

# X10 Deployment and User's Guide for Linux x86-64

Open Navigation

## 15 Managing an Oracle Database Appliance KVM Deployment

KVM virtualization uses a kernel-based virtual machine (KVM) to create a virtualized environment for your Linux applications.


Understand the Oracle Database Appliance KVM architecture, and procedures to deploy a guest virtual machine, manage high availability, manage CPU settings for Oracle Database Appliance KVM deployment.

- [About Oracle Database Appliance KVM Deployment](#)  
You can use Oracle KVM to optimize your system resource use for databases and applications.
- [About Oracle Database Appliance KVM Virtualization Architecture](#)  
Review this topic to understand how Oracle Database Appliance KVM deployment works with Oracle Database Appliance hardware.
- [About Virtual Machines and Oracle Database Appliance KVM Deployments](#)  
Oracle Database Appliance KVM deployment is designed to run and deploy virtual machines to manage system resources.
- [About KVM DB Systems on Oracle Database Appliance](#)  
Understand how you can deploy KVM-based DB systems on Oracle Database Appliance to run your Oracle Database environments.
- [Deploying Multiple Databases on DB Systems on Oracle Database Appliance](#)  
Understand the guidelines and procedure to deploy multiple databases on a DB system on Oracle Database Appliance.
- [Managing DB Systems in KVM Deployment](#)  
Use ODACLI to create, list, describe, start, stop, and delete DB systems in an Oracle Database Appliance KVM deployment.
- [Setting Up and Configuring a KVM Deployment for Applications](#)  
Understand the steps to set up and configure KVM deployment for applications.
- [Managing VM Storage in KVM Deployment](#)  
Use ODACLI to create, view, modify, and delete VM storage in an Oracle Database Appliance KVM deployment.
- [Managing Virtual Networks in KVM Deployment](#)  
Use ODACLI to create, view, start, stop, and delete virtual networks in an Oracle Database Appliance KVM deployment.
- [Managing Virtual Disks in KVM Deployment](#)  
Use ODACLI to create, view, clone, modify, and delete virtual disks on Oracle Database Appliance KVM deployment.
- [Managing Virtual Machines in KVM Deployment](#)  
Use ODACLI to create, view, clone, modify, start, stop, and delete virtual machines in an Oracle Database Appliance KVM deployment.
- [About Overcommitting Memory or CPUs in an Oracle Database Appliance KVM System](#)  
Understand performance and other considerations before overcommitting CPU and memory for application KVM.
- [Example JSON File to Create a Single-Node DB System](#)  
Follow the JSON file example to create a JSON file to deploy a single-node DB System, with role separation, with the command `odacli create-`
- [Example JSON File to Create a High-Availability DB System](#)  
Follow the JSON file example to create a JSON file to deploy a high-availability DB System, with role separation, with the command `odacli crea`

### About Oracle Database Appliance KVM Deployment

You can use Oracle KVM to optimize your system resource use for databases and applications.

You can deploy a Kernel-based Virtual Machine (KVM) virtual platform on Oracle Database Appliance. With Oracle Database Appliance KVM deployment, Oracle KVM to effectively allocate resources to databases and applications running on the same physical Oracle Database Appliance. Rather than simply

you can use the excess capacity to host other workloads. This enables consolidation of both databases and applications, while retaining the ease of deployment with Oracle Database Appliance. 

#### See Also:

For more information about supported operating systems, see the *Oracle Linux KVM User's Guide*: <https://docs.oracle.com/en/operating-systems/oracle-linux/7.7/using-kvm-to-run-oracle-database-appliance.html>

The KVM feature provides a set of modules that enable you to use the Oracle Linux kernel as a hypervisor. KVM supports x86\_64 processor architecture Unbreakable Enterprise Kernel (UEK) release. KVM features are actively developed and may vary depending on platform and kernel release. If you are unsure, you should refer to the release notes for the kernel release of your Oracle Database Appliance to obtain information about features and any known issues supported on Oracle Linux 7.

### What are the differences between KVM and Oracle VM Virtualization?

Oracle KVM makes it easy to setup and manage the virtualized environment with little virtualization expertise. With the KVM deployment, you can consolidate within a single Oracle Database Appliance system.

The following are some of the advantages of deploying Oracle Database Appliance with the KVM option:

- Deploy database and applications in a single hardware environment.
- Use your CPU cores (and memory) efficiently.
- Use virtual local area networks (VLANs) to provide separate networks to different virtual machines.
- Use VM storage to grow storage for the virtual machine repository.
- Enables easy resource management per VM and maintenance of quality of service (QoS)
- Provides solution-in-a-box with application, middleware, and databases.
- Improves data center efficiency by increasing space utilization and reducing energy consumption and cooling costs.
- Oracle KVM virtualization is available on bare metal deployments of Oracle Database Appliance, whereas Oracle VM virtualization on Oracle Database Appliance Virtualized Platform setup.
- Oracle Database Appliance supports KVM on all hardware models, whereas Virtualized Platform deployments using Oracle VM are supported on only High-Availability models.
- Oracle Database Appliance KVM deployments use Type 2 host operating system-based hypervisor, whereas Oracle Database Appliance Virtualized Platform uses Type 1 bare metal hypervisor.
- KVM is the virtualization technology used in Oracle Cloud Infrastructure (OCI), whereas Oracle Database Appliance Virtualized Platform is based on Oracle VM.

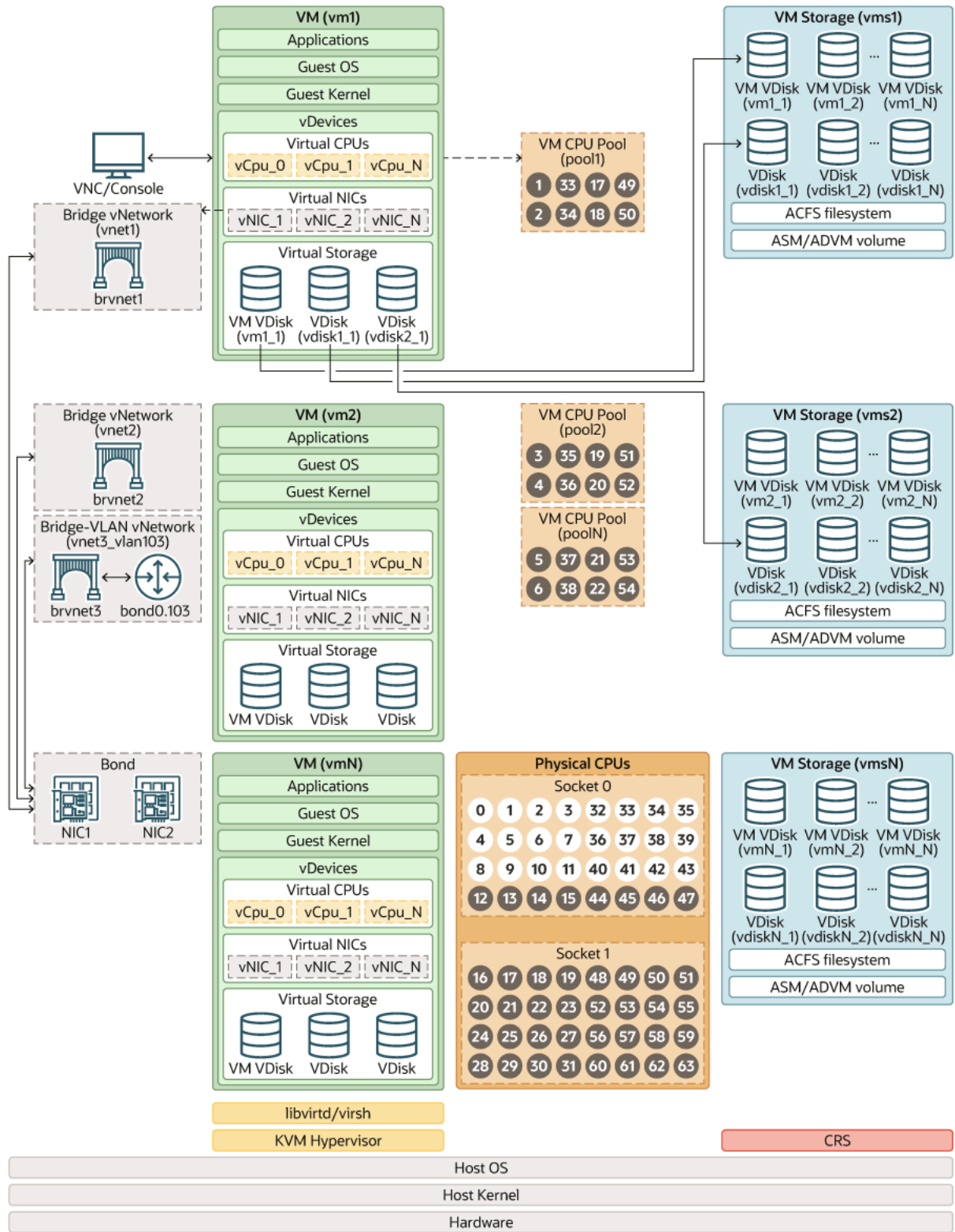
**Parent topic:** [Managing an Oracle Database Appliance KVM Deployment](#)

## About Oracle Database Appliance KVM Virtualization Architecture

Review this topic to understand how Oracle Database Appliance KVM deployment works with Oracle Database Appliance hardware.

Oracle Database Appliance KVM deployment provides virtualization technology that enables multiple applications to share the same physical server. This architecture is engineered specifically to leverage the Oracle Database Appliance hardware capabilities. Oracle Database Appliance KVM stack is integrated with Oracle VM Virtualization. VM resources such as VM storages and Virtual Machines are registered as CRS resources and are automatically managed by CRS for high availability.

The Oracle Database Appliance KVM virtualization architecture uses the virtual machine components shown in the following illustration:



Description of the illustration oda\_kvm\_architecture.png

Parent topic: Managing an Oracle Database Appliance KVM Deployment

## About Virtual Machines and Oracle Database Appliance KVM Deployments

Oracle Database Appliance KVM deployment is designed to run and deploy virtual machines to manage system resources.

Configure virtual machines on Oracle Database Appliance KVM deployment to manage the use of resources, such as the CPU pool, memory, and other :

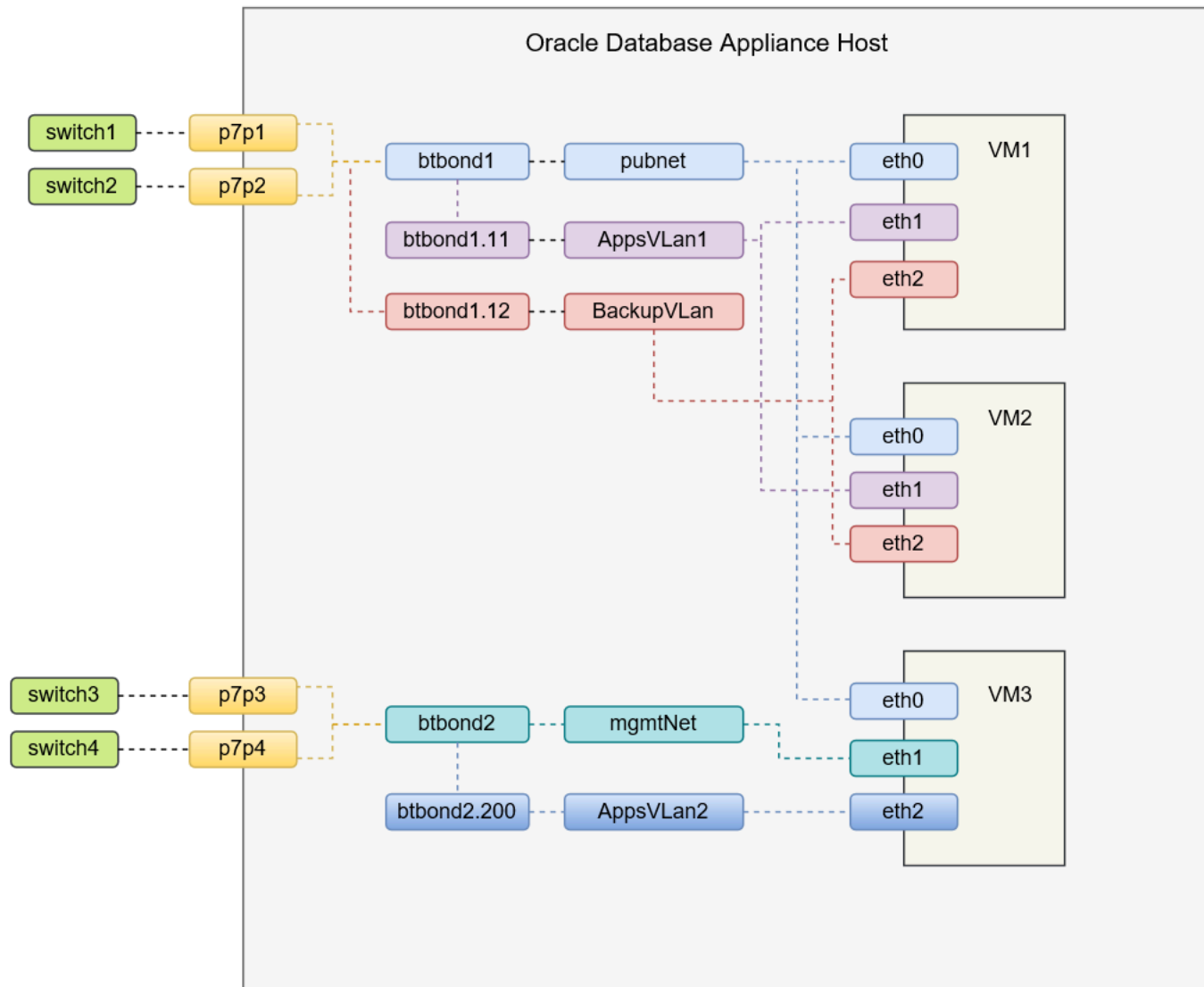
Understand the terminology of the various components you need to set up for an Oracle Database Appliance KVM deployment. The ODACLI tooling hai for your Oracle Database Appliance KVM deployment.

### VM Storage

A VM storage is a central location for storing resources that are essential to creating and managing virtual machines. These resources include ISO files (configuration files, and virtual disks. The VM storage is configured on an Oracle Automatic Storage Management Cluster File System (Oracle ACFS). Ora storage to optimize available disk space usage in the environment, and for easy reallocation of virtual machines if a physical server fails. The virtual mac disks, providing shared storage for the virtual machines. You can:

- Create one or more virtual machines on the VM storage.
- Use ODACLI commands to create and manage VM storage, virtual disks, and their virtual machines, and the underlying architecture shown in the

## Virtual Networks



Description of the illustration [kvm\\_network.png](#)

Oracle Database Appliance KVM virtual network supports two type of networks, bridged and bridged-vlan. The above figure is an example of the KVM virtual network architecture for Oracle Database Appliance X8-2 hardware models.

In a bridged network, a Linux bridge is created and the network interface or bond interface, is attached to the bridge. A default bridge network named pubnet is created during the appliance deployment. Any VM that wants to access this public network can be attached to this pubnet vnetwork. This public network is selected during the appliance deployment. Any VM that wants to access this public network can be attached to this pubnet vnetwork. In the above diagram, btbond1 is used for the public network, and default vnetwork pubnet is created with btbond1. eth0 of VMs are attached to the default pubnet bridge, no other bridged network is allowed to be created on this public network interface. Additional bridged vnetwork can be created and attached to the public network. In the above diagram, the mgmtNet bridged network is created with btbond2. eth1 of VM3 is attached to this mgmtNet vnetwork.

For example:

```
odacli create-vnetwork --name mgmtNet --bridge mgmtNet --type bridged --interface btbond2 --ip ip_address --gate
```

In a bridged-vlan network, VLAN can be created on all available public interfaces, including the interface where public network is already configured. For both `btbond1` and `btbond2`. Follow proper procedures to configure the VLAN on the switch before creating the bridged-vlan network. In the figure above, the network was created from `btbond1` and attached to `eth1` and `eth2` of VM1 and VM2 respectively.

For example:

```
odaccli create-vnetwork --name backupvlan --bridge backupvlan --type bridged-vlan --vlan-id 12 --interface btbond1
```

vnetwork is not supported on private interfaces and secondary interfaces of the bond interface.

## Virtual Disks

In addition to virtual machines, you can create virtual disks in VM storage. Virtual disks provide additional storage options for virtual machines by enabling storage to your virtual machines. Similarly, you can detach the disk if you no longer need the additional space. You can use virtual disks to expand existing virtual machine by adding the new virtual disk to an existing logical volume, or by creating a new file system on a virtual disk. Virtual disks can optionally be attached to multiple virtual machines.

## Virtual Machines

A *virtual machine* is granted virtual resources, and can be started, stopped, cloned, and restarted independently. By default, virtual machines are created for high-availability models. During failover, the VM is automatically started, and there will be an attempt to restart once before failing over to a different node. You can enable or disable autostart and failover using the `odaccli modify-vm` command options. The option to autostart is also available on single-node configurations.

**Parent topic:** [Managing an Oracle Database Appliance KVM Deployment](#)

# About KVM DB Systems on Oracle Database Appliance

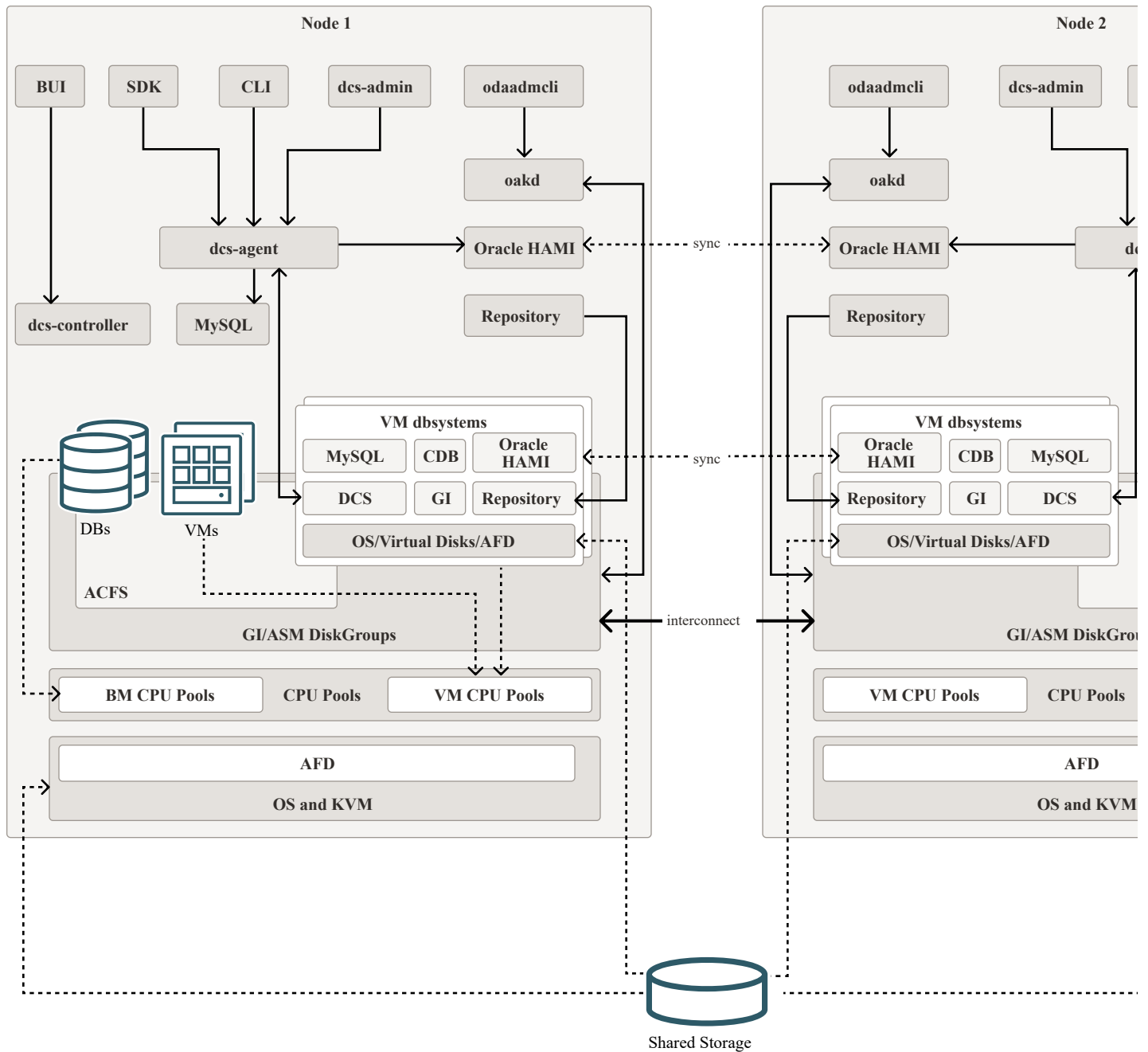
Understand how you can deploy KVM-based DB systems on Oracle Database Appliance to run your Oracle Database environments.

## About KVM DB Systems

KVM DB systems enable hard partitioning for Oracle Database licensing, where each KVM DB system has its own CPU pool that is automatically assigned. Oracle Database Appliance simplifies the management of KVM DB systems with the built-in Browser User Interface (BUI) or ODACLI Command Line Interface.

## About KVM DB System Architecture

You can provision and configure a DB system on an Oracle KVM. The database can be a single-instance Oracle Database or an Oracle RAC Database with multiple databases in each DB system.



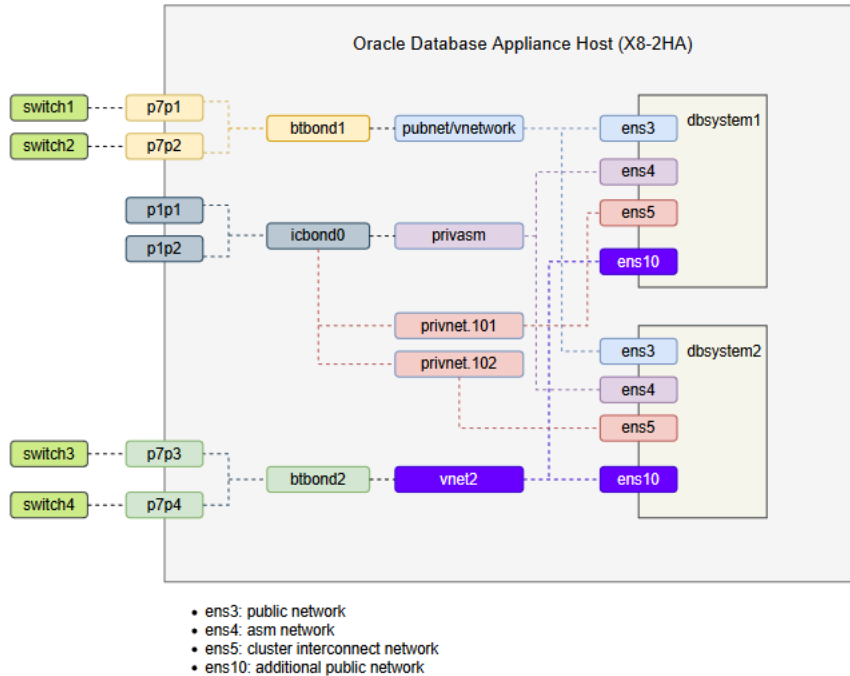
[Description of the illustration oda\\_kvm\\_db\\_system.svg](#)

## About KVM DB System Networks

Each Oracle Database Appliance DB system on KVM has the following networks created by default:

- Public network:** The default vnetwork pubnet is created on the bare metal system, based on the public interface you select for Oracle Grid Infrastructure. When you create a DB system, the public network of the DB system is attached to the pubnet vnetwork created on the bare metal system. The pubnet vnetwork is the public network of the DB system. You can also create a different vnetwork and use it as public network for the dbsystem.
- Oracle ASM network:** Oracle Database Appliance DB system uses Oracle ASM to manage the database storage. The Oracle ASM instance runs on the DB system. The database on the DB System uses the Oracle ASM listener running on the bare metal system to communicate with Oracle ASM. For example, privasm in the network diagram below, is created to facilitate this communication between the database on the DB system and the Oracle ASM on the bare metal system. This network is also used for communications between the DCS agent on the DB system and DCS agent on the bare metal system, and between the central repository on bare metal system with the DB systems.
- DB system cluster interconnect network:** When DB system is created as a two-node cluster on high-availability model, a cluster interconnect network is created and it is deleted when the DB system is deleted. Each cluster owns its cluster interconnect. Different DB system cannots communicate with each other on the same network. 192.168.16.0/24 and 192.168.17.0/24 are reserved for the use of these two internal networks. So, ensure that the IP CIDR 192.168.16.0/24 and 192.168.17.0/24 are available for use by Oracle Database Appliance DB system on KVM.

- Starting with Oracle Database Appliance release 19.12, you can create a vnetwork, either a bridged or bridged-vlan vnetwork, such as vnet2 in the `odacli modify-dbsystem` command to attach the vnetwork to the dbsystem.



[Description of the illustration dbsystems\\_network.png](#)

## Restrictions When Deploying KVM DB System

The following restrictions apply when you deploy KVM DB system on Oracle Database Appliance:

- You cannot use Oracle ACFS for storage within the DB system. The database running inside the dbsystem uses Oracle ASM storage. The VM is created on storage.
- On Oracle Database Appliance DB systems, the Oracle Grid Infrastructure software installed is of the same version as the Oracle Database version specified in the DB system JSON payload, with the attribute `version` for `database`.
- You cannot configure CPU pools or run VM commands within the DB system.

**Parent topic:** [Managing an Oracle Database Appliance KVM Deployment](#)

## Deploying Multiple Databases on DB Systems on Oracle Database Appliance

Understand the guidelines and procedure to deploy multiple databases on a DB system on Oracle Database Appliance.

### Supported DB System Shapes

In earlier releases of Oracle Database Appliance, where the DB system supported creation of only one database, the DB system shape was the same as the Oracle Database Appliance release 19.23, you can create multiple databases on the DB system, with a separate DB system shape `dbsX`, where X is the number of VMs. X is an even number that ranges from 2 to N-2 where N is the maximum number of CPU cores on the bare metal system. You can view the value of `num_cpus` command on the bare metal system. The default memory of the DB system is eight times that of the CPU cores of the DB system. You can create the DB system that is larger than the default memory size by specifying the memory size in the JSON file or BUI when you create the DB system.

### Restrictions and Guidelines for Deploying Multiple Databases on DB Systems

You can provision and configure a DB system on an Oracle KVM. The database can be a single-instance Oracle Database or an Oracle RAC Database with Oracle Database Appliance release 19.23, you can create multiple databases in each DB system. For an existing appliance, after you patch the bare metal system to release 19.23, you can create new DB systems that support multiple databases. For existing DB systems to support multiple databases, you must patch the appliance to release 19.23 first. To check whether a DB system supports multiple databases, run the `odacli list-dbsystems` command and verify that the shape of the DB system is `multiple`.

- You can create multiple databases on DB systems with Oracle Database release 19c version, but cannot create different major version databases within a DB system that supports 19c database, specify a 19c starter database during the DB system creation. If no starter database is specified in the DB system, a starter database that supports 19c database is created.
- You can use the `odacli create-database` command to create the first database if you did not choose to create a starter database, or additional databases. The Oracle Grid Infrastructure home and database home are configured on the local file system `/u01` of the DB system.

- When you create a database in DB system with the `odacli create-database` command, the default database shape is `odb1`.
- DB system must be sized properly based on the sizing requirement of databases running in the DB system. Taken together, the shapes of databases be no greater than the shape of the DB system, for example, databases `odbA`, `odbB`, ..., `odbN` running in the DB system shape `dbsx`, then  $A+B+...N \leq X$ . You can change the DB system shapes to meet the sizing requirement of the databases in the DB system.
- When you create a DB system, if you specify the starter database, then Oracle Grid Infrastructure of the same release as the database is installed. If you do not specify the starter database when you create a DB system, then Oracle Grid Infrastructure of the same release as the Oracle Database Appliance is installed.
- When you delete a DB system, and you use the `--force` option with the `odacli delete-dbsystem` command, all database data files in the DB system are deleted.
- You can move databases in the DB system across different database homes using the `odacli move-database` command.
- You can restore databases with the `odacli irestore-database` command to a DB system with support for multiple databases. This may result in the DB system shape supporting the additional databases restored. If not, change the DB system shape to accommodate the additional databases.
- Starting with Oracle Database Appliance release 19.23, the `odacli modify-dbsystem --shape` command only changes the shape of the DB system. It does not change the shapes of the database inside the DB system any more. You must modify database shapes accordingly based on the new shape of the DB system. For example, scale up database shapes after the DB system shape is scaled up. Similarly, scale down database shapes before DB system shape is scaled down, otherwise because of reduced DB system memory.

## Creating Multiple Databases on DB Systems

You can create databases on the DB system in one of the following ways:

- When you create the DB system using the `odacli create-dbsystem` command, you can optionally choose to create the starter database. The starter database software is installed and the started database is created. You specify the starter database details in the JSON file.
- If you do not specify the starter database details in the JSON file when you create the DB system, then only Oracle Grid Infrastructure is installed, but no database is created. After the `odacli create-dbsystem` command completes successfully, you can then create multiple databases using the `odacli create-database` command.

## Example JSON files to create databases on DB systems

Example JSON file to create DB system with starter database:

```
{
  "system": {
    "name": "dbsystem1",
    "diskGroup": "DATA",
    "systemPassword": "password",
    "timeZone": "Pacific/Majuro",
    "enableRoleSeparation": false,
    "shape": "dbs4",
    "customRoleSeparation": {
      "users": [
        {
          "name": "oracle",
          "id": 1618,
          "role": "oracleUser"
        }
      ]
    },
    "groups": [
      {
        "name": "oinstall",
        "id": 1018,
        "role": "oinstall"
      },
      {
        "name": "dbaoper",
        "id": 1019,
        "role": "dbaoper"
      },
      {
        "name": "dba",
        "id": 1020,
        "role": "dba"
      },
      {
        "name": "asmadmin",
        "id": 1021,
        "role": "asmadmin"
      }
    ]
  }
}
```



```

        "name": "asmoper",
        "id": 1022,
        "role": "asmoper"
    },
    {
        "name": "asmdba",
        "id": 1023,
        "role": "asmdba"
    }
]
},
"network": {
    "domainName": "us.oracle.com",
    "ntpServers": [
        "xx.xx.xxx.xx"
    ],
    "dnsServers": [
        "xxx.xxx.xx.x",
        "xxx.xx.xxx.xx"
    ],
    "scanName": "dbssystem1-scan",
    "scanIps": [
        "xx.xx.xxx.xxx",
        "xx.xx.xxx.xxx"
    ],
    "nodes": [
        {
            "name": "node1",
            "ipAddress": "xx.xx.xx.xxx",
            "netmask": "xxx.xxx.xxx.x",
            "gateway": "xx.xx.xxx.x",
            "number": 0,
            "vipName": "node1-vip",
            "vipAddress": "xx.xx.xxx.xxx"
        },
        {
            "name": "node2",
            "ipAddress": "xx.xx.xxx.xxx",
            "netmask": "xxx.xxx.xxx.0",
            "gateway": "xx.xx.xxx.x",
            "number": 1,
            "vipName": "node2-vip",
            "vipAddress": "xx.xx.xxx.xxx"
        }
    ]
},
"grid": {
    "language": "en",
    "enableAFD": false
},
"database": {
    "name": "rZWuZTw4",
    "uniqueName": "rZWuZTw4U",
    "domainName": "test_domain",
    "adminPassword": "password",
    "version": "19.23.0.0.240416",
    "edition": "EE",
    "type": "RAC",
    "dbClass": "IMDB",
    "shape": "odb2",
    "role": "PRIMARY",
    "targetNodeNumber": null,
    "enableDbConsole": false,
    "enableUnifiedAuditing": true,
    "redundancy": null,
    "characterSet": {
        "characterSet": "AL32UTF8",
        "nlsCharacterSet": "AL16UTF16",
        "dbTerritory": "AMERICA",
        "dbLanguage": "AMERICAN"
    },
    "rmanBackupPassword": null,

```

```

    "level0BackupDay": null,
    "enableTDE": false,
    "tdePassword": null,
    "isCdb": true,
    "pdbName": "test_pdb",
    "pdbAdminUser": "test_pdb_admin"
  }
}

```

Example JSON file to create DB system without starter database:

```

{
  "system": {
    "name": "dbssystem1",
    "diskGroup": "DATA",
    "systemPassword": "password",
    "timeZone": "Pacific/Majuro",
    "enableRoleSeparation": false,
    "shape": "dbs4",
    "customRoleSeparation": {
      "users": [
        {
          "name": "oracle",
          "id": 1618,
          "role": "oracleUser"
        }
      ],
      "groups": [
        {
          "name": "oinstall",
          "id": 1018,
          "role": "oinstall"
        },
        {
          "name": "dbaoper",
          "id": 1019,
          "role": "dbaoper"
        },
        {
          "name": "dba",
          "id": 1020,
          "role": "dba"
        },
        {
          "name": "asmadmin",
          "id": 1021,
          "role": "asmadmin"
        },
        {
          "name": "asmoper",
          "id": 1022,
          "role": "asmoper"
        },
        {
          "name": "asmdba",
          "id": 1023,
          "role": "asmdba"
        }
      ]
    }
  },
  "network": {
    "domainName": "testdomain",
    "ntpServers": [
      "xx.xx.xxx.xx"
    ],
    "dnsServers": [
      "xxx.xxx.xx.x",
      "xx.xx.xxx.xx"
    ],
    "scanName": "dbssystem1-scan",
    "scanIps": [

```

```

        "xx.xx.xxx.xxx",
        "xx.xx.xxx.xxx"
    ],
    "nodes": [
        {
            "name": "node1",
            "ipAddress": "xx.xx.xx.xx",
            "netmask": "xxx.xxx.xxx.x",
            "gateway": "xx.xx.xxx.x",
            "number": 0,
            "vipName": "node1-vip",
            "vipAddress": "xx.xx.xxx.xxx"
        },
        {
            "name": "node2",
            "ipAddress": "xx.xx.xx.xx",
            "netmask": "xxx.xxx.xxx.x",
            "gateway": "xx.xx.xxx.x",
            "number": 1,
            "vipName": "node2-vip",
            "vipAddress": "xx.xx.xxx.xxx"
        }
    ]
},
"grid": {
    "language": "en",
    "enableAFD": false
},
"database": null
}

```

**Parent topic:** [Managing an Oracle Database Appliance KVM Deployment](#)

## Managing DB Systems in KVM Deployment

Use ODACLI to create, list, describe, start, stop, and delete DB systems in an Oracle Database Appliance KVM deployment.

- [Creating a DB System in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to create a DB system in a KVM deployment.
- [Listing DB Systems in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to list DB systems in a KVM deployment.
- [Describing a DB System in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to describe a DB system in a KVM deployment.
- [Modifying a DB System in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to modify a DB system in a KVM deployment.
- [Attaching and Detaching a Network for a DB System in a KVM Deployment](#)  
Use ODACLI commands to attach or detach networks for DB system in a KVM deployment.
- [Starting a DB System in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to start a DB system in a KVM deployment.
- [Stopping a DB System in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to stop a DB system in a KVM deployment.
- [Deleting a DB System in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to delete a DB system in a KVM deployment.
- [Managing Backup, Restore, and Recovery on a DB System in a KVM Deployment](#)  
Understand the backup, restore, and recovery operations supported on a DB system in a KVM deployment.
- [Managing Shared CPU Pool with DB System in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to manage shared CPU pool with DB system in a KVM deployment.
- [Managing Oracle Data Guard on a DB System in a KVM Deployment](#)  
Understand the Oracle Data Guard operations supported on a DB system in a KVM deployment.

Parent topic: [Managing an Oracle Database Appliance KVM Deployment](#)

## Creating a DB System in a KVM Deployment

Use ODACLI commands or the Browser User Interface to create a DB system in a KVM deployment.



**Important:** Oracle Grid Infrastructure of the same version as the database version is installed in the DB system. Use the command `odacli` check the supported database versions for the DB system. Oracle Database Appliance release 19.23 supports creation of 23ai DB system through 23ai and Oracle Database 23ai. To create 23ai DB system, download and update the repository with DBVM 23ai template (patch 3652423ai clone (patch 36524627), and Oracle Database 23ai clones (patch 36524642). Specify the starter Oracle Database 23ai version in the `odacli describe-dbsystem-image` command to check the exact Oracle Database 23ai version. The 23ai DB system supports running database.



**Remember:** To create the DB system, access control must be enabled in Oracle ASM running on bare metal system. When the bare metal system is patched to Oracle Database Appliance release 19.23, access control is already enabled, which is a prerequisite for setting appropriate file ownership on DB systems and on the bare metal system. Before you create the DB system, the files on Oracle ASM disk groups that do not have file ownership. If ownership is not set, then creation of DB system fails. Use the `odacli modify-dbfileattributes` command to set up the

### Using ODACLI to Create a KVM DB System

Use the command `odacli create-dbsystem` to create a KVM DB system.

Follow these steps:

1. Download the Oracle Database Appliance KVM DB System Image from My Oracle Support to a temporary location on an external client. Refer to the software for the latest release.  
For example, to create a 23ai DB system, use patch number 36524660. For 19c DB system, use patch number 32451228.

```
patch_number_1923000_Linux-x86-64.zip
```

2. Unzip the software — it contains README.html and one or more zip files for the patch.

```
unzip patch_number_1923000_Linux-x86-64.zip
odacli-dcs-19.23.0.0.0-date-ODAVM-19.23.0.0.0.zip
```

3. Update the repository with the image.

```
# /opt/oracle/dcs/bin/odacli update-repository -f /tmp/odacli-dcs-19.23.0.0.0-date-ODAVM-19.23.0.0.0.zip
```

4. Run the `odacli describe-dbsystem-image` command to query the supported Oracle Grid Infrastructure and Oracle Database versions for the Oracle Grid Infrastructure version that supports the Oracle Database version is installed.

```
# odacli describe-dbsystem-image
DB System Image details
-----
Component Name Supported Versions Available Versions
-----
DBVM 23.5.0.0.0 23.5.0.0.0
GI 23.5.0.24.07 23.5.0.24.07
   23.5.0.24.07 23.5.0.24.07
DB 23.5.0.24.07 23.5.0.24.07
   23.5.0.24.07 23.5.0.24.07
DBVM 19.24.0.0.0 19.24.0.0.0
   GI 19.24.0.0.240716 19.24.0.0.240716
                                19.23.0.0.240416 not-available
                                19.22.0.0.240116 not-available
                                19.21.0.0.231017 not-available
                                19.20.0.0.230718 not-available
                                19.19.0.0.230418 not-available
                                19.18.0.0.230117 not-available
                                19.17.0.0.221018 not-available
                                19.16.0.0.220719 not-available
                                19.15.0.0.220419 not-available
```

```

19.14.0.0.220118 not-available
19.13.0.0.211019 not-available
19.12.0.0.210720 not-available
19.11.0.0.210420 not-available
21.8.0.0.221018 not-available
21.7.0.0.220719 not-available
21.6.0.0.220419 not-available
21.5.0.0.220118 not-available
21.4.0.0.211019 not-available
21.3.0.0.210720 not-available
DB 19.24.0.0.240716 19.24.0.0.240716
19.23.0.0.240416 not-available
19.22.0.0.240116 not-available
19.21.0.0.231017 not-available
19.20.0.0.230718 not-available
19.19.0.0.230418 not-available
19.18.0.0.230117 not-available
19.17.0.0.221018 not-available
19.16.0.0.220719 not-available
19.15.0.0.220419 not-available
19.14.0.0.220118 not-available
19.13.0.0.211019 not-available
19.12.0.0.210720 not-available
19.11.0.0.210420 not-available
21.8.0.0.221018 not-available
21.7.0.0.220719 not-available
21.6.0.0.220419 not-available
21.5.0.0.220118 not-available
21.4.0.0.211019 not-available
21.3.0.0.210720 not-available

```

5. If the `odacli describe-dbsystem-image` command output shows any component as `not-available` on the system, then download the cor versions and run the `odacli update-repository` to import the component to the repository.

To deploy Oracle Database release 19.23, deploy Oracle Grid Infrastructure release 19.23. To deploy 23ai DB system, ensure that the DBVM 23.4.x 23.4.x, and Oracle Database 23.4.x clones are shown available in the `odacli describe-dbsystem-image` command output.

6. Create the `prov.json` file as per examples provided in this chapter.

Oracle Database 21c and later supports only Container Databases (CDB). Ensure that you set the parameter `"isCdb": true`, and provide both the `pdbAdminUser` in the JSON file if provisioning an Oracle Database 21c or later DB system.

7. Run the `odacli create-dbsystem` command with the `prov.json` file on the host.

```
# odacli create-dbsystem -p prov.json
```

8. If the `odacli create-dbsystem` command displays the message `ASM ACL setup is not completed`, please run `'odacli modify-command`, then run the `odacli modify-dbfileattributes` command, and make sure the job finishes successfully before running the `odacli` command.

9. If the `odacli create-dbsystem` command displays the message `css_critical` configuration should be different on both nodes environment, then follow the instructions below to set `css_critical` and ensure that the DB system functions properly when interconnect fails.

- Run `crsctl set server css_critical yes` on the first bare metal host. You must restart the Oracle Clusterware stack on the node for effect.
- Run `crsctl set server css_critical no` on the second bare metal host. You must restart the Oracle Clusterware stack on the node for effect.
- Run `crsctl get server css_critical` to verify the value.

Refer to *Oracle Clusterware Administration and Deployment Guide* for more details about setting `css_critical`.

## Using Browser User Interface to Create a KVM DB System

Follow these steps:

- Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
  3. Click **DB Systems** to display the DB Systems page.
  4. In the page, click **Create DB System**.
  5. In the Create DB System page, provide the DB system values.
  6. If you want to use an existing configuration file, click **Browse** and select the configuration file. The system information from the file is populated in
  7. In the System Information tab, specify the values as follows.
    - **DB System Name:** Enter the DB System name. The name cannot end with a dash (-). Do not exceed 15 characters.
    - **Domain Name:** Enter the domain name.
    - **Region:** Select the region of the world where the Oracle Database Appliance is located.
    - **Time Zone:** Select the time zone where the Oracle Database Appliance is located.
    - Select **Specify Multi-User Access Option** to enable multi-user access on the DB system. You can choose either the **Enable Multi-User Access** or **Passwordless Multi-User Access** option. For more information about these features, see the chapter *Implementing Multi-User Access on Oracle Database Appliance* in this guide.
      - If you choose to enable multi-user access, then specify and confirm the **ODA Administrator User Password**, **Oracle User Password**, and **Grid User Password**. Click **Configure Multi-User Access Settings** and set the **User Password Expiry Duration**, **Session Expiration for CLI**, and **Login Attempts**.
      - If you choose to enable passwordless multi-user access, then you do not need to specify any of the above passwords.
    - **System Password and Confirm Password:** Enter the system password in both fields. The system password is the password set for UNIX and the password must contain at least two characters each from: uppercase letters, lowercase letters, numbers (0-9), and allowed special characters. The password must have a minimum of 9 characters and a maximum of 30 characters. Select the **Assign same password for admin, oracle, and grid users** option to keep the same password for all users.
    - **Disk Group for VM Storage:** Select the disk group that is used for hosting the KVM storage. Note that this is different from the database file disk group that runs in the KVM, which is displayed automatically from the correct Oracle ASM disk group based on the database file type.
    - **CPU Pool Name:** Select the CPU pool to be associated with the DB system.
    - Select **Force Run** if you want to allow the DB system to use this CPU pool even if selecting the CPU pool leads to oversubscription of the CPU pool. Select **Reserved CPU Cores** to use reserved CPU cores.
    - **Memory Size:** Specify the memory to be allocated to the DB system. The memory size can be in KB, MB, GB, or TB. The default is GB.
    - **VM Storage Redundancy:** Specify the redundancy for the VM storage, either **Mirror** or **High**.
    - (Optional) **DNS Servers:** Enter addresses for one or more DNS servers.
    - (Optional) **NTP Servers:** Enter addresses for one or more NTP servers.
    - **Public Network:** Select from the existing virtual networks.
  8. In the Network Information tab, specify the client access network details.
    - **Node Name:** For Node0, enter the host name for the primary client access network.
    - **IP Address:** Enter the virtual IP address that is shared between the nodes.
    - **Subnet Mask:** Enter the subnet mask address for the primary client access network.
    - **Gateway:** Enter the gateway address for the primary client access network.
- For two-node deployments, provide the above values for both nodes. In addition, you must also specify the following:
- **VIP Name and VIP Address:** Specify the Virtual IP name and address.
  - **SCAN Name and SCAN IP Address:** Specify the Single Client Access Name (SCAN) and SCAN IP address.
9. In the User and Group Selection tab, configure your users and groups and specify whether or not you want to allow operating system role separation.
    - Two users with six groups: Customize Users and Groups, select **No**. Allow OS Role Separation, select **Yes**. This is the default configuration.
    - Two customized users with six customized groups: Customize Users and Groups, select **Yes**. Allow OS Role Separation, select **Yes**.
    - Single user with two groups: Customize Users and Groups, select **No**. Allow OS Role Separation, select **No**.
    - Single user with six groups: Customize Users and Groups, select **Yes**. Allow OS Role Separation, select **No**.
    - Specify the **GI User**, **DB User**, **Install Group**, **DBA Oper Group**, **DBA Group**, **ASM Admin Group**, **ASM Oper Group**, **ASM DBA Group**, and the **ASM Backup Group**.
  10. In the Database Information tab, specify the following information to configure the database:
    - **DB Name:** Enter a name for the database. The name must contain alphanumeric characters and cannot exceed 8 characters.
    - **DB Version:** Specify the database version.

- (Optional) **DB Unique Name:** Enter a globally unique name for the database.  
Databases with the same DB Name within the same domain (for example, copies of a database created for reporting or a physical standby) must have a Unique Name that is unique within the enterprise. The name must begin with a lowercase or uppercase letter, and contain only alphanumeric characters and underscores(\_). The name must not contain dollar (\$), and pound (#) characters. The name cannot exceed 30 characters.
- **CDB:** Select **Yes** or **No** to specify whether or not you want a Container Database (CDB).
- **PDB Name:** Enter a name for the pluggable database (PDB).
- **PDB Admin User:** Enter an Admin user name for the pluggable database (PDB). The name must begin with an alphanumeric character. You can use only alphanumeric characters and underscore (\_) in the name.
- **Database Edition:** Select the Oracle Database edition, either Standard Edition and Enterprise Edition. Your license determines which database editions you can create in the DB System.
- **Deployment:** Select the type of Deployment, whether **RAC**, **RAC One**, or **SI**.  
For Standard Edition Oracle Database 19c or later, you can only create single-instance Oracle Database. For Standard Edition Oracle Database 12c, you can choose to enable high availability for single-instance database.  
For Enterprise Edition Oracle Database 19.15 or later or Oracle Database 21.6 or later, you can choose to enable high availability for single-instance database. For Enterprise Edition Oracle Database 19.15 or later or Oracle Database 21.6 or later, if you choose to create a single-instance database, then the high availability is disabled by default. To enable high-availability, set the value in the **Enable High Availability** field to **Yes**.
- **Sys and PDB Admin User Password and Confirm Password:** Provide a password for the database.
- **Shape:** Select a database shape from the list.
- **Database Redundancy:** If disk group redundancy is FLEX, then select **HIGH** or **MIRROR** from the drop down list for database redundancy. If disk group redundancy is not FLEX, then the database redundancy field is not available and is set to the same as the disk group redundancy internally.
- In the **Database Class** field, select a database class from the drop-down list. If an option is not available in the list, it is not supported for the Oracle Database Appliance or the version that you selected. The default is OLTP.
- **Configure EM Express:** Select **Yes** or **No**. Select **Yes** to configure the Oracle Enterprise Manager Database Express (EM Express) console.
- **Character set:** Select a character set.
- **National Character set:** Select a national character set.
- **Language:** Select the database language.
- **Territory:** Select a territory or location from the list.
- For Oracle Database Enterprise Edition 19c or later, you can choose to enable Transparent Database Encryption (TDE). Select **Yes** or **No** in the **Specify and confirm the TDE Password**. By default, the TDE option is disabled.
- **Data Files on Flash Storage:** Select **Yes** or **No**. This option is only available if the high-availability system has HDD storage drives.

11. Click **Create**.

12. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.

13. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-job`.

[Implementing Multi-User Access on Oracle Database Appliance](#)

[Example JSON File to Create a High-Availability DB System](#)

[Example JSON File to Create a Single-Node DB System](#)

`odacli describe-dbsystem-image`

`odacli modify-dbfileattributes`

**Parent topic:** [Managing DB Systems in KVM Deployment](#)

## Listing DB Systems in a KVM Deployment

Use ODACLI commands or the Browser User Interface to list DB systems in a KVM deployment.

### Using ODACLI to List DB Systems

Use the command `odacli list-dbsystems` to list DB systems.

```
# odacli list-dbsystems
```

### Using Browser User Interface to List DB Systems

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **DB Systems** to display the DB Systems page.
4. In the page, click on the DB system for which you want to view details.

[odacli list-dbsystems](#)

**Parent topic:** [Managing DB Systems in KVM Deployment](#)

## Describing a DB System in a KVM Deployment

Use ODACLI commands or the Browser User Interface to describe a DB system in a KVM deployment.

### Using ODACLI to Describe a DB System

Use the command `odacli describe-dbsystem` to describe a DB system.

```
# odacli describe-dbsystem -n dbsystem_name
```

### Using Browser User Interface to Describe a DB System

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **DB Systems** to display the DB Systems page.
4. In the page, click on the DB System for which you want to view details or click on the **Actions** drop down and select **View Details** to view the DB System details.
5. Click on the System Information tab for the DB System details, associated CPU Pool, Storage details, and associated VMs.
6. Click on the Database Information tab to view the associated database details.
7. Click on the Network Information tab to view the network details for single node or high-availability deployments.

[odacli describe-dbsystem](#)

**Parent topic:** [Managing DB Systems in KVM Deployment](#)

## Modifying a DB System in a KVM Deployment

Use ODACLI commands or the Browser User Interface to modify a DB system in a KVM deployment.

### Using ODACLI to Modify a DB System

Use the command `odacli modify-dbsystem --shape` to change shape of the DB system. Starting with Oracle Database Appliance release 19.23, the `shape` command only changes the shape of the DB system; it does not change the shapes of the database inside the DB system any more. You must migrate the database to a new shape based on the new shape of the DB system. Ensure that you scale up database shapes after the DB system shape is scaled up. Similarly, scale down database shapes after the DB system shape is scaled down; otherwise databases could fail to start because of reduced DB system memory.

```
# odacli modify-dbsystem -n dbsystem_name -s dbs4 [-dvn v_networks_to_detach] [-en] [-gw gateway] [-ip ip_address]
```

You can attach a CPU pool to the DB system, or remove the CPU pool from the DB system.

Use the command `odacli modify-dbsystem -m` to increase the DB system memory. This does not change the database memory configuration. You must migrate the database to a new shape based on the new shape of the DB system. Ensure that you scale up database shapes after the DB system shape is scaled up. Similarly, scale down database shapes after the DB system shape is scaled down; otherwise databases could fail to start because of reduced DB system memory.



after you change the DB system memory resets the memory of the DB system.

You can increase the DB system memory with the `odacli modify-dbsystem` command using the `-m` option.

```
# odacli modify-dbsystem -n dbsystem_name -m 24G
```

To reduce the DB system memory, use the `--shape` option with the command `odacli modify-dbsystem`. The DB system is automatically restarted

```
# odacli modify-dbsystem --name dbsystem1 --shape dbs2
```

Use the command `odacli modify-dbsystem` to attach or detach a new network to the DB system. This new network can be used for database backup purposes in the DB system.

You can use `odacli modify-dbsystem` to attach or detach the shared DB system CPU pool. Attaching a shared DB system CPU pool removes the internal system creation time. When detaching shared CPU pool from the DB system, an internal CPU pool is automatically created and attached to the DB system. It can be attached to multiple DB systems. Oversubscribing to shared DB system CPU pool is allowed, but there could be potential performance impact if s



**Note:** The `odacli modify-dbsystem -s shape` command only works on Oracle Database Appliance release 19.13 and later DB systems available on Oracle Database Appliance release 19.11 and later DB systems.

In Oracle Database Appliance release 19.23 and later, the `odacli modify-dbsystem -s shape` command changes the DB system shape database shape. Run the `odacli modify-database` command in the DB system to change the database shape in the DB system.

### Using Browser User Interface to Modify a KVM DB System

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **DB Systems** to display the DB Systems page.
4. In the page, click on the DB System for which you want to view details or click on the **Actions** drop down and select **Modify** to change the following:
  - DB System shape
  - Attach or detach CPU pool for the DB system
  - Attach or detach networks
  - Enable or disable NUMA for the DB system
  - Memory size allocated to the DB system
  - VM storage redundancy (either Mirror or High)
5. Select the new shape and CPU pool and click **Modify**.
6. Click **Enable NUMA** to set NUMA capabilities, and click **Modify**.
7. Click the Detach Networks field to view the networks, select the network you want to detach, and click **Modify**.
8. To attach a network, specify the **Name**, **IP Address**, **Subnet Mask**, **Gateway**, **Network Type**, **Default Network**, and click **Modify**.
9. Confirm your action to submit the job to modify the DB System. You can manually run the `odacli modify-database` command inside the DB system shape.

[odacli modify-dbsystem](#)

[odacli modify-vm](#)

[odacli remap-cpupools](#)

[odacli create-cpupool](#)

**Parent topic:** [Managing DB Systems in KVM Deployment](#)

### Attaching and Detaching a Network for a DB System in a KVM Deployment

Use ODACLI commands to attach or detach networks for DB system in a KVM deployment.

### Using ODACLI to Attach or Detach a Network to a DB System

You can attach a new vnetwork to the DB system and use it for database backup, Oracle Data Guard configuration, and other options. Use the `odaccli` `attach-network` or `detach-network` for a DB system in a KVM deployment. The vnetwork must exist before you can attach it to the dbssystem.

Follow these steps to attach a vnetwork to a DB system:

1. Identify the name of vnetwork bridge to attach to the DB system using the `odaccli list-vnetworks` and `odaccli describe-vnetwork -n vnetwork_name` commands.
2. Use the `odaccli modify-dbsystem` command to attach vnetwork to DB system:

```
# odaccli modify-dbsystem -n dbsystem1 -avn vnet2 -t Dataguard -ip 192.168.10.119,192.168.10.120 -nm 255.255.255.0 -gw 192.168.10.1 -sn scan1 -vip 192.168.10.119
```

The IP address, netmask (nm) and gateway (gw) values are used to configure the new interface of the DB system. The SCAN name (sn), SCAN IP address (vip) are used when the vnetwork type is `database` or `dataguard`.

To detach a vnetwork in the DB system, run the following command:

```
# odaccli modify-dbsystem -n name -dvn vnetwork_name
```

### Using Browser User Interface to Modify a DB System

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **DB Systems** to display the DB Systems page.
4. In the page, click on the DB System for which you want to view details or click on the **Actions** drop down and select **Modify** to attach or detach vnetwork.
5. Click the Detach Networks field to view the vnetworks, select the vnetwork you want to detach, and click **Modify**.
6. To attach a network, specify the **Name**, **IP Address**, **Subnet Mask**, **Gateway**, **Network Type**, **Default Network**, and click **Modify**.
7. Confirm your action to submit the job to modify the DB system.

Parent topic: [Managing DB Systems in KVM Deployment](#)

## Starting a DB System in a KVM Deployment

Use ODACLI commands or the Browser User Interface to start a DB system in a KVM deployment.

### Using ODACLI to start a DB System

Use the command `odaccli start-dbsystem` to start a DB system.

```
# odaccli start-dbsystem -n dbsystem_name
```

### Using Browser User Interface to Start a DB System

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.

3. Click **DB Systems** to display the DB Systems page.
4. In the page, click on the DB System which you want to start.
5. Click on the System Information tab and then click **Start**.
6. For high-availability deployments, select the node on which you want to start the DB System. Click **Yes**.
7. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
8. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-job`.
9. When the DB System starts, the state displays as ONLINE.

[odacli start-dbsystem](#)

**Parent topic:** [Managing DB Systems in KVM Deployment](#)

## Stopping a DB System in a KVM Deployment

Use ODACLI commands or the Browser User Interface to stop a DB system in a KVM deployment.

### Using ODACLI to Stop a DB System

Use the command `odacli stop-dbsystem` to stop a DB system.

```
# odacli stop-dbsystem -n dbsystem_name
```

### Using Browser User Interface to Stop a DB System

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **DB Systems** to display the DB Systems page.
4. In the page, click on the DB System which you want to stop.
5. Click on the System Information tab and then click **Stop**.
6. For high-availability deployments, select the node on which you want to stop the DB System. Click **Yes**.
7. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
8. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-job`.
9. When the DB system stops, the state displays as OFFLINE.

[odacli stop-dbsystem](#)

**Parent topic:** [Managing DB Systems in KVM Deployment](#)

## Deleting a DB System in a KVM Deployment

Use ODACLI commands or the Browser User Interface to delete a DB system in a KVM deployment.

### Using ODACLI to Delete a KVM DB System

Use the command `odacli delete-dbsystem` to delete a KVM DB system.

```
# odacli delete-dbsystem -n dbsystem_name -f
```

If you do not specify the `-f` option, then the operation deletes the DB system and Oracle Clusterware files, and retains the database files. With the `-f` option, Oracle Clusterware files, and the database files.

### Using Browser User Interface to Delete a KVM DB System

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **DB Systems** to display the DB Systems page.
4. In the page, for DB System which you want to delete, click on the **Actions** drop down list and select **Delete**.
5. Confirm that you want to delete the DB System. Select **Force Delete** if you want to delete the DB system, Oracle Clusterware files, and the database operation. Click **Yes**.
6. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
7. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-job odacli delete-dbsystem`

**Parent topic:** [Managing DB Systems in KVM Deployment](#)

## Managing Backup, Restore, and Recovery on a DB System in a KVM Deployment

Understand the backup, restore, and recovery operations supported on a DB system in a KVM deployment.

### About Backup, Restore, and Recovery on a DB System

Similar to the bare metal systems, you can use ODACL CLI commands to configure and perform backup, restore, and recovery operations on DB systems. For backup and recovery on Oracle Database Appliance, see the chapter *Backup, Recover and Restore* in this guide.

### Related Topics

- [Backup, Restore and Recover Databases](#)

**Parent topic:** [Managing DB Systems in KVM Deployment](#)

## Managing Shared CPU Pool with DB System in a KVM Deployment

Use ODACL CLI commands or the Browser User Interface to manage shared CPU pool with DB system in a KVM deployment.

### Using ODACL CLI to Manage Shared CPU Pool with DB system

Creating a DB system automatically creates an internal DB system CPU pool associated with this DB system. This DB system pool is managed internally and cannot be deleted manually. You can create a new type of shared CPU pool using the `odacli create-cpupool --dbssystem/-dbs -n cores` command. The shared CPU pool must not overlap with other CPU pools. In high-availability environments, the DB system CPU pool cannot be local, it must exist on both nodes.

### Associating Shared DB System CPU Pool

A shared DB system CPU pool can be associated to a new DB system or existing DB system. To create a new DB system with shared DB system CPU pool, use the `create-dbsystem json` file. The associated internal CPU pool is automatically deleted after a shared CPU pool is attached to a DB system. Use the `odacli modify-dbsystem` command to associate the CPU pool to an existing DB system.

```
# odacli modify-dbsystem -n dbssystem_name -cp dbspool1
```

The cores of the DB System (based on its shape) must fit into the shared DB System CPU pool. Associating a shared DB system CPU pool to an existing DB system is not supported. A single shared DB System CPU pool can be shared by multiple DB Systems. Oversubscription to the DB system CPU pool can be impacted if the CPU pool is oversubscribed. The list of associated DB systems can be queried using `odacli list-cpupool` or `odacli describe-cpupool`.

Use `odacli describe-dbsystem` to check the details of the CPU pool associated to the DB system:

```
# odacli describe-dbsystem -n dbs4e912c
CPU Pool
-----
Name:      ce3f42bb28
Number of cores: 2
```

```

Host: n1
Effective CPU set: 9-10,29-30
Online CPUs: 9, 10, 29, 30
Offline CPUs: NONE

Host: n2
Effective CPU set: 9-10,29-30
Online CPUs: 9, 10, 29, 30
Offline CPUs: NONE

```

Use `odacli list-cpupools` to view the DB system CPU pool type and associated DB systems. The internal DB system CPU pool has type "DB\_SYSTE" has type "DB\_SYSTEM\_SHARED".

```

# odacli list-cpupools
Name                Type                Configured on        Cores  Associated resources  Cre
-----
bmpool1             BM                  n1, n2               2      NONE                 2021-03-30 17:!!
vmppool1            VM                  n1, n2               2      NONE                 2021-03-30 17:!!
dbspool1            DB_SYSTEM_SHARED    n1, n2               4      NONE                 2021-03-30 17:!!
ce3f42bb28          DB_SYSTEM           n1, n2               2      dbs4e912c            2021-03-30 17:!!

```

Use `odacli describe-cpupool` to find out more details about DB system CPU pool:

```

# odacli describe-cpupool -n ce3f42bb28
CPU Pool details
-----
ID: 8e62933b-b394-4bcf-9c32-6a4cea2e0360
Name: ce3f42bb28
Created: 2021-03-30 17:57:43 UTC
Updated: 2021-03-30 18:01:27 UTC
Type: DB_SYSTEM
Number of cores: 2
Associated resources: dbs4e912c

CPU Allocations
-----
Node: n1
Effective CPU set: 9-10,29-30
Online CPUs: 9, 10, 29, 30
Offline CPUs: NONE

Node: n2
Effective CPU set: 9-10,29-30
Online CPUs: 9, 10, 29, 30
Offline CPUs: NONE

```

### Dissociating Shared DB System CPU Pool

A DB System with a shared DB System CPU pool could have the shared CPU pool detached from it and an internal DB System CPU Pool is created. If the be created, then the `odacli modify-dbsystem` command fails.

Dissociating shared DB System CPU Pool:

```
# odacli modify-dbsystem -n dbs4e912c -no-cp
```

### Modifying Shared DB System CPU Pool

A shared DB system CPU pool could be resized if all the associated DB systems shapes still fit into the cores of the pool (CPU pool size >= individual DB : immediately on running DB system VMs.

### Modifying a DB System Associated with a Shared DB System CPU Pool

If a DB System is associated with a shared DB System CPU Pool and its shape is modified with the `odacli modify-dbsystem` command, then the new cores of the pool.

### Deleting Shared DB System CPU Pool

A shared DB system CPU Pool can be deleted using the `odacli delete-cpupool` command only if it has no associated DB systems.

[Oracle Database Appliance KVM Hard Partitioning Compliance](#)

[odacli modify-dbsystem](#)

[Remapping CPU Pools in a Bare Metal or KVM Deployment](#)

[odacli modify-vm](#)

[odacli remap-cpupools](#)

[odacli create-cpupool](#)

**Parent topic:** [Managing DB Systems in KVM Deployment](#)

## Managing Oracle Data Guard on a DB System in a KVM Deployment

Understand the Oracle Data Guard operations supported on a DB system in a KVM deployment.

### About Using Oracle Data Guard on a DB System

Similar to the bare metal systems, you can configure and use Oracle Data Guard on DB systems using ODACLI commands. You can set up Oracle Data Guard and DB system, or between two DB systems. Integrated Oracle Data Guard can also be used for migrating from bare metal system to DB system, or between Oracle Data Guard on Oracle Database Appliance, see the chapter *Using Oracle Data Guard for Disaster Management and Recovery on Oracle Database Appliance*.

#### Related Topics

- [Using Oracle Data Guard for Disaster Management and Recovery on Oracle Database Appliance](#)

**Parent topic:** [Managing DB Systems in KVM Deployment](#)

## Setting Up and Configuring a KVM Deployment for Applications

Understand the steps to set up and configure KVM deployment for applications.

Follow these steps to set up a KVM deployment on Oracle Database Appliance

1. Create a VM storage.
2. Create a virtual network.
3. Create virtual disks.
4. Create virtual machines.

The following links provide information about performing these steps:

[Creating a VM Storage in a KVM Deployment](#)

[Creating a Virtual Network in a KVM Deployment](#)

[Creating a Virtual Disk in a KVM Deployment](#)

[Creating a Virtual Machine in a KVM Deployment](#)

**Parent topic:** [Managing an Oracle Database Appliance KVM Deployment](#)

## Managing VM Storage in KVM Deployment

Use ODACLI to create, view, modify, and delete VM storage in an Oracle Database Appliance KVM deployment.

- [Creating a VM Storage in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to create a VM storage in a KVM deployment.
- [Viewing VM Storage in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to view all VM storage or details about a VM storage in a KVM deployment.
- [Modifying VM Storage in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to modify VM storage in a KVM deployment.
- [Deleting a VM Storage in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to delete a VM storage in a KVM deployment.

**Parent topic:** [Managing an Oracle Database Appliance KVM Deployment](#)

## Creating a VM Storage in a KVM Deployment

Use ODACLI commands or the Browser User Interface to create a VM storage in a KVM deployment.

### Using ODACLI to Create VM Storage

Use the command `odacli create-vmstorage` to create a VM Storage.

Create a VM storage named `share1` of 8 GB.

```
# odacli create-vmstorage -n share1 -s 8G
```

### Using Browser User Interface to Create VM Storage

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **Show VM Instances** to display the VM Instances page.
4. In the page, select the **Create VM Storage** option and click **Next**.
5. In the Create VM Storage page, specify the following:
  - **Storage Name:** Name of the VM storage
  - **Storage Size:** Size of the storage to be allocated
  - **ASM Disk group:** Select the Oracle ASM disk group
  - **Redundancy:** Select the VM storage redundancy, either Mirror or High
6. Click **Create**.
7. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
8. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-`

**Parent topic:** [Managing VM Storage in KVM Deployment](#)

## Viewing VM Storage in a KVM Deployment

Use ODACLI commands or the Browser User Interface to view all VM storage or details about a VM storage in a KVM deployment.

### Using ODACLI to View VM Storage

The command `odacli describe-vmstorage` displays details about VM storage. Use the command `odacli list-vmstorages` to view all VM stor.

```
# odacli list-vmstorages
```

```
# odacli describe-vmstorage
```

### Using Browser User Interface to View VM Storage

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the VM Details page.
4. Click the **Show VM Storage** tab to view the list of all configured VM storages.
5. Click on a VM Storage to view the details of the VM storage.

**Parent topic:** [Managing VM Storage in KVM Deployment](#)

## Modifying VM Storage in a KVM Deployment

Use ODACLI commands or the Browser User Interface to modify VM storage in a KVM deployment.

### Using ODACLI to Modify VM Storages

Use the command `odacli modify-vmstorage` to modify VM storage.

Increase the size of a VM storage named `share1` by 10 gigabytes.

```
# odacli modify-vmstorage -n share1 -i 10G
```

Decrease the size of a VM storage named `share1` by 5 gigabytes.

```
# odacli modify-vmstorage -n share1 -s 5G
```

### Using Browser User Interface to Modify VM Storage

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the VM Details page.
4. Click the **Show VM Storage** tab.
5. In the page, select a VM Storage, select the **Modify** option and click **Next**.
6. In the Modify VM Storage page, specify **Increment** in size, the VM storage **Redundancy** as either **Mirror** or **High**, and click **Modify**.
7. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
8. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-`

**Parent topic:** [Managing VM Storage in KVM Deployment](#)

## Deleting a VM Storage in a KVM Deployment

Use ODACLI commands or the Browser User Interface to delete a VM storage in a KVM deployment.

### Using ODACLI to Delete VM Storage

Use the command `odacli delete-vmstorage` to delete a VM storage.

Delete a VM storage named `vs1`.

```
# odacli delete-vmstorage -n vs1
```



## Using Browser User Interface to Modify VM Storage

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the VM Details page.
4. Click the **Show VM Storage** tab.
5. In the page, select a VM Storage, select the **Delete** option.
6. Click Yes to confirm your choice.
7. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
8. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-`

**Parent topic:** [Managing VM Storage in KVM Deployment](#)

## Managing Virtual Networks in KVM Deployment

Use ODACLI to create, view, start, stop, and delete virtual networks in an Oracle Database Appliance KVM deployment.

- [Creating a Virtual Network in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to create a virtual network in a KVM deployment.
- [Viewing Virtual Networks in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to view all virtual networks or details about a virtual network in a KVM deployment.
- [Starting and Stopping Virtual Networks in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to start or stop virtual networks in a KVM deployment.
- [Modifying a Virtual Network in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to modify a virtual network in a KVM deployment.
- [Deleting a Virtual Network in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to delete a virtual network in a KVM deployment.

**Parent topic:** [Managing an Oracle Database Appliance KVM Deployment](#)

### Creating a Virtual Network in a KVM Deployment

Use ODACLI commands or the Browser User Interface to create a virtual network in a KVM deployment.

#### Using ODACLI to Create Virtual Network

Use the command `odacli create-vnetwork` to create a virtual network.

Create a VM network of type `bridged` using the interface `btbond2`.

```
# odacli create-vnetwork --name mgmtNet --bridge mgmtNet --type bridged --interface btbond2 --ip 192.168.120.26
```

Create a VM network of type `bridged-vlan` using the interface `btbond1`.

```
# odacli create-vnetwork --name backupvlan --bridge backupvlan --type bridged-vlan --vlan-id 12 --interface btbond1
```

#### Using Browser User Interface to Create Virtual Network

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **Show VM Instances** to display the VM Instances page.
4. In the page, select the **Create Virtual Network** option and click **Next**.
5. In the Create Virtual Network page, specify the following:
  - Name: Name of the virtual network
  - IP Address: Virtual network IP address
  - Subnet Mask and Gateway: Virtual network subnet mask and gateway
  - Network Type: Virtual network type
  - Interface: Virtual network interface
  - Bridge Name: Name of the network bridge
  - VLAN ID: ID of the VLAN network
6. Click **Create**.
7. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
8. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-`

**Parent topic:** [Managing Virtual Networks in KVM Deployment](#)

## Viewing Virtual Networks in a KVM Deployment

Use ODACLI commands or the Browser User Interface to view all virtual networks or details about a virtual network in a KVM deployment.

### Using ODACLI to View Virtual Networks

The command `odacli describe-vnetwork` displays details about a virtual network. Use the command `odacli list-vnetworks` to view all virtual networks.

```
# odacli list-vnetworks
```

```
# odacli describe-vnetwork -n vnet1
```

### Using Browser User Interface to Create Virtual Networks

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the VM Details page.
4. Click the **Show Virtual Networks** tab to view the list of all configured virtual networks.
5. Click on a Virtual Network to view the details of the virtual network.

**Parent topic:** [Managing Virtual Networks in KVM Deployment](#)

## Starting and Stopping Virtual Networks in a KVM Deployment

Use ODACLI commands or the Browser User Interface to start or stop virtual networks in a KVM deployment.

### Using ODACLI to Start and Stop Virtual Networks

The command `odacli start-vnetwork` starts a virtual network. Use the command `odacli stop-vnetwork` to stop a virtual network in the deployment.

```
# odacli start-vnetwork -n vnet1
```

```
# odacli stop-vnetwork -n vnet1
```

### Using Browser User Interface to Start and Stop Virtual Networks

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the VM Details page.
4. Click the **Show Virtual Networks** tab to view the list of all configured virtual networks.
5. To start a virtual network, click on a Virtual Network, and then click **Start**.
6. To stop a virtual network, click on a virtual network, and then click **Stop**.

**Parent topic:** [Managing Virtual Networks in KVM Deployment](#)

## Modifying a Virtual Network in a KVM Deployment

Use ODACLI commands or the Browser User Interface to modify a virtual network in a KVM deployment.

### Using ODACLI to Modify Virtual Network

Use the command `odacli modify-vnetwork` to modify a virtual network.

```
# odacli modify-vnetwork -n vnet1 -g 10.11.44.41
```

### Using Browser User Interface to Modify Virtual Network

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the VM Details page.
4. Click the **Show Virtual Networks** tab.
5. In the page, select a Virtual Network, select the **Modify** option and click **Next**.
6. In the Modify Virtual Network page, specify any changes in the Subnet Mask, Gateway, or IP Address, and click **Modify**.
7. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
8. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-job`.

**Parent topic:** [Managing Virtual Networks in KVM Deployment](#)

## Deleting a Virtual Network in a KVM Deployment

Use ODACLI commands or the Browser User Interface to delete a virtual network in a KVM deployment.

### Using ODACLI to Delete Virtual Networks

Use the command `odacli delete-vnetwork` to delete a virtual network.

Delete a virtual network named `vnet1`.

```
# odacli delete-vnetwork -n vnet1
```

### Using Browser User Interface to Modify Virtual Networks

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the VM Details page.
4. Click the **Show Virtual Networks** tab.
5. In the page, select a virtual network, select the **Delete** option.
6. Click Yes to confirm your choice.
7. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
8. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-`

**Parent topic:** [Managing Virtual Networks in KVM Deployment](#)

## Managing Virtual Disks in KVM Deployment

Use ODACLI to create, view, clone, modify, and delete virtual disks on Oracle Database Appliance KVM deployment.

- [Creating a Virtual Disk in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to create a virtual disk in a KVM deployment.
- [Viewing Virtual Disks in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to view all virtual disks or details about a virtual disk in a KVM deployment.
- [Cloning a Virtual Disk in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to clone a virtual disk in a KVM deployment.
- [Modifying a Virtual Disk in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to modify a virtual disk in a KVM deployment.
- [Deleting a Virtual Disk in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to delete a virtual disk in a KVM deployment.

**Parent topic:** [Managing an Oracle Database Appliance KVM Deployment](#)

### Creating a Virtual Disk in a KVM Deployment

Use ODACLI commands or the Browser User Interface to create a virtual disk in a KVM deployment.

Ensure that a VM storage exists before you create a virtual disk.

#### Using ODACLI to Create Virtual Disks

Use the command `odacli create-vdisk` to create a virtual disk.

Specify the name of the virtual disk in the command `odacli create-vdisk -n name`. Identify the storage in which you create the virtual disk by using option `-s size` to specify the virtual disk size. If you want to configure the virtual disk as a shared disk, then use the option `-sh`.

The vdisk name must start with a letter followed by underscores, hyphens, or alphanumeric characters and can have a maximum length of 30 character

Create a 2 GB non-sparse and shareable virtual disk named `vdisk1` inside the VM storage `vms1`.

```
odacli create-vdisk -n vdisk1 -st vms1 -s 2G -sh
```

### Using Browser User Interface to Create Virtual Disks

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the Virtual Disk page.
4. Click **Create Virtual Disk**.
5. In the page, select the **Create Virtual Disk** option and click **Next**.
6. In the Create Virtual Disk page, specify the following:
  - **Virtual Disk Name:** Name assigned to the virtual disk that is unique within the name repository
  - **VM Storage Name:** Name of the VM storage where the virtual disk will be created
  - **Disk Size:** Size of the virtual disk
  - **Shared:** Specify if you want to share the virtual disk
  - **Sparse or Non-Sparse:** Specify if the virtual disk is sparse
7. Click **Create**.
8. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
9. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-`

**Parent topic:** [Managing Virtual Disks in KVM Deployment](#)

## Viewing Virtual Disks in a KVM Deployment

Use ODACLI commands or the Browser User Interface to view all virtual disks or details about a virtual disk in a KVM deployment.

### Using ODACLI to View Virtual Disks

The command `odacli describe-vdisk` displays details about a virtual disk. Use the command `odacli list-vdisks` to view all virtual disks in the

```
# odacli list-vdisks
```

```
# odacli describe-vdisk -n vdisk_name
```

### Using Browser User Interface to Create Virtual Disks

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the VM Details page.
4. Click the **Show Virtual Disks** tab to view the list of all configured virtual disks.
5. Click on a virtual disk to view the details of the virtual disk.

**Parent topic:** [Managing Virtual Disks in KVM Deployment](#)

## Cloning a Virtual Disk in a KVM Deployment

Use ODACLI commands or the Browser User Interface to clone a virtual disk in a KVM deployment.

### Using ODACLI to Clone Virtual Disks

Use the command `odacli clone-vdisk` to clone an existing virtual disk.

Specify the source `vdisk` from which you want to clone with the `-n` option, and specify the name of the cloned virtual disk with the `-cn` option.

Create a clone of a virtual disk named `vdisk1`, with the name `vdisk1_clone`. The cloned disk is created on the same storage as `vdisk1`.

```
# odacli clone-vdisk -n vdisk1 -cn vdisk1_clone
```

### Using Browser User Interface to Create Virtual Disks

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the Virtual Disk page.
4. Click **Create Virtual Disk**.
5. In the page, select the **Clone Virtual Disk** option and click **Next**.
6. In the Clone Virtual Disk page, specify the following:
  - Select the source virtual disk you want to clone
  - Specify the name of the cloned virtual disk
7. Click **Create**.
8. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
9. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-`

**Parent topic:** [Managing Virtual Disks in KVM Deployment](#)

## Modifying a Virtual Disk in a KVM Deployment

Use ODACLI commands or the Browser User Interface to modify a virtual disk in a KVM deployment.

### Using ODACLI to Modify Virtual Disks

Use the command `odacli modify-vdisk` to increase the size of a virtual disk.

Increase the size of a virtual disk named `vdisk1` by 4 gigabytes.

```
# odacli modify-vdisk -n vdisk1 -i 4G
```

### Using Browser User Interface to Modify Virtual Disks

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the VM Details page.
4. Click the **Show Virtual Disks** tab.
5. In the page, select a Virtual Disk, select the **Modify Virtual Disk** option and click **Next**.
6. In the Modify Virtual Disk page, specify Increment in size and if you want to share the virtual disk.
7. Click **Modify**.
8. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
9. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-`

**Parent topic:** [Managing Virtual Disks in KVM Deployment](#)

## Deleting a Virtual Disk in a KVM Deployment

Use ODACLI commands or the Browser User Interface to delete a virtual disk in a KVM deployment.

### Using ODACLI to Delete Virtual Disks

Use the command `odacli delete-vdisk` to delete a virtual disk.

Delete a virtual disk named `vdisk1`.

```
# odacli delete-vdisk -n vdisk1
```

### Using Browser User Interface to Modify Virtual Disks

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the VM Details page.
4. Click the **Show Virtual Disks** tab.
5. In the page, select a Virtual Disk, select the **Delete** option.
6. Click Yes to confirm your choice.
7. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
8. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-`

**Parent topic:** [Managing Virtual Disks in KVM Deployment](#)

## Managing Virtual Machines in KVM Deployment

Use ODACLI to create, view, clone, modify, start, stop, and delete virtual machines in an Oracle Database Appliance KVM deployment.

- [Creating a Virtual Machine in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to create a virtual machine in a KVM deployment.
- [Cloning a Virtual Machine in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to clone a virtual machine instance in a KVM deployment.
- [Modifying a Virtual Machine in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to modify a virtual machine in a KVM deployment.
- [Viewing Virtual Machines in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to view all virtual machines or details about a virtual machine in a KVM deployment.
- [Starting and Stopping Virtual Machines in a KVM Deployment](#)  
Use ODACLI commands or the Browser User Interface to start or stop virtual machines in a KVM deployment.

- [Deleting a Virtual Machine in a KVM Deployment](#)

Use ODACLI commands or the Browser User Interface to delete a virtual machine in a KVM deployment.

**Parent topic:** [Managing an Oracle Database Appliance KVM Deployment](#)

## Creating a Virtual Machine in a KVM Deployment

Use ODACLI commands or the Browser User Interface to create a virtual machine in a KVM deployment.

Ensure that a VM storage exists before you create a VM instance.

### Using ODACLI to Create Virtual Machine Instance

Use the command `odacli create-vm` to create a VM instance. Specify the preferred node on which to start the virtual machine after creation (`-pn` or

Following is an example command to create a VM named `vm1` with 8 vcpus and 8 GB memory. A VM disk of size 10 GB is created automatically as `/dev` system installation. The VM disk is stored in the VM storage `vms1`. The source used for the installation is located on `/u01/OL7.iso`. If you want to attach a VM, you can specify `--vdisks vdisk_name`, where `vdisk_name` is an existing virtual disk. This virtual disk when attached, is shown as `/dev/vdb` in:

```
# odacli create-vm -n vm1 -vc 8 -m 8G -vms vms1 -src /u01/OL7.iso -size 10G
```

To use a kickstart configuration file to create the VM instance, include the file in the `--extra-args` parameter. Create a kickstart configuration file, include the guest VM. Ensure that you specify the network details in the `--extra-args` parameter so that the VM bootstrap can use it to access the kickstart configuration file. The IP must belong to the vnetwork to be attached to the VM, and the kickstart file must be available over this vnetwork.

For example:

```
odacli create-vm --name odaksvm --vnetwork pubnet --memory 4G --source /u01/kvm/images/OL77_x86_64.iso --vmstorage vms1
```

If you do not use a kickstart configuration file to create the VM instance, then complete the VM instance creation as follows:

1. Run the `odacli create-vm` command and check that it completes successfully.
2. Use the `odacli describe-vm` command to check the VNC port.
3. Log in to the VM instance using the VNC port: `host:vncport`. See the section *Connecting to the VM instance Using VNC* for the steps to connect.
4. Complete the installation process by providing your values for the installation screens.
5. Log in again to the VM instance using the VNC port: `host:vncport`. See the section *Connecting to the VM instance Using VNC* for the steps to connect to VNC.
6. If you specified `--vnetwork` in the `odacli create-vm` command, then you can run `ifconfig -a` to view the network interface in the VM. Corresponding interface would be on a Linux system.
7. After the network is configured, log into the VM using the network and customize the VM instance.
8. If you did not specify `--vnetwork` in the `odacli create-vm` command, then the VM network is created using the default Network Address Translation (NAT) bridge `virbr0`. `eth0` is configured as DHCP inside the VM.

### Connecting to the VM instance Using VNC: Method 1

1. Find out the VNC display port of the VM from the command `odacli describe-vm -n vm_name: 127.0.0.1:1`

```
# odacli describe-vm -n vm1
VM details
-----
ID: c280af13-997c-49b1-97ce-0617610535f1
Name: vm1
...
Graphics settings: vnc,listen=127.0.0.1
Display Port: 127.0.0.1:1 <<<<
...
```

2. Run `vncserver` on the Oracle Database Appliance host. Note down the vncserver address `odahost:11`.



```
# vncserver
New 'odahost:11 (root)' desktop is odahost:11
```

3. Launch `vncviewer` from your desktop, login to the Oracle Database Appliance host using the `vncserver` address created in step 2: `odahost:11`. This into the Oracle Database Appliance host.
4. From `vncviewer` launched in step 3, run the `vncviewer` command with the display port from the `odacli describe-vm` output. For example, in `describe-vm` command shows the display port is `127.0.0.1:1`, so type `vncviewer 127.0.0.1:1`.

### Connecting to the VM instance Using VNC: Method 2

1. Find out the VNC display port of the VM from the command `odacli describe-vm`. For example, the Display Port is `Display Port: 127.0.0.myodahost1`.

```
# odacli describe-vm -n vm1
VM details
-----
ID: c280af13-997c-49b1-97ce-0617610535f1
Name: vm1
...
Graphics settings: vnc,listen=127.0.0.1
Display Port: 127.0.0.1:1 <<<<
...
Status
-----
Current node: myodahost1 <<<
Current state: ONLINE
Target state: ONLINE
```

2. On Linux or macOS operating systems where you want to launch the `vncviewer`, use the following command to create the SSH tunnel:

```
ssh -L localport:127.0.0.1:vncport root@odahost
```

where `localport` is an available port number greater than 1024 on your local machine (for example, 12345) where `vnc viewer` is launched. `vncport` is the `Display Port` from the `odacli describe-vm` command plus 5900. For example, if `odacli describe-vm` returns the display port as `127.0.0.1:1`, then use 5901 as the name of the Oracle Database Appliance host where the VM is running. 127.0.0.1 is the address where `vnc server` is listening at. Do not change the example:

```
ssh -L 12345:127.0.0.1:5901 root@odahost1
```

On Windows, use PuTTY to create the SSH tunnel.

3. Launch `vncviewer` from your desktop, enter VNC server address as `localhost:localport`, where `localport` is one used in the SSH tunnel in step example:

```
vncserver: localhost:12345
```

### How to find VM network interface attached to the vnetwork

If you specified `--vnetwork` in the `odacli create-vm` command, then you can run `ifconfig -a` to view the network interface in the VM. Find out is attached to the `vnetwork` by matching the interface MAC address to the MAC address displayed in the `odacli describe-vm` command.

For example:

```
# odacli describe-vm -n vm1
VM details
-----
ID: c280af13-997c-49b1-97ce-0617610535f1
Name: vm1
...

```

```
Parameters
```

```
...
```

```
Config Live
```

```
Memory: 2.00 GB 2.00 GB
Max Memory: 2.00 GB 2.00 GB
```

```
...
```

```
vNetworks: pubnet:52:54:00:15:b5:c4 pubnet:52:54:00:15:b5:c4 <<<<
```

Run "ip link show" inside the VM, find the interface name whose MAC address (the link/ether field in the ip command output: link/ether 52:54:00:15:b5:c4 address in the describe-vm output "pubnet:52:54:00:15:b5:c4", in this case it is eth0. So eth0 is the interface attached to the vnetwork "pubnet".

```
# ip link show
...
3: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group default qlen 1000
    link/ether 52:54:00:15:b5:c4 brd ff:ff:ff:ff:ff:ff
```

Configure the network interface eth0 as you would on a Linux system.

### Using Browser User Interface to Create Virtual Machine Instance

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.

3. Click **Show VM Instances** to display the VM Instances page.

4. In the page, select the **Create VM Instance** option and click **Next**.

5. In the Create Virtual VM page, specify the following:

- VM Name: Name assigned to the VM instance that is unique within the name repository
- VM Storage Name: Name of the VM storage where the VM instance will be created
- Source Installation: The source from which you want to create the VM
- Preferred Node: Node where you want to run the VM instance
- Memory Size: Size of the memory to be allocated
- CPU Pool Name: Select the CPU Pool
- Number of vCPUs to Use: Number of virtual CPUs to be allocated

6. Click **Create**.

7. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.

8. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-`

**Parent topic:** [Managing Virtual Machines in KVM Deployment](#)

## Cloning a Virtual Machine in a KVM Deployment

Use ODACLI commands or the Browser User Interface to clone a virtual machine instance in a KVM deployment.

### Using ODACLI to Clone Virtual Machine Instance

Use the command `odacli clone-vm` to clone an existing virtual machine.

The name of the VM you create is defined by the command `odacli clone-vm -cn name`. Specify the source VM from which you want to clone.

Create a clone of a virtual machine named `vm1`, with the name `vm1_clone`. The cloned VM is created on the same storage as `vm1`.

```
# odacli clone-vm -n vm1 -cn vm1_clone
```

## Using Browser User Interface to Clone Virtual Machine Instance

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the VM Instance page.
4. Click **Create VM Instance**.
5. In the page, select the **Clone VM Instance** option and click **Next**.
6. In the Clone VM Instance page, specify the following:
  - Select the Source VM you want to clone
  - Specify the name and description of the cloned VM instance
7. Click **Create**.
8. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
9. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-`

**Parent topic:** [Managing Virtual Machines in KVM Deployment](#)

## Modifying a Virtual Machine in a KVM Deployment

Use ODACLI commands or the Browser User Interface to modify a virtual machine in a KVM deployment.

### Using ODACLI to Modify Virtual Machines

Use the command `odacli modify-vm` to modify a virtual machine.

Update the configuration of a VM named `vm1`, setting the CPU count to 6 and the maximum memory to 6 gigabytes on both the running VM and subse

```
# odacli modify-vm -n vm1 -c 6 -mm 6G --live --config
```

Modify a VM to attach a vnetwork:

```
# odacli modify-vm -n vm_name -avn vnetwork_name
```

- When an application VM is created, a default network interface is created by default in the VM. This interface is attached to the default kvm bridge interface is intended for convenient communication between the host and VM. This interface does not attach to any physical network interface at interface to use for any external communication.
- You must create a vnetwork using the command `odacli create-vnetwork` and attach the vnetwork (`odacli modify-vm -avn`) to the VM. T in the VM. Configure this network interface for your network requirement.
- The network interface name in the VM depends on the operating system.

### Using Browser User Interface to Modify Virtual Machines

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the VM Details page.
4. Click the **Show VM Instances** tab.

5. In the page, select a Virtual Machine, select the **Modify** option and click **Next**.
6. In the Modify VM page, specify Increment in size, and if you want to Auto Start, Set Failover, or Enable NUMA.
7. You can also modify the CPU Pool, Number of vCPUs to use, Memory Size, and attach and detach virtual disks and virtual networks.
8. Select if you want to save the configuration or apply the configuration to the running VM.
9. Click **Modify**.
10. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
11. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-`

**Parent topic:** [Managing Virtual Machines in KVM Deployment](#)

## Viewing Virtual Machines in a KVM Deployment

Use ODACLI commands or the Browser User Interface to view all virtual machines or details about a virtual machine in a KVM deployment.

### Using ODACLI to View Virtual Machines

The command `odacli describe-vm` displays details about a virtual machine. Use the command `odacli list-vms` to view all virtual machines in the

```
# odacli list-vms
```

```
# odacli describe-vm -n vm_name
```

### Using Browser User Interface to Create Virtual Machines

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the VM Details page.
4. Click the **Show VM Instances** tab to view the list of all configured virtual machines.
5. Click on a VM Instance to view the details of the virtual machine.

**Parent topic:** [Managing Virtual Machines in KVM Deployment](#)

## Starting and Stopping Virtual Machines in a KVM Deployment

Use ODACLI commands or the Browser User Interface to start or stop virtual machines in a KVM deployment.

### Using ODACLI to Start and Stop Virtual Machines

The command `odacli start-vm` starts a virtual machine. Use the command `odacli stop-vm` to stop a virtual machine in the deployment.

If `pref-node` is defined for the VM, then the VM starts on the `pref-node`. If `pref-node` is not defined for the VM, then the VM can start on any node. However, if you specify the node name `-n`, then the VM starts on the specified node, even if the preferred node is defined.

```
# odacli start-vm -n vm1
```

```
# odacli stop-vm -n vm1
```

### Using Browser User Interface to Start and Stop Virtual Machines

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the VM Details page.
4. Click the **Show VM Instances** tab to view the list of all configured virtual machines.
5. To start a virtual machine, click on a VM Instance, and then click **Start** and select the node to start the virtual machine.
6. To stop a virtual machine, click on a VM Instance, and then click **Stop** to stop the virtual machine. Click **Force Stop** to close all running processes a

**Parent topic:** [Managing Virtual Machines in KVM Deployment](#)

## Deleting a Virtual Machine in a KVM Deployment

Use ODACLI commands or the Browser User Interface to delete a virtual machine in a KVM deployment.

### Using ODACLI to Delete Virtual Machines

Use the command `odacli delete-vm` to delete a virtual machine.

Delete a virtual machine named `vm1`.

```
# odacli delete-vm -n vm1
```

### Using Browser User Interface to Modify Virtual Machines

Follow these steps:

1. Log into the Browser User Interface:

```
https://host-ip-address:7093/mgmt/index.html
```

2. Click the **Appliance** tab.
3. Click **VM Instances** to display the VM Details page.
4. Click the **Show VM Instances** tab.
5. In the page, select a VM Instance, select the **Delete** option.
6. Click Yes to confirm your choice.
7. When you submit the job, the job ID and a link to the job appears. Click the link to display the job status and details.
8. Validate that the job completed. You can track the job in the **Activity** tab in the Browser User Interface, or run the command `odacli describe-`

**Parent topic:** [Managing Virtual Machines in KVM Deployment](#)

## About Overcommitting Memory or CPUs in an Oracle Database Appliance KVM System

Understand performance and other considerations before overcommitting CPU and memory for application KVM.

Oracle Database Appliance does not restrict overcommitting of CPU and memory for application KVM. However, it is not recommended to overcommit C can lead to slow performance of the VMs and the host. Overcommitting memory can cause the system to run out of memory (OOM), which may lead to important system processes. Before you decide to overcommit CPU or memory, ensure that you test your systems. When sizing the application VMs, it i GB memory and 2 CPU cores for bare metal system KVM host when there is no database running on the bare metal system host.

Oracle Database Appliance validates oversubscription of shared CPU pools for DB systems as well as memory associated with DB systems. Starting with 19.15, the commands `odacli start-dbsystem`, `odacli modify-dbsystem`, `odacli create-dbsystem`, and `odacli modify-cpupool displa` overcommitting of the CPU pool or memory for the DB system. You can use the `--force/-f` option to allow overcommit of resources for DB systems. T not for application VMs.

**Parent topic:** [Managing an Oracle Database Appliance KVM Deployment](#)

## Example JSON File to Create a Single-Node DB System

Follow the JSON file example to create a JSON file to deploy a single-node DB System, with role separation, with the command `odacli create-dbsy:`

Use the example JSON file to create a file for your environment.



**Note:** It is important to review the readme and the examples carefully before creating your JSON file.

### Example 15-1 JSON File to Create a Single-Node DB system with Role Separation

The following is an example of a JSON file that creates a single-node DB 23ai DB system on Oracle Database Appliance bare metal system. To create a JSON, and update the version in the database section to DB 23ai version. Use the `odacli describe-dbsystem-image` command to view the sup example to create your JSON file, change the definitions to match your environment. The password must meet password requirements.

```
{
  "system": {
    "name": "odan1",
    "diskGroup": "RECO",
    "systemPassword": "xx",
    "timeZone": "Asia/Chita",
    "enableRoleSeparation": true,
    "customRoleSeparation": {
      "users": [
        {
          "name": "oracle",
          "id": 1712,
          "role": "oracleUser"
        },
        {
          "name": "grid",
          "id": 1713,
          "role": "gridUser"
        }
      ],
      "groups": [
        {
          "name": "oracle",
          "id": 1813,
          "role": "oinstall"
        },
        {
          "name": "grid",
          "id": 1814,
          "role": "dbaoper"
        },
        {
          "name": "dba",
          "id": 1815,
          "role": "dba"
        },
        {
          "name": "asmadmin",
          "id": 1816,
          "role": "asmadmin"
        },
        {
          "name": "asmoper",
          "id": 1817,
          "role": "asmoper"
        },
        {
          "name": "asmdba",
          "id": 1818,
          "role": "asmdba"
        }
      ]
    }
  },
  "shape": "dbs2"
},
"network": {
  "domainName": "exampledomain.com",
```

```

    "ntpServers": [
      "xx.xx.xx.xx"
    ],
    "dnsServers": [
      "xx.xx.xx.xx"
    ],
    "nodes": [
      {
        "name": "node1",
        "ipAddress": "xx.xx.xx.xx",
        "netmask": "xx.xx.xx.xx",
        "gateway": "xx.xx.xx.xx",
        "number": 0
      }
    ]
  },
  "grid": {
    "language": "en",
    "enableAFD": true
  },
  "database": {
    "name": "UEyuQgrD",
    "uniqueName": "UEyuQgrDU",
    "domainName": "us.oracle.com",
    "adminPassword": "xxx",
    "version": "23.5.0.24.07",
    "edition": "EE",
    "type": "SI",
    "dbClass": "OLTP",
    "shape": "odb2",
    "role": "PRIMARY",
    "targetNodeNumber": null,
    "enableUnifiedAuditing": true,
    "redundancy": "HIGH",
    "characterSet": {
      "characterSet": "AL32UTF8",
      "nlsCharacterSet": "AL16UTF16",
      "dbTerritory": "AMERICA",
      "dbLanguage": "AMERICAN"
    },
    "rmanBackupPassword": null,
    "level0BackupDay": null,
    "enableTDE": true,
    "tdePassword": "xxx",
    "isCdb": true,
    "pdbName": "UEyuQgrD_pdb",
    "pdbAdminUser": "UEyuQgrD_pdb_admin"
  }
}

```

#### Example 15-2 JSON File to Create a Single-Node DB system with Role Separation

The following is an example of a JSON file that creates a single-node DB system on Oracle Database Appliance bare metal platform. The example uses r example to create your JSON file, change the definitions to match your environment. The password must meet password requirements.

```

{
  "system": {
    "name": "test_example",
    "shape": "odb2",
    "customMemorySize": "24G",
    "timeZone": "America/Los_Angeles",
    "diskGroup": "DATA",
    "cpuPoolName": "shared_dbssystem_cpupool",
    "enableRoleSeparation": true,
    "customRoleSeparation": {
      "groups": [
        {
          "name": "oinstall",
          "id": 1001,
          "role": "oinstall"
        },
        {
          "name": "dbaoper",

```

```

        "id": 1002,
        "role": "dbaoper"
    },
    {
        "name": "dba",
        "id": 1003,
        "role": "dba"
    },
    {
        "name": "asmadmin",
        "id": 1004,
        "role": "asmadmin"
    },
    {
        "name": "asmoper",
        "id": 1005,
        "role": "asmoper"
    },
    {
        "name": "asmdba",
        "id": 1006,
        "role": "asmdba"
    }
],
"users": [
    {
        "name": "grid",
        "id": 1000,
        "role": "gridUser"
    },
    {
        "name": "oracle",
        "id": 1001,
        "role": "oracleUser"
    }
]
},
"database": {
    "name": "db19",
    "uniqueName": "db19",
    "domainName": "example.com",
    "version": "19.23.0.0.240416",
    "edition": "EE",
    "type": "SI",
    "dbClass": "OLTP",
    "shape": "odb2",
    "role": "PRIMARY",
    "targetNodeNumber": null,
    "enableDbConsole": false,
    "enableFlashStorage": false,
    "enableFlashCache": false,
    "enableUnifiedAuditing": true,
    "enableEEHA": true,
    "enableSEHA": false,
    "redundancy": null, <<< if diskgroup redundancy is FLEX, then database redundancy must be set to one of '
    "characterSet": {
        "characterSet": "AL32UTF8",
        "nlsCharacterSet": "AL16UTF16",
        "dbTerritory": "AMERICA",
        "dbLanguage": "ENGLISH"
    },
    "enableTDE": false,
    "isCdb": true,
    "pdbName": "pdb1",
    "pdbAdminUser": "pdbadmin"
},
"network": {
    "domainName": "test_domain",
    "ntpServers": [
        "xx.xxx.xx.xxx"
    ],
    "dnsServers": [

```



```

        "xx.xxx.xx.xxx"
    ],
    "nodes": [
        {
            "name": "node1",
            "ipAddress": "xx.xx.xx.xxx",
            "netmask": "xxx.xxx.xxx.x",
            "gateway": "xx.xx.xx.x",
            "number": 0
        }
    ],
    "publicVNetwork": "vnet1"
  },
  "grid": {
    "language": "en",
    "enableAFD": true
  }
}

```

### Example 15-3 JSON File to Create a Single-Node DB system without Role Separation

The following is an example of a JSON file that creates a single-node DB system on Oracle Database Appliance bare metal platform, without role separation. To create your JSON file, change the definitions to match your environment. The password must meet password requirements.

```

{
  "system": {
    "name": "test_example",
    "shape": "dbs2",
    "customMemorySize": "24G",
    "timeZone": "America/Los_Angeles",
    "diskGroup": "DATA",
    "cpuPoolName": "shared_dbsystem_cpupool",
  },
  "database": {
    "name": "db19",
    "uniqueName": "db19",
    "domainName": "example.com",
    "version": "19.23.0.0.240416",
    "edition": "EE",
    "type": "SI",
    "dbClass": "OLTP",
    "shape": "odb4",
    "role": "PRIMARY",
    "targetNodeNumber": null,
    "enableDbConsole": false,
    "enableUnifiedAuditing": true,
    "redundancy": "HIGH", <<< if diskgroup redundancy is FLEX, then database redundancy must be set to one of:
    "characterSet": {
      "characterSet": "AL32UTF8",
      "nlsCharacterSet": "AL16UTF16",
      "dbTerritory": "AMERICA",
      "dbLanguage": "ENGLISH"
    },
    "enableTDE": false,
    "isCdb": true,
    "pdbName": "pdb1",
    "pdbAdminUser": "pdbadmin"
  },
  "network": {
    "domainName": "test_domain",
    "ntpServers": [],
    "dnsServers": [
      "xx.xxx.xx.xxx"
    ],
    "nodes": [
      {
        "name": "node1",
        "ipAddress": "xx.xx.xx.xxx",
        "netmask": "xxx.xxx.xxx.x",
        "gateway": "xx.xx.xx.x",
        "number": 0
      }
    ]
  }
}

```

```

    },
    "publicVNetwork": "vnet1"
  },
  "grid": {
    "language": "en",
    "enableAFD": true
  }
}

```

#### Example 15-4 JSON File to Create a Single-Node DB system with the --template/-t option

The following is an example of a JSON file template sample that creates a single-node DB system on Oracle Database Appliance bare metal platform. If you run the `odacli create-dbsystem` command with the `--template/-t` option. When using the example to create your JSON file, change the environment. The password must meet password requirements.

```

{
  "system": {
    "name": "test_dbsystem",
    "shape": "dbs2",
    "customMemorySize": "24G",
    "timeZone": "America/Los_Angeles",
    "diskGroup": "DATA",
    "cpuPoolName": "shared_dbsystem_cpupool",
    "enableRoleSeparation": true,
    "customRoleSeparation": {
      "groups": [
        {
          "name": "oinstall",
          "id": 1001,
          "role": "oinstall"
        }
      ],
      "users": [
        {
          "name": "grid",
          "id": 1000,
          "role": "gridUser"
        },
        {
          "name": "oracle",
          "id": 1001,
          "role": "oracleUser"
        }
      ]
    }
  },
  "database": {
    "name": "dbtest",
    "uniqueName": "dbtest",
    "domainName": "example.com",
    "version": "19.16.0.0.220719",
    "edition": "EE",
    "type": "SI",
    "dbClass": "OLTP",
    "shape": "odb2",
    "role": "PRIMARY",
    "targetNodeNumber": "0",
    "enableDbConsole": false,
    "enableFlashStorage": false,
    "enableFlashCache": false,
    "enableUnifiedAuditing": true,
    "enableEEHA": true,
    "enableSEHA": false,
    "redundancy": "MIRROR",
    "characterSet": {
      "characterSet": "AL32UTF8",
      "nlsCharacterSet": "AL16UTF16",
      "dbTerritory": "AMERICA",
      "dbLanguage": "ENGLISH"
    }
  }
}

```

```

    "enableTDE": false,
    "isCdb": true,
    "pdbName": "pdb1",
    "pdbAdminUser": "pdbadmin"
  },
  "network": {
    "domainName": "example.com",
    "ntpServers": [
      "xxx.xxx.xxx.xxx"
    ],
    "dnsServers": [
      "xxx.xxx.xxx.xxx"
    ],
    "nodes": [
      {
        "name": "node1",
        "ipAddress": "xxx.xxx.xxx.xxx",
        "netmask": "xxx.xxx.xxx.xxx",
        "gateway": "xxx.xxx.xxx.xxx",
        "number": 0
      }
    ],
    "publicVNetwork": "vnet1"
  },
  "grid": {
    "language": "en",
    "enableAFD": true
  }
}

```

#### Example 15-5 JSON File to Create a Single-Node DB system with the --template-annotated/-ta option

The following is an example of a JSON file that creates a single-node DB system on Oracle Database Appliance bare metal platform with the --template-annotated/-ta option. To use the example to create your JSON file, change the definitions to match your environment. The password must meet password requirements.

```

{
  "system" : {
    "name" : "test_dbssystem",
    "shape" : "dbs2",
    "customMemorySize" : "24G",
    "timeZone" : "America/Los_Angeles",
    "diskGroup" : "DATA",
    "cpuPoolName" : "shared_dbssystem_cpupool",
    "enableRoleSeparation" : true,
    "customRoleSeparation" : {
      "groups" : [
        {
          "name" : "oinstall",
          "id" : 1001,
          "role" : "oinstall",
        }
      ],
      "users" : [
        {
          "name" : "grid",
          "id" : 1000,
          "role" : "gridUser",
        },
        {
          "name" : "oracle",
          "id" : 1001,
          "role" : "oracleUser",
        }
      ]
    }
  },
  "database" : {
    "name" : "dbtest",
    "uniqueName" : "dbtest",
    "domainName" : "example.com",
    "version" : "19.23.0.0.240416",
  }
}

```

---> The name for the DB System  
 ---> The shape for the DB System  
 ---> The memory size for the DB System  
 ---> The timezone for the DB System  
 ---> The ASM disk group to be used for the DB System vol  
 ---> The CPU Pool for the DB System  
 ---> Whether or not to enable Role Separation. If true, 1  
 ---> Name of the group to create, must be unique  
 ---> ID of the group to create, must be unique  
 ---> Role of the group  
 ---> Name of the user to create, must be unique  
 ---> ID of the user to create, must be unique  
 ---> Role of the user  
 ---> Name of the user to create, must be unique  
 ---> ID of the user to create, must be unique  
 ---> Role of the user  
 ---> Name for the DB System database  
 ---> The unique name for the DB System database  
 ---> The domain name that the DB System database will hav  
 ---> The version for the DB System database. Execute 'odacli

```
--> Enter 'EE' for Enterprise Edition, or 'SE' for Standalone
--> Enter a database deployment type. Allowed options: 'STANDALONE', 'RAC', 'SI'
--> The class for the DB System database. Standard Edition Only: 'SE1', 'SE2'. Enterprise Edition: 'EE1', 'EE2'
--> The DB System database shape, it must be same as the database shape
--> The role for the DB System database. Allowed options: 'PRIMARY', 'DATA GUARD', 'ARCHIVELOG'
--> Use '0' for Node0 and '1' for Node1. For RAC or RAC Data Guard, use '0' for Node0 and '1' for Node1
--> Whether or not to create dbconsole or EM express. By default, dbconsole will be created
--> Whether or not to enable the flash storage for the database
--> Whether or not to enable the flash cache for the database
--> Whether or not to enable unified auditing for the database
--> Whether or not to enable HA for EE SI database. enabled
--> Whether or not to enable HA for SE SI database. enabled
--> If diskgroup redundancy is FLEX, then database redundancy must be FLEX
--> The character set for the DB System database
--> The NLS character set for the DB System database
--> The territory for the DB System database
--> The language for the DB System database
--> Whether or not to enable TDE. Allowed options: <true>, <>false>
--> Enter 'true' if this database is container DB. For non-container DB, enter 'false'
--> Enter pdbName if isCdb parameter is 'true', use 'null' if isCdb parameter is 'false'
--> Enter PDB admin user name, use 'null' if isCdb parameter is 'false'
--> The domain name for the DB System network
--> The name used to configure the hostname
--> The IP address for this node
--> The mask of the network for this node
--> The gateway address for this node
--> 0 (Use 0 for the first node of the DB System instance)
--> The name of the public virtual network. The ipAddress is the IP address of the public virtual network
```

**Parent topic:** [Managing an Oracle Database Appliance KVM Deployment](#)

## Example JSON File to Create a High-Availability DB System

Follow the JSON file example to create a JSON file to deploy a high-availability DB System, with role separation, with the command `odacli create-dl`

Use the example JSON file to create a file for your environment.

### Example 15-6 JSON File to create a High-Availability DB system with Role Separation

The following is an example of a JSON file that creates a high-availability DB system on Oracle Database Appliance bare metal platform. The example uses the `dbconfig` command to create the database. To use this example to create your JSON file, change the definitions to match your environment. The password must meet password requirements.

```
{
  "system": {
    "name": "test_system",
    "shape": "dbs2",
    "customMemorySize": "24G",
    "timeZone": "America/Los_Angeles",
    "diskGroup": "DATA",
    "cpuPoolName": "shared_dbsystem_cpupool",
  },
}
```

```

"database": {
  "name": "dbtest",
  "uniqueName": "dbtest",
  "domainName": "test_domain",
  "version": "23.5.0.24.07",
  "edition": "EE",
  "type": "SI",
  "dbClass": "OLTP",
  "shape": "odb2",
  "role": "PRIMARY",
  "targetNodeNumber": "0",
  "enableDbConsole": false,
  "enableUnifiedAuditing": true,
  "enableEEHA": true,
  "redundancy": "MIRROR", <<< if diskgroup redundancy is FLEX, then database redundancy must be set to one
  "characterSet": {
    "characterSet": "AL32UTF8",
    "nlsCharacterSet": "AL16UTF16",
    "dbTerritory": "AMERICA",
    "dbLanguage": "AMERICAN"
  },
  "enableTDE": false,
  "isCdb": true,
  "pdbName": "pdb1",
  "pdbAdminUser": "pdbadmin"
},
"network": {
  "domainName": "test_domain",
  "ntpServers": [],
  "dnsServers": [
    "xx.xxx.xx.xxx"
  ],
  "nodes": [
    {
      "name": "node1",
      "ipAddress": "xx.xx.xxx.xx",
      "netmask": "xxx.xxx.xxx.x",
      "gateway": "xx.xx.xxx.x",
      "number": 0,
      "vipName": "node1-vip",
      "vipAddress": "xx.xx.xxx.xx"
    },
    {
      "name": "node2",
      "ipAddress": "xx.xx.xxx.xx",
      "netmask": "xxx.xxx.xxx.x",
      "gateway": "xx.xx.xxx.x",
      "number": 1,
      "vipName": "node2-vip",
      "vipAddress": "xx.xx.xxx.xx"
    }
  ],
  "publicVNetwork": "vnet1",
  "scanName": "test-scan",
  "scanIps": [
    "xx.xx.xxx.xx",
    "xx.xx.xxx.xx"
  ]
},
"grid": {
  "language": "en",
  "enableAFD": true
}
}

```

#### Example 15-7 JSON File to create High-Availability DB system without Role Separation

The following is an example of a JSON file that creates high-availability DB system on Oracle Database Appliance bare metal platform, without role separation. To create your JSON file, change the definitions to match your environment. The password must meet password requirements.

```

{
  "system": {
    "name": "test_system",
    "shape": "dbs2",
    "customMemorySize": "24G",
    "timeZone": "America/Los_Angeles",
    "diskGroup": "DATA",
    "cpuPoolName": "shared_dbsystem_cpupool",
  },
  "database": {
    "name": "dbtest",
    "uniqueName": "dbtest",
    "domainName": "test_domain",
    "version": "19.23.0.0.240416",
    "edition": "EE",
    "type": "SI",
    "dbClass": "OLTP",
    "shape": "odb2",
    "role": "PRIMARY",
    "targetNodeNumber": "0",
    "enableDbConsole": false,
    "enableUnifiedAuditing": true,
    "enableEEHA": true,
    "redundancy": "MIRROR", <<< if diskgroup redundancy is FLEX, then database redundancy must be set to one
    "characterSet": {
      "characterSet": "AL32UTF8",
      "nlsCharacterSet": "AL16UTF16",
      "dbTerritory": "AMERICA",
      "dbLanguage": "AMERICAN"
    },
    "enableTDE": false,
    "isCdb": true,
    "pdbName": "pdb1",
    "pdbAdminUser": "pdbadmin"
  },
  "network": {
    "domainName": "test_domain",
    "ntpServers": [],
    "dnsServers": [
      "xx.xxx.xx.xxx"
    ],
    "nodes": [
      {
        "name": "node1",
        "ipAddress": "xx.xx.xxx.xx",
        "netmask": "xxx.xxx.xxx.x",
        "gateway": "xx.xx.xxx.x",
        "number": 0,
        "vipName": "node1-vip",
        "vipAddress": "xx.xx.xxx.xx"
      },
      {
        "name": "node2",
        "ipAddress": "xx.xx.xxx.xx",
        "netmask": "xxx.xxx.xxx.x",
        "gateway": "xx.xx.xxx.x",
        "number": 1,
        "vipName": "node2-vip",
        "vipAddress": "xx.xx.xxx.xx"
      }
    ],
    "publicVNetwork": "vnet1",
    "scanName": "test-scan",
    "scanIps": [
      "xx.xx.xxx.xx",
      "xx.xx.xxx.xx"
    ]
  },
  "grid": {
    "language": "en",
    "enableAFD": true
  }
}

```

```
}
}
```

### Example 15-8 JSON File to Create High-Availability DB system with Role Separation

The following is an example of a JSON file that creates high-availability DB system on Oracle Database Appliance bare metal platform. The example use example to create your JSON file, change the definitions to match your environment. The password must meet password requirements.

```
{
  "system": {
    "name": "test_system",
    "shape": "dbs2",
    "customMemorySize": "24G",
    "timeZone": "America/Los_Angeles",
    "diskGroup": "DATA",
    "cpuPoolName": "shared_dbssystem_cpupool",
    "enableRoleSeparation": true,
    "customRoleSeparation": {
      "groups": [
        {
          "name": "oinstall",
          "id": 1001,
          "role": "oinstall"
        },
        {
          "name": "dbaoper",
          "id": 1002,
          "role": "dbaoper"
        },
        {
          "name": "dba",
          "id": 1003,
          "role": "dba"
        },
        {
          "name": "asmadmin",
          "id": 1004,
          "role": "asmadmin"
        },
        {
          "name": "asmoper",
          "id": 1005,
          "role": "asmoper"
        },
        {
          "name": "asmdba",
          "id": 1006,
          "role": "asmdba"
        }
      ],
      "users": [
        {
          "name": "grid",
          "id": 1000,
          "role": "gridUser"
        },
        {
          "name": "oracle",
          "id": 1001,
          "role": "oracleUser"
        }
      ]
    }
  },
  "database": {
    "name": "dbtest",
    "uniqueName": "dbtest",
    "domainName": "test_domain",
    "version": "19.23.0.0.240416",
    "edition": "EE",
    "type": "SI",
  }
}
```

```

    "dbClass": "OLTP",
    "shape": "odb2",
    "role": "PRIMARY",
    "targetNodeNumber": "0",
    "enableDbConsole": false,
    "enableUnifiedAuditing": true,
    "enableEEHA": true,
    "redundancy": null, <<< if diskgroup redundancy is FLEX, then database redundancy must be set to one of
    "characterSet": {
        "characterSet": "AL32UTF8",
        "nlsCharacterSet": "AL16UTF16",
        "dbTerritory": "AMERICA",
        "dbLanguage": "AMERICAN"
    },
    "enableTDE": false,
    "isCdb": true,
    "pdbName": "pdb1",
    "pdbAdminUser": "pdbadmin"
  },
  "network": {
    "domainName": "test_domain",
    "ntpServers": [],
    "dnsServers": [
        "xx.xxx.xx.xxx"
    ],
    "nodes": [
        {
            "name": "node1",
            "ipAddress": "xx.xx.xxx.xx",
            "netmask": "xxx.xxx.xxx.x",
            "gateway": "xx.xx.xxx.x",
            "number": 0,
            "vipName": "node1-vip",
            "vipAddress": "xx.xx.xxx.xx"
        },
        {
            "name": "node2",
            "ipAddress": "xx.xx.xxx.xx",
            "netmask": "xxx.xxx.xxx.x",
            "gateway": "xx.xx.xxx.x",
            "number": 1,
            "vipName": "node2-vip",
            "vipAddress": "xx.xx.xxx.xx"
        }
    ],
    "publicVNetwork": "vnet1",
    "scanName": "test-scan",
    "scanIps": [
        "xx.xx.xxx.xx",
        "xx.xx.xxx.xx"
    ]
  },
  "grid": {
    "language": "en",
    "enableAFD": true
  }
}

```

#### Example 15-9 JSON File to Create High-Availability DB system with Standard Edition High-Availability Enabled

The following is an example of a JSON file that creates high-availability DB system on Oracle Database Appliance bare metal platform. When using the c change the definitions to match your environment. The password must meet password requirements.

```

{
  "system": {
    "name": "test_system",
    "shape": "dbs2",
    "customMemorySize": "24G",
    "timeZone": "America/Los_Angeles",
    "diskGroup": "DATA",
    "cpuPoolName": "shared_dbsystem_cpupool",

```



```

    },
    "database": {
      "name": "dbtest",
      "uniqueName": "dbtest",
      "domainName": "test_domain",
      "version": "19.23.0.0.240416",
      "edition": "SE",
      "type": "SI",
      "dbClass": "OLTP",
      "shape": "odb2",
      "role": "PRIMARY",
      "targetNodeNumber": "0",
      "enableDbConsole": false,
      "enableUnifiedAuditing": true,
      "enableSEHA": true,
      "redundancy": "MIRROR", <<< if diskgroup redundancy is FLEX, then database redundancy must be set to one
      "characterSet": {
        "characterSet": "AL32UTF8",
        "nlsCharacterSet": "AL16UTF16",
        "dbTerritory": "AMERICA",
        "dbLanguage": "AMERICAN"
      },
      "enableTDE": false,
      "isCdb": true,
      "pdbName": "pdb1",
      "pdbAdminUser": "pdbadmin"
    },
    "network": {
      "domainName": "test_domain",
      "ntpServers": [],
      "dnsServers": [
        "xx.xxx.xx.xxx"
      ],
      "nodes": [
        {
          "name": "node1",
          "ipAddress": "xx.xx.xxx.xx",
          "netmask": "xxx.xxx.xxx.x",
          "gateway": "xx.xx.xxx.x",
          "number": 0,
          "vipName": "node1-vip",
          "vipAddress": "xx.xx.xxx.xx"
        },
        {
          "name": "node2",
          "ipAddress": "xx.xx.xxx.xx",
          "netmask": "xxx.xxx.xxx.x",
          "gateway": "xx.xx.xxx.x",
          "number": 1,
          "vipName": "node2-vip",
          "vipAddress": "xx.xx.xxx.xx"
        }
      ],
      "publicVNetwork": "vnet1",
      "scanName": "test-scan",
      "scanIps": [
        "xx.xx.xxx.xx",
        "xx.xx.xxx.xx"
      ]
    },
    "grid": {
      "language": "en"
      "enableAFD": true
    }
  }
}

```

#### Example 15-10 JSON File to Create High-Availability DB system with Oracle RAC Database

The following is an example of a JSON file that creates high-availability DB system on Oracle Database Appliance bare metal platform. When using the `oacore` command, change the definitions to match your environment. The password must meet password requirements.

```

{
  "system": {
    "name": "test_system",
    "shape": "dbs2",
    "customMemorySize": "24G",
    "timeZone": "America/Los_Angeles",
    "diskGroup": "DATA",
    "cpuPoolName": "shared_dbsystem_cpupool",
  },
  "database": {
    "name": "dbtest",
    "uniqueName": "dbtest",
    "domainName": "test_domain",
    "version": "19.23.0.0.240416",
    "edition": "EE",
    "type": "RAC",
    "dbClass": "OLTP",
    "shape": "odb2",
    "role": "PRIMARY",
    "targetNodeNumber": "0",
    "enableDbConsole": false,
    "enableUnifiedAuditing": true,
    "redundancy": "MIRROR", <<< if diskgroup redundancy is FLEX, then database redundancy must be set to one
    "characterSet": {
      "characterSet": "AL32UTF8",
      "nlsCharacterSet": "AL16UTF16",
      "dbTerritory": "AMERICA",
      "dbLanguage": "AMERICAN"
    },
    "enableTDE": false,
    "isCdb": true,
    "pdbName": "pdb1",
    "pdbAdminUser": "pdbadmin"
  },
  "network": {
    "domainName": "test_domain",
    "ntpServers": [],
    "dnsServers": [
      "xx.xxx.xx.xxx"
    ],
    "nodes": [
      {
        "name": "node1",
        "ipAddress": "xx.xx.xxx.xx",
        "netmask": "xxx.xxx.xxx.x",
        "gateway": "xx.xx.xxx.x",
        "number": 0,
        "vipName": "node1-vip",
        "vipAddress": "xx.xx.xxx.xx"
      },
      {
        "name": "node2",
        "ipAddress": "xx.xx.xxx.xx",
        "netmask": "xxx.xxx.xxx.x",
        "gateway": "xx.xx.xxx.x",
        "number": 1,
        "vipName": "node2-vip",
        "vipAddress": "xx.xx.xxx.xx"
      }
    ],
    "publicVNetwork": "vnet1",
    "scanName": "test-scan",
    "scanIps": [
      "xx.xx.xxx.xx",
      "xx.xx.xxx.xx"
    ]
  },
  "grid": {
    "language": "en"
    "enableAFD": true
  }
}

```

**Example 15-11 JSON File to Create A High-Availability DB system with the --template/-t option**

The following is an example of a JSON file that creates high-availability DB system on Oracle Database Appliance bare metal platform. This template `sarodacli create-dbsystem` command with the `--template/-t` option. When using the example to create your JSON file, change the definitions to `r` password must meet password requirements.

```
{
  "system": {
    "name": "test_dbsystem",
    "shape": "dbs2",
    "customMemorySize": "24G",
    "timeZone": "America/Los_Angeles",
    "diskGroup": "DATA",
    "cpuPoolName": "shared_dbsystem_cpupool",
    "enableRoleSeparation": true,
    "customRoleSeparation": {
      "groups": [
        {
          "name": "oinstall",
          "id": 1001,
          "role": "oinstall"
        }
      ],
      "users": [
        {
          "name": "grid",
          "id": 1000,
          "role": "gridUser"
        },
        {
          "name": "oracle",
          "id": 1001,
          "role": "oracleUser"
        }
      ]
    }
  },
  "database": {
    "name": "dbtest",
    "uniqueName": "dbtest",
    "domainName": "example.com",
    "version": "19.23.0.0.240416",
    "edition": "EE",
    "type": "SI",
    "dbClass": "OLTP",
    "shape": "odb2",
    "role": "PRIMARY",
    "targetNodeNumber": "0",
    "enableDbConsole": false,
    "enableFlashStorage": false,
    "enableFlashCache": false,
    "enableUnifiedAuditing": true,
    "enableEEHA": true,
    "enableSEHA": false,
    "redundancy": "MIRROR",
    "characterSet": {
      "characterSet": "AL32UTF8",
      "nlsCharacterSet": "AL16UTF16",
      "dbTerritory": "AMERICA",
      "dbLanguage": "AMERICAN"
    },
    "enableTDE": false,
    "isCdb": true,
    "pdbName": "pdb1",
    "pdbAdminUser": "pdbadmin"
  },
  "network": {
    "domainName": "example.com",
    "ntpServers": [
```

```

        "xxx.xxx.xxx.xxx"
    ],
    "dnsServers": [
        "xxx.xxx.xxx.xxx"
    ],
    "nodes": [
        {
            "name": "node1",
            "ipAddress": "xxx.xxx.xxx.xxx",
            "netmask": "xxx.xxx.xxx.xxx",
            "gateway": "xxx.xxx.xxx.xxx",
            "number": 0,
            "vipName": "node1-vip",
            "vipAddress": "xxx.xxx.xxx.xxx"
        },
        {
            "name": "node2",
            "ipAddress": "xxx.xxx.xxx.xxx",
            "netmask": "xxx.xxx.xxx.xxx",
            "gateway": "xxx.xxx.xxx.xxx",
            "number": 1,
            "vipName": "node2-vip",
            "vipAddress": "xxx.xxx.xxx.xxx"
        }
    ],
    "publicVNetwork": "vnet1",
    "scanName": "dbtest-scan",
    "scanIps": [
        "xxx.xxx.xxx.xxx",
        "xxx.xxx.xxx.xxx"
    ]
  },
  "grid": {
    "language": "en",
    "enableAFD": true
  }
}

```

#### Example 15-12 JSON File to Create A High-Availability DB system with the --template/-t option

The following is an example of a JSON file that creates high-availability DB system on Oracle Database Appliance bare metal platform. This template is used with the `odacli create-dbsystem` command with the `--template/t` option. When using the example to create your JSON file, change the definitions to meet password requirements.

```

{
  "system": {
    "name": "test_dbsystem",
    "shape": "dbs2",
    "customMemorySize": "24G",
    "timeZone": "America/Los_Angeles",
    "diskGroup": "DATA",
    "cpuPoolName": "shared_dbsystem_cpupool",
    "enableRoleSeparation": true,
    "customRoleSeparation": {
      "groups": [
        {
          "name": "oinstall",
          "id": 1001,
          "role": "oinstall"
        }
      ],
      "users": [
        {
          "name": "grid",
          "id": 1000,
          "role": "gridUser"
        },
        {
          "name": "oracle",
          "id": 1001,
          "role": "oracleUser"
        }
      ]
    }
  }
}

```

```

    }
  }
},
"database": {
  "name": "dbtest",
  "uniqueName": "dbtest",
  "domainName": "example.com",
  "version": "19.23.0.0.240416",
  "edition": "EE",
  "type": "SI",
  "dbClass": "OLTP",
  "shape": "odb2",
  "role": "PRIMARY",
  "targetNodeNumber": "0",
  "enableDbConsole": false,
  "enableFlashStorage": false,
  "enableFlashCache": false,
  "enableUnifiedAuditing": true,
  "enableEEHA": true,
  "enableSEHA": false,
  "redundancy": "MIRROR",
  "characterSet": {
    "characterSet": "AL32UTF8",
    "nlsCharacterSet": "AL16UTF16",
    "dbTerritory": "AMERICA",
    "dbLanguage": "AMERICAN"
  },
  "enableTDE": false,
  "isCdb": true,
  "pdbName": "pdb1",
  "pdbAdminUser": "pdbadmin"
},
"network": {
  "domainName": "example.com",
  "ntpServers": [
    "xxx.xxx.xxx.xxx"
  ],
  "dnsServers": [
    "xxx.xxx.xxx.xxx"
  ],
  "nodes": [
    {
      "name": "node1",
      "ipAddress": "xxx.xxx.xxx.xxx",
      "netmask": "xxx.xxx.xxx.xxx",
      "gateway": "xxx.xxx.xxx.xxx",
      "number": 0,
      "vipName": "node1-vip",
      "vipAddress": "xxx.xxx.xxx.xxx"
    },
    {
      "name": "node2",
      "ipAddress": "xxx.xxx.xxx.xxx",
      "netmask": "xxx.xxx.xxx.xxx",
      "gateway": "xxx.xxx.xxx.xxx",
      "number": 1,
      "vipName": "node2-vip",
      "vipAddress": "xxx.xxx.xxx.xxx"
    }
  ],
  "publicVNetwork": "vnet1",
  "scanName": "dbtest-scan",
  "scanIps": [
    "xxx.xxx.xxx.xxx",
    "xxx.xxx.xxx.xxx"
  ]
},
"grid": {
  "language": "en",
  "enableAFD": true
}
}

```

**Example 15-13 JSON File to Create A High-Availability DB system with the --template-annotated/-ta option**

The following is an example of a JSON file that creates high-availability DB system on Oracle Database Appliance bare metal platform. This template sar  
 odacli create-dbsystem command with the --template-annotated/-ta option. When using the example to create your JSON file, change the  
 environment. The password must meet password requirements.

```
{
  "system" : {
    "name" : "test_dbsystem",
    "shape" : "dbs2",
    "customMemorySize" : "24G",
    "timeZone" : "America/Los_Angeles",
    "diskGroup" : "DATA",
    "cpuPoolName" : "shared_dbsystem_cpupool",
    "useReservedCores" : false,
    "enableRoleSeparation" : true,
    "customRoleSeparation" : {
      "groups" : [
        {
          "name" : "oinstall",
          "id" : 1001,
          "role" : "oinstall",
        }
      ],
      "users" : [
        {
          "name" : "grid",
          "id" : 1000,
          "role" : "gridUser",
        },
        {
          "name" : "oracle",
          "id" : 1001,
          "role" : "oracleUser",
        }
      ]
    }
  },
  "database" : {
    "name" : "dbtest",
    "uniqueName" : "dbtest",
    "domainName" : "example.com",
    "version" : "19.23.0.0.240416",
    "edition" : "EE",
    "type" : "SI",
    "dbClass" : "OLTP",
    "shape" : "odb2",
    "role" : "PRIMARY",
    "dbStorage" : "ASM",
    "level0BackupDay" : "wednesday",
    "targetNodeNumber" : "0",
    "enableDbConsole" : false,
    "enableFlashStorage" : false,
    "enableFlashCache" : false,
    "enableUnifiedAuditing" : true,
    "enableEEHA" : true,
    "enableSEHA" : false,
    "redundancy" : "MIRROR",
    "characterSet" : {
      "characterSet" : "AL32UTF8",
      "nlsCharacterSet" : "AL16UTF16",
      "dbTerritory" : "AMERICA",
      "dbLanguage" : "AMERICAN",
    },
    "enableTDE" : false,
    "isCdb" : true,
    "pdbName" : "pdb1",
    "pdbAdminUser" : "pdbadmin",
  },
  "network" : {
```

---> The name for the DB System  
 ---> The shape for the DB System  
 ---> The memory size for the DB System  
 ---> The timezone for the DB System  
 ---> The ASM disk group to be used for the DB System vol  
 ---> The CPU Pool for the DB System  
 ---> Whether or not to use reserved CPU cores. Allowed o  
 ---> Whether or not to enable Role Separation. If true, i  
 ---> Name of the group to create, must be unique  
 ---> ID of the group to create, must be unique  
 ---> Role of the group  
 ---> Name of the user to create, must be unique  
 ---> ID of the user to create, must be unique  
 ---> Role of the user  
 ---> Name of the user to create, must be unique  
 ---> ID of the user to create, must be unique  
 ---> Role of the user  
 ---> Name for the DB System database  
 ---> The unique name for the DB System database  
 ---> The domain name that the DB System database will hav  
 ---> The version for the DB System database. Execute 'oda  
 ---> Enter 'EE' for Enterprise Edition, or 'SE' for Stand  
 ---> Enter a database deployment type. Allowed options: <tr  
 ---> The class for the DB System database. Standard Edit  
 ---> The DB System database shape, it must be same as the  
 ---> The role for the DB System database. Allowed option  
 ---> Storage type of the DB System database. The default  
 ---> Backup day of the DB System database. Allowed option  
 ---> Use '0' for Node0 and '1' for Node1. For RAC or RAC  
 ---> Whether or not to create dbconsole or EM express. B  
 ---> Whether or not to enable the flash storage for the l  
 ---> Whether or not to enable the flash cache for the DB  
 ---> Whether or not to enable unified auditing for the DI  
 ---> Whether or not to enable HA for EE SI database. enal  
 ---> Whether or not to enable HA for SE SI database. enal  
 ---> If diskgroup redundancy is FLEX, then database redu  
 ---> The character set for the DB System database  
 ---> The NLS character set for the DB System database  
 ---> The territory for the DB System database  
 ---> The language for the DB System database  
 ---> Whether or not to enable TDE. Allowed options: <true  
 ---> Enter 'true' if this database is container DB. For  
 ---> Enter pdbName if isCdb parameter is 'true', use 'nul  
 ---> Enter PDB admin user name, use 'null' if isCdb parame

```

"domainName" : "example.com",
"ntpServers" : [
    "xxx.xxx.xxx.xxx"
],
"dnsServers" : [
    "xxx.xxx.xxx.xxx"
],
"nodes" : [
    {
        "name" : "node1",
        "ipAddress" : "xxx.xxx.xxx.xxx",
        "netmask" : "xxx.xxx.xxx.xxx",
        "gateway" : "xxx.xxx.xxx.xxx",
        "number" : 0,
        "vipName" : "node1-vip",
        "vipAddress" : "xxx.xxx.xxx.xxx",
    },
    {
        "name" : "node2",
        "ipAddress" : "xxx.xxx.xxx.xxx",
        "netmask" : "xxx.xxx.xxx.xxx",
        "gateway" : "xxx.xxx.xxx.xxx",
        "number" : 1,
        "vipName" : "node2-vip",
        "vipAddress" : "xxx.xxx.xxx.xxx",
    }
],
"publicVNetwork" : "vnet1",
"scanName" : "dbtest-scan",
"scanIps" : [
    "xxx.xxx.xxx.xxx",
    "xxx.xxx.xxx.xxx"
]
},
"grid" : {
    "language" : "en",
    "enableAFD" : true,
}
}

```

---> The domain name for the DB System network

---> The name used to configure the hostname

---> The IP address for this node

---> The mask of the network for this node

---> The gateway address for this node

---> 0 (Use 0 for the first node of the DB System instance)

---> The VIP name for this first node, only for HA case

---> The VIP address for this first node, only for HA case

---> The name used to configure the hostname

---> The IP address for this node

---> The mask of the network for this node

---> The gateway address for this node

---> 1 (Use 1 for the second node of the DB System instance)

---> The VIP name for this second node, only for HA case

---> The VIP address for this second node, only for HA case

---> The name of the public virtual network. The ipAddress is the IP address of the public virtual network.

---> The scan name of the DB System. It should resolve to the IP address of the public virtual network.

---> The language used for GI (Grid Infrastructure) installation.

---> Whether or not to enable ASM Filter Driver, enabled

Parent topic: [Managing an Oracle Database Appliance KVM Deployment](#)

[< Previous Page](#)

[Next Page >](#)