...
[grid@c01n01 ~]$
```

9. Reexamine the disks in the `DATA` disk group. Compare this output with what you observed in step 3. Notice that the disk named `DATA_0000` is deleted and the disk associated with `/dev/xvdg3` is added.

```
[grid@c01n01 ~]$ asmcmd lsdsk -k -G data
Total_MB  Free_MB  OS_MB  Name       Failgroup  Failgroup_Type
Library  Label  UDID  Product  Redund   Path
    2870     1588   2870  DATA_0001  DATA_0001  REGULAR
System                          UNKNOWN  /dev/xvdf10
    2870     1627   2870  DATA_0002  DATA_0002  REGULAR
System                          UNKNOWN  /dev/xvdf11
    2870     1594   2870  DATA_0003  DATA_0003  REGULAR
System                          UNKNOWN  /dev/xvdf12
    2870     1594   2870  DATA_0004  DATA_0004  REGULAR
System                          UNKNOWN  /dev/xvdf2
    2870     1629   2870  DATA_0005  DATA_0005  REGULAR
System                          UNKNOWN  /dev/xvdf3
    2870     1627   2870  DATA_0006  DATA_0006  REGULAR
System                          UNKNOWN  /dev/xvdf5
    2870     1621   2870  DATA_0007  DATA_0007  REGULAR
System                          UNKNOWN  /dev/xvdf6
    2870     1625   2870  DATA_0008  DATA_0008  REGULAR
System                          UNKNOWN  /dev/xvdf7
    2870     1629   2870  DATA_0009  DATA_0009  REGULAR
System                          UNKNOWN  /dev/xvdf8
    2870     1640   2870  DATA_0010  DATA_0010  REGULAR
System                          UNKNOWN  /dev/xvdg3
[grid@c01n01 ~]$
```

In the second part of this practice, you will replace an ASM disk by using the new disk replacement command available in release 12.1.

10. Connect to ASM as an ASM administrator.

```
[grid@c01n01 ~]$ sqlplus / as sysasm

SQL*Plus: Release 12.1.0.1.0 Production...

SQL>
```

11. Configure the SQL*Plus session to display timing information.

```
SQL> set timing on
SQL>
```

12. To replace a disk by using the new method, the disk must first be taken offline. However to take a disk offline, the disk group attribute `compatible.rdbms` must be set to `11.1.0.0.0` or later. Using the following commands, set the disk group attribute `compatible.rdbms` to `12.1.0.0.0` and take one of the disks offline.

```
SQL> alter diskgroup data set attribute 'compatible.rdbms' =
'12.1.0.0.0';

Diskgroup altered.

Elapsed: 00:00:01.35
SQL> alter diskgroup data offline disk data_0010;

Diskgroup altered.

Elapsed: 00:00:00.06
SQL>
```

13. Use the following command to replace the offline disk. Note the amount of time taken to complete the operation (1 minute and 44 seconds in the example below).

```
SQL> alter diskgroup data replace disk data_0010 with
'/dev/xvdf1' wait;

Diskgroup altered.

Elapsed: 00:01:44.94
SQL>
```

    You should observe that this method of disk replacement is quicker than the drop-and-add method that you observed earlier in the practice. This is because the drop-and-add method requires the entire disk group to be rebalanced, while the new method requires only the data on the replacement disk to be written.

14. Exit your SQL*Plus session.

```
SQL> exit
Disconnected from Oracle Database 12c...
[grid@c01n01 ~]$
```

15. Reexamine the disks in the `DATA` disk group. Compare this output with what you observed in step 9. Notice that the disk named `DATA_0010` is now associated with `/dev/xvdf1` instead of `/dev/xvdg3`.

```
[grid@c01n01 ~]$ asmcmd lsdsk -k -G data
Total_MB  Free_MB  OS_MB  Name       Failgroup  Failgroup_Type
Library  Label  UDID  Product  Redund   Path
    2870     1634   2870  DATA_0010  DATA_0010  REGULAR
System                        UNKNOWN  /dev/xvdf1
    2870     1592   2870  DATA_0001  DATA_0001  REGULAR
System                        UNKNOWN  /dev/xvdf10
    2870     1626   2870  DATA_0002  DATA_0002  REGULAR
System                        UNKNOWN  /dev/xvdf11
    2870     1595   2870  DATA_0003  DATA_0003  REGULAR
System                        UNKNOWN  /dev/xvdf12
    2870     1594   2870  DATA_0004  DATA_0004  REGULAR
System                        UNKNOWN  /dev/xvdf2
    2870     1627   2870  DATA_0005  DATA_0005  REGULAR
System                        UNKNOWN  /dev/xvdf3
    2870     1627   2870  DATA_0006  DATA_0006  REGULAR
System                        UNKNOWN  /dev/xvdf5
    2870     1620   2870  DATA_0007  DATA_0007  REGULAR
System                        UNKNOWN  /dev/xvdf6
    2870     1624   2870  DATA_0008  DATA_0008  REGULAR
System                        UNKNOWN  /dev/xvdf7
    2870     1629   2870  DATA_0009  DATA_0009  REGULAR
System                        UNKNOWN  /dev/xvdf8
[grid@c01n01 ~]$
```

Congratulations! You have now used the new capabilities that enable more efficient disk replacement in Oracle Database 12*c*.

Practice 7-2: Managing ASM-Based Password Files

Overview

In this practice, you will examine the default-created password files inside ASM. You will then delete and re-create the ASM-based password file for a database by using the `orapwd` utility.

Assumptions

This practice relies on the configuration established in Practice 2-1.

Tasks

1. Establish a terminal session connected to `c01n01` by using the `grid` OS user.

```
$ ssh grid@c01n01
Password: <oracle>
[grid@c01n01 ~]$
```

2. Configure the environment by using the `oraenv` script. Enter `+ASM1` when you are prompted for an ORACLE_SID value.

```
[grid@c01n01 ~]$ . oraenv
ORACLE_SID = [grid] ? +ASM1
The Oracle base has been set to /u01/app/grid
[grid@c01n01 ~]$
```

3. With Oracle Clusterware release 12.1, password files are stored on an ASM disk group by default. The exact location is stored as an attribute of the cluster resource that is associated with the corresponding database or ASM. Use the following command to locate the ASM password file.

```
[grid@c01n01 ~]$ crsctl status resource ora.asm -f | grep PWFILE
PWFILE=+DATA/orapwASM
[grid@c01n01 ~]$
```

4. Use the following command to locate the password file for the `orcl` database.

```
[grid@c01n01 ~]$ crsctl status resource ora.orcl.db -f | grep
PWFILE
PWFILE=+DATA/orcl/orapworcl
[grid@c01n01 ~]$
```

5. Start the ASM command utility ASMCMD. Use the `-p` option to ensure that the current working directory in displayed in the prompt.

```
[grid@c01n01 ~]$ asmcmd -p
ASMCMD [+] >
```

6. ASMCMD provides the `pwget` command, which you can use to locate ASM-based password files. Use the following command to locate the ASM password file.

```
ASMCMD [+] > pwget --asm
+DATA/orapwASM
ASMCMD [+] >
```

7. Use the following command to locate the password file for the `orcl` database.

```
ASMCMD [+] > pwget --dbuniquename orcl
+DATA/orcl/orapworcl
ASMCMD [+] >
```

8. Navigate to the directory where `orapworcl` is located.

```
ASMCMD [+] > cd +DATA/orcl
ASMCMD [+DATA/orcl] >
```

9. Execute the following command. Notice that `orapworcl` is not actually a file. Rather it is a link that points to the true location of the password file. This link is provided so that administrators do not have to enter the full name of the file (`pwdorcl.268.800448661` in this example), thus simplifying access to it.

```
ASMCMD [+DATA/orcl] > ls -l orapworcl
Type      Redund  Striped  Time             Sys  Name
PASSWORD  HIGH    COARSE   NOV 27 10:00:00  N    orapworcl =>
+DATA/ORCL/PASSWORD/pwdorcl.268.800448661
ASMCMD [+DATA/orcl] >
```

10. Navigate to the directory where the password file is located and examine it.

```
ASMCMD [+DATA/orcl] > cd password
ASMCMD [+DATA/orcl/password] > ls -l
Type      Redund  Striped  Time             Sys  Name
PASSWORD  HIGH    COARSE   NOV 27 10:00:00  Y
pwdorcl.268.800448661
ASMCMD [+DATA/orcl/password] >
```

11. Navigate back to the root directory.

```
ASMCMD [+DATA/orcl/password] > cd +
ASMCMD [+] >
```

12. Establish another terminal session connected to `c01n01` as the `oracle` OS user.

```
$ ssh oracle@c01n01
Password: <oracle>
[oracle@c01n01 ~]$
```

13. Configure the environment by using the `oraenv` script. Enter `orcl` when you are prompted for an ORACLE_SID value.

```
[oracle@c01n01 ~]$ . oraenv
ORACLE_SID = [oracle] ? orcl
The Oracle base has been set to /u01/app/oracle
[oracle@c01n01 ~]$
```

14. Connect to the database as a database administrator by using a network login. This sort of connection requires authentication using the password file. The success of this connection demonstrates the ASM-based password file in action.

```
[oracle@c01n01 ~]$ sqlplus sys@orcl as sysdba

SQL*Plus: Release 12.1.0.1.0 Production...

Enter password: <oracle_4U>

Connected to:
Oracle Database 12c Enterprise Edition...

SQL>
```

15. Quit the newly established SQL*Plus session.

```
SQL> exit
Disconnected from Oracle Database 12c...
[oracle@c01n01 ~]$
```

16. With release 12.1, administrators still have the `orapwd` utility to help manage password files, including password files stored inside ASM. Use the `orapwd` utility to delete the password file for the `orcl` database.

```
[oracle@c01n01 ~]$ orapwd dbuniquename='orcl' delete=y
[oracle@c01n01 ~]$
```

17. Back in your ASMCMD session, verify that the ASM-based password file for the `orcl` database has been deleted.

```
ASMCMD [+] > pwget --dbuniquename orcl
Password file location has not been set for DB instance
ASMCMD [+] >
```

18. Back in your `oracle` terminal session, try to connect to the database as a database administrator by using a network login. This time the connection is denied because there is no password file associated with the database.

```
[oracle@c01n01 ~]$ sqlplus sys@orcl as sysdba

SQL*Plus: Release 12.1.0.1.0 Production...

Enter password: <oracle_4U>
ERROR:
ORA-01017: invalid username/password; logon denied


Enter user-name: ^C
[oracle@c01n01 ~]$
```

19. Use the `orapwd` utility to re-create the database password file.

```
[oracle@c01n01 ~]$ orapwd file='+data/orcl/orapworcl'
dbuniquename='orcl' password='oracle_4U'
[oracle@c01n01 ~]$
```

20. Back in your ASMCMD session, verify that the password file has been re-created.

```
ASMCMD [+] > pwget --dbuniquename orcl
+DATA/orcl/orapworcl
ASMCMD [+] >
```

21. Back in your `oracle` terminal session, try to connect to the database as a database administrator by using a network login. This time the connection succeeds because a valid password file exists.

```
[oracle@c01n01 ~]$ sqlplus sys@orcl as sysdba
SQL*Plus: Release 12.1.0.1.0 Production...

Enter password: <oracle_4U>

Connected to:
Oracle Database 12c Enterprise Edition...

SQL>
```

22. Quit the SQL*Plus session.

```
SQL> exit
Disconnected from Oracle Database 12c...
[oracle@c01n01 ~]$
```

Congratulations! You have now used and managed ASM-based password files.

Practices for Lesson 8: Cloud FS New Features

Chapter 8

Practices for Lesson 8: Overview

Practices Overview

In these practices, you will:

- Configure and use HANFS
- Use Cloud FS to store and manage Oracle Database files
- Configure and use Cloud FS auditing
- Implement node-specific file system dependencies

Practice 8-1: Configuring and Using HANFS

Overview

In this practice, you will configure and use High Availability NFS (HANFS). You will also shut down (crash) the node running the HANFS service and watch it migrate to a surviving node.

Assumptions

This practice relies on the configuration performed in Practice 2-1.

Tasks

1. Establish a terminal session connected to `c01n01` using the `root` OS user.

```
$ ssh root@c01n01
root@c01n01's password: <oracle>
[root@c01n01 ~]$
```

2. Configure the environment using the `oraenv` script. Enter `+ASM1` when you are prompted for an ORACLE_SID value.

```
[root@c01n01 ~]$ . oraenv
ORACLE_SID = [root] ? +ASM1
The Oracle base has been set to /u01/app/grid
[root@c01n01 ~]$
```

3. HANFS requires a running NFS service on each node that can host the HANFS services. Use the following command to ensure that NFS is running on `c01n01`.

```
[root@c01n01 ~]# service nfs restart
Shutting down NFS daemon:                                  [  OK  ]
Shutting down NFS mountd:                                  [  OK  ]
Shutting down NFS quotas:                                  [  OK  ]
Starting NFS services:                                     [  OK  ]
Starting NFS quotas:                                       [  OK  ]
Starting NFS mountd:                                       [  OK  ]
Stopping RPC idmapd:                                       [  OK  ]
Starting RPC idmapd:                                       [  OK  ]
Starting NFS daemon:                                       [  OK  ]
[root@c01n01 ~]#
```

4. Ensure that NFS is also running on `c01n02`.

```
[root@c01n01 ~]# ssh c01n02 service nfs restart
Shutting down NFS daemon: [  OK  ]
Shutting down NFS mountd: [  OK  ]
Shutting down NFS quotas: [  OK  ]
Starting NFS services:  [  OK  ]
Starting NFS quotas: [  OK  ]
Starting NFS mountd: [  OK  ]
Stopping RPC idmapd: [  OK  ]
```

```
Starting RPC idmapd: [  OK  ]
Starting NFS daemon: [  OK  ]
[root@c01n01 ~]#
```

5. Soon, you will create a new Cloud FS file system. In preparation for the new file system, create a mount point directory on `c01n01`.

```
[root@c01n01 ~]# mkdir -p /mnt/acfsmounts/acfs1
[root@c01n01 ~]#
```

6. Create the mount point directory on `c01n02`.

```
[root@c01n01 ~]# ssh c01n02 mkdir -p /mnt/acfsmounts/acfs1
[root@c01n01 ~]#
```

7. Become the `grid` OS user.

```
[root@c01n01 ~]# su grid
[grid@c01n01 root]$
```

8. Start the ASM Command Utility (ASMCMD).

```
[grid@c01n01 root]$ asmcmd
ASMCMD>
```

9. Modify the `DATA` disk group to enable all the new ASM Dynamic Volume (ADVM) features included in release 12.1.

```
ASMCMD> setattr -G DATA compatible.advm 12.1.0.0.0
ASMCMD>
```

10. Create a new volume. Place the volume in the `DATA` disk group and set the volume size to 300 MB. Name the volume `VOL1`.

```
ASMCMD> volcreate -G DATA -s 300m VOL1
ASMCMD>
```

11. Examine the newly created volume and take note of the volume device associated with it. Note that your volume device will be different from that shown below (`/dev/asm/vol1-999`). Make note of the different volume device because you will require it numerous times in the following steps.

```
ASMCMD> volinfo -G DATA VOL1
Diskgroup Name: DATA

         Volume Name: VOL1
         Volume Device: /dev/asm/vol1-999
         State: ENABLED
         Size (MB): 320
         Resize Unit (MB): 32
         Redundancy: MIRROR
         Stripe Columns: 4
         Stripe Width (K): 128
         Usage:
         Mountpath:
```

```
ASMCMD>
```

12. Exit ASMCMD.

```
ASMCMD> exit
[grid@c01n01 root]$
```

13. Exit your `grid` OS session.

```
[grid@c01n01 root]$ exit
exit
[root@c01n01 ~]#
```

14. Make an `acfs` file system on the newly created volume. Use the volume device that you identified in step 11.

```
[root@c01n01 ~]# mkfs -t acfs /dev/asm/vol1-999
mkfs.acfs: version                   = 12.1.0.1.0
mkfs.acfs: on-disk version           = 39.0
mkfs.acfs: volume                    = /dev/asm/vol1-999
mkfs.acfs: volume size               = 335544320
mkfs.acfs: Format complete.
[root@c01n01 ~]#
```

15. Create a new Cloud FS file system resource using the volume device that you identified in step 11 along with the mount points that you created at the beginning of the practice.

```
[root@c01n01 ~]# srvctl add filesystem -m /mnt/acfsmounts/acfs1
-d /dev/asm/vol1-999
[root@c01n01 ~]#
```

16. Start the new Cloud FS file system.

```
[root@c01n01 ~]# srvctl start filesystem -d /dev/asm/vol1-999
[root@c01n01 ~]#
```

17. Confirm that the new file system is mounted on `c01n01` and `c01n02`.

```
[root@c01n01 ~]# srvctl status filesystem
ACFS file system /mnt/acfsmounts/acfs1 is mounted on nodes
c01n01,c01n02
[root@c01n01 ~]#
```

18. Create a small text file inside the new Cloud FS file system.

```
[root@c01n01 ~]# echo "Test File on ACFS" >
/mnt/acfsmounts/acfs1/testfile.txt
[root@c01n01 ~]#
```

19. Access the file from another node to demonstrate that the Cloud FS file system is working correctly.

```
[root@c01n01 ~]# ssh c01n02 cat
/mnt/acfsmounts/acfs1/testfile.txt
Test File on ACFS
[root@c01n01 ~]#
```

20. Modify the access privileges for your new file to enable access by any user.

```
[root@c01n01 ~]# chmod 777 /mnt/acfsmounts/acfs1/testfile.txt
[root@c01n01 ~]#
```

At this point, you have created and tested a new Cloud FS file system. In the next part of this practice, you will publish it using HANFS.

21. Your environment is preconfigured with a hostname and IP address that you will use to configure HANFS. Examine the IP address associated with the hostname `c01vip`.

```
[root@c01n01 ~]# nslookup c01vip.example.com
Server:         192.0.2.1
Address:        192.0.2.1#53

Name:   c01vip.example.com
Address: 192.0.2.15

[root@c01n01 ~]#
```

22. Create a new `havip` cluster resource using the hostname `c01vip`. Use `havip1` as the identifier for the new `havip` resource.

```
[root@c01n01 ~]# srvctl add havip -address c01vip -id havip1
[root@c01n01 ~]#
```

23. Create a new `exportfs` cluster resource. The `exportfs` resource publishes the specified file system using HANFS. Following is a summary of the options used:
    - `-id havip1`: Specifies the `havip` resource used to export the file system
    - `-path /mnt/acfsmounts/acfs1`: Specifies the file system being exported
    - `-name export1`: Specifies the name used to identify the `exportfs` resource
    - `-options rw`: Specifies the NFS options for the exported file system
    - `-clients *.export.com`: Specifies the clients permitted to access the exported file system

```
[root@c01n01 ~]# srvctl add exportfs -id havip1 -path
/mnt/acfsmounts/acfs1 -name export1 -options rw -clients
*.example.com
[root@c01n01 ~]#
```

24. Start the newly created `exportfs` resource.

```
[root@c01n01 ~]# srvctl start exportfs -name export1
[root@c01n01 ~]#
```

25. Confirm that the `exportfs` resource is running. Note the server that the file system is exported on (`c01n02` in the example below).

```
[root@c01n01 ~]# srvctl status exportfs
export file system export1 is enabled
export file system export1 is exported on node c01n02
[root@c01n01 ~]#
```

26. Confirm that the `havip` resource is also running. The `havip` is started whenever an associated `exportfs` resource is started. Note that the `havip` resource is located on the same server as the `exportfs` resource.

```
[root@c01n01 ~]# srvctl status havip
HAVIP ora.havip1.havip is enabled
HAVIP ora.havip1.havip is running on nodes c01n02
[root@c01n01 ~]#
```

27. Establish another terminal session connected to `c01n04` as the `root` OS user. In the remainder of this practice, you will use `c01n04` as an NFS client.

    Note that you may see additional messages relating to server identities. Answer `yes` if you are prompted to acknowledge server authenticity.

```
$ ssh root@c01n04
root@c01n04's password: <oracle>
[root@c01n04 ~]#
```

28. Create an empty directory to use as an NFS mount point.

```
[root@c01n04 ~]# mkdir -p /mnt/hanfs1
[root@c01n04 ~]#
```

29. Mount the HANFS exported file system.

```
[root@c01n04 ~]# mount c01vip:/mnt/acfsmounts/acfs1 /mnt/hanfs1
[root@c01n04 ~]#
```

30. Execute the `df` command. Examine the output and confirm that the HANFS exported file system is mounted.

```
[root@c01n04 ~]# df
Filesystem           1K-blocks      Used Available Use% Mounted
on
/dev/xvda2            10068136   3907468   5649232  41% /
tmpfs                  4194304         0   4194304   0% /dev/shm
/dev/xvda1              247919     84819    150300  37% /boot
/dev/xvdc1             2063504     37236   1921448   2% /home
/dev/xvdd1            20153140  11650160   7479240  61% /u01
192.0.2.1:/stage      10317888  10277152         0 100% /stage
c01vip:/mnt/acfsmounts/acfs1
                        327680    144384    183296  45%
/mnt/hanfs1
[root@c01n04 ~]#
```

31. Become the `grid` OS user.

```
[root@c01n04 ~]# su - grid
[grid@c01n04 ~]$
```

32. Using your HANFS mount, edit the text file that you created earlier in this practice.

```
[grid@c01n04 ~]$ vi /mnt/hanfs1/testfile.txt
```

33. Add some text to the file and leave the file open. If you are unfamiliar with `vi`, type `o` to add a new line and then type some text.

```
Test File on ACFS
Here is some more text...
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --                                    2,26          All
```

34. Back in your `root` terminal session, stop Clusterware on the server running the HANFS services (that is, the server running the `exportfs` and `havip` resources, which you identified in steps 25 and 26).

```
[root@c01n01 ~]# crsctl stop cluster -n c01n02 -f
CRS-2673: Attempting to stop 'ora.crsd' on 'c01n02'
CRS-2790: Starting shutdown of Cluster Ready Services-managed
resources on 'c01n02'
CRS-2673: Attempting to stop 'ora.havip1.havip' on 'c01n02'
CRS-2673: Attempting to stop 'ora.SCRUBTEST.dg' on 'c01n02'
CRS-2673: Attempting to stop 'ora.LISTENER.lsnr' on 'c01n02'
```

```
CRS-2673: Attempting to stop 'ora.LISTENER_SCAN1.lsnr' on
'c01n02'
CRS-2673: Attempting to stop 'ora.orcl.db' on 'c01n02'
CRS-2677: Stop of 'ora.havip1.havip' on 'c01n02' succeeded
...
CRS-2673: Attempting to stop 'ora.net1.network' on 'c01n02'
CRS-2677: Stop of 'ora.net1.network' on 'c01n02' succeeded
CRS-2792: Shutdown of Cluster Ready Services-managed resources
on 'c01n02' has completed
CRS-2677: Stop of 'ora.crsd' on 'c01n02' succeeded
CRS-2673: Attempting to stop 'ora.ctssd' on 'c01n02'
CRS-2673: Attempting to stop 'ora.evmd' on 'c01n02'
CRS-2673: Attempting to stop 'ora.storage' on 'c01n02'
CRS-2677: Stop of 'ora.storage' on 'c01n02' succeeded
CRS-2673: Attempting to stop 'ora.asm' on 'c01n02'
CRS-2677: Stop of 'ora.ctssd' on 'c01n02' succeeded
CRS-2677: Stop of 'ora.asm' on 'c01n02' succeeded
CRS-2673: Attempting to stop 'ora.cluster_interconnect.haip' on
'c01n02'
CRS-2677: Stop of 'ora.cluster_interconnect.haip' on 'c01n02'
succeeded
CRS-2677: Stop of 'ora.evmd' on 'c01n02' succeeded
CRS-2673: Attempting to stop 'ora.cssd' on 'c01n02'
CRS-2677: Stop of 'ora.cssd' on 'c01n02' succeeded
[root@c01n01 ~]#
```

35. Confirm that the HANFS services have migrated to another server (`c01n01` in the example below).

```
[root@c01n01 ~]# srvctl status exportfs
export file system export1 is enabled
export file system export1 is exported on node c01n01
[root@c01n01 ~]# srvctl status havip
HAVIP ora.havip1.havip is enabled
HAVIP ora.havip1.havip is running on nodes c01n01
[root@c01n01 ~]#
```

36. Back in the NFS client session, save the file and exit `vi` (press Esc, type `:wq`, and press Enter). You may notice a delay while the NFS connection is reestablished.

```
Test File on ACFS
Here is some more text...
~
~
~
~
~
```

```
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:wq
```

37. Examine the file that you just edited to confirm that your changes were saved. Now you have seen HANFS in action.

```
[grid@c01n04 ~]$ cat /mnt/hanfs1/testfile.txt
Test File on ACFS
Here is some more text...
[grid@c01n04 ~]$
```

38. Back in your `root` terminal session, restart Clusterware on the server where you stopped it in step 34.

```
[root@c01n01 ~]# crsctl start cluster -n c01n02
CRS-2672: Attempting to start 'ora.cssdmonitor' on 'c01n02'
CRS-2672: Attempting to start 'ora.evmd' on 'c01n02'
CRS-2676: Start of 'ora.cssdmonitor' on 'c01n02' succeeded
CRS-2672: Attempting to start 'ora.cssd' on 'c01n02'
CRS-2672: Attempting to start 'ora.diskmon' on 'c01n02'
CRS-2676: Start of 'ora.diskmon' on 'c01n02' succeeded
CRS-2676: Start of 'ora.evmd' on 'c01n02' succeeded
CRS-2676: Start of 'ora.cssd' on 'c01n02' succeeded
CRS-2672: Attempting to start 'ora.ctssd' on 'c01n02'
CRS-2672: Attempting to start 'ora.cluster_interconnect.haip' on
'c01n02'
CRS-2676: Start of 'ora.ctssd' on 'c01n02' succeeded
CRS-2676: Start of 'ora.cluster_interconnect.haip' on 'c01n02'
succeeded
CRS-2672: Attempting to start 'ora.asm' on 'c01n02'
CRS-2676: Start of 'ora.asm' on 'c01n02' succeeded
```

```
CRS-2672: Attempting to start 'ora.storage' on 'c01n02'
CRS-2676: Start of 'ora.storage' on 'c01n02' succeeded
CRS-2672: Attempting to start 'ora.crsd' on 'c01n02'
CRS-2676: Start of 'ora.crsd' on 'c01n02' succeeded
[root@c01n01 ~]#
```

39. To ensure that there are no stale connections from earlier in the practice, restart the NFS services on the server where you just restarted Clusterware.

```
[root@c01n01 ~]# ssh c01n02 service nfs restart
Shutting down NFS daemon: [  OK  ]
Shutting down NFS mountd: [  OK  ]
Shutting down NFS quotas: [  OK  ]
Starting NFS services:  [  OK  ]
Starting NFS quotas: [  OK  ]
Starting NFS mountd: [  OK  ]
Stopping RPC idmapd: [  OK  ]
Starting RPC idmapd: [  OK  ]
Starting NFS daemon: [  OK  ]
[root@c01n01 ~]#
```

So far, you have seen how HANFS services are automatically migrated when Clusterware is stopped (or a server fails). However, HANFS services can also be manually relocated, which may be useful when you wish to prepare for a period of planned maintenance for example.

40. Manually relocate the `havip` resource. Specify the server where you just restarted Clusterware as the relocation target (using the `-n` option). Note that the `exportfs` resource is automatically relocated when the `havip` is relocated. If you receive an error message indicating that the relocation target is not online, wait a few seconds and try again.

```
[root@c01n01 ~]# srvctl relocate havip -id havip1 -n c01n02 -f
HAVIP was relocated successfully
[root@c01n01 ~]#
```

41. Confirm that the HANFS services are relocated.

```
[root@c01n01 ~]# srvctl status exportfs
export file system export1 is enabled
export file system export1 is exported on node c01n02
[root@c01n01 ~]# srvctl status havip
HAVIP ora.havip1.havip is enabled
HAVIP ora.havip1.havip is running on nodes c01n02
[root@c01n01 ~]#
```

42. Back in the NFS client session, examine the HANFS mounted file that you have been using throughout this practice. Again, you may notice a delay while the NFS connection is reestablished.

```
[grid@c01n04 ~]$ cat /mnt/hanfs1/testfile.txt
Test File on ACFS
Here is some more text...
[grid@c01n04 ~]$
```

43. Exit the `grid` terminal session on `c01n04` and return to the `root` terminal session.

```
[grid@c01n04 ~]$ exit
logout
[root@c01n04 ~]#
```

44. Unmount the NFS mount on `c01n04`.

```
[root@c01n04 ~]# umount /mnt/hanfs1
[root@c01n04 ~]#
```

45. Back in your `root` terminal session on `c01n01`, stop the HANFS services.

```
[root@c01n01 ~]# srvctl stop exportfs -name export1 -f
[root@c01n01 ~]#
```

46. Stop the Cloud FS file system that you have used throughout this practice.

```
[root@c01n01 ~]# srvctl stop filesystem -d /dev/asm/vol1-999
[root@c01n01 ~]#
```

Congratulations! You have successfully configured and used High Availability NFS (HANFS).

Practice 8-2: Using Cloud FS to Store Oracle Database Files

Overview

In this practice, you will use DBCA to create a new Oracle database that uses Cloud FS to store all its data files, control files, log files, etc. You will then use snapshots, including the new Snaps-of-Snaps feature, to perform a point-in-time backup of the database and quickly revert to the snapshot copy.

Assumptions

This practice relies on the configuration performed in Practice 2-1.

Tasks

1. Establish a terminal session connected to `c01n01` using the `root` OS user.

```
$ ssh root@c01n01
root@c01n01's password: <oracle>
[root@c01n01 ~]$
```

2. Configure the environment by using the `oraenv` script. Enter `+ASM1` when you are prompted for an ORACLE_SID value.

```
[root@c01n01 ~]$ . oraenv
ORACLE_SID = [grid] ? +ASM1
The Oracle base has been set to /u01/app/grid
[root@c01n01 ~]$
```

3. Soon, you will create a new Cloud FS file system to use in this practice. In preparation for the new file system, create a mount point directory on `c01n01`.

```
[root@c01n01 ~]# mkdir -p /mnt/acfsmounts/acfs2
[root@c01n01 ~]#
```

4. Create the mount point directory on `c01n02`.

```
[root@c01n01 ~]# ssh c01n02 mkdir -p /mnt/acfsmounts/acfs2
[root@c01n01 ~]#
```

5. Become the `grid` OS user.

```
[root@c01n01 ~]# su grid
[grid@c01n01 root]$
```

6. Start the ASM Command Utility (ASMCMD).

```
[grid@c01n01 root]$ asmcmd
ASMCMD>
```

7. Modify the `DATA` disk group to ensure that all the new ASM Dynamic Volume (ADVM) features included in release 12.1 are enabled.

```
ASMCMD> setattr -G DATA compatible.advm 12.1.0.0.0
ASMCMD>
```

8. Create a new volume. Place the volume in the DATA disk group and set the volume size to 5 GB. Name the volume VOL2.

```
ASMCMD> volcreate -G DATA -s 5g --column 1 VOL2
ASMCMD>
```

9. Examine the newly created volume and take note of the volume device associated with it. Note that your volume device will be different from that shown below (/dev/asm/vol2-*999*). Make note of the different volume device because you will require it numerous times in the following steps.

```
ASMCMD> volinfo -G DATA VOL2
Diskgroup Name: DATA

         Volume Name: VOL2
         Volume Device: /dev/asm/vol2-999
         State: ENABLED
         Size (MB): 5120
         Resize Unit (MB): 8
         Redundancy: MIRROR
         Stripe Columns: 1
         Stripe Width (K): 8192
         Usage:
         Mountpath:

ASMCMD>
```

10. Exit ASMCMD.

```
ASMCMD> exit
[grid@c01n01 root]$
```

11. Exit your grid OS session.

```
[grid@c01n01 root]$ exit
exit
[root@c01n01 ~]#
```

12. Make an acfs file system on the newly created volume. Use the volume device that you identified in step 9.

```
[root@c01n01 ~]# mkfs -t acfs /dev/asm/vol2-999
mkfs.acfs: version                   = 12.1.0.1.0
mkfs.acfs: on-disk version           = 39.0
mkfs.acfs: volume                    = /dev/asm/vol2-999
mkfs.acfs: volume size               = 5368709120
mkfs.acfs: Format complete.
[root@c01n01 ~]#
```

13. Create a new Cloud FS file system resource using the volume device that you identified in step 9 along with the mount points that you created at the beginning of the practice.

```
[root@c01n01 ~]# srvctl add filesystem -m /mnt/acfsmounts/acfs2
-d /dev/asm/vol2-999
[root@c01n01 ~]#
```

14. Start the new Cloud FS file system.

```
[root@c01n01 ~]# srvctl start filesystem -d /dev/asm/vol2-999
[root@c01n01 ~]#
```

15. Confirm that the new file system is mounted on `c01n01` and `c01n02`.

```
[root@c01n01 ~]# srvctl status filesystem -d /dev/asm/vol2-999
ACFS file system /mnt/acfsmounts/acfs2 is mounted on nodes
c01n01,c01n02
[root@c01n01 ~]#
```

16. Modify the access privileges for your new file system to enable access by any user.

```
[root@c01n01 ~]# chmod 777 /mnt/acfsmounts/acfs2
[root@c01n01 ~]#
```

17. Modify the newly created Cloud FS file system to enable full control by members of the `dba` OS group (Oracle Database Administrators).

```
[root@c01n01 ~]# crsctl modify resource ora.data.vol2.acfs -attr
"ACL='owner:root:rwx,pgrp:dba:rwx,other::r--'"
[root@c01n01 ~]#
```

18. Establish another terminal session connected to `c01n01` using the `oracle` OS user.

```
$ ssh oracle@c01n01
oracle@c01n01's password: <oracle>
[oracle@c01n01 ~]$
```

19. Configure the environment using the `oraenv` script. Enter `orcl` when you are prompted for an ORACLE_SID value.

```
[oracle@c01n01 ~]$ . oraenv
ORACLE_SID = [oracle] ? orcl
The Oracle base has been set to /u01/app/oracle
[oracle@c01n01 ~]$
```

20. Stop the `orcl` database. This is required to ensure that you have access to the required server resources (particularly memory and CPU) later in this practice.

```
[oracle@c01n01 ~]$ srvctl stop database -d orcl
[oracle@c01n01 ~]$
```

21. Create a new database using `dbca` and the response file provided at `/stage/labs/8_2/dbca.rsp`.

```
[oracle@c01n01 ~]$ dbca -silent -createDatabase -responseFile
/stage/labs/8_2/dbca.rsp
```

22. Wait while the database is created. The database creation process should take approximately 25 minutes to complete.

```
[oracle@c01n01 ~]$ dbca -silent -createDatabase -responseFile
/stage/labs/8_2/dbca.rsp
Copying database files
1% complete
3% complete
9% complete
15% complete
21% complete
27% complete
30% complete
Creating and starting Oracle instance
32% complete
36% complete
40% complete
44% complete
48% complete
49% complete
52% complete
Creating cluster database views
54% complete
72% complete
Completing Database Creation
75% complete
78% complete
87% complete
96% complete
100% complete
Look at the log file
"/u01/app/oracle/cfgtoollogs/dbca/cfsdb/cfsdb.log" for further
details.
[oracle@c01n01 ~]$
```

23. Confirm that the newly created database (named `cfsdb`) is running.

```
[oracle@c01n01 ~]$ srvctl status database -d cfsdb
Instance cfsdb_1 is running on node c01n02
Instance cfsdb_2 is running on node c01n01
[oracle@c01n01 ~]$
```

24. Using SQL*Plus, connect to the newly created database as the `system` user.

```
[oracle@c01n01 ~]$ sqlplus system/oracle_4U@cfsdb

SQL*Plus: Release 12.1.0.1.0 Production...

SQL>
```

25. Examine the location of the online redo log files. Notice that they are located on the Cloud FS file system that you created at the beginning of the practice.

```
SQL> select member from v$logfile;

MEMBER
------------------------------------------------------------------------
/mnt/acfsmounts/acfs2/oradata/cfsdb/redo02.log
/mnt/acfsmounts/acfs2/oradata/cfsdb/redo01.log
/mnt/acfsmounts/acfs2/oradata/cfsdb/redo03.log
/mnt/acfsmounts/acfs2/oradata/cfsdb/redo04.log

SQL>
```

26. Examine the location of the database control files. Notice that they are located on the Cloud FS file system that you created at the beginning of the practice.

```
SQL> select name from v$controlfile;

NAME
--------------------------------------------------------------------------------
/mnt/acfsmounts/acfs2/oradata/CFSDB/controlfile/o1_mf_8cxf1kjc_.ctl
/mnt/acfsmounts/acfs2/fast_recovery_area/CFSDB/controlfile/o1_mf_8cxf1lz5_.ctl

SQL>
```

27. Examine the location of the database data files. Notice that they too are located on the Cloud FS file system that you created at the beginning of the practice.

```
SQL> select file_name from dba_data_files;

FILE_NAME
--------------------------------------------------------------------------------
/mnt/acfsmounts/acfs2/oradata/CFSDB/datafile/o1_mf_system_8cxdx31b_.dbf
/mnt/acfsmounts/acfs2/oradata/CFSDB/datafile/o1_mf_sysaux_8cxdt3jt_.dbf
/mnt/acfsmounts/acfs2/oradata/CFSDB/datafile/o1_mf_undotbs1_8cxf0g2o_.dbf
/mnt/acfsmounts/acfs2/oradata/CFSDB/datafile/o1_mf_users_8cxf0doc_.dbf
/mnt/acfsmounts/acfs2/oradata/CFSDB/datafile/o1_mf_undotbs2_8cxfpsz3_.dbf
/mnt/acfsmounts/acfs2/oradata/CFSDB/datafile/o1_mf_example_8cxf3ffx_.dbf

6 rows selected.

SQL>
```

At this point, you have an entire RAC database that is located inside Cloud FS.

28. Query the `sh.sales` table to verify its existence. Leave your SQL*Plus session running.

```
SQL> select count(*) from sh.sales;

  COUNT(*)
----------
    918843

SQL>
```

29. Back in your `root` terminal session, create a snapshot of the Cloud FS file system that contains your newly created database files.

```
[root@c01n01 ~]# acfsutil snap create dbsnap
/mnt/acfsmounts/acfs2
acfsutil snap create: Snapshot operation is complete.
[root@c01n01 ~]#
```

30. Examine the snapshots in your Cloud FS file system. Notice that the snapshot you created in the previous step (named `dbsnap`) is a read-only snapshot.

```
[root@c01n01 ~]# acfsutil snap info /mnt/acfsmounts/acfs2
snapshot name:               dbsnap
RO snapshot or RW snapshot:  RO
parent name:                 /mnt/acfsmounts/acfs2
snapshot creation time:      Wed Dec  5 03:01:56 2012

    number of snapshots:  1
    snapshot space usage: 17874944
[root@c01n01 ~]#
```

31. Back in your SQL*Plus session, drop the `sh` user including all of its schema objects.

```
SQL> drop user sh cascade;

User dropped.

SQL>
```

32. Requery the `sh.sales` table and verify that it no longer exists.

```
SQL> select count(*) from sh.sales;
select count(*) from sh.sales
                        *
ERROR at line 1:
ORA-00942: table or view does not exist


SQL>
```

33. Exit SQL*Plus.

```
SQL> exit
Disconnected from Oracle Database 12c...
[oracle@c01n01 ~]$
```

In the next part of this practice, you will use the snapshot capabilities inside Cloud FS to recover the database. You will use the snapshot that you created in step 29 to restore the `sh` user and schema objects.

34. Change directory to your Cloud FS file system mount directory.

```
[oracle@c01n01 ~]$ cd /mnt/acfsmounts/acfs2
[oracle@c01n01 acfs2]$
```

35. List the database files inside Cloud FS. These are the current database files (after dropping the `sh` user).

```
[oracle@c01n01 acfs2]$ ls -l oradata/CFSDB/datafile
total 2240700
-rw-r----- 1 oracle asmadmin 338829312 Dec  5 02:43
o1_mf_example_8cxf3ffx_.dbf
-rw-r----- 1 oracle asmadmin 817897472 Dec  5 02:53
o1_mf_sysaux_8cxdt3jt_.dbf
-rw-r----- 1 oracle asmadmin 817897472 Dec  5 02:46
o1_mf_system_8cxdx31b_.dbf
-rw-r----- 1 oracle asmadmin  99622912 Dec  5 02:59
o1_mf_temp_8cxf28mj_.tmp
-rw-r----- 1 oracle asmadmin 183508992 Dec  5 02:55
o1_mf_undotbs1_8cxf0g2o_.dbf
-rw-r----- 1 oracle asmadmin  26222592 Dec  5 02:52
o1_mf_undotbs2_8cxfpsz3_.dbf
-rw-r----- 1 oracle asmadmin   5251072 Dec  5 02:40
o1_mf_users_8cxf0doc_.dbf
[oracle@c01n01 acfs2]$
```

36. List the database files inside the Cloud FS snapshot that you captured in step 29. Remember that these files were captured before you dropped the `sh` user.

```
[oracle@c01n01 acfs2]$ ls -l
.ACFS/snaps/dbsnap/oradata/CFSDB/datafile
total 2240700
-rw-r----- 1 oracle asmadmin 338829312 Dec  5 02:43
o1_mf_example_8cxf3ffx_.dbf
-rw-r----- 1 oracle asmadmin 817897472 Dec  5 02:53
o1_mf_sysaux_8cxdt3jt_.dbf
-rw-r----- 1 oracle asmadmin 817897472 Dec  5 02:46
o1_mf_system_8cxdx31b_.dbf
-rw-r----- 1 oracle asmadmin  99622912 Dec  5 02:59
o1_mf_temp_8cxf28mj_.tmp
-rw-r----- 1 oracle asmadmin 183508992 Dec  5 02:55
o1_mf_undotbs1_8cxf0g2o_.dbf
-rw-r----- 1 oracle asmadmin  26222592 Dec  5 02:52
o1_mf_undotbs2_8cxfpsz3_.dbf
-rw-r----- 1 oracle asmadmin   5251072 Dec  5 02:40
o1_mf_users_8cxf0doc_.dbf
[oracle@c01n01 acfs2]$
```

37. Stop the `cfsdb` database instances.

```
[oracle@c01n01 acfs2]$ srvctl stop database -d cfsdb
[oracle@c01n01 acfs2]$
```

38. Remove the current copy of the database files and fast recovery area.

```
[oracle@c01n01 acfs2]$ rm -rf oradata fast_recovery_area
[oracle@c01n01 acfs2]$
```

At this point, you could simply copy the snapshot copies of all the database files into the original locations and restart the database in order to revert the database back to the point at which the snapshot was taken. However, in the remainder of this practice, you will see how you can use the new Snaps-of-Snaps feature to provide a more flexible solution.

39. Back in your `root` terminal session, create a new read-write snapshot (called `writable`) based on your first snapshot (`dbsnap`).

```
[root@c01n01 ~]# acfsutil snap create -w -p dbsnap writable
/mnt/acfsmounts/acfs2
acfsutil snap create: Snapshot operation is complete.
[root@c01n01 ~]#
```

40. Examine the snapshots in your Cloud FS file system to confirm the existence of the writable snapshot that you created in the previous step.

```
[root@c01n01 ~]# acfsutil snap info /mnt/acfsmounts/acfs2
snapshot name:               dbsnap
RO snapshot or RW snapshot:  RO
parent name:                 /mnt/acfsmounts/acfs2
snapshot creation time:      Wed Dec  5 03:01:56 2012

snapshot name:               writable
RO snapshot or RW snapshot:  RW
parent name:                 dbsnap
snapshot creation time:      Wed Dec  5 03:09:32 2012

    number of snapshots:  2
    snapshot space usage: 2544349184
[root@c01n01 ~]#
```

41. In your `oracle` terminal session, use the following command to create a link that exposes the writable snapshot database files in their original locations.

```
[oracle@c01n01 acfs2]$ ln -s .ACFS/snaps/writable/oradata .
[oracle@c01n01 acfs2]$
```

42. Use the following command to create a link that exposes the writable snapshot fast recovery area files in their original locations.

```
[oracle@c01n01 acfs2]$ ln -s
.ACFS/snaps/writable/fast_recovery_area .
[oracle@c01n01 acfs2]$
```

43. Reexamine the database files. The files reside inside the writable snapshot and are exposed in their original locations using the links that you just created.

```
[oracle@c01n01 acfs2]$ ls -l oradata/CFSDB/datafile
total 2240700
-rw-r----- 1 oracle asmadmin 338829312 Dec  5 02:43
o1_mf_example_8cxf3ffx_.dbf
-rw-r----- 1 oracle asmadmin 817897472 Dec  5 02:53
o1_mf_sysaux_8cxdt3jt_.dbf
-rw-r----- 1 oracle asmadmin 817897472 Dec  5 02:46
o1_mf_system_8cxdx31b_.dbf
-rw-r----- 1 oracle asmadmin  99622912 Dec  5 02:59
o1_mf_temp_8cxf28mj_.tmp
-rw-r----- 1 oracle asmadmin 183508992 Dec  5 02:55
o1_mf_undotbs1_8cxf0g2o_.dbf
-rw-r----- 1 oracle asmadmin  26222592 Dec  5 02:52
o1_mf_undotbs2_8cxfpsz3_.dbf
-rw-r----- 1 oracle asmadmin   5251072 Dec  5 02:40
o1_mf_users_8cxf0doc_.dbf
[oracle@c01n01 acfs2]$
```

44. Restart the `cfsdb` database.

```
[oracle@c01n01 acfs2]$ srvctl start database -d cfsdb
[oracle@c01n01 acfs2]$
```

45. Using SQL*Plus, reconnect to the `cfsdb` database as the `system` user.

```
[oracle@c01n01 acfs2]$ sqlplus system/oracle_4U@cfsdb

SQL*Plus: Release 12.1.0.1.0 Production...

SQL>
```

46. Query the `sh.sales` table. Notice that the `sh` user and schema objects are again present in the database.

```
SQL> select count(*) from sh.sales;

  COUNT(*)
----------
    918843

SQL>
```

At this point, the database is available for full read and write operation. If you ever wanted to revert the database back to this point again, you could easily create another snapshot, based on the original `dbsnap` snapshot. This demonstrates how you can use Cloud FS snapshots to quickly and easily restore databases to a prior state.

47. Exit your SQL*Plus session.

```
SQL> exit
Disconnected from Oracle Database 12c...
[oracle@c01n01 acfs2]$
```

48. Change back to the `oracle` user home directory.

```
[oracle@c01n01 acfs2]$ cd
[oracle@c01n01 ~]$
```

49. Stop the `cfsdb` database.

```
[oracle@c01n01 ~]$ srvctl stop database -d cfsdb
[oracle@c01n01 ~]$
```

50. Back in your `root` terminal session, stop the Cloud FS file system that you have been using throughout this practice.

```
[root@c01n01 ~]$ srvctl stop filesystem -d /dev/asm/vol2-999
[root@c01n01 ~]$
```

Congratulations! You have successfully created a new Oracle database inside Cloud FS and you have used snapshots, including the new Snaps-of-Snaps feature, to perform a point-in-time backup of the database and quickly revert to the snapshot copy.

Practice 8-3: Configuring and Using Cloud FS Auditing

Overview

In this practice, you will go through the process of configuring Cloud FS auditing. After configuration, you will interact with Cloud FS to generate audit records. Finally, you will also exercise the audit trail management procedure.

Assumptions

This practice relies on the configuration performed in Practice 2-1. It also uses the file system created in Practice 8-1.

Tasks

1. Establish a terminal session connected to `c01n01` using the `root` OS user.

```
$ ssh root@c01n01
root@c01n01's password: <oracle>
[root@c01n01 ~]$
```

2. Configure the environment by using the `oraenv` script. Enter `+ASM1` when you are prompted for an ORACLE_SID value.

```
[root@c01n01 ~]$ . oraenv
ORACLE_SID = [grid] ? +ASM1
The Oracle base has been set to /u01/app/grid
[root@c01n01 ~]$
```

3. Restart the file system that you created in Practice 8-1 and confirm that the file system is mounted on `c01n01` and `c01n02`.

```
[root@c01n01 ~]# srvctl start filesystem -d /dev/asm/vol1-999
[root@c01n01 ~]# srvctl status filesystem -d /dev/asm/vol1-999
ACFS file system /mnt/acfsmounts/acfs1 is mounted on nodes
c01n01,c01n02
[root@c01n01 ~]#
```

4. Create a new directory inside the Cloud FS file system. Later in this practice, you will configure a security realm for the contents of this directory.

```
[root@c01n01 ~]# mkdir /mnt/acfsmounts/acfs1/Prac8-3
[root@c01n01 ~]#
```

5. Modify the permissions on the directory that you created in the previous step so that all users have full access to the directory.

```
[root@c01n01 ~]# chmod 777 /mnt/acfsmounts/acfs1/Prac8-3
[root@c01n01 ~]#
```

6. To configure the Cloud FS security and auditing features, you must designate specific OS users and groups, which are granted various administrative privileges. Examine the script at `/stage/labs/8_3/users.sh`. The script contains commands that create three OS groups. In turn, a corresponding OS user account is also created and associated with each of the three OS groups. The purpose of these groups and user accounts will be described later in the practice when they are being used.

```
[root@c01n01 ~]# cat /stage/labs/8_3/users.sh
groupadd -g 9997 secadmin
useradd -g secadmin -m -u 9997 secadmin

groupadd -g 9999 auditmgr
groupadd -g 9998 auditor
useradd -g auditmgr -m -u 9999 auditmgr
useradd -g auditor -m -u 9998 auditor
[root@c01n01 ~]#
```

7. Execute `/stage/labs/8_3/users.sh` on `c01n01`.

```
[root@c01n01 ~]# /stage/labs/8_3/users.sh
[root@c01n01 ~]#
```

8. Examine the user accounts to confirm that they are created and that each account is associated with the corresponding OS group.

```
[root@c01n01 ~]# id secadmin
uid=9997(secadmin) gid=9997(secadmin) groups=9997(secadmin)
[root@c01n01 ~]# id auditmgr
uid=9999(auditmgr) gid=9999(auditmgr) groups=9999(auditmgr)
[root@c01n01 ~]# id auditor
uid=9998(auditor) gid=9998(auditor) groups=9998(auditor)
[root@c01n01 ~]#
```

9. The user and group definitions that you just created are required on every cluster node where the Cloud FS file system is mounted. Execute `/stage/labs/8_3/users.sh` on `c01n02`.

```
[root@c01n01 ~]# ssh c01n02 /stage/labs/8_3/users.sh
[root@c01n01 ~]#
```

10. Initialize Cloud FS security. Specify the `secadmin` user and `secadmin` group to receive the privileges required to administer Cloud FS security. Enter `oracle_4U` when you are prompted to enter a security administrator password. Note that this password is required every time a security administration operation is performed.

```
[root@c01n01 ~]# acfsutil sec init -u secadmin -g secadmin
Password for new ACFS Security administrator: <oracle_4U>
Re-enter password for new ACFS Security administrator:
<oracle_4U>
acfsutil sec init: Security wallet created.
[root@c01n01 ~]#
```

11. Assume the role of Cloud FS security administrator.

```
[root@c01n01 ~]# su secadmin
[secadmin@c01n01 root]$
```

12. Prepare the Cloud FS file system (mounted at `/mnt/acfsmounts/acfs1`) for Cloud FS security.

```
[secadmin@c01n01 root]$ acfsutil sec prepare -m
/mnt/acfsmounts/acfs1
ACFS Security administrator password: <oracle_4U>
System realm 'SYSTEM_SecurityMetadata' created.
System realm 'SYSTEM_Logs' created.
System realm 'SYSTEM_BackupOperators' created.
[secadmin@c01n01 root]$
```

13. Create a security realm. Specify `myrealm1` as the realm name. Include the following options:
    - `-m /mnt/acfsmounts/acfs1`: Specifies the file system
    - `-e off`: Specifies that encryption is off for the realm
    - `-o enable`: Enables security for the realm

```
[secadmin@c01n01 root]$ acfsutil sec realm create myrealm1 -m
/mnt/acfsmounts/acfs1 -e off -o enable
ACFS Security administrator password: <oracle_4U>
[secadmin@c01n01 root]$
```

14. Add the `root` user and your practice directory (created in step 4) to the realm that you just created.

```
[secadmin@c01n01 root]$ acfsutil sec realm add myrealm1 -m
/mnt/acfsmounts/acfs1 -u root -f -r /mnt/acfsmounts/acfs1/Prac8-
3
ACFS Security administrator password: <oracle_4U>
[secadmin@c01n01 root]$
```

15. Query the newly created realm. Confirm that the realm is enabled and that it includes the `root` user.

```
[secadmin@c01n01 root]$ acfsutil sec info -m
/mnt/acfsmounts/acfs1 -n myrealm1
ACFS Security administrator password: <oracle_4U>
Realm status: ENABLED

Users present in realm 'myrealm1' are as follows :
        root

Groups present in realm 'myrealm1' are as follows :

Filters present in realm 'myrealm1' are as follows :

Encryption status : OFF

Realm description : ''
[secadmin@c01n01 root]$
```

16. Exit the `secadmin` terminal session.

```
[secadmin@c01n01 root]$ exit
exit
[root@c01n01 ~]#
```

At this point, you have a Cloud FS file system configured with Cloud FS security. In the next part of this practice, you will configure Cloud FS auditing. This involves three tasks:

- Initialize auditing across the system.
- Enable auditing for the file system.
- Enable auditing of command rules for files in a security realm.

17. As the system administrator, initialize Cloud FS auditing. This is a one-time configuration step where you specify the OS groups that are associated with the audit manager and auditor roles. By splitting various administrative tasks between these two roles, separation of duties is enforced so that no single administrator has all the privileges required to tamper with an audit trail.

```
[root@c01n01 ~]# acfsutil audit init -M auditmgr -A auditor
[root@c01n01 ~]#
```

18. Confirm the role assignments that you made in the previous step. Note that once they are made, these assignments cannot be changed. That is why it is recommended that you create dedicated groups for the audit manager and auditor roles.

```
[root@c01n01 ~]# acfsutil audit info
Audit manager OS group : 'auditmgr'
Auditor OS group       : 'auditor'
[root@c01n01 ~]#
```

In the remainder of this practice, you will perform various tasks associated with the audit manager and auditor roles. To achieve this, you will be required to switch between the `auditmgr` and `auditor` user accounts.

19. Become the audit manager.

```
[root@c01n01 ~]# su auditmgr
[auditmgr@c01n01 root]$
```

20. Enable Cloud FS security auditing on the file system. After this command is executed, auditing commences on the file system.

```
[auditmgr@c01n01 root]$ acfsutil audit enable -m
/mnt/acfsmounts/acfs1 -s sec
[auditmgr@c01n01 root]$
```

21. Confirm that security auditing is enabled for the file system. Note that it is also possible to enable auditing for encryption; however, you will not exercise this option in this practice.

```
[auditmgr@c01n01 root]$ acfsutil audit info -m
/mnt/acfsmounts/acfs1
Auditing information for mount point '/mnt/acfsmounts/acfs1':
Maximum Audit trail size  : 10 MB
Archive file      : 'NOT PRESENT'
Audit sources:
         Security   : 'ENABLED'
         Encryption : 'DISABLED'
[auditmgr@c01n01 root]$
```

22. Exit the audit manager terminal session.

```
[auditmgr@c01n01 root]$ exit
exit
[root@c01n01 ~]#
```

23. Become the security administrator.

```
[root@c01n01 ~]# su secadmin
[secadmin@c01n01 root]$
```

24. In addition to the core auditing capabilities that you enabled in step 20, you can enable auditing of command rules for files in a security realm. Execute the following command to enable auditing of command rules for the `myrealm1` realm that you created earlier in the practice. In this case, you will audit realm authorizations (using the `-a` option) and realm violations (using the `-v` option).

```
[secadmin@c01n01 root]$ acfsutil sec realm audit enable myrealm1
-m /mnt/acfsmounts/acfs1 -a -v
ACFS Security administrator password: <oracle_4U>
[secadmin@c01n01 root]$
```

25. Exit the security administrator terminal session.

```
[secadmin@c01n01 root]$ exit
exit
[root@c01n01 ~]#
```

At this point, you have gone through the process of configuring Cloud FS auditing. In a real production environment, you may have multiple file systems and security realms, in which case you would need to repeat some of the tasks that you have just performed.

Next, you will perform an action that will generate some audit records. Then, in the final part of this practice, you will perform the tasks required to manage the Cloud FS audit trail.

26. Change directory to the root of your Cloud FS file system.

```
[root@c01n01 ~]# cd /mnt/acfsmounts/acfs1
[root@c01n01 acfs1]#
```

27. As mentioned earlier, Cloud FS auditing implements a separation of duties policy so that specific privileges are granted to different administrative roles. As part of this arrangement, access to the files that make up the audit trail is controlled, and even the system administrator (`root`) cannot access them. Confirm this by attempting to access the directory that houses the audit files.

```
[root@c01n01 acfs1]# ls -l .Security/audit
ls: cannot open directory .Security/audit: Permission denied
[root@c01n01 acfs1]#
```

28. As the auditor, examine the audit directory. Note the size of the current audit log.

```
[root@c01n01 acfs1]# su auditor -c "ls -l .Security/audit"
total 4
----rw-r-- 1 root auditmgr 331 Dec 10 04:42 audit-c01n01-
1632357021.log
[root@c01n01 acfs1]#
```

29. Use the following command to create a new file inside your practice directory. This action will generate a series of audit records that you will examine later.

```
[root@c01n01 acfs1]# cat testfile.txt > Prac8-3/audittest.txt
[root@c01n01 acfs1]#
```

30. As the auditor, reexamine the audit directory. Note that the action that you performed in the previous step has caused the current audit log file to grow.

```
[root@c01n01 acfs1]# su auditor -c "ls -l .Security/audit"
total 4
----rw-r-- 1 root auditmgr 1276 Dec 10 04:45 audit-c01n01-
1632357021.log
[root@c01n01 acfs1]#
```

Active audit files should not be interrogated because this could interrupt auditing or result in the loss of auditing data. To examine the records in an audit file, it must first be archived. This occurs automatically when the audit file size reaches 10 MB, or the audit manager can manually archive the audit trail at any time.

31. As the audit manager, archive the audit trail.

```
[root@c01n01 acfs1]# su auditmgr -c "acfsutil audit archive -m
/mnt/acfsmounts/acfs1"
acfsutil audit archive: ACFS-10356: waiting for the operation to
complete...
[root@c01n01 acfs1]#
```

32. Become the auditor.

```
[root@c01n01 acfs1]# su auditor
[auditor@c01n01 acfs1]$
```

33. Examine the audit directory. Notice that there are now three files. The file with the `.bak` extension is the archived audit log file, and the `.xml` file is a representation of the audit log that is ready to be consumed by Oracle Audit Vault if it is configured in the environment. The `.log` file is the next (current) audit log.

```
[auditor@c01n01 acfs1]$ ls -l .Security/audit
total 12
----rw-r-- 1 root     auditmgr 337  Dec 10 04:47 audit-c01n01-
1632357021.log
----rw-r-- 1 root     auditmgr 1276 Dec 10 04:45 audit-c01n01-
1632357021.log.bak
-rw-rw-r-- 1 auditmgr auditmgr 2933 Dec 10 04:47 audit-c01n01-
1632357021.xml
[auditor@c01n01 acfs1]$
```

34. Examine the archived audit log file. Your file name will differ from the example shown below but it will end with the `.bak` extension. The first audit record (containing `Event:ACFS_AUDIT_ENABLE`) was generated when you enabled auditing for the file system in step 20. The remaining records were generated by the action that you performed in step 29. Note that the auditor can use any available tools to examine the archived audit file. The auditor can also copy the contents of the archived audit file to another location if desired.

```
[auditor@c01n01 acfs1]$ more .Security/audit/audit-c01n01-
1632357021.log.bak
Timestamp: 12/10/12 04:42:29:623 UTC
Event:ACFS_AUDIT_ENABLE
Source:ACFS Security
User:9999
Group:9999
Host:c01n01.example.com
Application:acfsutil.bin
Evaluation Result:ACFS_CMD_SUCCESS
Process:23533
FileSystem-ID:1632357021
Message:acfsutil audit enable: ACFS-10991: Auditing is enabled
on mount point '/
mnt/acfsmounts/acfs1'.
```

```
Timestamp: 12/10/12 04:45:32:040 UTC
Event:ACFS_AUDIT_CREATEFILE_OP
Source:ACFS Security
User:0
Group:0
Host:c01n01.example.com
Application:bash
Realm:myrealm1
File:Prac8-3
Evaluation Result:ACFS_AUDIT_REALM_AUTH
Process:23646
FileSystem-ID:1632357021
Message:Realm authorization succeeded for file ops CREATEFILE

Timestamp: 12/10/12 04:45:32:040 UTC
Event:ACFS_AUDIT_OPENFILE_OP
Source:ACFS Security
User:0
Group:0
Host:c01n01.example.com
Application:bash
Realm:myrealm1
File:audittest.txt
Evaluation Result:ACFS_AUDIT_REALM_AUTH
Process:23646
FileSystem-ID:1632357021
Message:Realm authorization succeeded for file ops OPENFILE

Timestamp: 12/10/12 04:45:32:063 UTC
Event:ACFS_AUDIT_WRITE_OP
Source:ACFS Security
User:0
Group:0
Host:c01n01.example.com
Application:cat
Realm:myrealm1
File:audittest.txt
Evaluation Result:ACFS_AUDIT_REALM_AUTH
Process:23646
FileSystem-ID:1632357021
Message:Realm authorization succeeded for file ops WRITE
```

```
[auditor@c01n01 acfs1]$
```

35. Mark the archived audit file as read. This action signals that the archived file is no longer required and can be purged.

```
[auditor@c01n01 acfs1]$ acfsutil audit read -m
/mnt/acfsmounts/acfs1
[auditor@c01n01 acfs1]$
```

36. Exit the auditor terminal session.

```
[auditor@c01n01 acfs1]$ exit
exit
[root@c01n01 acfs1]#
```

37. As the audit manager, purge the audit trail. This action is required so that the next (current) audit log file can be archived.

```
[root@c01n01 acfs1]# su auditmgr -c "acfsutil audit purge -m
/mnt/acfsmounts/acfs1"
[root@c01n01 acfs1]#
```

38. As the auditor, examine the audit directory. Confirm that the archived log files (`.bak` and `.xml`) have been deleted.

```
[root@c01n01 acfs1]# su auditor -c "ls -l .Security/audit"
total 4
----rw-r-- 1 root     auditmgr 1014 Dec 10 04:47 audit-c01n01-
1632357021.log
[root@c01n01 acfs1]#
```

39. As the system administrator, change back to the home directory and stop the Cloud FS file system that you have been using throughout this practice.

```
[root@c01n01 acfs1]# cd
[root@c01n01 ~]# srvctl stop filesystem -d /dev/asm/vol1-999
[root@c01n01 ~]#
```

Congratulations! You have successfully gone through the process of configuring Cloud FS auditing. You have also interacted with Cloud FS to generate audit records, and finally you exercised the audit trail management procedure.

Practice 8-4: Implementing Node-Specific File System Dependencies

Overview

In this practice, you will use the enhanced file system resource definitions to implement node-specific file system dependencies. This ensures that application resources on different nodes have access to separate node-specific filesystems whenever they are running.

Assumptions

This practice relies on the configuration performed in Practice 2-1.

Tasks

1. Establish a terminal session connected to `c01n01` using the `root` OS user.

```
$ ssh root@c01n01
Password: <oracle>
[root@c01n01 ~]$
```

2. Configure the environment using the `oraenv` script. Enter `+ASM1` when you are prompted for an ORACLE_SID value.

```
[root@c01n01 ~]$ . oraenv
ORACLE_SID = [grid] ? +ASM1
The Oracle base has been set to /u01/app/grid
[root@c01n01 ~]$
```

3. On `c01n01`, create a new node-specific file system mount point.

```
[root@c01n01 ~]# mkdir /mnt/acfsmounts/myappn01
[root@c01n01 ~]#
```

4. Use the following command to create a link to the directory that you created in the previous step. You will later use the link location to provide a consistent location on all the cluster nodes.

```
[root@c01n01 ~]# ln -s /mnt/acfsmounts/myappn01
/mnt/acfsmounts/myapp
[root@c01n01 ~]#
```

5. On `c01n02`, create a node-specific file system mount point and link it to the same location that you specified in the previous step.

```
[root@c01n01 ~]# ssh c01n02 "mkdir /mnt/acfsmounts/myappn02; ln
-s /mnt/acfsmounts/myappn02 /mnt/acfsmounts/myapp"
[root@c01n01 ~]#
```

6. Become the `grid` user.

```
[root@c01n01 ~]# su grid
[grid@c01n01 root]$
```

7. Start the ASM Command Utility (ASMCMD).

```
[grid@c01n01 root]$ asmcmd
ASMCMD>
```

8. Modify the DATA disk group to ensure that all the new ASM Dynamic Volume (ADVM) features included in release 12.1 are enabled.

```
ASMCMD> setattr -G DATA compatible.advm 12.1.0.0.0
ASMCMD>
```

9. Create a new volume. Place the volume in the DATA disk group and set the volume size to 200 MB. Name the volume VOLN01.

```
ASMCMD> volcreate -G data -s 200m VOLN01
ASMCMD>
```

10. Create another new volume. Place the volume in the DATA disk group and set the volume size to 200 MB. Name this volume VOLN02.

```
ASMCMD> volcreate -G data -s 200m VOLN02
ASMCMD>
```

11. Examine both of the newly created volumes and take note of the volume device associated with each volume. Note that your volume devices will be different from that shown below (/dev/asm/voln01-*999* and /dev/asm/voln02-*999*). Make note of the different volume devices because you will require them numerous times in the following steps.

```
ASMCMD> volinfo -G DATA VOLN01
Diskgroup Name: DATA

         Volume Name: VOLN01
         Volume Device: /dev/asm/voln01-999
         State: ENABLED
         Size (MB): 224
         Resize Unit (MB): 32
         Redundancy: MIRROR
         Stripe Columns: 4
         Stripe Width (K): 128
         Usage:
         Mountpath:

ASMCMD> volinfo -G DATA VOLN02
Diskgroup Name: DATA

         Volume Name: VOLN02
         Volume Device: /dev/asm/voln02-999
         State: ENABLED
         Size (MB): 224
         Resize Unit (MB): 32
         Redundancy: MIRROR
```

```
          Stripe Columns: 4
          Stripe Width (K): 128
          Usage:
          Mountpath:

ASMCMD>
```

12. Exit ASMCMD.

```
ASMCMD> exit
[grid@c01n01 root]$
```

13. Exit your `grid` OS session.

```
[grid@c01n01 root]$ exit
exit
[root@c01n01 ~]#
```

14. Make an `acfs` file system on each of the newly created volumes. Use the volume devices that you identified in step 11.

```
[root@c01n01 ~]# mkfs -t acfs /dev/asm/voln01-999
mkfs.acfs: version                   = 12.1.0.1.0
mkfs.acfs: on-disk version           = 39.0
mkfs.acfs: volume                    = /dev/asm/voln01-999
mkfs.acfs: volume size               = 234881024
mkfs.acfs: Format complete.
[root@c01n01 ~]# mkfs -t acfs /dev/asm/voln02-999
mkfs.acfs: version                   = 12.1.0.1.0
mkfs.acfs: on-disk version           = 39.0
mkfs.acfs: volume                    = /dev/asm/voln02-999
mkfs.acfs: volume size               = 234881024
mkfs.acfs: Format complete.
[root@c01n01 ~]#
```

15. Create a new node-specific Cloud FS file system resource on `c01n01` using the first volume device that you identified in step 11 along with the node-specific mount point that you created in step 3. Use the new `appid` option to associate an application identifier (`LOGFS`) with the file system.

```
[root@c01n01 ~]# srvctl add filesystem -node c01n01 -m
/mnt/acfsmounts/myappn01 -appid LOGFS -d /dev/asm/voln01-999
[root@c01n01 ~]#
```

16. Create a new node-specific Cloud FS file system resource on `c01n02` using the second volume device you identified in step 11 along with the node-specific mount point that you created in step 5. Use the same `appid` setting that you used in the previous step.

```
[root@c01n01 ~]# srvctl add filesystem -node c01n02 -m
/mnt/acfsmounts/myappn02 -appid LOGFS -d /dev/asm/voln02-999
[root@c01n01 ~]#
```

17. Confirm that the file systems that you created in the previous two steps are currently not mounted.

```
[root@c01n01 ~]# srvctl status filesystem | grep myapp
ACFS file system /mnt/acfsmounts/myappn01 is not mounted
ACFS file system /mnt/acfsmounts/myappn02 is not mounted
[root@c01n01 ~]#
```

18. Because you specified the `appid` option when you created the file systems in steps 15 and 16, a type definition containing the `appid` in the type name was created. Locate the full name of the type definition using the following command.

```
[root@c01n01 ~]# crsctl status type | grep LOGFS
TYPE_NAME=ora.LOGFS_fs.type
[root@c01n01 ~]#
```

19. Navigate to the directory that contains the files associated with this practice.

```
[root@c01n01 ~]# cd /stage/labs/8_4
[root@c01n01 8_4]#
```

20. Examine the file named `create_res.sh`. This file contains a command that you will use to add a simple application resource. Notice that the application resource contains a hard dependency on the `ora.LOGFS_fs.type` type. This dependency, coupled with the configuration that you performed earlier in the practice, results in a separate node-specific Cloud FS file system being started on each node hosting the application resource.

```
[root@c01n01 8_4]# cat create_res.sh
crsctl add resource my_application -type cluster_resource
-attr "ACTION_SCRIPT=/stage/labs/8_4/action.scr, CARDINALITY=2,
PLACEMENT=restricted, SERVER_POOLS=ora.orcldb,
START_DEPENDENCIES=hard(type:ora.LOGFS_fs.type)
pullup(type:ora.LOGFS_fs.type)"
[root@c01n01 8_4]#
```

21. Examine the action script associated with the resource definition that you observed in the previous step. Note that the application resource writes log information to a file under `/mnt/acfsmounts/myapp`. This is the location of the links that you created in steps 3 and 5.

```
[root@c01n01 8_4]# more action.scr
#!/bin/sh
TOUCH=/bin/touch
RM=/bin/rm
PATH_NAME=/tmp/$_CRS_NAME
HOSTNAME=`hostname -s`
LOG=/mnt/acfsmounts/myapp/$_CRS_NAME-$HOSTNAME.log

$TOUCH $LOG

#
# These messages go into the application log file.
echo " *******   `date` ********** " >> $LOG
```

```
echo "Action script '$_CRS_ACTION_SCRIPT' on '$HOSTNAME' for
resource [$_CRS_NAME] called for action
$1" >> $LOG
#

case "$1" in
  'start')
     echo "START entry point has been called.." >> $LOG
     echo "Creating the file: $PATH_NAME" >> $LOG
     $TOUCH $PATH_NAME
     exit 0
     ;;

  'stop')
     echo "STOP entry point has been called.." >> $LOG
     echo "Deleting the file: $PATH_NAME" >> $LOG
     $RM $PATH_NAME
     exit 0
     ;;

  'check')
    echo "CHECK entry point has been called.." >> $LOG
    if [ -e $PATH_NAME ]; then
        echo "Check -- $PATH_NAME exists" >> $LOG
        exit 0
    else
        echo "Check -- $PATH_NAME does not exist" >> $LOG
        exit 1
    fi
    ;;

  'clean')
     echo "CLEAN entry point has been called.." >> $LOG
     echo "Deleting the file: $PATH_NAME" >> $LOG
     $RM -f $PATH_NAME
     exit 0
     ;;

esac

[root@c01n01 8_4]#
```

22. Execute `create_res.sh` to create the application resource.

```
[root@c01n01 8_4]# ./create_res.sh
[root@c01n01 8_4]#
```

23. Start the newly created application resource. Notice that a node-specific file system is started on each node that hosts the application resource.

```
[root@c01n01 8_4]# crsctl start resource my_application
CRS-2672: Attempting to start 'ora.data.voln01.acfs' on 'c01n01'
CRS-2672: Attempting to start 'ora.data.voln02.acfs' on 'c01n02'
CRS-2676: Start of 'ora.data.voln01.acfs' on 'c01n01' succeeded
CRS-2672: Attempting to start 'my_application' on 'c01n01'
CRS-2676: Start of 'ora.data.voln02.acfs' on 'c01n02' succeeded
CRS-2672: Attempting to start 'my_application' on 'c01n02'
CRS-2676: Start of 'my_application' on 'c01n01' succeeded
CRS-2676: Start of 'my_application' on 'c01n02' succeeded
[root@c01n01 8_4]#
```

24. Examine `/mnt/acfsmounts/myapp` on `c01n01`. There you will find a log file named `my_application-c01n01.log`.

```
[root@c01n01 8_4]# ls -l /mnt/acfsmounts/myapp/
total 68
drwx------ 2 root root 65536 Dec 10 04:56 lost+found
-rw-r--r-- 1 root root   469 Dec 10 04:56 my_application-
c01n01.log
[root@c01n01 8_4]#
```

25. Examine `my_application-c01n01.log`.

```
[root@c01n01 8_4]# cat /mnt/acfsmounts/myapp/my_application-
c01n01.log
 *******   Mon Dec 10 04:56:42 UTC 2012 **********
Action script '/stage/labs/8_4/action.scr' on 'c01n01' for
resource [my_application] called for action start
START entry point has been called..
Creating the file: /tmp/my_application
 *******   Mon Dec 10 04:56:42 UTC 2012 **********
Action script '/stage/labs/8_4/action.scr' on 'c01n01' for
resource [my_application] called for action check
CHECK entry point has been called..
Check -- /tmp/my_application exists
[root@c01n01 8_4]#
```

26. Examine `/mnt/acfsmounts/myapp` on `c01n02`. There you will find a separate log file named `my_application-c01n02.log`.

```
[root@c01n01 8_4]# ssh c01n02 ls -l /mnt/acfsmounts/myapp/
total 68
drwx------ 2 root root 65536 Dec 10 04:57 lost+found
-rw-r--r-- 1 root root   469 Dec 10 04:58 my_application-
c01n02.log
[root@c01n01 8_4]#
```

27. Examine `my_application-c01n02.log`.

```
[root@c01n01 8_4]# ssh c01n02 cat
/mnt/acfsmounts/myapp/my_application-c01n02.log
*******   Mon Dec 10 04:57:01 UTC 2012 **********
Action script '/stage/labs/8_4/action.scr' on 'c01n02' for
resource [my_application] called for action start
START entry point has been called..
Creating the file: /tmp/my_application
 *******   Mon Dec 10 04:57:01 UTC 2012 **********
Action script '/stage/labs/8_4/action.scr' on 'c01n02' for
resource [my_application] called for action check
CHECK entry point has been called..
Check -- /tmp/my_application exists
[root@c01n01 8_4]#
```

Now you have seen how an application resource can use separate node-specific file systems on each node hosting the application. You have also seen how you can define a dependency between the application and the file systems by using the new `appid` file system attribute. This ensures that the required file system is available whenever the application resource starts, regardless of which node is hosting the application. This capability is useful for clustered applications where you want to keep the files associated with each application instance separate from each other.

28. Stop the application and file system resources that you used throughout this practice.

```
[root@c01n01 8_4]# crsctl stop resource my_application
CRS-2673: Attempting to stop 'my_application' on 'c01n01'
CRS-2673: Attempting to stop 'my_application' on 'c01n02'
CRS-2677: Stop of 'my_application' on 'c01n01' succeeded
CRS-2677: Stop of 'my_application' on 'c01n02' succeeded
[root@c01n01 8_4]# srvctl stop filesystem -d /dev/asm/voln01-999
[root@c01n01 8_4]# srvctl stop filesystem -d /dev/asm/voln02-999
[root@c01n01 8_4]#
```

Congratulations! You have successfully used the enhanced file system resource definitions in Cloud FS to implement node-specific file system dependencies.

Practices for Lesson 9: Application Continuity

Chapter 9

Practices for Lesson 9: Overview

Practices Overview

In this practice, you will explore Application Continuity.

Practice 9-1: Using Application Continuity

Overview

In this practice, you will use Application Continuity against a RAC database to demonstrate how Application Continuity helps an application to seamlessly recover after the failure of a RAC instance.

Assumptions

This practice relies on the configuration performed in Practice 2-1.

Tasks

1. Establish a terminal session connected to `c01n01` using the `oracle` OS user.

```
$ ssh oracle@c01n01
oracle@c01n01's password: <oracle>
[oracle@c01n01 ~]$
```

2. Configure the environment by using the `oraenv` script. Enter `orcl` when you are prompted for an ORACLE_SID value.

```
[oracle@c01n01 ~]$ . oraenv
ORACLE_SID = [oracle] ? orcl
The Oracle base has been set to /u01/app/oracle
[oracle@c01n01 ~]$
```

3. Start the `orcl` database.

```
[oracle@c01n01 ~]$ srvctl start database -d orcl
[oracle@c01n01 ~]$
```

4. Confirm that both instances of the RAC database are up and running.

```
[oracle@c01n01 ~]$ srvctl status database -d orcl
Instance orcl_1 is running on node c01n02
Instance orcl_2 is running on node c01n01
[oracle@c01n01 ~]$
```

5. Navigate to the directory that contains the files for this practice.

```
[oracle@c01n01 ~]$ cd /stage/labs/9_1
[oracle@c01n01 9_1]$
```

6. Examine the file named `create_service`. This file contains a command to create a database service on the `orcl` database. This service is configured for use in conjunction with Application Continuity.

```
[oracle@c01n01 9_1]$ cat create_service
srvctl add service -db orcl -service actest \
       -serverpool ora.orcldb \
       -cardinality singleton \
       -failovertype TRANSACTION \
       -commit_outcome TRUE \
       -failoverretry 50 \
       -failoverdelay 5 \
       -retention 86400 \
       -replay_init_time 1800 \
       -notification TRUE
[oracle@c01n01 9_1]$
```

7. Execute `create_service` to create the database service.

```
[oracle@c01n01 9_1]$ ./create_service
[oracle@c01n01 9_1]$
```

8. Start the service that you created in the previous step.

```
[oracle@c01n01 9_1]$ srvctl start service -db orcl -service
actest
[oracle@c01n01 9_1]$
```

9. Examine the status of the newly created service. Take note of the node it is running on (`c01n02` in this case), because it may be different in your environment.

```
[oracle@c01n01 9_1]$ srvctl status service -db orcl -service
actest
Service actest is running on nodes: c01n02
[oracle@c01n01 9_1]$
```

10. Using SQL*Plus, connect to the `orcl` database as the `system` user.

```
[oracle@c01n01 9_1]$ sqlplus system/oracle_4U@orcl

SQL*Plus: Release 12.1.0.1.0 Production...

SQL>
```

11. Configure the `scott` database user as shown below. This is required for the application that you will use later. After the user is configured, exit your SQL*Plus session.

```
SQL> alter user scott identified by tiger account unlock;

User altered.

SQL> exit
Disconnected from Oracle Database 12c...
[oracle@c01n01 9_1]$
```

12. Establish another terminal session connected to `c01n01` using the `oracle` OS user. To differentiate this session from your primary session, it will be referred to as the ADMIN session for the rest of the practice.

```
$ ssh oracle@c01n01
oracle@c01n01's password: <oracle>
[oracle@c01n01 ~]$
```

13. Configure the prompt in your ADMIN session as shown below. This will help you to differentiate between your terminal sessions as you progress through this practice.

```
[oracle@c01n01 ~]$ export PS1='ADMIN $ '
ADMIN $
```

14. Configure the environment in your ADMIN session by using the `oraenv` script. Enter `orcl` when you are prompted for an ORACLE_SID value.

```
ADMIN $ . oraenv
ORACLE_SID = [oracle] ? orcl
The Oracle base has been set to /u01/app/oracle
ADMIN $
```

15. Back in your primary session, examine the scripts that you will soon use to execute the practice application. Notice that both scripts execute the same application code (in `actest.jar`). The only difference is that each script references a different properties file.

```
[oracle@c01n01 9_1]$ cat runnoreplay
java -classpath
./actest.jar:$ORACLE_HOME/ucp/lib/ucp.jar:$ORACLE_HOME/jdbc/lib/
ojdbc6.jar actest.ACTest actest_noreplay.properties
[oracle@c01n01 9_1]$ cat runreplay
java -classpath
./actest.jar:$ORACLE_HOME/ucp/lib/ucp.jar:$ORACLE_HOME/jdbc/lib/
ojdbc6.jar actest.ACTest actest_replay.properties
[oracle@c01n01 9_1]$
```

16. Examine the properties files. Notice that the only difference is the `datasource` specification.

```
[oracle@c01n01 9_1]$ cat actest_noreplay.properties
username=scott
password=tiger
autoCommit=false
```

```
# Use standard  12.1 datasource no replay
datasource=oracle.jdbc.pool.OracleDataSource

url=jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=
cluster01-
scan.cluster01.example.com)(PORT=1521))(CONNECT_DATA=(SERVICE_NA
ME=actest.cluster01.example.com)))

# UCP setting:
ucp_pool_size=2
ucp_validate_connection_on_borrow=true
ucp_connection_wait_timeout=60

# Think Time taken to process the results from the database.
Time in milliseconds.
# -1 means no sleep.
thread_think_time=20

# Number of concurrent threads running in the application
# UCP is tuned to have MAX and MIN limit set to this
number_of_threads=6

verbose=true
[oracle@c01n01 9_1]$ cat actest_replay.properties
username=scott
password=tiger
autoCommit=false

# Use new 12.1 replay datasource
datasource=oracle.jdbc.replay.OracleDataSourceImpl

url=jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=
cluster01-
scan.cluster01.example.com)(PORT=1521))(CONNECT_DATA=(SERVICE_NA
ME=actest.cluster01.example.com)))

# UCP setting:
ucp_pool_size=2
ucp_validate_connection_on_borrow=true
ucp_connection_wait_timeout=60

# Think Time taken to process the results from the database.
Time in milliseconds.
```

```
# -1 means no sleep.
thread_think_time=20

# Number of concurrent threads running in the application
# UCP is tuned to have MAX and MIN limit set to this
number_of_threads=6

verbose=true
[oracle@c01n01 9_1]$ diff actest_replay.properties
actest_noreplay.properties
5,6c5,6
< # Use new 12.1 replay datasource
< datasource=oracle.jdbc.replay.OracleDataSourceImpl
---
> # Use standard  12.1 datasource no replay
> datasource=oracle.jdbc.pool.OracleDataSource
[oracle@c01n01 9_1]$
```

Next, you will execute the practice Java application twice. Once without the benefit of Application Continuity, and once with Application Continuity enabled. Notice that you will execute the same application and the only difference is the JDBC data source that is used on each occasion. The source files containing the application code are contained in the `src` directory. Feel free to examine the application code if you like.

17. Execute the practice application without the benefit of Application Continuity. Notice that while the application runs, a periodic status message is displayed.

```
[oracle@c01n01 9_1]$ ./runnoreplay
#########################################################
Connecting to
jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=clus
ter01-
scan.cluster01.example.com)(PORT=1521))(CONNECT_DATA=(SERVICE_NA
ME=actest.cluster01.example.com)))
  # of Threads             : 6
  UCP pool size            : 2
  Thread think time        : 20 ms
#########################################################

2 active connections, avg response time from db 38 ms
2 active connections, avg response time from db 23 ms
```

18. While the application continues to execute in the primary window, return to your ADMIN session and remind yourself about which node is running the `actest` service. Then, abort the database instance running the `actest` service (`c01n02` in the example shown below). Ensure that you abort the instance on the node running the service and not the other database node.

```
ADMIN $ srvctl status service -d orcl -s actest
Service actest is running on nodes: c01n02
ADMIN $ srvctl stop instance -db orcl -node c01n02 -stopoption
ABORT -force
ADMIN $
```

19. Return to your primary window and you should see a series of errors caused by the aborting the database instance. This is typical of applications that do not use Application Continuity. Press Ctrl + C to abort the application.

```
...
        at
oracle.ucp.jdbc.oracle.OracleJDBCConnectionPool.borrowConnection
(OracleJDBCConnectionPool.java:1441)
        at
oracle.ucp.jdbc.oracle.OracleConnectionConnectionPool.borrowConn
ection(OracleConnectionConnectionPool.java:81)
        at
oracle.ucp.jdbc.PoolDataSourceImpl.getConnection(PoolDataSourceI
mpl.java:1027)
        ... 4 more
.Exception occurred while getting connection:
oracle.ucp.UniversalConnectionPoolException: Cannot get
Connection from Datasource: java.sql.SQLRecoverableException:
Listener refused the connection with the following error:
ORA-12514, TNS:listener does not currently know of service
requested in connect descriptor
 .
0 active connections, avg response time from db 150418377 ms
^C[oracle@c01n01 9_1]$
```

20. Restart the aborted instance and confirm the both RAC database instances are up and running again.

```
[oracle@c01n01 9_1]$ srvctl start instance -d orcl -n c01n02
[oracle@c01n01 9_1]$ srvctl status database -d orcl
Instance orcl_1 is running on node c01n02
Instance orcl_2 is running on node c01n01
[oracle@c01n01 9_1]$
```

21. Reexamine the status of the `actest` service. You should observe that the service is running on a different node compared to what you observed earlier. This is because the service was migrated when you aborted the database instance earlier in the practice.

```
[oracle@c01n01 9_1]$ srvctl status service -d orcl -s actest
Service actest is running on nodes: c01n01
[oracle@c01n01 9_1]$
```

22. Execute the practice application with Application Continuity enabled. You should see the same period status messages as before while the application is running.

```
[oracle@c01n01 9_1]$ ./runreplay
#####################################################
Connecting to
jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=clus
ter01-
scan.cluster01.example.com)(PORT=1521))(CONNECT_DATA=(SERVICE_NA
ME=actest.cluster01.example.com)))
  # of Threads             : 6
  UCP pool size            : 2
  Thread think time        : 20 ms
#####################################################

2 active connections, avg response time from db 44 ms
2 active connections, avg response time from db 23 ms
2 active connections, avg response time from db 20 ms
```

23. While the application continues to execute in the primary window, return to your ADMIN session and remind yourself about which node is now running the `actest` service. Then, abort the database instance running the `actest` service (which is now `c01n01` in the example shown below). Ensure that you abort the instance on the node running the service and not the other database node.

```
ADMIN $ srvctl status service -d orcl -s actest
Service actest is running on nodes: c01n01
ADMIN $ srvctl stop instance -db orcl -node c01n01 -stopoption
ABORT -force
ADMIN $
```

24. Return to your primary window and you should see that the application continued in spite of aborted database instance. You should see a brief spike in the response time, which coincides with the time when the database instance was aborted. Now you have seen how Application Continuity masks the effect of database instance loss in a RAC database environment. Press Ctrl + C to abort the application.

```
[oracle@c01n01 9_1]$ ./runreplay
#####################################################
Connecting to
jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=clus
ter01-
scan.cluster01.example.com)(PORT=1521))(CONNECT_DATA=(SERVICE_NA
ME=actest.cluster01.example.com)))
  # of Threads             : 6
  UCP pool size            : 2
  Thread think time        : 20 ms
#####################################################

2 active connections, avg response time from db 44 ms
2 active connections, avg response time from db 23 ms
2 active connections, avg response time from db 20 ms
2 active connections, avg response time from db 16 ms
2 active connections, avg response time from db 14 ms
2 active connections, avg response time from db 12 ms
2 active connections, avg response time from db 12 ms
2 active connections, avg response time from db 15 ms
2 active connections, avg response time from db 425 ms
2 active connections, avg response time from db 16 ms
2 active connections, avg response time from db 13 ms
2 active connections, avg response time from db 12 ms
2 active connections, avg response time from db 12 ms
2 active connections, avg response time from db 13 ms
^C[oracle@c01n01 9_1]$
```

25. Confirm that the database instance aborted and that the service migrated to the other node as expected.

```
[oracle@c01n01 9_1]$ srvctl status database -d orcl
Instance orcl_1 is running on node c01n02
Instance orcl_2 is not running on node c01n01
[oracle@c01n01 9_1]$ srvctl status service -d orcl -s actest
Service actest is running on nodes: c01n02
[oracle@c01n01 9_1]$
```

26. Restart the stopped database instance.

```
[oracle@c01n01 9_1]$ srvctl start instance -d orcl -n c01n01
[oracle@c01n01 9_1]$
```

Congratulations! You have now seen how to configure and use Application Continuity.

Practices for Lesson 10: RAC New Features

Chapter 10

Practices for Lesson 10

Practices Overview

There are no practices for this lesson.

Practices for Lesson 11: Oracle Data Guard New Features

Chapter 11

Practices for Lesson 11

Practices Overview

There are no practices for this lesson.

Practices for Lesson 12: Oracle Global Data Services Overview

Chapter 12

Practices for Lesson 12

Practices Overview

There are no practices for this lesson.