ORACLE®

Oracle® Real Application Clusters

Administration and Deployment Guide

12*c* Release 1 (12.1)

E48838-12

July 2017

ORACLE®

Oracle Real Application Clusters Administration and Deployment Guide, 12*c* Release 1 (12.1)

E48838-12

Primary Author: Richard Strohm

Contributing Authors: Janet Stern

Contributors: Troy Anthony, Lance Ashdown, Ram Avudaiappan, Prasad Bagal, Mark Bauer, Anand Beldalker, Eric Belden, Gajanan Bhat, David Brower, George Claborn, Maria Colgan, Carol Colrain, Jonathan Creighton, Rajesh Dasari, Steve Fogel, Richard Frank, GP Prabhaker Gongloor, Wei Hu, Yong Hu, Dominique Jeunot, Sameer Joshi, Raj K. Kammend, Sana Karam, Roland Knapp, Karen Li, Barb Lundhild, Venkat Maddali, Bill Manry, John McHugh, Saar Maoz, Matthew Mckerley, Markus Michalewicz, Anil Nair, Philip Newlan, Michael Nowak, Muthu Olagappan, Bharat Paliwal, Hanlin Qian, Mark Ramacher, Kevin Reardon, Dipak Saggi, Daniel Semler, Ara Shakian, Cathy Shea, Khethavath P. Singh, Kesavan Srinivasan, Leo Tominna, Peter Wahl, Tak Wang, Richard Wessman, Douglas Williams, Mike Zampiceni, Michael Zoll

Contents

14 Converting Single-Instance Oracle Databases to Oracle RAC and Oracle RAC One Node

A Server Control Utility Reference

B Troubleshooting Oracle RAC

Glossary

Index

List of Tables

Preface

The *Oracle Real Application Clusters Administration and Deployment Guide* describes the Oracle Real Application Clusters (Oracle RAC) architecture and provides an overview of this product, including Oracle Real Application Clusters One Node (Oracle RAC One Node). This book also describes administrative and deployment topics for Oracle RAC.

Information in this manual applies to Oracle RAC as it runs on all platforms unless otherwise noted. In addition, the content of this manual supplements administrative and deployment topics for noncluster Oracle databases that appear in other Oracle documentation. Where necessary, this manual refers to platform-specific documentation. This Preface contains these topics:

- Audience

- Documentation Accessibility

- Related Documents

- Conventions

Audience

The *Oracle Real Application Clusters Administration and Deployment Guide* is intended for database administrators, network administrators, and system administrators who perform the following tasks:

- Install and configure an Oracle RAC database

- Administer and manage Oracle RAC databases

- Manage and troubleshoot clusters and networks that use Oracle RAC

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Related Documents

This book, the *Oracle Real Application Clusters Administration and Deployment Guide*, provides administration and application deployment information that is specific to Oracle RAC. The discussions herein assume a knowledge of Oracle Clusterware.

For more information, see the Oracle resources listed in this section.

- *Oracle Database 2 Day + Real Application Clusters Guide*

  This task-oriented guide helps you understand the basic steps required to install, configure, and administer Oracle Clusterware and Oracle RAC on a two-node system using Red Hat Linux system.

- *Oracle Clusterware Administration and Deployment Guide*

  This is an essential companion book that describes Oracle Clusterware components such as the voting disks and the Oracle Cluster Registry (OCR).

- Platform-specific Oracle Clusterware and Oracle RAC installation guides

  Each platform-specific Oracle Database installation media contains a copy of an Oracle Clusterware and Oracle RAC platform-specific installation and configuration guide in HTML and PDF formats. These installation books contain the preinstallation, installation, and postinstallation information for the various UNIX, Linux, and Windows platforms on which Oracle Clusterware and Oracle RAC operate.

- *Oracle Database 2 Day DBA*

- *Oracle Database Administrator's Guide*

- *Oracle Database Net Services Administrator's Guide*

- *Oracle Database Platform Guide for Microsoft Windows*

- *Oracle Database Administrator's Reference for Linux and UNIX-Based Operating Systems*

- *Oracle Database 11g Administrator's Reference Release 1 (11.1) for UNIX Systems: AIX Systems, HP-UX, Linux, and the Solaris Operating System (SPARC)*

  > **Note:**
  >
  > Additional information for this release may be available in the Oracle Database 12*c* README or Release Notes. If these documents are available for this release, then they are on your Oracle product installation media.

Database error messages descriptions are available online or by way of a Tahiti documentation search.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Changes in This Release for Oracle Real Application Clusters Administration and Deployment Guide

This preface contains:

-

Changes in Oracle Real Application Clusters 12*c* Release 1 (12.1)

The following are changes in *Oracle Real Application Clusters Administration and Deployment Guide* for Oracle Real Application Clusters (Oracle RAC) 12*c*:

-

-

-

-

Changes in Oracle Real Application Clusters 12*c* Release 1 (12.1.0.2)

The following features are new in this release:

- **In-Memory Column Store**

  The In-Memory Column Store is an optional area in the SGA that stores whole tables, table partitions, and individual columns in a compressed columnar format. The database uses special techniques, including SIMD vector processing, to scan columnar data extremely rapidly. The In-Memory Column Store is a supplement to, rather than a replacement for, the database buffer cache.

  > **See Also:**
  >
  > *Oracle Database Data Warehousing Guide* for more information

- **In-Memory Transaction Manager**

  The In-Memory Transaction Manager is an independent engine that automatically provides read consistency for transactions that apply changes to the In-Memory Column Store. This engine is necessary because tables and partitions residing in

the In-Memory Column Store are stored in columnar format in memory and in row-major format in the data files and database buffer cache.

> **See Also:**
>
> *Oracle Database Concepts* for more information

- **Rapid Home Provisioning**

  Rapid Home Provisioning enables you to deploy Oracle homes based on images stored in a catalog of precreated software homes.

  > **See Also:**
  >
  > *Oracle Clusterware Administration and Deployment Guide* for more information

- **Full Database In-Memory Caching**

  In this release you can cache an entire database in memory. Use this feature when the buffer cache size of each instance is greater than the size of the whole database. In Oracle RAC systems, for well-partitioned applications, you can use this feature when the combined buffer caches of all database instances (with some extra space to handle duplicate cached blocks between instances) is greater than the database size.

  > **See Also:**
  >
  > *Oracle Database Performance Tuning Guide* for more information

- **Memory Guard Does Not Require Oracle Database QoS Management to be Active**

  With this release, Memory Guard is enabled by default independent of whether you use Oracle Database Quality of Service Management (Oracle Database QoS Management). Memory Guard detects memory stress on a node and causes new sessions to be directed to other instances until the existing workload drains and frees memory. When free memory increases on the node, then services are enabled again to automatically accept new connections.

Changes in Oracle Real Application Clusters 12*c* Release 1 (12.1.0.1)

The following features are new in this release:

- **Application Continuity**

  Before this release, application developers were required to deal explicitly with outages of the underlying software, hardware, and communications layers if they wanted to mask outages from end users.

  In Oracle Database 10*g*, Fast Application Notification (FAN) quickly delivered exception conditions to applications. However, neither FAN nor earlier Oracle technology reported the outcome of the last transaction to the application, or recovered the in-progress request from an application perspective. As a result, outages were exposed leading to user inconvenience and lost revenue. Users could unintentionally make duplicate purchases and submit multiple payments

for the same invoice. In the problematic cases, the administrator needed to reboot the mid-tier to deal with the incoming problems this caused.

Application Continuity is an application-independent feature that attempts to recover incomplete requests from an application perspective and masks many system, communication, hardware failures, and storage outages from the end user.

See Also:

– *Oracle Database Concepts* for more information

– "How Application Continuity Works for Applications"

- **Transaction Guard for Java**

  This feature exposes the new Application Continuity infrastructure to Java. It provides support for:

  – At-most-once transaction execution protocol, such as transaction idempotence

  – API for retrieving logical transaction ID (LTXID)

  – Attribute to get Connection or Session status

  See Also:

  Oracle Database JDBC Developer's Guide for more information

- **Transaction Idempotence**

  This feature delivers a general purpose, application-independent infrastructure that enables recovery of work from an application perspective and masks most system, communication, and hardware failures from the user. Transaction idempotence ensures that your transactions are executed on time and, at most, once.

  See Also:

  Oracle Database Development Guide for more information

- **Oracle Flex Clusters**

  Large clusters consisting of, potentially, thousands of nodes, provide a platform for Oracle RAC to support massive parallel query operations.

  See Also:

  Oracle Clusterware Administration and Deployment Guide for more information about Oracle Flex Clusters

- **Shared Oracle ASM Password File in a Disk Group**

This feature implements the infrastructure needed to address the bootstrapping issues of storing an Oracle Automatic Storage Management (Oracle ASM) shared password file in an Oracle ASM disk group.

> **See Also:**
>
> *Oracle Automatic Storage Management Administrator's Guide*

- **Global Data Services**

  Similar to the way Oracle RAC supports a database service and enables service-level workload management across database instances in a cluster, Global Data Services provides Oracle RAC-like connect-time and run-time load balancing, failover, and centralized service management for a set of replicated databases that offer common services. The set of databases can include Oracle RAC and nonclustered Oracle databases interconnected with Oracle Data Guard, Oracle GoldenGate, or any other replication technology.

  > **See Also:**
  >
  > *Oracle Database Global Data Services Concepts and Administration Guide* for more information

- **Shared Grid Naming Service**

  One instance of Grid Naming Service (GNS) can service any number of clusters.

  > **See Also:**
  >
  > *Oracle Clusterware Administration and Deployment Guide* for more information

- **What-If Command Evaluation for Oracle RAC**

  This feature of Oracle Clusterware improves resource management and availability, through a mechanism that provides a policy response to a hypothetical planned or unplanned event, without modifying the state of the system.

  In Oracle RAC, enhancements to SRVCTL aid you in determining the impact of certain commands before you run them to determine the potential impact of the command.

  > **See Also:**
  >
  > - "SRVCTL Usage Information" for a list of SRVCTL commands with What-If functionality
  >
  > - *Oracle Clusterware Administration and Deployment Guide* for a list of Oracle Clusterware Control (CRSCTL) utility commands with similar enhancements

- **Restricting Service Registration for Oracle RAC Deployments**

This feature allows listener registration only from local IPs by default and provides the ability to configure and dynamically update a set of IP addresses or subnets from which registration requests are allowed by the listener.

> **See Also:**
>
> *Oracle Database Net Services Administrator's Guide* for more information

- **Restricting Service Registration with Valid Node Checking**

  This feature enables the network administrator to specify a list of nodes and subnet information from which the Single Client Access Name (SCAN) listener accepts registration. You can specify the nodes and subnets information using SRVCTL, and SRVCTL stores the information in the SCAN listener resource profile and this information is also written to the listener.ora file. Restricting client access to a database makes Oracle RAC even more secure and less vulnerable to security threads and attacks.

- **Pluggable Databases**

  Pluggable Databases enables an Oracle database to contain a portable collection of schemas, schema objects, and nonschema objects that appears to an Oracle Net client as a separate database. This self-contained collection is called a pluggable database (PDB). A container database (CDB) is an Oracle database that includes zero, one, or many user-created pluggable databases (PDBs). You can unplug a PDB from a CDB and plug it into a different CDB.

  > **See Also:**
  >
  > *Oracle Database Administrator's Guide*

- **Support of Oracle Home User on Windows**

  Starting with Oracle Database 12*c*, Oracle Database supports the use of an Oracle home user, which you can specify at installation time. The Oracle home user is associated with a Windows domain user. The Windows domain user should be a low-privileged (non-Administrator) account to ensure that the Oracle home user has a limited set of privileges, thus ensuring that the Oracle Database services have only those privileges required to run Oracle products.

  Windows Administrator user privileges are still required to perform Oracle software maintenance tasks including installation, upgrade, patching, and so on. Oracle Database administrative tools have been enhanced to ask for the password of the Oracle home user, if needed. In Oracle RAC environments, you can store the password for the Oracle home user in a secure wallet. If such a wallet exists, then the Oracle Database administrative tools automatically use the password from the wallet and do not require the user to enter the password for the Oracle home user.

  > **See Also:**
  >
  > *Oracle Real Application Clusters Installation Guide for Microsoft Windows x64 (64-Bit)*

- **Cluster Resources for Oracle ACFS and Oracle ADVM**

Oracle Clusterware resource support includes enhancements for Oracle homes stored on Oracle Automatic Storage Management Cluster File System (Oracle ACFS), Oracle ACFS General Purpose file systems for Grid homes, and Oracle ASM Dynamic Volume Manager (Oracle ADVM) volumes. These resources, that Oracle Clusterware manages, support automatic loading of Oracle ACFS, Oracle ADVM and OKS drivers, disk group mounts, dynamic volume enablement, and automatic Oracle ACFS file system mounts.

See Also:

Oracle Automatic Storage Management Administrator's Guide for more information

- **Oracle Highly Available NFS**

  You can configure Oracle ACFS as a highly available, exported file system service. This service uses Oracle ACFS' clusterwide data consistency and coherency, in combination with virtual IP addresses, to provide failover capability for NFS exports. By mounting the NFS export from this virtual IP address, a client can be assured that, if one node of the cluster is available, then the NFS export will be available.

  See Also:

  Oracle Automatic Storage Management Administrator's Guide for more information

- **Policy-Based Cluster Management and Administration**

  Oracle Grid Infrastructure allows running multiple applications in one cluster. Using a policy-based approach, the workload introduced by these applications can be allocated across the cluster using a policy. In addition, a policy set enables different policies to be applied to the cluster over time as required. You can define policy sets using a web-based interface or a command-line interface.

  Hosting various workloads in the same cluster helps to consolidate the workloads into a shared infrastructure that provides high availability and scalability. Using a centralized policy-based approach allows for dynamic resource reallocation and prioritization as the demand changes.

  See Also:

  Oracle Clusterware Administration and Deployment Guide for more information

- **Online Resource Attribute Modification**

  Oracle Clusterware manages hardware and software components for high availability using a resource model. You use resource attributes to define how Oracle Clusterware manages those resources. You can modify certain resource attributes and implement those changes without having to restart the resource using online resource attribute modification. You manage online resource attribute modification with certain SRVCTL and CRSCTL commands.

Deprecated Features

Deprecation of single-letter SRVCTL CLI options

All SRVCTL commands have been enhanced to accept full-word options instead of the single-letter options. All new SRVCTL command options added in this release support full-word options, only, and do not have single-letter equivalents. The use of single-letter options with SRVCTL commands might be desupported in a future release.

Desupported Features

> **See Also:**
>
> *Oracle Database Upgrade Guide* for more information

- **Oracle Cluster File System for Windows**

  Oracle no longer supports Oracle Cluster File System (OCFS) on Windows.

- **Raw (block) storage devices for Oracle Database and related technologies**

  Oracle Database 12c release 1 (12.1) and its related grid technologies, such as Oracle Clusterware, no longer support the direct use of raw or block storage devices. You must move existing files from raw or block devices to Oracle ASM before you upgrade to Oracle Clusterware 12c release 1 (12.1).

1

Introduction to Oracle RAC

This chapter introduces Oracle Real Application Clusters (Oracle RAC) and describes how to install, administer, and deploy Oracle RAC.

This chapter includes the following topics:

- Overview of Oracle RAC

- Overview of Oracle Real Application Clusters One Node

- Overview of Oracle Clusterware for Oracle RAC

- Overview of Oracle RAC Architecture and Processing

- Overview of Automatic Workload Management with Dynamic Database Services

- Overview of Server Pools and Policy-Managed Databases

- Overview of Installing Oracle RAC

- Overview of Oracle Multitenant with Oracle RAC

- Overview of In-Memory Column Store with Oracle RAC

- Overview of Managing Oracle RAC Environments

Overview of Oracle RAC

Noncluster Oracle databases have a one-to-one relationship between the Oracle database and the instance. Oracle RAC environments, however, have a one-to-many relationship between the database and instances. An Oracle RAC database can have up to 100 instances,[1] all of which access one database. All database instances must use the same interconnect, which can also be used by Oracle Clusterware.

Oracle RAC databases differ architecturally from noncluster Oracle databases in that each Oracle RAC database instance also has:

- At least one additional thread of redo for each instance

- An instance-specific undo tablespace

The combined processing power of the multiple servers can provide greater throughput and Oracle RAC scalability than is available from a single server.

A **cluster** comprises multiple interconnected computers or servers that appear as if they are one server to end users and applications. The Oracle RAC option with Oracle

---

[1] With Oracle Database 10*g* release 2 (10.2) and later releases, Oracle Clusterware supports 100 nodes in an Oracle Clusterware standard Cluster, with the option to run 100 database instances belonging to one production database on these nodes.

Database enables you to cluster Oracle databases. Oracle RAC uses Oracle Clusterware for the infrastructure to bind multiple servers so they operate as a single system.

Oracle Clusterware is a portable cluster management solution that is integrated with Oracle Database. Oracle Clusterware is a required component for using Oracle RAC that provides the infrastructure necessary to run Oracle RAC. Oracle Clusterware also manages resources, such as **Virtual Internet Protocol (VIP)** addresses, databases, listeners, services, and so on. In addition, Oracle Clusterware enables both noncluster Oracle databases and Oracle RAC databases to use the Oracle high-availability infrastructure. Oracle Clusterware along with Oracle Automatic Storage Management (Oracle ASM) (the two together comprise the **Oracle Grid Infrastructure**) enables you to create a clustered pool of storage to be used by any combination of noncluster and Oracle RAC databases.

Oracle Clusterware is the only clusterware that you need for most platforms on which Oracle RAC operates. If your database applications require vendor clusterware, then you can use such clusterware in conjunction with Oracle Clusterware if that vendor clusterware is certified for Oracle RAC.

See Also:

Oracle Clusterware Administration and Deployment Guide and *Oracle Grid Infrastructure Installation Guide* for more details

---

Figure 1-1 shows how Oracle RAC is the Oracle Database option that provides a single system image for multiple servers to access one Oracle database. In Oracle RAC, each Oracle instance must run on a separate server.

Figure 1-1   Oracle Database with Oracle RAC Architecture
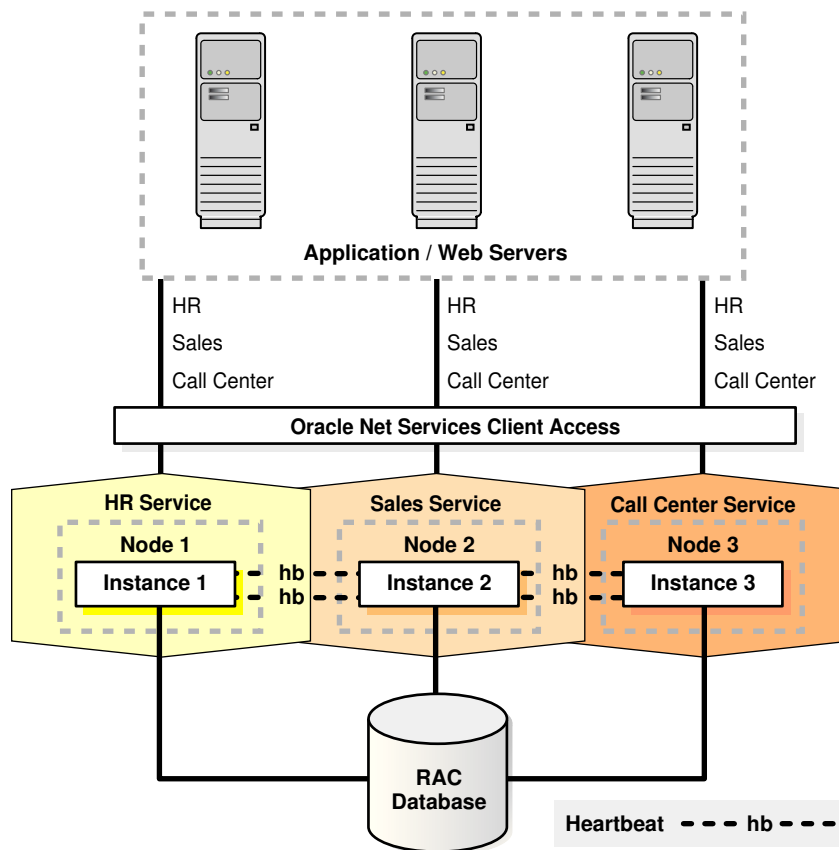
Traditionally, an Oracle RAC environment is located in one data center. However, you can configure Oracle RAC on an **extended distance cluster**, which is an architecture that provides extremely fast recovery from a site failure and allows for all nodes, at all sites, to actively process transactions as part of a single database cluster. In an extended cluster, the nodes in the cluster are typically dispersed, geographically, such as between two fire cells, between two rooms or buildings, or between two different data centers or cities. For availability reasons, the data must be located at both sites, thus requiring the implementation of disk mirroring technology for storage.

If you choose to implement this architecture, you must assess whether this architecture is a good solution for your business, especially considering distance, latency, and the degree of protection it provides. Oracle RAC on extended clusters provides higher availability than is possible with local Oracle RAC configurations, but an extended cluster may not fulfill all of the disaster-recovery requirements of your organization. A feasible separation provides great protection for some disasters (for example, local power outage or server room flooding) but it cannot provide protection against all types of outages. For comprehensive protection against disasters—including protection against corruptions and regional disasters—Oracle recommends the use of Oracle Data Guard with Oracle RAC, as described in the *Oracle Database High Availability Overview* and on the Maximum Availability Architecture (MAA) Web site at

```
http://www.oracle.com/technology/deploy/availability/htdocs/
maa.htm
```

Oracle RAC is a unique technology that provides high availability and scalability for all application types. The Oracle RAC infrastructure is also a key component for implementing the Oracle enterprise grid computing architecture. Having multiple

instances access a single database prevents the server from being a single point of failure. Oracle RAC enables you to combine smaller commodity servers into a cluster to create scalable environments that support mission critical business applications. Applications that you deploy on Oracle RAC databases can operate without code changes.

Overview of Oracle Real Application Clusters One Node

Oracle Real Application Clusters One Node (Oracle RAC One Node) is an option to Oracle Database Enterprise Edition available since Oracle Database 11*g* release 2 (11.2). Oracle RAC One Node is a single instance of an Oracle RAC-enabled database running on one node in the cluster, only, under normal operations. This option adds to the flexibility that Oracle offers for database consolidation while reducing management overhead by providing a standard deployment for Oracle databases in the enterprise. Oracle RAC One Node database requires Oracle Grid Infrastructure and, therefore, requires the same hardware setup as an Oracle RAC database.

Oracle supports Oracle RAC One Node on all platforms on which Oracle RAC is certified. Similar to Oracle RAC, Oracle RAC One Node is certified on Oracle Virtual Machine (Oracle VM). Using Oracle RAC or Oracle RAC One Node with Oracle VM increases the benefits of Oracle VM with the high availability and scalability of Oracle RAC.

With Oracle RAC One Node, there is no limit to server scalability and, if applications grow to require more resources than a single node can supply, then you can upgrade your applications online to Oracle RAC. If the node that is running Oracle RAC One Node becomes overloaded, then you can relocate the instance to another node in the cluster. With Oracle RAC One Node you can use the Online Database Relocation feature to relocate the database instance with no downtime for application users. Alternatively, you can limit the CPU consumption of individual database instances per server within the cluster using Resource Manager Instance Caging and dynamically change this limit, if necessary, depending on the demand scenario.

Using the Single Client Access Name (SCAN) to connect to the database, clients can locate the service independently of the node on which it is running. Relocating an Oracle RAC One Node instance is therefore mostly transparent to the client, depending on the client connection. Oracle recommends to use either Application Continuity and Oracle Fast Application Notification or Transparent Application Failover to minimize the impact of a relocation on the client.

Oracle RAC One Node databases are administered slightly differently from Oracle RAC or noncluster databases. For administrator-managed Oracle RAC One Node databases, you must monitor the candidate node list and make sure a server is always available for failover, if possible. Candidate servers reside in the Generic server pool and the database and its services will fail over to one of those servers.

For policy-managed Oracle RAC One Node databases, you must ensure that the server pools are configured such that a server will be available for the database to fail over to in case its current node becomes unavailable. In this case, the destination node for online database relocation must be located in the server pool in which the database is located. Alternatively, you can use a server pool of size 1 (one server in the server pool), setting the minimum size to 1 and the importance high enough in relation to all other server pools used in the cluster, to ensure that, upon failure of the one server used in the server pool, a new server from another server pool or the Free server pool is relocated into the server pool, as required.

> **Note:**
>
> - Oracle RAC One Node supports Transaction Guard and Application Continuity for failing clients over.
>
> - To prepare for all failure possibilities, you must add at least one Dynamic Database Service (Oracle Clusterware-managed database service) to an Oracle RAC One Node database.

> **See Also:**
>
> - *Oracle Real Application Clusters Installation Guide for Linux and UNIX* for information about installing Oracle RAC One Node databases
>
> - "Transaction Guard for Improving Client Failover"

Overview of Oracle Clusterware for Oracle RAC

Oracle Clusterware provides a complete, integrated clusterware management solution on all Oracle Database platforms. This clusterware functionality provides all of the features required to manage your cluster database including node membership, group services, global resource management, and high availability functions.

You can install Oracle Clusterware independently or as a prerequisite to the Oracle RAC installation process. Oracle Database features, such as services, use the underlying Oracle Clusterware mechanisms to provide advanced capabilities. Oracle Database also continues to support select third-party clusterware products on specified platforms.

Oracle Clusterware is designed for, and tightly integrated with, Oracle RAC. You can use Oracle Clusterware to manage high-availability operations in a cluster. When you create an Oracle RAC database using any of the management tools, the database is registered with and managed by Oracle Clusterware, along with the other required components such as the VIP address, the Single Client Access Name (SCAN) (which includes the SCAN VIPs and the SCAN listener), Oracle Notification Service, and the Oracle Net listeners. These resources are automatically started when the node starts and automatically restart if they fail. The Oracle Clusterware daemons run on each node.

Anything that Oracle Clusterware manages is known as a *CRS resource*. A CRS resource can be a database, an instance, a service, a listener, a VIP address, or an application process. Oracle Clusterware manages CRS resources based on the resource's configuration information that is stored in the Oracle Cluster Registry (OCR). You can use SRVCTL commands to administer any Oracle-defined CRS resources. Oracle Clusterware provides the framework that enables you to create CRS resources to manage any process running on servers in the cluster which are not predefined by Oracle. Oracle Clusterware stores the information that describes the configuration of these components in OCR that you can administer as described in the *Oracle Clusterware Administration and Deployment Guide*.

Overview of Oracle RAC Architecture and Processing

At a minimum, Oracle RAC requires Oracle Clusterware software infrastructure to provide concurrent access to the same storage and the same set of data files from all nodes in the cluster, a communications protocol for enabling interprocess communication (IPC) across the nodes in the cluster, enable multiple database instances to process data as if the data resided on a logically combined, single cache, and a mechanism for monitoring and communicating the status of the nodes in the cluster.

The following sections describe these concepts in more detail:

- Understanding Cluster-Aware Storage Solutions

- Oracle RAC and Network Connectivity

- Overview of Using Dynamic Database Services to Connect to Oracle Databases

- Restricted Service Registration in Oracle RAC

- About Oracle RAC Software Components

- About Oracle RAC Background Processes

Understanding Cluster-Aware Storage Solutions

An Oracle RAC database is a **shared everything** database. All data files, control files, SPFILEs, and redo log files in Oracle RAC environments must reside on cluster-aware shared disks, so that all of the cluster database instances can access these storage components. Because Oracle RAC databases use a shared everything architecture, Oracle RAC requires cluster-aware storage for all database files.

In Oracle RAC, the Oracle Database software manages disk access and is certified for use on a variety of storage architectures. It is your choice how to configure your storage, but you must use a supported cluster-aware storage solution. Oracle Database provides the following storage options for Oracle RAC:

- Oracle Automatic Storage Management (Oracle ASM)

  Oracle recommends this solution to manage your storage.

- A certified cluster file system

  – Oracle recommends Oracle Automatic Storage Management Cluster File System (Oracle ACFS).

  – A third-party cluster file system on a cluster-aware volume manager that is certified for Oracle RAC. For example:

    * Oracle OCFS2 (Linux, only)

    * IBM GPFS (IBM AIX, only)

- Certified network file system (NFS) solution

Oracle RAC and Network Connectivity

All **node**s in an Oracle RAC environment must connect to at least one Local Area Network (LAN) (commonly referred to as the *public network*) to enable users and

applications to access the database. In addition to the public network, Oracle RAC requires private network connectivity used exclusively for communication between the nodes and database instances running on those nodes. This network is commonly referred to as the *interconnect*.

The interconnect network is a private network that connects all of the servers in the cluster. The interconnect network must use at least one switch and a Gigabit Ethernet adapter.

Note:

- Oracle supports interfaces with higher bandwidth but *does not* support using crossover cables with the interconnect.

- Do not use the interconnect (the private network) for user communication, because **Cache Fusion** uses the interconnect for interinstance communication.

You must configure User Datagram Protocol (UDP) for the cluster interconnect, except in a Windows cluster. Windows clusters use the TCP protocol. On Linux and UNIX systems, you can configure Oracle RAC to use either the UDP or Reliable Data Socket (RDS) protocols for inter-instance communication on the interconnect. Oracle Clusterware uses the same interconnect using the UDP protocol, but cannot be configured to use RDS.

An additional network connectivity is required when using **Network Attached Storage (NAS)**. Network attached storage can be typical NAS devices, such as NFS filers, or can be storage that is connected using Fibre Channel over IP, for example. This additional network communication channel should be independent of the other communication channels used by Oracle RAC (the public and private network communication). If the storage network communication must be converged with one of the other network communication channels, then you must ensure that storage-related communication gets first priority.

Overview of Using Dynamic Database Services to Connect to Oracle Databases

Applications should use the Dynamic Database Services feature to connect to an Oracle database over the public network. Dynamic Database Services enable you to define rules and characteristics to control how users and applications connect to database instances. These characteristics include a unique name, workload balancing and failover options, and high availability characteristics.

Users can access an Oracle RAC database using a client/server configuration or through one or more middle tiers, with or without connection pooling. By default, a user connection to an Oracle RAC database is established using the TCP/IP protocol but Oracle supports other protocols. Oracle RAC database instances must be accessed through the SCAN for the cluster.

See Also:

"Overview of Automatic Workload Management with Dynamic Database Services" for more information

---

Overview of Virtual IP Addresses

Oracle Clusterware hosts node virtual IP (VIP) addresses on the public network. Node VIPs are VIP addresses that clients use to connect to an Oracle RAC database. A typical connect attempt from a database client to an Oracle RAC database instance can be summarized, as follows:

1. The database client connects to SCAN (which includes a SCAN VIP on a public network), providing the SCAN listener with a valid service name.

2. The SCAN listener then determines which database instance hosts this service and routes the client to the local or node listener on the respective node.

3. The node listener, listening on a node VIP and a given port, retrieves the connection request and connects the client to the an instance on the local node.

If multiple public networks are used on the cluster to support client connectivity through multiple subnets, then the preceding operation is performed within a given subnet.

If a node fails, then the VIP address fails over to another node on which the VIP address can accept TCP connections, but it does not accept connections to the Oracle database. Clients that attempt to connect to a VIP address not residing on its home node receive a *rapid connection refused* error instead of waiting for TCP connect timeout messages. When the network on which the VIP is configured comes back online, Oracle Clusterware fails back the VIP to its home node, where connections are accepted. Generally, VIP addresses fail over when:

- The node on which a VIP address runs fails

- All interfaces for the VIP address fail

- All interfaces for the VIP address are disconnected from the network

Oracle RAC 12*c* supports multiple public networks to enable access to the cluster through different subnets. Each network resource represents its own subnet and each database service uses a particular network to access the Oracle RAC database. Each network resource is a resource managed by Oracle Clusterware, which enables the VIP behavior previously described.

SCAN is a single network name defined either in your organization's Domain Name Server (DNS) or in the Grid Naming Service (GNS) that round robins to three IP addresses. Oracle recommends that all connections to the Oracle RAC database use the SCAN in their client connection string. Incoming connections are load balanced across the active instances providing the requested service through the three SCAN listeners. With SCAN, you do not have to change the client connection even if the configuration of the cluster changes (nodes added or removed). Unlike in previous releases, SCAN in Oracle RAC 12*c* fully supports multiple subnets, which means you can create one SCAN for each subnet in which you want the cluster to operate.

Restricted Service Registration in Oracle RAC

The valid node checking feature provides the ability to configure and dynamically update a set of IP addresses or subnets from which registration requests are allowed by the listener. Database instance registration with a listener succeeds only when the request originates from a valid node. The network administrator can specify a list of valid nodes, excluded nodes, or disable valid node checking. The list of valid nodes explicitly lists the nodes and subnets that can register with the database. The list of

excluded nodes explicitly lists the nodes that cannot register with the database. The control of dynamic registration results in increased manageability and security of Oracle RAC deployments.

By default, valid node checking for registration (VNCR) is enabled. In the default configuration, registration requests from all nodes within the subnet of the SCAN listener can register with the listener. Non-SCAN listeners only accept registration from instances on the local node. Remote nodes or nodes outside the subnet of the SCAN listener must be included on the list of valid nodes by using the `registration_invited_nodes_alias` parameter in the `listener.ora` file or by modifying the SCAN listener using SRVCTL.

> **See Also:**
>
> *Oracle Database Net Services Administrator's Guide* for more information about VNCR

About Oracle RAC Software Components

Oracle RAC databases generally have two or more database instances that each contain memory structures and background processes. An Oracle RAC database has the same processes and memory structures as a noncluster Oracle database and additional processes and memory structures that are specific to Oracle RAC. Any one instance's database view is nearly identical to any other instance's view in the same Oracle RAC database; the view is a single system image of the environment.

Each instance has a buffer cache in its System Global Area (SGA). Using Cache Fusion, Oracle RAC environments logically combine each instance's buffer cache to enable the instances to process data as if the data resided on a logically combined, single cache.

> **Note:**
>
> - The In-Memory Transaction Manager integrates with the Cache Fusion protocol.
>
> - The SGA size requirements for Oracle RAC are greater than the SGA requirements for noncluster Oracle databases due to Cache Fusion.

> **See Also:**
>
> *Oracle Database Concepts* for more information about the In-Memory Transaction Manager

To ensure that each Oracle RAC database instance obtains the block that it requires to satisfy a query or transaction, Oracle RAC instances use two processes, the Global Cache Service (GCS) and the Global Enqueue Service (GES). The GCS and GES maintain records of the statuses of each data file and each cached block using a Global Resource Directory (GRD). The GRD contents are distributed across all of the active instances, which effectively increases the size of the SGA for an Oracle RAC instance.

After one instance caches data, any other instance within the same cluster database can acquire a block image from another instance in the same database faster than by

reading the block from disk. Therefore, Cache Fusion moves current blocks between instances rather than re-reading the blocks from disk. When a consistent block is needed or a changed block is required on another instance, Cache Fusion transfers the block image directly between the affected instances. Oracle RAC uses the private interconnect for interinstance communication and block transfers. The GES Monitor and the Instance Enqueue Process manage access to Cache Fusion resources and enqueue recovery processing.

About Oracle RAC Background Processes

The GCS and GES processes, and the GRD collaborate to enable Cache Fusion. The Oracle RAC processes and their identifiers are as follows:

- ACMS: Atomic Controlfile to Memory Service (ACMS)

  In an Oracle RAC environment, the ACMS per-instance process is an agent that contributes to ensuring a distributed SGA memory update is either globally committed on success or globally aborted if a failure occurs.

- GTX0-j: Global Transaction Process

  The GTX0-j process provides transparent support for XA global transactions in an Oracle RAC environment. The database autotunes the number of these processes based on the workload of XA global transactions.

- LMON: Global Enqueue Service Monitor

  The LMON process monitors global enqueues and resources across the cluster and performs global enqueue recovery operations.

- LMD: Global Enqueue Service Daemon

  The LMD process manages incoming remote resource requests within each instance.

- LMS: Global Cache Service Process

  The LMS process maintains records of the data file statuses and each cached block by recording information in a Global Resource Directory (GRD). The LMS process also controls the flow of messages to remote instances and manages global data block access and transmits block images between the buffer caches of different instances. This processing is part of the Cache Fusion feature.

- LCK0: Instance Enqueue Process

  The LCK0 process manages non-Cache Fusion resource requests such as library and row cache requests.

- RMSn: Oracle RAC Management Processes (RMSn)

  The RMSn processes perform manageability tasks for Oracle RAC. Tasks accomplished by an RMSn process include creation of resources related to Oracle RAC when new instances are added to the clusters.

- RSMN: Remote Slave Monitor manages background slave process creation and communication on remote instances. These background slave processes perform tasks on behalf of a coordinating process running in another instance.

> **Note:**
>
> Many of the Oracle Database components that this section describes are in
> addition to the components that are described for noncluster Oracle databases
> in *Oracle Database Concepts*.

Overview of Automatic Workload Management with Dynamic Database Services

Services represent groups of applications with common attributes, service level thresholds, and priorities. Application functions can be divided into workloads identified by services. For example, Oracle E-Business Suite can define a service for each responsibility, such as general ledger, accounts receivable, order entry, and so on. A service can span one or more instances of an Oracle database, multiple databases in a global cluster, and a single instance can support multiple services. The number of instances that are serving the service is transparent to the application. Services provide a single system image to manage competing applications, and allow each workload to be managed as a unit.

Middle tier applications and clients select a service by specifying the service name as part of the connection in the TNS connect string. For example, data sources for Oracle WebLogic Server are set to route to a service. Using Net Easy*Connection, this connection comprises simply the service name and network address, as follows: *user_name*/*password*@SCAN/*service_name*. Server-side work such as Oracle Scheduler, Parallel Query, and Oracle Streams queues set the service name as part of the workload definition. For Oracle Scheduler, jobs are assigned to job classes and job classes execute within services. For Parallel Query and Parallel DML, the query coordinator connects to a service and the parallel query slaves inherit the service for the duration of the parallel execution. For Oracle Streams, streams queues are accessed using services. Work executing under a service inherits the thresholds and attributes for the service and is measured as part of the service.

Oracle Database Resource Manager binds services to consumer groups and priorities. This allows the database to manage the services in the order of their importance. For example, the DBA can define separate services for high priority online users and lower priority internal reporting applications. Likewise, the DBA can define Gold, Silver and Bronze services to prioritize the order in which requests are serviced for the same application. When planning the services for a system, the plan should include the priority of each service relative to the other services. In this way, Oracle Database Resource Manager can satisfy the priority-one services first, followed by the priority-two services, and so on.

> **See Also:**
>
> *Oracle Database Administrator's Guide* for more information about Oracle
> Database Resource Manager

When users or applications connect to a database, Oracle recommends that you use a service specified in the CONNECT_DATA portion of the connect string. Oracle Database automatically creates one database service when the database is created but the behavior of this service is different from that of database services that you subsequently create. To enable more flexibility in the management of a workload using the database, Oracle Database enables you to create multiple services and

specify on which instances (or in which server pools) the services start. If you are interested in greater workload management flexibility, then continue reading this chapter to understand the added features that you can use with services.

> **Note:**
>
> The features discussed in this chapter do not work with the following default database services: DB_NAME, DB_UNIQUE_NAME, PDB_NAME, SYS $BACKGROUND, and SYS$USERS. Oracle strongly recommends that you do not use these services for applications to connect to the database. You must create cluster managed services to take advantage of these features. You can only manage the services that you create. Any service that the database create automatically is managed by the database server.

Dynamic Database Services

Dynamic database services enable you to manage workload distributions to provide optimal performance for users and applications. Dynamic database services offer the following features:

- **Services**: Oracle Database provides a powerful automatic workload management facility, called services, to enable the enterprise grid vision. Services are entities that you can define in Oracle RAC databases that enable you to group database workloads, route work to the optimal instances that are assigned to offer the service, and achieve high availability for planned and unplanned actions.

- **High Availability Framework**: An Oracle RAC component that enables Oracle Database to always maintain components in a running state.

- **Fast Application Notification (FAN)**: Provides information to Oracle RAC applications and clients about cluster state changes and Load Balancing Advisory events, such as UP and DOWN events for instances, services, or nodes. FAN has two methods for publishing events to clients, the Oracle Notification Service daemon, which is used by Java Database Connectivity (JDBC) clients including the Oracle Application Server, and Oracle Streams Advanced Queueing, which is used by Oracle Call Interface (OCI) and Oracle Data Provider for .NET (ODP.NET) clients.

  > **Note:**
  >
  > In previous releases of Oracle Clusterware, OCI and ODP.NET clients received FAN events through Oracle Streams Advanced Queuing.

- **Transaction Guard**: A tool that provides a protocol and an API for at-most-once execution of transactions in case of unplanned outages and duplicate submissions.

- **Application Continuity**: Provides a general purpose infrastructure that replays the in-flight request when a recoverable error is received, masking many system, communication, and storage outages, and hardware failures. Unlike existing recovery technologies, this feature attempts to recover the transactional and non-transactional session states beneath the application, so that the outage appears to the application as a delayed execution.

- **Connection Load Balancing**: A feature of Oracle Net Services that balances incoming connections across all of the instances that provide the requested database service.

- **Load Balancing Advisory**: Provides information to applications about the current service levels that the database and its instances are providing. The load balancing advisory makes recommendations to applications about where to direct application requests to obtain the best service based on the management policy that you have defined for that service. Load balancing advisory events are published through Oracle Notification Service.

- **Automatic Workload Repository (AWR)**: Tracks service-level statistics as **metric**s. Server generated alerts can be created for these metrics when they exceed or fail to meet certain thresholds.

- **Fast Connection Failover**: This is the ability of Oracle Clients to provide rapid failover of connections by subscribing to FAN events.

- **Runtime Connection Load Balancing**: This is the ability of Oracle Clients to provide intelligent allocations of connections in the connection pool based on the current service level provided by the database instances when applications request a connection to complete some work.

- **Single Client Access Name (SCAN)**: Provides a single name to the clients connecting to Oracle RAC that does not change throughout the life of the cluster, even if you add or remove nodes from the cluster. Clients connecting with SCAN can use a simple connection string, such as a thin JDBC URL or EZConnect, and still achieve the load balancing and client connection failover.

You can deploy Oracle RAC and noncluster Oracle database environments to use dynamic database service features in many different ways. Depending on the number of nodes and your environment's complexity and objectives, your choices for optimal automatic workload management and high-availability configuration depend on several considerations that are described in this chapter.

> **See Also:**
>
> Workload Management with Dynamic Database Services

Overview of Server Pools and Policy-Managed Databases

You can create Oracle RAC databases, whether multinode or Oracle Real Application Clusters One Node (Oracle RAC One Node), using the following deployment models:

- Administrator-managed deployment is based on the Oracle RAC deployment types that existed before Oracle Database 11*g* release 2 (11.2) and requires that you statically configure each database instance to run on a specific node in the cluster, and that you configure database services to run on specific instances belonging to a certain database using the `preferred` and `available` designation.

- Policy-managed deployment is based on *server pools*, where database services run within a server pool as *singleton* or *uniform* across all of the servers in the server pool. Databases are deployed in one or more server pools and the size of the server pools determine the number of database instances in the deployment.

This section includes the following topics:

- Introduction to Server Pools

- Deploying Policy-Managed Databases

- Managing Policy-Managed Databases

- Policy-Based Cluster Management

Introduction to Server Pools

Server pools logically apportion a cluster into groups of servers offering database or application services. Server pool properties control the scalability and availability of those databases and applications. You can configure each server pool with a minimum and maximum size, which determines scalability. Oracle Clusterware manages availability between server pools, and you can further regulate availability by configuring the importance value of individual server pools.

Servers are not assigned to server pools by name but by number. Therefore, you must configure any server to run any database. If you cannot configure servers due to, for example, heterogeneous servers or storage connectivity, then you can restrict servers by using server category definitions to determine server pool membership eligibility.

See Also:

Oracle Clusterware Administration and Deployment Guide for more information about creating and managing server pools

---

Examples of Using Server Pools

This section includes the following examples of using server pools:

- Minimum and Maximum Number of Servers

- IMPORTANCE Attribute of Server Pools

- Consolidation of Databases

Minimum and Maximum Number of Servers

Consider a four-node cluster configured into two server pools named `online` and `backoffice`. A database named `dbsales` runs in the `online` server pool offering the `browse`, `search`, and `salescart` services. A database named `dberp` runs in the `backoffice` server pool and offers the `inventory` and `shipping` services, as shown in Figure 1-2. During normal business hours the enterprise requires a minimum of two instances of the `dbsales` database and one instance of the `dberp` database to meet normal demand.

Figure 1-2   Server Placement by Minimum and Maximum Limits
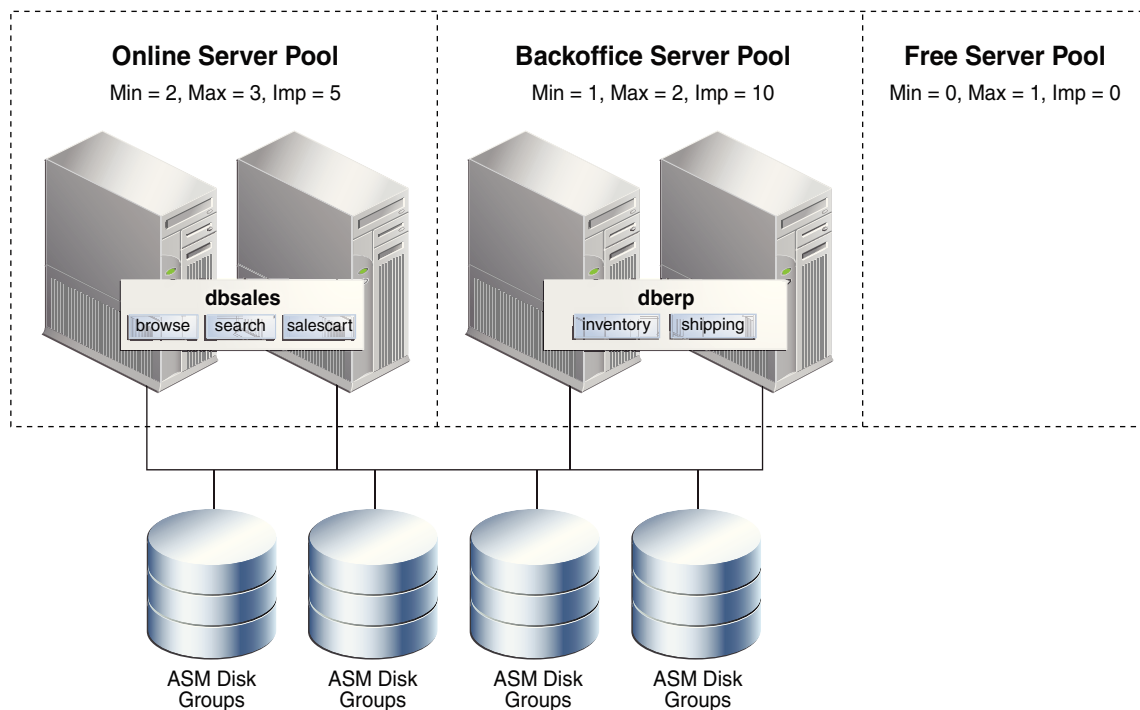
In this policy-managed deployment, the value of the MIN_SIZE server pool attribute for the online server pool is 2, while the value of the MIN_SIZE server pool attribute for the backoffice server pool is 1. Configured this way, Oracle Clusterware ensures that there are always two servers in the online server pool and one server in the backoffice server pool. Because this is a four-node cluster, there is one server left not assigned to either server pool. Where that last server gets deployed is determined by the MAX_SIZE server pool parameter of each server pool. If the sum of the values of the MAX_SIZE server pool attribute for each server pool is less than the total number of servers in the cluster, then the remaining servers stay in the Free server pool awaiting a failure of a deployed node.

If the value of MAX_SIZE is greater than that of MIN_SIZE, then the remaining server will be deployed into the server pool whose importance value is greatest, as shown in Figure 1-2, and fully discussed in the next section. In this case, the server is a shareable resource that can be relocated online to join the server pool where it is required. For example, during business hours the server could be given to the online server pool to add an instance of the dbsales database but after hours could be relocated to the backoffice server pool, adding a dberp database instance. All such movements are online and instances are shut down, transactionally.

These two policy-managed databases are running only the instances that are required and they can be dynamically increased or decreased to meet demand or business requirements.

IMPORTANCE Attribute of Server Pools

The IMPORTANCE server pool attribute is used at cluster startup and in response to a node failure or eviction. In contrast to administrator-managed databases, you can configure server pools with different importance levels to determine which databases are started first and which databases remain online in case there is a multinode outage.

Consider a four-node cluster that hosts a database named `dbapps` in two server pools, `sales` and `backoffice`. Two services, `orderentry` and `billing`, run in the `sales` server pool, while two other services, `erp` and `reports`, run in the `backoffice` server pool, as shown in Figure 1-3. By configuring the value of the `IMPORTANCE` server pool attribute of the `sales` server pool higher than that of the `backoffice` server pool, the services in `sales` start first when the cluster starts and are always available, even if there is only one server left running after a multinode failure. The `IMPORTANCE` server pool attribute enables you to rank services and also eliminates the requirement to run a service on all nodes in a cluster to ensure that it is always available.

Figure 1-3   Server Pool Importance
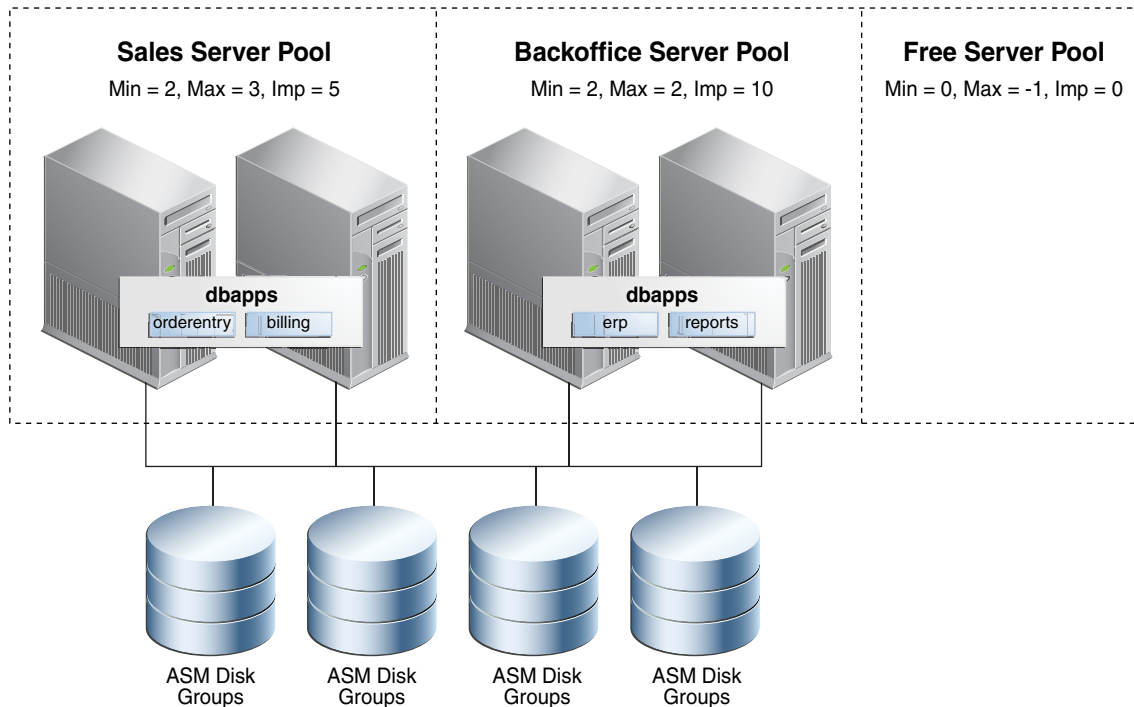
Consolidation of Databases

You can use several different approaches, either discretely or combined, to consolidate Oracle databases. Policy-managed deployments facilitate consolidation. In the case of schema consolidation, where multiple applications are being hosted in a single database separated into discrete schemas or pluggable databases (PDBs), you can use server pools to meet required capacity. Because of the dynamic scaling property of server pools, you can increase or decrease the number of database instances to meet current demand or business requirements. Since server pools also determine which services run together or separately, you can configure and maintain required affinity or isolation.

When it is not possible to use schema consolidation because of, for example, version requirements, you can host multiple databases on a single set of servers. Using policy-managed databases facilitates this database consolidation because they can share the same server pool by making use of **instance caging**, which enables you to dynamically increase or decrease databases, both horizontally (using server pool size) and vertically (using the `CPU_COUNT` server configuration attribute) to meet demand or business policies and schedules.

> **See Also:**
>
> *Oracle Clusterware Administration and Deployment Guide* for more information about the CPU_COUNT server configuration attribute

By contrast, with administrator-managed databases, you are required to reserve capacity on each server to absorb workload failing over should a database instance or server fail. With policy-managed databases, however, you can effectively rank server pools by the business necessity of the workloads that they are running using the MIN_SIZE, MAX_SIZE, and IMPORTANCE server pool attributes.

When the failure of a server brings a server pool to below its configured minimum number of servers, another server will move from a less important server pool to take its place and bring the number of servers back up to the configured minimum. This eliminates the risk of cascade failures due to overloading the remaining servers and enables you to significantly reduce or even eliminate the need to reserve capacity for handling failures.

Migrating or converting to policy-managed databases also enables cluster consolidation and creates larger clusters that have greater availability and scalability because of the increased number of servers available to host and scale databases. Because policy-managed databases do not require binding their instance names to a particular server and binding services to particular instances, the complexity of configuring and managing large clusters is greatly reduced.

An example deployment is shown in Figure 1-4 where the previous two cluster examples (shown in Figure 1-2 and Figure 1-3) are consolidated into a single cluster, making use of both database consolidation (using instance caging) and cluster consolidation (using server pools) configured so that workloads are properly sized and prioritized.

Figure 1-4   Consolidating Databases
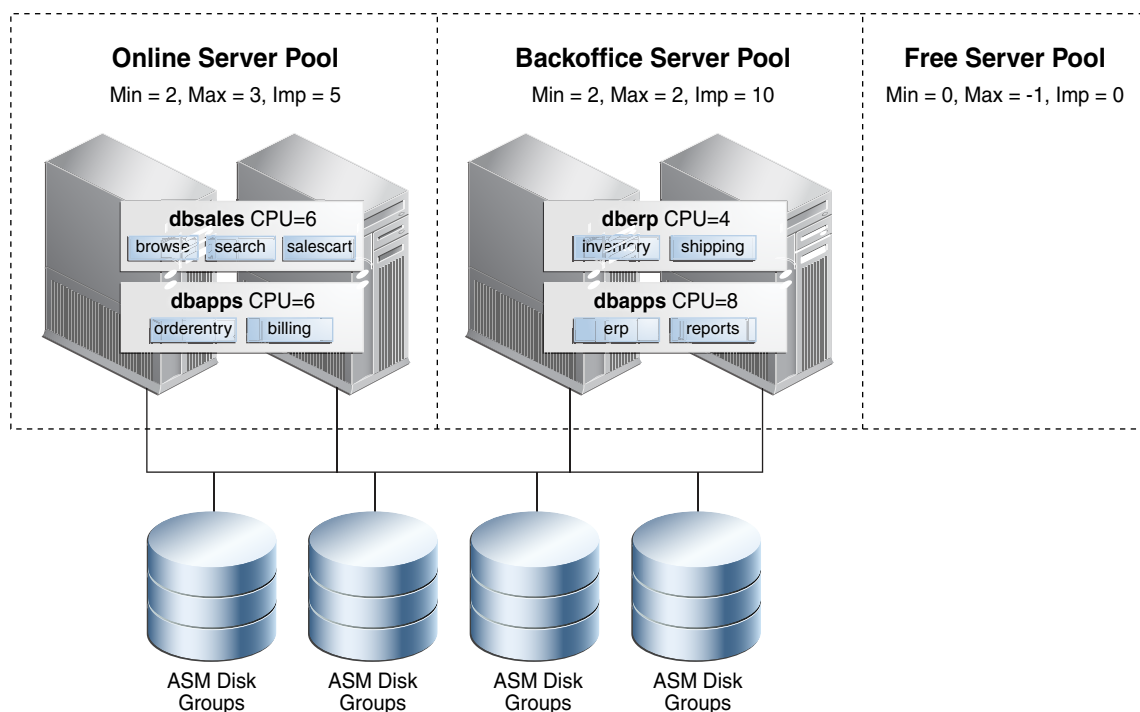
Deploying Policy-Managed Databases

When you deploy a policy-managed database you must first determine the services and their required sizing, taking into account that services cannot span server pools. If you are going to collocate this database with other databases, then you should factor in its CPU requirements relative to the other hosted databases, and also factor in the value of its `CPU_COUNT` attribute for instance caging, so that you can size the database both vertically and horizontally in one or more server pools.

If you are going to collocate the server pools for this database with other server pools, then consider configuring the server pools to adjust the server pool sizes on a calendar or event basis to optimize meeting demand and business requirements. Once you have determined the sizes of the server pools, and configured the appropriate values for the `MIN_SIZE` and `MAX_SIZE` server pool attributes, you can then determine the relative importance of each server pool.

> **See Also:**
>
> *Oracle Clusterware Administration and Deployment Guide* for more information about server pool attributes

You, as the cluster administrator, create policy-managed database server pools using the `srvctl add serverpool` command. You can modify the properties of the server pool using the `srvctl modify serverpool` command in the Oracle Grid Infrastructure home.

> **See Also:**
>
> *Oracle Database Quality of Service Management User's Guide* for more information about creating server pools

While it is possible to create a server pool using DBCA, Oracle recommends this only for small, single server pool deployments, because DBCA will fail if servers are already allocated to other server pools. Additionally, if the cluster is made up of servers with different capacities, such as old and new servers, Oracle recommends that you set up server category definitions defining the minimum server requirements for a server to join each server pool.

> **See Also:**
>
> *Oracle Clusterware Administration and Deployment Guide* for more information about server categorization

After you create the server pools, you can run DBCA from the appropriate database home. Depending on the database type and task, you will be presented with different default options. For all new Oracle RAC and Oracle RAC One Node databases, including container databases (CDBs), the Policy-Managed option is the default and the option that Oracle recommends.

If you are upgrading your database from an administrator-managed database or a database earlier than Oracle Database 11*g* release 2 (11.2), then you will not have the

option to directly upgrade to a policy-managed database. After you upgrade, however, you can convert the database to policy managed using the `srvctl modify database` command.

> **See Also:**
>
> "Converting an Administrator-Managed Database to a Policy-Managed Database"

When you convert from an administrator-managed database to a policy-managed database, the instance names are automatically updated to include the underscore (for example: `orcl1` becomes `orcl_1`). The underscore is required so that the database can automatically create instances when a server pool grows in size.

Managing Policy-Managed Databases

Managing a policy-managed database requires less configuration and reconfiguration steps than an administrator-managed one with respect to creation, sizing, patching, and load balancing. Also, because any server in the server pools within the cluster can run any of the databases, you do not have to create and maintain database instance-to-node-name mappings. You can perform maintenance tasks such as patching by relocating servers into the Free pool or by adjusting the server pool minimum and maximum sizes, thereby retaining required availability.

> **See Also:**
>
> "srvctl modify srvpool" and "srvctl relocate server" for more information

Policy-managed databases also facilitate the management of services, because they are assigned to a single server pool and run as singletons or uniform across all servers in the pool. You no longer have to create or maintain explicit preferred and available database instance lists for each service. If a server moves into a server pool because of manual relocation or a high availability event, all uniform services and their dependent database instances are automatically started. If a server hosting one or more singleton services goes down, those services will automatically be started on one or more of the remaining servers in the server pool. In the case of Oracle RAC One Node, the corresponding database instance will also be started automatically.

Managing services relative to each other is improved by making use of the importance attribute of each server pool. Each service running in a server pool inherits the server pool's importance relative to the other server pool-hosted services in the cluster. If the minimum size of the most important server pool is greater than zero, then the services and associated database instances in that server pool are started first on cluster startup and will be the last services and database instances running, as long as there is one server running in the cluster. You can offer services not critical to the business in the least important server pool, ensuring that, should sufficient resources not be available due to demand or failures, those services will eventually be shut down and the more business-critical services remain available.

Because many management tasks may involve making changes that can affect multiple databases, services, or server pools in a consolidated environment, you can use the evaluate mode for certain SRVCTL commands to get a report of the resource impact of a command.

Consider the following example, that evaluates the effect on the system of modifying a server pool:

```
$ srvctl modify srvpool -l 3 -g online -eval
Service erp1 will be stopped on node test3
Service reports will be stopped on node test3
Service inventory will be stopped on node test3
Service shipping will be stopped on node test3
Database dbsales will be started on node test3
Service orderentry will be started on node test3
Service billing will be started on node test3
Service browse will be started on node test3
Service search will be started on node test3
Service salescart will be started on node test3
Server test3 will be moved from pool backoffice to pool online
```

As shown in the preceding example, modifying a server pool can result in many resource state changes. You can use a *policy set* through either Oracle Clusterware or Oracle Database Quality of Service Management.

See Also:

- "SRVCTL Usage Information" for a list of SRVCTL commands which use the evaluate mode

- *Oracle Clusterware Administration and Deployment Guide* for information about the Oracle Clusterware policy set

- *Oracle Database Quality of Service Management User's Guide* for information about the Oracle Database Quality of Service Management policy set

---

Policy-Based Cluster Management

In Oracle Database 12*c*, Oracle Clusterware supports the management of a cluster configuration policy set as a native Oracle Clusterware feature. A **cluster configuration policy** contains one definition for each server pool that is defined in the system. A cluster configuration policy also specifies resource placement and cluster node availability. A **cluster configuration policy set** defines the names of all of the server pools that are configured in a cluster, and contains one or more configuration policies.

There is always only one configuration policy in effect at any one time. However, administrators typically create several configuration policies to reflect the different business needs and demands based on calendar dates or time of day parameters. For instance, morning hours during business days are typically when most users log in and download their email; email-related workloads are usually light at nighttime and on weekends. In such cases, you can use cluster configuration policies to define the server allocation based on the expected demand. More specifically for this example, a configuration policy that allocates more servers to OLTP workloads is in effect during workday mornings, and another configuration policy allocates more servers to batch workloads on weekends and workday evenings.

Using cluster configuration policies can also help manage clusters that comprise servers of different capabilities, such as different computer and memory sizes (heterogeneous). To create management and availability policies for clusters comprised of heterogeneous server types, the cluster administrator can create server

categories based on server attributes. These attributes can restrict which servers can be assigned to which server pools. For example, if you have some servers in a cluster that run older hardware, then you can use an attribute to specify that these servers should only be allocated to the server pools that support batch jobs and testing, instead of allocating them to the server pools that are used for online sales or other business-critical applications.

> **See Also:**
>
> *Oracle Clusterware Administration and Deployment Guide* for more information

Overview of Installing Oracle RAC

Install Oracle Grid Infrastructure and Oracle Database software using Oracle Universal Installer, and create your database with Database Configuration Assistant (DBCA). This ensures that your Oracle RAC environment has the optimal network configuration, database structure, and parameter settings for the environment that you selected.

Alternatively, you can install Oracle RAC using Rapid Home Provisioning.

> **See Also:**
>
> *Oracle Real Application Clusters Installation Guide* for your platform for more information about installing Oracle RAC using Rapid Home Provisioning

This section introduces the installation processes for Oracle RAC under the following topics:

- Understanding Compatibility in Oracle RAC Environments
- Oracle RAC Database Management Styles and Database Installation
- Oracle RAC Database Management Styles and Database Creation
- Overview of Extending an Oracle RAC Cluster

> **Note:**
>
> You must first install Oracle Grid Infrastructure before installing Oracle RAC.

> **See Also:**
>
> *Oracle Grid Infrastructure Installation Guide* for your platform

Understanding Compatibility in Oracle RAC Environments

To run Oracle RAC in configurations with different versions of Oracle Database in the same cluster, you must first install Oracle Grid Infrastructure, which must be the same version, or higher, as the highest version of Oracle Database that you want to deploy in this cluster. For example, to run an Oracle RAC 11*g* release 2 (11.2) database and an Oracle RAC 12*c* database in the same cluster, you must install Oracle Grid

Infrastructure 12*c*. Contact My Oracle Support for more information about version compatibility in Oracle RAC environments.

> **Note:**
>
> Oracle does not support deploying an Oracle9*i* cluster in an Oracle Grid Infrastructure 12*c* environment.

Oracle RAC Database Management Styles and Database Installation

Before installing the Oracle RAC database software and creating respective databases, decide on the management style you want to apply to the Oracle RAC databases, as described in "Overview of Server Pools and Policy-Managed Databases".

The management style you choose impacts the software deployment and database creation. If you choose the administrator-managed database deployment model, using a per-node installation of software, then it is sufficient to deploy the Oracle Database software (the database home) on only those nodes on which you plan to run Oracle Database.

If you choose the policy-managed deployment model, using a per-node installation of software, then you must deploy the software on all nodes in the cluster, because the dynamic allocation of servers to server pools, in principle, does not predict on which server a database instance can potentially run. To avoid instance startup failures on servers that do not host the respective database home, Oracle strongly recommends that you deploy the database software on all nodes in the cluster. When you use a shared Oracle Database home, accessibility to this home from all nodes in the cluster is assumed and the setup needs to ensure that the respective file system is mounted on all servers, as required.

Oracle Universal Installer will only allow you to deploy an Oracle Database home across nodes in the cluster if you previously installed and configured Oracle Grid Infrastructure for the cluster. If Oracle Universal Installer does not give you an option to deploy the database home across all nodes in the cluster, then check the prerequisite, as stated, by Oracle Universal Installer.

During installation, you can choose to create a database during the database home installation. Oracle Universal Installer runs DBCA to create your Oracle RAC database according to the options that you select.

> **See Also:**
>
> "Oracle RAC Database Management Styles and Database Creation" for more information if you choose this option

> **Note:**
>
> Before you create a database, a default listener must be running in the Oracle Grid Infrastructure home. If a default listener is not present in the Oracle Grid Infrastructure home, then DBCA returns an error instructing you to run NETCA from the Oracle Grid Infrastructure home to create a default listener.

> **See Also:**
>
> *Oracle Database Net Services Administrator's Guide* for more information about NETCA

The Oracle RAC software is distributed as part of the Oracle Database installation media. By default, the Oracle Database software installation process installs the Oracle RAC option when it recognizes that you are performing the installation on a cluster. Oracle Universal Installer installs Oracle RAC into a directory structure referred to as the Oracle home, which is separate from the Oracle home directory for other Oracle software running on the system. Because Oracle Universal Installer is cluster aware, it installs the Oracle RAC software on all of the nodes that you defined to be part of the cluster.

Oracle RAC Database Management Styles and Database Creation

Part of Oracle Database deployment is the creation of the database. You can choose to create a database as part of the database software deployment, as described in "Oracle RAC Database Management Styles and Database Installation", or you can choose to only deploy the database software, first, and then, subsequently, create any database that is meant to run out of the newly created Oracle home by using DBCA. In either case, you must consider the management style that you plan to use for the Oracle RAC databases.

For administrator-managed databases, you must ensure that the database software is deployed on the nodes on which you plan to run the respective database instances. You must also ensure that these nodes have access to the storage in which you want to store the database files. Oracle recommends that you select Oracle ASM during database installation to simplify storage management. Oracle ASM automatically manages the storage of all database files within disk groups. If you plan to use Oracle Database Standard Edition to create an Oracle RAC database, then you *must* use Oracle ASM to store all of the database files.

For policy-managed databases, you must ensure that the database software is deployed on all nodes on which database instances can potentially run, given your active server pool setup. You must also ensure that these nodes have access to the storage in which you want to store the database files. Oracle recommends using Oracle ASM, as previously described for administrator-managed databases.

Server pools are a feature of Oracle Grid Infrastructure (specifically Oracle Clusterware). There are different ways you can set up server pools on the Oracle Clusterware level, and Oracle recommends you create server pools for database management before you create the respective databases. DBCA, however, will present you with a choice of either using precreated server pools or creating a new server pool, when you are creating a policy-managed database. Whether you can create a new server pool during database creation depends on the server pool configuration that is active at the time.

> **Note:**
>
> You must configure Oracle ASM separately before you create an Oracle RAC database.

> **See Also:**
>
> *Oracle Clusterware Administration and Deployment Guide* for more information on managing server pools

By default, DBCA creates one service for your Oracle RAC installation. This is the default database service and should not be used for user connectivity. The default database service is typically identified using the combination of the DB_NAME and DB_DOMAIN initialization parameters: *db_name.db_domain*. The default service is available on all instances in an Oracle RAC environment, unless the database is in restricted mode.

> **Note:**
>
> Oracle recommends that you reserve the default database service for maintenance operations and create dynamic database services for user or application connectivity as a post-database-creation step, using either SRVCTL or Oracle Enterprise Manager. DBCA no longer offers a dynamic database service creation option for Oracle RAC databases. For Oracle RAC One Node databases, you must create at least one dynamic database service.

Overview of Extending an Oracle RAC Cluster

If you want to extend the Oracle RAC cluster (also known as cloning) and add nodes to the existing environment after your initial deployment, then you must to do this on multiple layers, considering the management style that you currently use in the cluster. Oracle provides various means of extending an Oracle RAC cluster. In principle, you can choose from two different approaches to extend the current environment:

- Cloning using cloning scripts

  > **See Also:**
  >
  > Cloning Oracle RAC to Nodes in a New Cluster

- Adding nodes using the addnode.sh (addnode.bat on Windows) script

  > **See Also:**
  >
  > Adding and Deleting Oracle RAC from Nodes on Linux and UNIX Systems or Adding and Deleting Oracle RAC from Nodes on Windows Systems depending on your platform

Both approaches are applicable, regardless of how you initially deployed the environment. Both approaches copy the required Oracle software on to the node that you plan to add to the cluster. Software that gets copied to the node includes the Oracle Grid Infrastructure software and the Oracle database homes.

For Oracle database homes, you must consider the management style deployed in the cluster. For administrator-managed databases, you must ensure that the database software is deployed on the nodes on which you plan to run the respective database

instances. For policy-managed databases, you must ensure that the database software is deployed on all nodes on which database instances can potentially run, given your active server pool setup. In either case, you must first deploy Oracle Grid Infrastructure on all nodes that are meant to be part of the cluster.

> **Note:**
>
> Oracle cloning is not a replacement for cloning using Oracle Enterprise Manager as part of the Provisioning Pack. When you clone Oracle RAC using Oracle Enterprise Manager, the provisioning process includes a series of steps where details about the home you want to capture, the location to which you want to deploy, and various other parameters are collected.

For new installations or if you install only one Oracle RAC database, use the traditional automated and interactive installation methods, such as Oracle Universal Installer, or the Provisioning Pack feature of Oracle Enterprise Manager. If your goal is to add or delete Oracle RAC from nodes in the cluster, you can use the procedures detailed in Adding and Deleting Oracle RAC from Nodes on Linux and UNIX Systems.

The cloning process assumes that you successfully installed an Oracle Clusterware home and an Oracle home with Oracle RAC on at least one node. In addition, all root scripts must have run successfully on the node from which you are extending your cluster database.

> **See Also:**
>
> Oracle Enterprise Manager online Help system for more information about the Provisioning Pack

Overview of Oracle Multitenant with Oracle RAC

Oracle Multitenant is an option with Oracle Database 12*c* that simplifies consolidation, provisioning, upgrades, and more. It is based on an architecture that allows a multitenant container database (CDB) to hold several pluggable databases (PDBs). You can adopt an existing database as a PDB without having to change the application tier. In this architecture, Oracle RAC provides the local high availability that is required when consolidating various business-critical applications on one system.

When using PDBs with Oracle RAC, the multitenant CDB is based on Oracle RAC. You can make each PDB available on either every instance of the Oracle RAC CDB or a subset of instances. In either case, access to and management of the PDBs are regulated using dynamic database services, which will also be used by applications to connect to the respective PDB, as they would in a single instance Oracle database using Oracle Net Services for connectivity.

If you create an Oracle RAC database as a CDB and plug one or more PDBs into the CDB, then, by default, a PDB is not started automatically on any instance of the Oracle RAC CDB. With the first dynamic database service assigned to the PDB (other than the default database service which has the same name as the database name), the PDB is made available on those instances on which the service runs.

Whether a PDB is available on more than one instance of an Oracle RAC CDB, the CDB is typically managed by the services running on the PDB. You can manually

enable PDB access on each instance of an Oracle RAC CDB by starting the PDB manually on that instance.

Overview of In-Memory Column Store with Oracle RAC

Each node in an Oracle RAC environment has its own In-Memory (IM) column store. Oracle recommends that you equally size the IM column stores on each Oracle RAC node. For any Oracle RAC node that does not require an IM column store, set the `INMEMORY_SIZE` parameter to 0.

It is possible to have completely different objects populated on every node, or to have larger objects distributed across all of the IM column stores in the cluster. It is also possible to have the same objects appear in the IM column store on every node (on engineered systems, only). The distribution of objects across the IM column stores in a cluster is controlled by two additional sub-clauses to the `INMEMORY` attribute; `DISTRIBUTE` and `DUPLICATE`.

In an Oracle RAC environment, an object that only has the `INMEMORY` attribute specified on it is automatically distributed across the IM column stores in the cluster. The `DISTRIBUTE` clause can be used to specify how an object is distributed across the cluster. By default, the type of partitioning used (if any) determines how the object is distributed. If the object is not partitioned it is distributed by rowid range. Alternatively, you can specify the `DISTRIBUTE` clause to over-ride the default behavior.

On an Engineered System, it is possible to duplicate or mirror objects populated in memory across the IM column store in the cluster. This provides the highest level of redundancy. The `DUPLICATE` clause is used to control how an object should be duplicated across the IM column stores in the cluster. If you specify just `DUPLICATE`, then one mirrored copy of the data is distributed across the IM column stores in the cluster. If you want to duplicate the entire object in each IM column store in the cluster, then specify `DUPLICATE ALL`.

> **Note:**
>
> When you deploy Oracle RAC on a non-Engineered System, the `DUPLICATE` clause is treated as `NO DUPLICATE`.

> **See Also:**
>
> *Oracle Database Concepts* for more information about IM column store

Overview of Managing Oracle RAC Environments

This section describes the following Oracle RAC environment management topics:

- About Designing and Deploying Oracle RAC Environments

- About Administrative Tools for Oracle RAC Environments

- About Monitoring Oracle RAC Environments

- About Evaluating Performance in Oracle RAC Environments

About Designing and Deploying Oracle RAC Environments

Any enterprise that is designing and implementing a high availability strategy with Oracle RAC must begin by performing a thorough analysis of the business drivers that require high availability. An analysis of business requirements for high availability combined with an understanding of the level of investment required to implement different high availability solutions enables the development of a high availability architecture that achieves both business and technical objectives.

See Also:

For help choosing and implementing the architecture that best fits your availability requirements:

- Design and Deployment Techniques provides a high-level overview you can use to evaluate the high availability requirements of your business.

- *Oracle Database High Availability Overview* describes how to select the most suitable architecture for your organization, describes several high availability architectures, and provides guidelines for choosing the one that best meets your requirements.

- *Oracle Database High Availability Overview* for information about the Oracle Maximum Availability Architecture

---

About Administrative Tools for Oracle RAC Environments

You administer a cluster database as a single-system image using the Server Control Utility (SRVCTL), Oracle Enterprise Manager, or SQL*Plus

- **Server Control Utility (SRVCTL)**: SRVCTL is a command-line interface that you can use to manage an Oracle RAC database from a single point. You can use SRVCTL to start and stop the database and instances and to delete or move instances and services. You can also use SRVCTL to manage configuration information, Oracle Real Application Clusters One Node (Oracle RAC One Node), Oracle Clusterware, and Oracle ASM.

- **Oracle Enterprise Manager**: Oracle Enterprise Manager Cloud Control GUI interface for managing both noncluster database and Oracle RAC database environments. Oracle recommends that you use Oracle Enterprise Manager to perform administrative tasks whenever feasible.

  You can use Oracle Enterprise Manager Cloud Control to also manage Oracle RAC One Node databases.

- **SQL*Plus**: SQL*Plus commands operate on the current instance. The current instance can be either the local default instance on which you initiated your SQL*Plus session, or it can be a remote instance to which you connect with Oracle Net Services.

- **Cluster Verification Utility (CVU)**: CVU is a command-line tool that you can use to verify a range of cluster and Oracle RAC components, such as shared storage devices, networking configurations, system requirements, and Oracle Clusterware, in addition to operating system groups and users. You can use CVU for preinstallation checks and for postinstallation checks of your cluster

environment. CVU is especially useful during preinstallation and during installation of Oracle Clusterware and Oracle RAC components. Oracle Universal Installer runs CVU after installing Oracle Clusterware and Oracle Database to verify your environment.

Install and use CVU before you install Oracle RAC to ensure that your configuration meets the minimum Oracle RAC installation requirements. Also, use CVU for verifying the completion of ongoing administrative tasks, such as node addition and node deletion.

- **DBCA**: The recommended utility for creating and initially configuring Oracle RAC, Oracle RAC One Node, and Oracle noncluster databases.

- **NETCA**: Configures the network for your Oracle RAC environment.

See Also:

- [Administering Database Instances and Cluster Databases](#) for an introduction to Oracle RAC administration using SRVCTL, Oracle Enterprise Manager, and SQL*Plus

- "[Monitoring Oracle RAC and Oracle Clusterware](#)"

- [Server Control Utility Reference](#) for SRVCTL reference information

- *Oracle Clusterware Administration and Deployment Guide* for information about the Cluster Verification Utility (CVU), in addition to other Oracle Clusterware tools, such as the OIFCFG tool for allocating and deallocating network interfaces and the OCRCONFIG command-line tool for managing OCR

- *Oracle Database Net Services Administrator's Guide* for more information about NETCA

About Monitoring Oracle RAC Environments

Web-based Oracle Enterprise Manager Cloud Control enables you to monitor an Oracle RAC database. Oracle Enterprise Manager Cloud Control is a central point of control for the Oracle environment that you access by way of a graphical user interface (GUI). See "[Monitoring Oracle RAC and Oracle Clusterware](#)" and the *Oracle Database 2 Day + Real Application Clusters Guide* for detailed information about using Oracle Enterprise Manager to monitor Oracle RAC environments.

Also, note the following recommendations about monitoring Oracle RAC environments:

- Use Oracle Enterprise Manager Cloud Control to initiate cluster database management tasks.

- Use Oracle Enterprise Manager Cloud Control to administer multiple or individual Oracle RAC databases.

- Use the global views (GV$ views), which are based on V$ views. The catclustdb.sql script creates the GV$ views. Run this script if you do not create your database with DBCA. Otherwise, DBCA runs this script for you.

For almost every `V$` view, there is a corresponding global `GV$` view. In addition to the `V$` information, each `GV$` view contains an extra column named `INST_ID`, which displays the instance number from which the associated `V$` view information was obtained.

- Use the sophisticated management and monitoring features of the Oracle Database Diagnostic and Tuning packs within Oracle Enterprise Manager that include the Automatic Database Diagnostic Monitor (ADDM) and Automatic Workload Repository (AWR).

> **Note:**
>
> Although Statspack is available for backward compatibility, Statspack provides reporting only. You must run Statspack at level 7 to collect statistics related to block contention and segment block waits.

> **See Also:**
>
> - *Oracle Database Performance Tuning Guide* for descriptions of the Oracle Database automatic features for performance diagnosing and tuning, including ADDM
>
> - *Oracle Database 2 Day + Real Application Clusters Guide* for more information about monitoring performance and troubleshooting

About Evaluating Performance in Oracle RAC Environments

You do not need to perform special tuning for Oracle RAC; Oracle RAC scales without special configuration changes. If your application performs well on a noncluster Oracle database, then it will perform well in an Oracle RAC environment. Many of the tuning tasks that you would perform on a noncluster Oracle database can also improve Oracle RAC database performance. This is especially true if your environment requires scalability across a greater number of CPUs.

Some of the performance features specific to Oracle RAC include:

- Dynamic resource allocation

    - Oracle Database dynamically allocates Cache Fusion resources as needed

    - The dynamic mastering of resources improves performance by keeping resources local to data blocks

- Cache Fusion enables a simplified tuning methodology

    - You do not have to tune any parameters for Cache Fusion

    - No application-level tuning is necessary

    - You can use a bottom-up tuning approach with virtually no effect on your existing applications

- More detailed performance statistics

    - More views for Oracle RAC performance monitoring

- – Oracle RAC-specific performance views in Oracle Enterprise Manager

2

Administering Storage in Oracle RAC

Oracle Automatic Storage Management (Oracle ASM) is a **volume manager** and a **file system** for Oracle database files that supports single-instance Oracle Database and Oracle Real Application Clusters (Oracle RAC) configurations. Oracle recommends Oracle ASM as a storage management solution that provides an alternative to conventional volume managers, file systems, and raw devices.

Oracle ASM uses disk groups to store data files; an Oracle ASM disk group is a collection of disks that Oracle ASM manages as a unit. Within a disk group, Oracle ASM exposes a file system interface for Oracle database files. The content of files that are stored in a disk group is evenly distributed to eliminate hot spots and to provide uniform performance across the disks. The performance is comparable to the performance of raw devices.

You can add or remove disks from a disk group while a database continues to access files from the disk group. When you add or remove disks from a disk group, Oracle ASM automatically redistributes the file contents and eliminates the need for downtime when redistributing the content.

> **See Also:**
>
> *Oracle Automatic Storage Management Administrator's Guide* for information about administering disk groups

The Oracle ASM volume manager functionality provides flexible server-based mirroring options. The Oracle ASM normal and high redundancy disk groups enable two-way and three-way mirroring respectively. You can use external redundancy to enable a Redundant Array of Independent Disks (RAID) storage subsystem to perform the mirroring protection function.

Oracle ASM also uses the Oracle Managed Files feature to simplify database file management. Oracle Managed Files automatically creates files in designated locations. Oracle Managed Files also names files and removes them while relinquishing space when tablespaces or files are deleted.

Oracle ASM reduces the administrative overhead for managing database storage by consolidating data storage into a small number of disk groups. The smaller number of disk groups consolidates the storage for multiple databases and provides for improved I/O performance.

Oracle ASM files can coexist with other storage management options such as raw disks and third-party file systems. This capability simplifies the integration of Oracle ASM into pre-existing environments.

Oracle ASM has easy to use management interfaces such as SQL*Plus, the Oracle ASM Command Line Utility (ASMCMD) command-line interface, and Oracle ASM Configuration Assistant (ASMCA).

> **See Also:**
>
> *Oracle Automatic Storage Management Administrator's Guide* for information about ASMCMD and ASMCA

This chapter includes the following topics:

- Overview of Storage Management for Oracle RAC

- Data File Access in Oracle RAC

- Redo Log File Storage in Oracle RAC

- Automatic Undo Management in Oracle RAC

- Oracle Automatic Storage Management with Oracle RAC

Overview of Storage Management for Oracle RAC

All data files (including an undo tablespace for each instance) and redo log files (at least two for each instance) for an Oracle RAC database must reside on shared storage. Oracle recommends that you use Oracle ASM to store these files in an Oracle ASM disk group.

Oracle supports alternative ways of using shared storage, such as certified **cluster file system**s. In addition, Oracle recommends that you use one shared server parameter file (SPFILE) with instance-specific entries. Oracle RAC 12*c* allows storing shared password files in Oracle ASM and storing Oracle Database files on Oracle Automatic Storage Management Cluster File System (Oracle ACFS).

> **Note:**
>
> Oracle Database and related technologies, such as Oracle Clusterware, no longer support the use of raw (block) storage devices. You must move files to Oracle ASM before upgrading to Oracle Clusterware 12*c*.

Unless otherwise noted, Oracle Database storage features such as Oracle ASM, Oracle Managed Files, automatic segment-space management, and so on, function the same in Oracle RAC environments as they do in noncluster Oracle database environments.

> **See Also:**
>
> For additional information about these storage features:
>
> - *Oracle Database 2 Day DBA*
>
> - *Oracle Automatic Storage Management Administrator's Guide*
>
> - *Oracle Database Administrator's Guide*

> **Note:**
>
> To create an Oracle RAC database using Oracle Database Standard Edition, you must use Oracle ASM for your database storage.

Data File Access in Oracle RAC

All Oracle RAC instances must be able to access all data files. If a data file must be recovered when the database is opened, then the first Oracle RAC instance to start is the instance that performs the recovery and verifies access to the file. As other instances start, they also verify their access to the data files. Similarly, when you add a tablespace or data file or bring a tablespace or data file online, all instances verify access to the file or files.

If you add a data file to a disk that other instances cannot access, then verification fails. Verification also fails if instances access different copies of the same data file. If verification fails for any instance, then diagnose and fix the problem. Then run the `ALTER SYSTEM CHECK DATAFILES` statement on each instance to verify data file access.

Redo Log File Storage in Oracle RAC

In an Oracle RAC database, each instance must have at least two groups of redo log files. You must allocate the redo log groups before enabling a new instance with the `ALTER DATABASE ENABLE INSTANCE instance_name` command. When you use DBCA to create the database, DBCA allocates redo log files to instances, as required, automatically. You can change the number of redo log groups and the size of the redo log files as required either during the initial database creation or as a post-creation step.

When the current group fills, an instance begins writing to the next log file group. If your database is in `ARCHIVELOG` mode, then each instance must save filled online log groups as archived redo log files that are tracked in the control file. During database recovery, all enabled instances are checked to see if recovery is needed. If you remove an instance from your Oracle RAC database, then you should disable the instance's thread of redo so that Oracle does not have to check the thread during database recovery.

Redo log management must be considered when the number of instances for a particular production Oracle RAC database changes. For example, if you increase the cardinality of a server pool for a policy-managed database and a new server is allocated to the server pool, then Oracle starts an instance on the new server. As soon as the database instance on the new server starts, it will require a set of redo log groups. Using Oracle Managed Files based on an Oracle ASM disk group, allocation of the required redo log thread and respective files is performed automatically. You should create redo log groups only if you use administrator-managed databases.

For administrator-managed databases, each instance has its own online redo log groups. Create these redo log groups and establish group members. To add a redo log group to a specific instance, specify the `INSTANCE` clause in the `ALTER DATABASE ADD LOGFILE` statement. If you do not specify the instance when adding the redo log group, then the redo log group is added to the instance to which you are currently connected.

> **See Also:**
>
> - "About Designing and Deploying Oracle RAC Environments" for more information about administrator and policy management for databases
>
> - *Oracle Database Administrator's Guide* for information about creating redo log groups and establishing group members
>
> - *Oracle Database SQL Language Reference* for information about the `ALTER DATABASE ADD LOGFILE` SQL statement

Each instance must have at least two groups of redo log files. You must allocate the redo log groups before enabling a new instance with the `ALTER DATABASE ENABLE INSTANCE instance_name` command. When the current group fills, an instance begins writing to the next log file group. If your database is in `ARCHIVELOG` mode, then each instance must save filled online log groups as archived redo log files that are tracked in the control file.

During database recovery, all enabled instances are checked to see if recovery is needed. If you remove an instance from your Oracle RAC database, then you should disable the instance's thread of redo so that Oracle does not have to check the thread during database recovery.

Automatic Undo Management in Oracle RAC

Oracle Database automatically manages undo segments within a specific undo tablespace that is assigned to an instance. Instances can always read all undo blocks throughout the **cluster** environment for consistent read purposes. Also, any instance can update any undo tablespace during transaction recovery, if that undo tablespace is not currently used by another instance for undo generation or transaction recovery.

You assign undo tablespaces in your Oracle RAC administrator-managed database by specifying a different value for the `UNDO_TABLESPACE` parameter for each instance in your SPFILE or individual PFILEs. For policy-managed databases, Oracle automatically allocates the undo tablespace when the instance starts if you have Oracle Managed Files enabled. You cannot simultaneously use automatic undo management and manual undo management in an Oracle RAC database. In other words, all instances of an Oracle RAC database must operate in the same undo mode.

> **See Also:**
>
> - "Setting SPFILE Parameter Values for Oracle RAC" for information about modifying SPFILE parameters
>
> - *Oracle Database Administrator's Guide* for detailed information about creating and managing undo tablespaces

Oracle Automatic Storage Management with Oracle RAC

Oracle ASM automatically maximizes I/O performance by managing the storage configuration across the disks that Oracle ASM manages. Oracle ASM does this by evenly distributing the database files across all of the available storage assigned to the disk groups within Oracle ASM. Oracle ASM allocates your total disk space

requirements into uniformly sized units across all disks in a disk group. Oracle ASM can also automatically mirror files to prevent data loss. Because of these features, Oracle ASM also significantly reduces your administrative overhead.

Oracle ASM instances are created on each node where you install Oracle Clusterware. Each Oracle ASM instance has either an SPFILE or PFILE type parameter file. Oracle recommends that you back up the parameter files and the TNS entries for nondefault Oracle Net listeners.

To use Oracle ASM with Oracle RAC, select Oracle ASM as your storage option when you create your database with the Database Configuration Assistant (DBCA). As in noncluster Oracle databases, using Oracle ASM with Oracle RAC does not require I/O tuning.

The following topics describe Oracle ASM and Oracle ASM administration, as follows:

- Storage Management in Oracle RAC

- Modifying Disk Group Configurations for Oracle ASM

- Oracle ASM Disk Group Management

- Configuring Preferred Mirror Read Disks in Extended Distance Clusters

- Converting Nonclustered Oracle ASM to Clustered Oracle ASM

- Administering Oracle ASM Instances with SRVCTL in Oracle RAC

See Also:

Oracle Automatic Storage Management Administrator's Guide for complete information about managing Oracle ASM

Storage Management in Oracle RAC

You can create Oracle ASM disk groups and configure mirroring for Oracle ASM disk groups using the Oracle ASM configuration assistant (ASMCA). Alternatively, you can use Oracle Enterprise Manager to administer Oracle ASM disk groups after you have discovered the respective servers with Oracle Enterprise Manager.

The Oracle tools that you use to manage Oracle ASM, including ASMCA, Oracle Enterprise Manager, and the silent mode install and upgrade commands, include options to manage Oracle ASM instances and disk groups.

You can use the Cluster Verification Utility (CVU) to verify the integrity of Oracle ASM across the **cluster**. Typically, this check ensures that the Oracle ASM instances on all nodes run from the same Oracle home and, if `asmlib` exists, that it is a valid version and has valid ownership. Run the following command to perform this check:

```
cluvfy comp asm [-n node_list] [-verbose]
```

Replace `node_list` with a comma-delimited list of node names on which the check is to be performed. Specify `all` to check all nodes in the cluster.

Use the `cluvfy comp ssa` command to locate shared storage.

> **See Also:**
>
> *Oracle Clusterware Administration and Deployment Guide* for more information about CVU

Modifying Disk Group Configurations for Oracle ASM

When you create a disk group for a cluster or add new disks to an existing clustered disk group, prepare the underlying physical storage on shared disks and give the Oracle user permission to read and write to the disk. The shared disk requirement is the only substantial difference between using Oracle ASM with an Oracle RAC database compared to using it with a noncluster Oracle database. Oracle ASM automatically redistributes the data files after you add or delete a disk or disk group.

In a cluster, each Oracle ASM instance manages its node's metadata updates to the disk groups. In addition, each Oracle ASM instance coordinates disk group metadata with other nodes in the cluster. As with noncluster Oracle databases, you can use Oracle Enterprise Manager, ASMCA, SQL*Plus, and the Server Control Utility (SRVCTL) to administer disk groups for Oracle ASM that are used by Oracle RAC. The *Oracle Automatic Storage Management Administrator's Guide* explains how to use SQL*Plus to administer Oracle ASM instances. Subsequent sections describe how to use the other tools.

> **Note:**
>
> When you start ASMCA, if there is not an Oracle ASM instance, then the utility prompts you to create one.

Oracle ASM Disk Group Management

To use Oracle ASM, you must first create disk groups with ASMCA before creating a database with DBCA. You can also use the disk group management commands to create and manage an Oracle ASM instance and its associated disk groups independently of creating a database. You can use Oracle Enterprise Manager or DBCA to add disks to a disk group, to mount a disk group or to mount all of the disk groups, or to create Oracle ASM instances. Additionally, you can use Oracle Enterprise Manager to dismount and drop disk groups or to delete Oracle ASM instances.

Oracle ASM instances are created when you install Oracle Clusterware. To create an Oracle ASM disk group, run ASMCA from the `Grid_home/bin` directory. You can also use the Oracle ASM Disk Groups page in ASMCA for Oracle ASM management. That is, you can configure Oracle ASM storage separately from database creation. For example, from the ASM Disk Groups page, you can create disk groups, add disks to existing disk groups, or mount disk groups that are not currently mounted.

> **See Also:**
>
> *Oracle Automatic Storage Management Administrator's Guide* for information about managing Oracle ASM

When you start ASMCA, if the Oracle ASM instance has not been created, then ASMCA prompts you to create the instance. ASMCA prompts you for the `sysasm` password and the `ASMSNMP` password.

Configuring Preferred Mirror Read Disks in Extended Distance Clusters

When you configure Oracle ASM **failure group**s, it may be more efficient for a node to read from an extent that is closest to the node, even if that extent is a secondary extent. You can configure Oracle ASM to read from a secondary extent if that extent is closer to the node instead of Oracle ASM reading from the primary copy which might be farther from the node. Using preferred read failure groups is most beneficial in an **extended distance cluster**.

To configure this feature, set the ASM_PREFERRED_READ_FAILURE_GROUPS initialization parameter to specify a list of failure group names as preferred read disks. Oracle recommends that you configure at least one mirrored extent copy from a disk that is local to a node in an extended cluster. However, a failure group that is preferred for one instance might be remote to another instance in the same Oracle RAC database. The parameter setting for preferred read failure groups is instance specific.

See Also:

- *Oracle Automatic Storage Management Administrator's Guide* for complete information about configuring preferred mirror read disks in extended distance clusters

- *Oracle Database Reference* for information about the ASM_PREFERRED_READ_FAILURE_GROUPS initialization parameter

Converting Nonclustered Oracle ASM to Clustered Oracle ASM

When installing Oracle Grid Infrastructure, any nonclustered Oracle ASM instances are automatically converted to clustered Oracle ASM.

See Also:

- *Oracle Database 2 Day + Real Application Clusters Guide* for information about using Oracle Enterprise Manager Cloud Control to convert nonclustered Oracle ASM to clustered Oracle ASM

- *Oracle Automatic Storage Management Administrator's Guide* for complete information about configuring preferred mirror read disks in extended distance clusters

- *Oracle Real Application Clusters Installation Guide* for your platform for detailed information about converting Oracle ASM using the rconfig command

Administering Oracle ASM Instances with SRVCTL in Oracle RAC

You can use the Server Control Utility (SRVCTL) to add or remove an Oracle ASM instance. To issue SRVCTL commands to manage Oracle ASM, log in as the operating system user that owns the Oracle Grid Infrastructure home and issue the SRVCTL commands from the bin directory of the Oracle Grid Infrastructure home.

Use the following syntax to add an Oracle ASM instance:

```
srvctl add asm
```

Use the following syntax to remove an Oracle ASM instance:

```
srvctl remove asm [-force]
```

You can also use SRVCTL to start, stop, and obtain the status of an Oracle ASM instance as in the following examples.

Use the following syntax to start an Oracle ASM instance:

```
srvctl start asm [-node node_name] [-startoption start_options]
```

Use the following syntax to stop an Oracle ASM instance:

```
srvctl stop asm [-node node_name] [-stopoption stop_options]
```

Use the following syntax to show the configuration of an Oracle ASM instance:

```
srvctl config asm -node node_name
```

Use the following syntax to display the state of an Oracle ASM instance:

```
srvctl status asm [-node node_name]
```

> **See Also:**
>
> - Server Control Utility Reference for more SRVCTL commands you can use to administer Oracle ASM
>
> - *Oracle Automatic Storage Management Administrator's Guide* for more information about administering Oracle ASM instances

3

Administering Database Instances and Cluster Databases

This chapter describes how to administer Oracle Real Application Clusters (Oracle RAC) databases and database instances.

The topics in this chapter include:

- Overview of Oracle RAC Database Administration

- Starting and Stopping Instances and Oracle RAC Databases

- Starting and Stopping PDBs in Oracle RAC

- Verifying That Instances are Running

- Terminating Sessions On a Specific Cluster Instance

- Overview of Initialization Parameter Files in Oracle RAC

- Initialization Parameter Use in Oracle RAC

- Converting an Administrator-Managed Database to a Policy-Managed Database

- Managing Memory Pressure for Database Servers

- Quiescing Oracle RAC Databases

- Administering Multiple Cluster Interconnects on Linux and UNIX Platforms

- Customizing How Oracle Clusterware Manages Oracle RAC Databases

- Advanced Oracle Enterprise Manager Administration

See Also:

The Oracle Enterprise Manager Cloud Control online help for more information about Oracle Enterprise Manager Cloud Control

---

Overview of Oracle RAC Database Administration

Oracle RAC databases support two different management styles and deployment models:

- Administrator-managed deployment is based on the Oracle RAC deployment types that existed before Oracle Database 11*g* release 2 (11.2) and requires that you statically configure each database instance to run on a specific node in the cluster,

and that you configure database services to run on specific instances belonging to a certain database using the `preferred` and `available` designation.

- Policy-managed deployment is based on *server pools*, where database services run within a server pool as *singleton* or *uniform* across all of the servers in the server pool. Databases are deployed in one or more server pools and the size of the server pools determine the number of database instances in the deployment.

You can manage databases with either the administrator-managed or policy-managed deployment model using the same commands or methods (such as DBCA or Oracle Enterprise Manager). All commands and utilities maintain backward compatibility to support the management of Oracle databases that only support administrator-based management (Oracle databases before Oracle Database 11*g* release 2 (11.2)).

In general, a database is defined as a resource in Oracle Clusterware. The database resource is automatically created when you create your database with DBCA, or you can manually create the database resource by adding your database with SRVCTL. The database resource contains the Oracle home, the SPFILE, one or more server pools, and one or more Oracle ASM disk groups required for the database. The database resource also has a *weak start dependency* on the listener type, which means that the resource tries to start all listeners for the node when the database instance starts. Oracle Clusterware tries to start listeners on the node where the database instance starts. Starting the listeners in turn starts the VIP for the node.

When you review the database resource for an administrator-managed database, you see a server pool defined with the same name as the Oracle database. This server pool is part of a special Oracle-defined server pool called *Generic*. Oracle RAC manages the Generic server pool to support administrator-managed databases. When you add or remove an administrator-managed database using either SRVCTL or DBCA, Oracle RAC creates or removes the server pools that are members of Generic. You cannot use SRVCTL or CRSCTL commands to modify the Generic server pool.

See Also:

Oracle Clusterware Administration and Deployment Guide for more information about defining resources, server pools, and resource dependencies

---

Use **policy-managed database**s to simplify management of dynamic systems. Policy management allows clusters and databases to expand or shrink as requirements change. If you use policy-managed databases, then you must install the Oracle home software on every node in your cluster. Policy-managed databases must use Oracle Database 11*g* release 2 (11.2) or higher software and cannot coexist on the same servers as administrator-managed databases.

Note:

You cannot run more than one instance of the same database on the same node.

A policy-managed database is defined by **cardinality**, which is the number of database instances you want running during normal operations. A policy-managed database runs in one or more database server pools that the cluster administrator creates in the cluster, and it can run on different servers at different times. Every server pool of a policy-managed database should have at least one database service. A database

instance starts on a server that is in the server pools defined for the database. If you are using Oracle Automatic Storage Management (Oracle ASM) with Oracle Managed Files for your database storage, then, when an instance starts and there is no redo thread available, Oracle RAC automatically enables one and creates the required redo log files and undo tablespace. Clients can connect to a policy-managed database using the same SCAN-based connect string no matter which servers they happen to be running on at the time.

> **See Also:**
>
> "Converting an Administrator-Managed Database to a Policy-Managed Database"

Using the Same Cluster for Administrator-Managed and Policy-Managed Databases

If you want to create an administrator-managed database on a cluster that already hosts policy-managed databases, then you must carefully select the nodes for the administrator-managed database. This is because the nodes that you select for an administrator-managed database that are in policy-managed server pools will be moved into the Generic server pool as part of this process.

If you select nodes that already run other policy-managed database instances, then DBCA prompts you with a message that lists the instances and services that will be shut down when DBCA creates the administrator-managed database. If you select the **Yes** button on the dialog box when DBCA asks "Do you want to continue?", then your policy-managed database instances and services will be shut down because of the administrator-managed database creation process.

> **Note:**
>
> This is also true if you use the `srvctl add instance` command, which returns a similar error message indicating that the databases would be shut down. If you also use the force option (`-f`) with the `srvctl add instance` command, then this is the same as choosing **Yes** on the DBCA dialog. Doing this shuts down any policy-managed databases that are running on the node before moving the node into the Generic server pool.

Tools for Administering Oracle RAC

The following sections introduce Oracle RAC administration using the three tools that you commonly use to manage Oracle RAC databases and instances: the SRVCTL utility, Oracle Enterprise Manager, and SQL*Plus. In many cases, you use these tools the same way to manage Oracle RAC environments as you would use them manage noncluster Oracle databases:

- Administering Oracle RAC with SRVCTL

- Administering Oracle RAC with Oracle Enterprise Manager

- Administering Oracle RAC with SQL*Plus

Administering Oracle RAC with SRVCTL

The Server Control Utility (SRVCTL) is a command-line interface that you can use to manage Oracle Databases in a centralized manner. Oracle made centralized, SRVCTL-based database management available in Oracle Database 11*g* release 2 (11.2) for single-instance Oracle Databases, using Oracle ASM in the Oracle Grid Infrastructure, for both a noncluster environment and Oracle RAC databases, based on Oracle Grid Infrastructure for a cluster. This enables homogeneous management of all Oracle Database types using SRVCTL. You can use SRVCTL to start and stop the database and instances, and to delete or move instances and services. You can also use SRVCTL to add services and manage configuration information, in addition to other resources in the cluster.

When you use SRVCTL to perform configuration operations on your cluster, SRVCTL stores configuration data in the Oracle Cluster Registry (OCR) in a cluster or Oracle Local Registry (OLR) in Oracle Restart environments. SRVCTL performs other operations, such as starting and stopping instances, by configuring and managing Oracle Clusterware resources, which define agents that perform database startup and shutdown operations using Oracle Call Interface APIs.

> **Note:**
>
> If you require your database (or database instance) to start using certain environment variables, then use the `srvctl setenv` command to set those variables for the database profile that is maintained for the database using SRVCTL. You do not need to set the `ORACLE_HOME` and `ORACLE_SID` environment variables, because SRVCTL maintains and sets those parameters, automatically.

> **See Also:**
>
> Server Control Utility Reference for more information about SRVCTL

Administering Oracle RAC with Oracle Enterprise Manager

Oracle Enterprise Manager provides a central point of control for the Oracle RAC environment, allowing you to perform administrative tasks simultaneously on multiple cluster databases. Based on the Oracle Enterprise Manager Cloud Control (Grid Control in Oracle Enterprise Manager 11*g*) graphical user interface (GUI), you can manage both non-clustered and Oracle RAC environments.

In Oracle Enterprise Manager, Oracle RAC-specific administrative tasks generally focus on two levels: tasks that affect an entire **cluster database** and tasks that affect specific instances. For example, you can use Oracle Enterprise Manager to start, stop, and monitor databases, cluster database instances, and their listeners, and to schedule jobs or set up alert thresholds for metrics. Or you can perform instance-specific commands such as setting parameters or creating resource plans. You can also use Oracle Enterprise Manager to manage schemas, security, and cluster database storage features.

> **See Also:**
>
> - *Oracle Database 2 Day + Real Application Clusters Guide* for a task-oriented guide that explains how to use Oracle Enterprise Manager to perform routine Oracle RAC database administrative tasks
>
> - "Advanced Oracle Enterprise Manager Administration" for advanced administration tasks not covered in *Oracle Database 2 Day + Real Application Clusters Guide*

Administering Oracle RAC with SQL*Plus

Unlike SRVCTL or Oracle Enterprise Manager, SQL*Plus is an instance-oriented management tool. This means that SQL*Plus commands operate on the current instance. The current instance can be either the local default instance on which you initiated your SQL*Plus session, or it can be a remote instance to which you connect with Oracle Net Services. For an Oracle RAC environment that runs multiple instances on one database at the same time, this implies that you need to consider the extent to which SQL*Plus can operate on this instance. Due to those restrictions, you should not use SQL*Plus to manage policy-managed databases.

For example, when using pluggable databases (PDBs)—regardless of whether those databases are managed in an administrator-managed or a policy-managed style—you must consider that any alteration performed on the PDB using a SQL*Plus connection will, by default, only affect the current instance. To make changes affecting all instances that belong to the PDB, you must use the `ALTER PLUGGABLE DATABASE` command with `instance=all`. When using PDBs you must connect, using a dynamic database service (`net_service_name`), to an instance, as PDBs represent themselves as dynamic database services associated with one or more instances of an Oracle RAC database.

> **See Also:**
>
> *Oracle Database Administrator's Guide* for more information

Because, by default, the SQL*Plus prompt does not identify the current instance, you should direct your commands to the correct instance. Starting a SQL*Plus session and connecting to the database without specifying an instance directs all SQL*Plus commands to the local instance. In this case, the default instance is also the current instance.

Since the SQL*Plus prompt does not identify the current instance by default, you should direct your commands to the correct instance. Starting a SQL*Plus session and connecting to the database without specifying an instance directs all SQL*Plus commands to the local instance. In this case, the default instance is also the current instance. To connect to a different instance in SQL*Plus, issue a new `CONNECT` command and specify a remote instance net service name, as shown in the following example, where *password* is the password:

```
CONNECT user_name@net_service_name
Enter password: password
```

Connecting as `SYSOPER` or `SYSDBA` enables you to perform privileged operations, such as instance startup and shutdown. Multiple SQL*Plus sessions can connect to the

same instance at the same time. SQL*Plus automatically disconnects you from the first instance whenever you connect to another one.

Note:

Use the SYSASM privilege instead of the SYSDBA privilege to connect to and administer an Oracle ASM instance. If you use the SYSDBA privilege to connect to an Oracle ASM instance, then Oracle Database writes warnings to the alert log files because commands that run using the SYSDBA privilege on an Oracle ASM instance are deprecated.

See *Oracle Automatic Storage Management Administrator's Guide* for more information.

See Also:

- *SQL*Plus User's Guide and Reference*

- *Oracle Database Net Services Administrator's Guide* for the proper specification of `net_service_name`

- *Oracle Database Administrator's Guide* for information about connecting to the database using SYSDBA or SYSOPER privileges

---

How SQL*Plus Commands Affect Instances

Most SQL statements affect the current instance. You can use SQL*Plus to start and stop instances in the Oracle RAC database. You do not need to run SQL*Plus commands as `root` on Linux and UNIX systems or as `Administrator` on Windows systems. You need only the proper database account with the privileges that you normally use for a noncluster Oracle database. Some examples of how SQL*Plus commands affect instances are:

- `ALTER SYSTEM CHECKPOINT LOCAL` affects only the instance to which you are currently connected, rather than the default instance or all instances.

- `ALTER SYSTEM CHECKPOINT` or `ALTER SYSTEM CHECKPOINT GLOBAL` affects *all* instances in the cluster database.

- `ALTER SYSTEM SWITCH LOGFILE` affects only the current instance.

  - To force a global log switch, use the `ALTER SYSTEM ARCHIVE LOG CURRENT` statement.

  - The `INSTANCE` option of `ALTER SYSTEM ARCHIVE LOG` enables you to archive each online redo log file for a specific instance.

Table 3-1 describes how SQL*Plus commands affect instances.

*Table 3-1   How SQL*Plus Commands Affect Instances*

SQL*Plus Command	Associated Instance
`ARCHIVE LOG`	Always affects the current instance.

Table 3-1   (Cont.) How SQL*Plus Commands Affect Instances

SQL*Plus Command	Associated Instance
CONNECT	Affects the default instance if no instance is specified in the CONNECT command.
HOST	Affects the node running the SQL*Plus session, regardless of the location of the current and default instances.
RECOVER	Does not affect any particular instance, but rather the database.
SHOW INSTANCE	Displays information about the current instance, which can be different from the default local instance if you have redirected your commands to a remote instance.
SHOW PARAMETER and SHOW SGA	Displays parameter and SGA information from the current instance.
STARTUP and SHUTDOWN	Always affects the current instance. These are privileged SQL*Plus commands.

Starting and Stopping Instances and Oracle RAC Databases

You can start and stop instances with Oracle Enterprise Manager, SQL*Plus or SRVCTL as described in the following sections. Both Oracle Enterprise Manager and SRVCTL provide options to start and stop all of the instances in an Oracle RAC database with a single step.

Using any tool, you can choose the startup state to which you want to start the database. The state of the database and database instance will determine what operations you can perform. You can perform certain operations only when the database is in the MOUNT (NOMOUNT) state. Performing other operations requires that the database be in the OPEN state.

> **Note:**
>
> Oracle does not support running more than one instance of the same database on the same node.

To start an Oracle RAC database instance on a node in the cluster, you must first start the Oracle Grid Infrastructure stack on the node. An Oracle RAC database instance will not start on a server on which the Oracle Grid Infrastructure stack is not running.

Oracle Database QoS Management Policy Workload Criticality Determines Database Startup Order

If a user-created Oracle Database Quality of Service Management (Oracle Database QoS Management) policy is active, then the ranked order of the performance classes determines the order in which the associated Oracle RAC databases start or request real-time LMS process slots. Using the performance class rankings ensures that mission-critical databases running in a consolidated environment have their LMS processes run in real time, thus eliminating a resource bottleneck within inter-node communication. Because the Oracle Database QoS Management policy specifies the rank of each workload, using the value of Max(Ranks) for each database provides a consistent expression of the expressed business criticality of each database.

The procedures in the following sections discuss starting and stopping Oracle RAC database instances:

- Starting One or More Instances and Oracle RAC Databases Using SRVCTL

- Stopping One or More Instances and Oracle RAC Databases Using SRVCTL

- Stopping All Databases and Instances Using CRSCTL

- Starting and Stopping Individual Instances Using SQL*Plus

> **See Also:**
>
> - *Oracle Database Concepts* for more information about database and instance startup states
>
> - *Oracle Database 2 Day + Real Application Clusters Guide* for step-by-step instructions on using Oracle Enterprise Manager to start or stop a cluster database instance or a cluster database.

Starting One or More Instances and Oracle RAC Databases Using SRVCTL

> **Note:**
>
> This section assumes that you are using an SPFILE for your database.

Enter the following SRVCTL syntax from the command line, providing the required database name and instance name, or include multiple instance names to start multiple specific instances:

- To start or stop your entire cluster database, that is, all of the instances and its enabled services, enter the following SRVCTL commands:

```
$ srvctl start database -db db_unique_name [-startoption start_options]
```

```
$ srvctl stop database -db db_unique_name [-o stop_options]
```

The following SRVCTL command, for example, mounts all of the non-running instances of an Oracle RAC database:

```
$ srvctl start database -db orcl -startoption mount
```

- To start administrator-managed databases, enter a comma-delimited list of instance names:

  ```
  $ srvctl start instance -db db_unique_name -instance instance_name_list [-
  startoption start_options]
  ```

  In Windows you must enclose a comma-delimited list in double quotation marks ("").

- To start policy-managed databases, enter a single node name:

  ```
  $ srvctl start instance -db db_unique_name -node node_name [-startoption
  start_options]
  ```

  Note that this command also starts all enabled and non-running services that have AUTOMATIC management policy, and for which the database role matches one of the service's roles.

- To stop one or more instances, enter the following SRVCTL syntax from the command line:

  ```
  $ srvctl stop instance -db db_unique_name [ -instance "instance_name_list" |
    -node node_name ] [ -stopoption stop_options ]
  ```

  You can enter either a comma-delimited list of instance names to stop several instances or you can enter a node name to stop one instance. In Windows you must enclose a comma-delimited list in double quotation marks ("").

This command also stops the services related to the terminated instances on the nodes where the instances were running. As an example, the following command shuts down the two instances, `orcl3` and `orcl4`, on the `orcl` database using the `immediate` stop option:

```
$ srvctl stop instance -db orcl -instance "orcl3,orcl4" -stopoption immediate
```

> **See Also:**
>
> Server Control Utility Reference for information about SRVCTL options and information about other administrative tasks that you can perform with SRVCTL

Stopping One or More Instances and Oracle RAC Databases Using SRVCTL

The procedure for shutting down Oracle RAC instances is identical to shutting down instances in noncluster Oracle databases, with the following exceptions:

- In Oracle RAC, shutting down one instance does not interfere with the operation of other running instances.

- To shut down an Oracle RAC database completely, shut down every instance that has the database open or mounted.

- After a NORMAL or IMMEDIATE shutdown, instance recovery is not required. Recovery is required, however, after you issue the SHUTDOWN ABORT command or after an instance terminates abnormally. An instance that is still running performs instance recovery for the instance that shut down. If no other instances are running, the next instance to open the database performs instance recovery for any instances needing it.

- Using the `SHUTDOWN TRANSACTIONAL` command with the `LOCAL` option is useful to shut down a particular Oracle RAC database instance. Transactions on other instances do not block this operation. If you omit the `LOCAL` option, then this operation waits until transactions on all other instances that started before you ran the `SHUTDOWN` command either commit or rollback, which is a valid approach, if you intend to shut down all instances of an Oracle RAC database.

> **Note:**
>
> `SHUTDOWN TRANSACTIONAL` and `SHUTDOWN TRANSACTIONAL LOCAL` both perform the same action on a nonclustered database but the two commands are different on an Oracle RAC database.

> **See Also:**
>
> *Oracle Database Concepts* for more information about shutting down a database

Use SRVCTL to shut down an Oracle RAC database or database instance. The respective SRVCTL commands (srvctl stop database or srvctl stop instance) provide shutdown options to perform an optimized transactional shutdown. Use the `TRANSACTIONAL` stop option with the `srvctl stop database` command and the `TRANSACTIONAL LOCAL` stop option with the `srvctl stop instance` command.

> **See Also:**
>
> *Oracle Database Administrator's Guide* for more information about shutting down Oracle databases

Stopping All Databases and Instances Using CRSCTL

When you want to stop an entire node or cluster (for maintenance purposes, for example), you run either the `crsctl stop crs` command on the node or the `crsctl stop cluster -all` command, provided you have the required cluster privileges. These commands stop all database instances running on a server or in the cluster and ensure that their state is recovered after you restart the cluster. Using CRSCTL also enables Oracle Clusterware to relocate services and other resources that can run elsewhere.

Using either of these CRSCTL commands to stop all database instances on a server or in the cluster can lead to the database instances being stopped similar to *shutdown abort*, which requires an instance recovery on startup. If you use SRVCTL to stop the database instances manually before stopping the cluster, then you can prevent a *shutdown abort*, but this requires that you manually restart the database instances after restarting Oracle Clusterware.

Starting and Stopping Individual Instances Using SQL*Plus

If you want to start or stop just one instance and you are connected to your local node, you should first ensure that your current environment includes the SID for the local instance. Note that any subsequent commands in your session, whether inside or outside a SQL*Plus session, are associated with that same SID.

> **Note:**
>
> This section assumes you are using an SPFILE.

To start or shutdown your local instance, initiate a SQL*Plus session and connect with the SYSDBA or SYSOPER privilege and then issue the required command. For example to start and mount an instance on your local node, run the following commands in your SQL*Plus session:

```
CONNECT / AS SYSDBA
STARTUP MOUNT
```

> **Note:**
>
> If you use Oracle ASM disk groups, then use the SYSASM privilege instead of the SYSDBA privilege to connect to and administer the Oracle ASM instances.
>
> Oracle recommends that you do not use SQL*Plus to manage Oracle ASM instances in an Oracle RAC environment. Oracle Clusterware automatically manages Oracle ASM instances, as required. If manual intervention is necessary, then use respective SRVCTL commands.

> **See Also:**
>
> *Oracle Automatic Storage Management Administrator's Guide* for more information

You can start multiple instances from a single SQL*Plus session on one node using Oracle Net Services. Connect to each instance in turn by using a Net Services connection string, typically an instance-specific alias from your tnsnames.ora file.

For example, you can use a SQL*Plus session on a local node to perform a transactional shutdown for two instances on remote nodes by connecting to each in turn using the instance's individual alias name. Assume the alias name for the first instance is db1 and that the alias for the second instance is db2. Connect to the first instance and shut it down as follows:

```
CONNECT /@db1 AS SYSDBA
SHUTDOWN TRANSACTIONAL
```

> **Note:**
>
> To ensure that you connect to the correct instance, you must use an alias in the connect string that is associated with just one instance. If you use a connect string that uses a TNS alias that connects to a service or an Oracle Net address that lists multiple IP addresses, then you might not be connected to the specific instance you want to shut down.

Then connect to and shutdown the second instance by entering the following from your SQL*Plus session:

```
CONNECT /@db2 AS SYSDBA
SHUTDOWN TRANSACTIONAL
```

It is not possible to start or stop multiple instances, simultaneously, with SQL*Plus, so you cannot start or stop all of the instances for a cluster database with a single SQL*Plus command. You may want to create a script that connects to each instance in turn and start it up and shut it down. However, you must maintain this script manually if you add or drop instances.

> **See Also:**
>
> *SQL*Plus User's Guide and Reference* for information about other startup and shutdown keywords, such as NOMOUNT, MOUNT, IMMEDIATE, and so on

Starting and Stopping PDBs in Oracle RAC

Administering an Oracle RAC-based multitenant container database (**CDB**) is somewhat similar to administering a non-CDB. The differences are only that some administrative tasks apply to the entire CDB, some apply only to the root, and some apply to specific pluggable databases (**PDB**s). Administering a pluggable database (PDB) involves a small subset of the tasks required to administer a non-CDB. In this subset of tasks, most are the same for a PDB and a non-CDB. There are some differences, however, such as when you modify the open mode of a PDB. Also, a PDB administrator is limited to managing a single PDB and is not affected by other PDBs in the CDB.

> **See Also:**
>
> *Oracle Database Administrator's Guide* for more information about managing CDBs and PDBs

You manage PDBs in an Oracle RAC-based CDB by managing services, regardless of whether the PDBs are policy managed or administrator managed. Assign one dynamic database service to each PDB to coordinate start, stop, and placement of PDBs across instances in a clustered container database.

For example, if you have a CDB called raccont with a policy-managed PDB called spark, which is in a server pool called prod, then assign a service called plug to this database using the following command:

```
srvctl add service –db raccont –pdb spark –service plug –serverpool prod
```

The service plug will be uniformly managed across all nodes in the server pool. If you want to have this service running as a singleton service in the same server pool, use the -cardinality singleton parameter with the preceding command.

To open the PDB spark, you must start the respective service, plug, as follows:

```
srvctl start service -db raccont -service plug
```

To stop the service, plug:

```
srvctl stop service -db raccont -service plug
```

The PDB spark remains open until you close the PDB using the SQL command ALTER PLUGGABLE DATABASE *PDB_NAME* CLOSE IMMEDIATE;. You can check the status of the database using the srvctl status service command.

Because PDBs are managed using dynamic database services, typical Oracle RAC-based management practices apply. So, if the service `plug` is in the ONLINE state when Oracle Clusterware is shut down on a server hosting this service, then the service will be restored to its original state after the restart of Oracle Clusterware on this server. This way, starting PDBs is automated as with any other Oracle RAC database.

> **Note:**
>
> Unlike SQL*Plus, SRVCTL operates on a cluster database, as a whole. Starting a PDB using services therefore applies to multiple instances of the clustered CDB at the same time, if the service is defined to run on multiple servers simultaneously, and the current status of the cluster allows for this placement.

> **See Also:**
>
> Workload Management with Dynamic Database Services for more information about dynamic database services

Verifying That Instances are Running

To verify that a database instance is available, use Oracle Enterprise Manager, SRVCTL, or SQL*Plus.

- Using SRVCTL to Verify That Instances are Running

- Using SQL*Plus to Verify That Instances are Running

> **See Also:**
>
> *Oracle Database 2 Day + Real Application Clusters Guide* for information about using Oracle Enterprise Manager to view the status of Oracle RAC database instances

Using SRVCTL to Verify That Instances are Running

You can use SRVCTL to verify that instances are running on a particular database.

The following command provides an example of using SRVCTL to check the status of the database instances for the Oracle RAC database named mail:

```
$ srvctl status database -db mail
```

This command returns output similar to the following:

```
Instance mail1 is running on node betal011Instance mail2 is running on node betal010
```

Additionally, you can check whether PDBs are running in the cluster by checking the availability of their assigned services, as follows:

```
$ srvctl status service -db db_unique_name -service service_name
```

Using SQL*Plus to Verify That Instances are Running

You can use SQL*Plus to verify that database instances are running.

1. On any node, from a SQL*Plus prompt, connect to a database instance by using a Net Services connection string, typically an instance-specific alias from your `tnsnames.ora` file.

   ```
   CONNECT /@db1 as SYSDBA
   ```

2. Query the V$ACTIVE_INSTANCES view, using the following statement:

   ```
   CONNECT SYS/as SYSDBA
   Enter password: password
   SELECT * FROM V$ACTIVE_INSTANCES;
   ```

   This query returns output similar to the following:

   ```
   INST_NUMBER INST_NAME
   ----------- -----------------
   1           db1-sun:db1
   2           db2-sun:db2
   3           db3-sun:db3
   ```

The output columns for this example are shown in Table 3-2.

Table 3-2   Descriptions of V$ACTIVE_INSTANCES Columns

Column	Description
INST_NUMBER	Identifies the instance number.
INST_NAME	Identifies the host name and instance name as *host_name*:*instance_name*.

Terminating Sessions On a Specific Cluster Instance

You can use the `ALTER SYSTEM KILL SESSION` statement to terminate a session on a specific instance.

When a session is terminated, any session active transactions are rolled back, and resources held by the session (such as locks and memory areas) are immediately released and available to other sessions.

Using the `ALTER SYSTEM KILL SESSION` statement enables you to maintain strict application service-level agreements in Oracle RAC environments. Often, the goal of a service-level agreement is to carry out a transaction in a specified time limit. In an Oracle RAC environment, this may require terminating a transaction on an instance, and retrying the transaction on another instance within a specified time frame.

> **Note:**
>
> You can use Application Continuity to hide the cancellation of a transaction from the user, if the application initially used an Application Continuity-enabled dynamic database service to connect to the database instance.
>
> For a more granular approach to service-level management, Oracle recommends that you use Oracle Database Quality of Service Management (Oracle Database QoS Management) for all Oracle RAC-based databases.

To terminate sessions, follow these steps:

1. Query the value of the `INST_ID` column in the `GV$SESSION` dynamic performance view to identify which session to terminate.

2. Issue the `ALTER SYSTEM KILL SESSION` and specify the session index number (SID) and serial number of a session that you identified with the `GV$SESSION` dynamic performance view.

   ```
   KILL SESSION 'integer1, integer2[, @integer3]'
   ```

   - For `integer1`, specify the value of the SID column.

   - For `integer2`, specify the value of the `SERIAL#` column.

   - For the optional `integer3`, specify the ID of the instance where the session to be killed exists. You can find the instance ID by querying the `GV$` tables.

   To use this statement, your instance must have the database open, and your session and the session to be terminated must be on the same instance unless you specify `integer3`.

If the session is performing some activity that must be completed, such as waiting for a reply from a remote database or rolling back a transaction, then Oracle Database waits for this activity to complete, marks the session as terminated, and then returns control to you. If the waiting lasts a minute, then Oracle Database marks the session to be terminated and returns control to you with a message that the session is marked to be terminated. The PMON background process then marks the session as terminated when the activity is complete.

Examples of Identifying and Terminating Sessions

The following examples provide three scenarios in which a user identifies and terminates a specific session. In each example, the SYSDBA first queries the `GV$SESSION` view for the `SCOTT` user's session to identify the session to terminate, and then runs the `ALTER SYSTEM KILL SESSION` statement to terminate the session on the instance.

Example 3-1   Identify and terminate the session on an busy instance

In this example, assume that the executing session is `SYSDBA` on the instance `INST_ID=1`. The `ORA-00031` message is returned because some activity must be completed before the session can be terminated.

```
SQL> SELECT SID, SERIAL#, INST_ID FROM GV$SESSION WHERE USERNAME='SCOTT';

       SID    SERIAL#    INST_ID
---------- ---------- ----------
        80          4          2
```

```
SQL> ALTER SYSTEM KILL SESSION '80, 4, @2';
alter system kill session '80, 4, @2'
*
ERROR at line 1:
ORA-00031: session marked for kill
SQL>
```

Example 3-2   Identify and terminate the session on an idle instance

In this example, assume that the executing session is SYSDBA on the instance INST_ID=1. The session on instance INST_ID=2 is terminated immediately when Oracle Database executes the statement within 60 seconds.

```
SQL> SELECT SID, SERIAL#, INST_ID FROM GV$SESSION WHERE USERNAME='SCOTT';

       SID    SERIAL#    INST_ID
---------- ---------- ----------
        80          6          2

SQL> ALTER SYSTEM KILL SESSION '80, 6, @2';

System altered.

SQL>
```

Example 3-3   Using the IMMEDIATE parameter

The following example includes the optional IMMEDIATE clause to immediately terminate the session without waiting for outstanding activity to complete.

```
SQL> SELECT SID, SERIAL#, INST_ID FROM GV$SESSION WHERE USERNAME='SCOTT';

       SID    SERIAL#    INST_ID
---------- ---------- ----------
        80          8          2

SQL> ALTER SYSTEM KILL SESSION '80, 8, @2' IMMEDIATE;

System altered.

SQL>
```

Related Topics:

Workload Management with Dynamic Database Services

See Also:

- *Oracle Database Administrator's Guide* for more information about terminating sessions

- *Oracle Database 2 Day + Performance Tuning Guide* for examples of these procedures using Oracle Enterprise Manager

- *Oracle Database Quality of Service Management User's Guide* for more information about service-level management

Overview of Initialization Parameter Files in Oracle RAC

When you create the database, Oracle Database creates an SPFILE in the file location that you specify. This location can be either an Oracle ASM disk group or a **cluster file system**. If you manually create your database, then Oracle recommends that you create an SPFILE from an initialization parameter file (PFILE).

> **Note:**
>
> Oracle RAC uses a traditional PFILE only if an SPFILE does not exist or if you specify `PFILE` in your `STARTUP` command. Oracle recommends that you use an SPFILE to simplify administration, to maintain parameter setting consistency, and to guarantee parameter setting persistence across database shutdown and startup events. In addition, you can configure RMAN to back up your SPFILE.

All instances in the cluster database use the same SPFILE at startup. Because the SPFILE is a binary file, do not directly edit the SPFILE with an editor. Instead, change SPFILE parameter settings using Oracle Enterprise Manager or `ALTER SYSTEM` SQL statements.

When creating an SPFILE, if you include the `FROM MEMORY` clause (for example, `CREATE PFILE FROM MEMORY` or `CREATE SPFILE FROM MEMORY`), then the `CREATE` statement creates a PFILE or SPFILE using the current system-wide parameter settings. In an Oracle RAC environment, the created file contains the parameter settings from each instance. Because the `FROM MEMORY` clause requires all other instances to send their parameter settings to the instance that is trying to create the parameter file, the total execution time depends on the number of instances, the number of parameter settings on each instance, and the amount of data for these settings.

This section includes the following topics:

- Setting SPFILE Parameter Values for Oracle RAC

- Parameter File Search Order in Oracle RAC

- Backing Up the Server Parameter File

Setting SPFILE Parameter Values for Oracle RAC

You can change SPFILE settings with Oracle Enterprise Manager or by using the `SET` clause of the `ALTER SYSTEM` statement.

> **Note:**
>
> Modifying the SPFILE using tools other than Oracle Enterprise Manager or SQL*Plus can corrupt the file and prevent database startup. To repair the file, you might be required to create a PFILE and then regenerate the SPFILE.

The examples in this section appear in ASCII text although the SPFILE is a binary file. Assume that you start an instance with an SPFILE containing the following entries:

```
*.OPEN_CURSORS=500
prod1.OPEN_CURSORS=1000
```

The value before the period (.) in an SPFILE entry identifies the instance to which the particular parameter value belongs. When an asterisk (*) precedes the period, the value is applied to all instances that do not have a subsequent, individual value listed in the SPFILE.

For the instance with the Oracle system identifier (SID) `prod1`, the `OPEN_CURSORS` parameter is set to `1000` even though it has a database-wide setting of `500`. Parameter file entries that have the asterisk (*) wildcard character only affect the instances without an instance-specific entry. This gives you control over parameter settings for instance `prod1`. These two types of settings can appear in any order in the parameter file.

If another DBA runs the following statement, then Oracle Database updates the setting on all instances except the instance with SID `prod1`:

```
ALTER SYSTEM SET OPEN_CURSORS=1500 sid='*' SCOPE=SPFILE;
```

The SPFILE now has the following entries for `OPEN_CURSORS`:

```
*.OPEN_CURSORS=1500
prod1.OPEN_CURSORS=1000
```

Run the following statement to reset `OPEN_CURSORS` to its default value for all instances except `prod1`:

```
ALTER SYSTEM RESET OPEN_CURSORS SCOPE=SPFILE;
```

The SPFILE now has just the following entry for `prod1`:

```
prod1.OPEN_CURSORS=1000
```

Run the following statement to reset the `OPEN_CURSORS` parameter to its default value for instance `prod1` only:

```
ALTER SYSTEM RESET OPEN_CURSORS SCOPE=SPFILE SID='prod1';
```

Parameter File Search Order in Oracle RAC

Oracle Database searches for your parameter file in a particular order depending on your platform. For Oracle RAC databases, you can easily determine the location of the parameter file by using the srvctl config database command.

On Linux and UNIX platforms, the search order is as follows:

1. `$ORACLE_HOME/dbs/spfilesid.ora`

2. `$ORACLE_HOME/dbs/spfile.ora`

3. `$ORACLE_HOME/dbs/initsid.ora`

On Windows platforms, the search order is as follows:

1. `%ORACLE_HOME%\database\spfilesid.ora`

2. `%ORACLE_HOME%\database\spfile.ora`

3. `%ORACLE_HOME%\database\initsid.ora`

> **Note:**
>
> Oracle recommends that you do not use the default SPFILE names because all instances must use the same file and they all have different SIDs. Instead, store the SPFILE on Oracle ASM. If you store the SPFILE on a cluster file system, then use the following naming convention for the SPFILE: `$ORACLE_HOME/dbs/spfile`*db_unique_name*`.ora`. Create a PFILE named `$ORACLE_HOME/dbs/init`*sid*`.ora` that contains the name `SPFILE=`*ORACLE_HOME*`/dbs/spfile`*db_unique_name*`.ora`.

Backing Up the Server Parameter File

Oracle recommends that you regularly back up the server parameter file for recovery purposes. Do this using Oracle Enterprise Manager (as described in the *Oracle Database 2 Day + Real Application Clusters Guide*) or use the `CREATE PFILE` statement. For example:

```
CREATE PFILE='/u01/oracle/dbs/test_init.ora'
FROM SPFILE='/u01/oracle/dbs/test_spfile.ora';
```

You can use Recovery Manager (RMAN) to create backups of the server parameter file. You can also recover an SPFILE by starting an instance using a client-side initialization parameter file. Then re-create the server parameter file using the `CREATE SPFILE` statement. Note that if the parameter file that you use for this operation was for a single instance, then the parameter file does not contain instance-specific values, even those that must be unique in Oracle RAC instances. Therefore, ensure that your parameter file contains the appropriate settings as described earlier in this chapter.

To ensure that your SPFILE (and control files) are automatically backed up by RMAN during typical backup operations, use Oracle Enterprise Manager or the RMAN `CONTROLFILE AUTOBACKUP` statement to enable the RMAN autobackup feature

> **See Also:**
>
> - *Oracle Database 2 Day + Real Application Clusters Guide* to perform backup jobs using Oracle Enterprise Manager
>
> - *Oracle Database SQL Language Reference* for more information about the `CREATE SPFILE` statement
>
> - *Oracle Database Backup and Recovery Reference* for more information about the `CONTROLFILE AUTOBACKUP` statement

Initialization Parameter Use in Oracle RAC

By default, most parameters are set to a default value and this value is the same across all instances. However, many initialization parameters can also have different values on different instances as described in Table 3-3. Other parameters *must* either be unique or identical as described in the following sections

- Parameters That Must Have Identical Settings on All Instances

- Parameters That Have Unique Settings on All Instances

- [Parameters That Should Have Identical Settings on All Instances](#)

Table 3-3 summarizes the initialization parameters used specifically for Oracle RAC databases.

See Also:

Oracle Database Reference for additional information about these and other initialization parameters

---

Table 3-3   Initialization Parameters Specific to Oracle RAC

Parameter	Description
ACTIVE_INSTANCE_COUNT	This initialization parameter was deprecated in Oracle RAC 11*g* release 2 (11.2). Instead, use a service with one preferred and one available instance.
ASM_PREFERRED_READ_FAILURE_GROUPS	Specifies a set of disks to be the preferred disks from which to read mirror data copies. The values you set for this parameter are instance specific and need not be the same on all instances.
CLUSTER_DATABASE	Enables a database to be started in cluster mode. Set this parameter to TRUE.
CLUSTER_DATABASE_INSTANCES	Oracle RAC uses this parameter to allocate adequate memory resources. It must be set to the same value on all instances. • For policy-managed databases, Oracle internally sets this parameter to 16 • For administrator-managed databases, Oracle internally sets it to the number of configured Oracle RAC instances You can set this parameter to a value that is *greater than* the current number of instances, if you are planning to add instances. For policy-managed databases, you should set this parameter to a higher value only if you intend to run a database with more than 16 instances. In this case, set the parameter to the expected maximum number of instances on which this database will run.
CLUSTER_INTERCONNECTS	Specifies an alternative cluster interconnect for the private network when there are multiple interconnects. **Notes:** • Oracle recommends that all Oracle databases and Oracle Clusterware use the same interconnect network. • Oracle does not recommend setting the CLUSTER_INTERCONNECTS parameter except in certain situations. See "Administering Multiple Cluster Interconnects on Linux and UNIX Platforms" for more details. • This parameter is stored in the Grid Plug and Play profile in a Grid Plug and Play environment.
DB_NAME	If you set a value for DB_NAME in instance-specific parameter files, the setting must be identical for all instances.

Table 3-3   (Cont.) Initialization Parameters Specific to Oracle RAC

Parameter	Description
DISPATCHERS	Set the DISPATCHERS parameter to enable a shared server configuration, that is a server that is configured to enable many user processes to share very few server processes. With shared server configurations, many user processes connect to a dispatcher. The DISPATCHERS parameter may contain many attributes. Oracle recommends that you configure at least the PROTOCOL and LISTENER attributes. PROTOCOL specifies the network protocol for which the dispatcher process generates a listening end point. LISTENER specifies an alias name for the Oracle Net Services listeners. Set the alias to a name that is resolved through a naming method such as a tnsnames.ora file. The tnsnames.ora file contains net service names. Clients, nodes, and the Oracle Performance Manager node need this file. Oracle Enterprise Manager does not require tnsnames.ora entries on the client for Cloud Control. **See Also:** *Oracle Database Net Services Administrator's Guide* for complete information about configuring the DISPATCHERS parameter and its attributes and for configuring the shared server
GCS_SERVER_PROCESSES	This static parameter specifies the initial number of server processes for an Oracle RAC instance's Global Cache Service (GCS). The GCS processes manage the routing of interinstance traffic among Oracle RAC instances. The default number of GCS server processes is calculated based on system resources with a minimum setting of 2. For systems with one CPU, there is one GCS server process. For systems with two to eight CPUs, there are two GCS server processes. For systems with more than eight CPUs, the number of GCS server processes equals the number of CPUs divided by 4, dropping any fractions. For example, if you have 10 CPUs, then 10 divided by 4 means that your system has 2 GCS processes. You can set this parameter to different values on different instances.
INSTANCE_NAME	Specifies the unique name of an instance. Clients can use this name to force their session to be connected to a specific instance in the cluster. The format of the INSTANCE_NAME parameter is generally *db_unique_name_instance_number*, such as orcldb_2. **Note:** In Grid Plug and Play environments, the INSTANCE_NAME parameter is not required and defaults to *db_unique_name_instance_number* if not specified.
RESULT_CACHE_MAX_SIZE	In a clustered database, you can either set RESULT_CACHE_MAX_SIZE=0 on every instance to disable the **result cache**, or use a nonzero value on every instance to enable the result cache. To switch between enabled and disabled result cache requires that you restart every instance: • **Enabling the result cache:** Set RESULT_CACHE_MAX_SIZE to a value greater than 0, or leave the parameter unset. You can size the cache differently on individual instances. • **Disabling the result cache:** Set RESULT_CACHE_MAX_SIZE=0 on all instances to disable the result cache. If you set RESULT_CACHE_MAX_SIZE=0 upon startup of any one instance, then you must set the parameter to zero on all instance startups because disabling the result cache must done clusterwide. Disabling the result cache on some instances may lead to incorrect results. If you do not set the RESULT_CACHE_MAX_SIZE parameter, the parameter resolves to a default, nonzero value.

Table 3-3 (Cont.) Initialization Parameters Specific to Oracle RAC

Parameter	Description
SERVICE_NAMES	When you use services, Oracle recommends that you do not set a value for the SERVICE_NAMES parameter but instead you should create cluster managed services through the Cluster Managed Services page in Oracle Enterprise Manager Cloud Control. This is because Oracle Clusterware controls the setting for this parameter for the services that you create and for the default database service. The service features described in Workload Management with Dynamic Database Services are not directly related to the features that Oracle provides when you set SERVICE_NAMES. In addition, setting a value for this parameter may override some benefits of using services.
	Note: Oracle recommends that client connections use services rather than instance names. Entries in the SERVICE_NAMES parameter may be used by client connections rather than the INSTANCE_NAME parameter value. The SERVICE_NAMES parameter may include one or more names and different instances may share one or more names with other instances, enabling a client to connect to either a specific instance or to any one of a set of instances, depending on the service name chosen in the connection string.
SPFILE	When you use an SPFILE, all Oracle RAC database instances must use the SPFILE and the file must be on shared storage.
THREAD	Specifies the number of the redo threads to be used by an instance. You can specify any available redo thread number if that thread number is enabled and is not used. If specified, this parameter must have unique values on all instances. The best practice is to use the INSTANCE_NAME parameter to specify redo log groups.

Parameters That Must Have Identical Settings on All Instances

Certain initialization parameters that are critical at database creation or that affect certain database operations must have the same value for every instance in an Oracle RAC database. Specify these parameter values in the SPFILE or in the individual PFILEs for each instance. The following list contains the parameters that must be identical on every instance:

```
COMPATIBLE
CLUSTER_DATABASE
CONTROL_FILES
DB_BLOCK_SIZE
DB_DOMAIN
DB_FILES
DB_NAME
DB_RECOVERY_FILE_DEST
DB_RECOVERY_FILE_DEST_SIZE
DB_UNIQUE_NAME
INSTANCE_TYPE (RDBMS or ASM)
PARALLEL_EXECUTION_MESSAGE_SIZE
REMOTE_LOGIN_PASSWORDFILE
UNDO_MANAGEMENT
```

The following parameters must be identical on every instance only if the parameter value is set to zero:

```
DML_LOCKS
RESULT_CACHE_MAX_SIZE
```

Parameters That Have Unique Settings on All Instances

When it is necessary to set parameters that have unique settings on a policy-managed database, you can ensure that instances always use the same name on particular nodes by running the `srvctl modify instance -n node_name -i instance_name` command for each server that can be assigned to the database's server pool. Then a unique value of the parameter can be specified for `instance_name` that is used whenever the database runs on `node_name`.

Specify the `ORACLE_SID` environment variable, which consists of the database name and the number of the `INSTANCE_NAME` assigned to the instance.

Use the `CLUSTER_INTERCONNECTS` initialization parameter to specify an alternative interconnect to the one Oracle Clusterware is using for the private network. Each instance of the Oracle RAC database gets a unique value when setting the `CLUSTER_INTERCONNECTS` initialization parameter.

> **See Also:**
>
> "Administering Multiple Cluster Interconnects on Linux and UNIX Platforms" for more information about the `CLUSTER_INTERCONNECTS` initialization parameter

Oracle Database uses the `INSTANCE_NUMBER` parameter to distinguish among instances at startup and the `INSTANCE_NAME` parameter to assign redo log groups to specific instances. The instance name can take the form `db_unique_name_instance_number` and when it has this form of name and number separated by an underscore, the number after the underscore is used as the `INSTANCE_NUMBER`. With Oracle Database 11.2 using Grid Plug and Play, you no longer have to explicitly assign instance numbers for policy-managed databases and the instance name defaults to `db_unique_name_instance_number`, where Oracle Database assigns the instance number.

When you specify `UNDO_TABLESPACE` with automatic undo management enabled, then set this parameter to a unique undo tablespace name for each instance.

If you use the `ROLLBACK_SEGMENTS` parameters, then Oracle recommends setting unique values for it by using the `SID` identifier in the SPFILE. However, you must set a unique value for `INSTANCE_NUMBER` for each instance and you cannot use a default value.

Using the `ASM_PREFERRED_READ_FAILURE_GROUPS` initialization parameter, you can specify a list of preferred read failure group names. The disks in those failure groups become the preferred read disks. Thus, every node can read from its local disks. This results in higher efficiency and performance and reduced network traffic. The setting for this parameter is instance-specific, and the values need not be the same on all instances.

Parameters That Should Have Identical Settings on All Instances

Oracle recommends that you set the values for the parameters in Table 3-4 to the same value on all instances. Although you can have different settings for these parameters

on different instances, setting each parameter to the same value on all instances simplifies administration.

Table 3-4   Parameters That Should Have Identical Settings on All Instances

Parameter	Description
`ARCHIVE_LAG_TARGET`	Different values for instances in your Oracle RAC database are likely to increase overhead because of additional automatic synchronization performed by the database processing.
	When using either Oracle Streams downstream capture or Oracle GoldenGate integrated capture mode in a downstream capture configuration with your Oracle RAC database, the value must be greater than zero.
	See Also: *Oracle Streams Replication Administrator's Guide* for more information about downstream capture configuration
`CLUSTER_DATABASE_INSTANCES`	While it is preferable for this parameter to have identical settings across all Oracle RAC database instances, it is not required.
`LICENSE_MAX_USERS`	Because this parameter determines a database-wide limit on the number of users defined in the database, it is useful to have the same value on all instances of your database so you can see the current value no matter which instance you are using. Setting different values may cause Oracle Database to generate additional warning messages during instance startup, or cause commands related to database user management to fail on some instances.
`LOG_ARCHIVE_FORMAT`	If you do not use the same value for all your instances, then you unnecessarily complicate media recovery. The recovering instance expects the required archive log file names to have the format defined by its own value of `LOG_ARCHIVE_FORMAT`, regardless of which instance created the archive log files.
	Databases that support Oracle Data Guard, either to send or receive archived redo log files, must use the same value of `LOG_ARCHIVE_FORMAT` for all instances.
`SPFILE`	If this parameter does not identify the same file to all instances, then each instance may behave differently and unpredictably in fail over, load-balancing, and during normal operations. Additionally, a change you make to the SPFILE with an `ALTER SYSTEM SET` or `ALTER SYSTEM RESET` command is saved only in the SPFILE used by the instance where you run the command. Your change is not reflected in instances using different SPFILEs.
	If the SPFILE values are different in instances for which the values were set by the server, then you should restart the instances that are not using the default SPFILE.
`TRACE_ENABLED`	If you want diagnostic trace information to be always available for your Oracle RAC database, you must set `TRACE_ENABLED` to `TRUE` on all of your database instances. If you trace on only some of your instances, then diagnostic information might not be available when required should the only accessible instances be those with `TRACE_ENABLED` set to `FALSE`.

Table 3-4 (Cont.) Parameters That Should Have Identical Settings on All Instances

Parameter	Description
UNDO_RETENTION	By setting different values for UNDO_RETENTION in each instance, you are likely to reduce scalability and encounter unpredictable behavior following a failover. Therefore, you should carefully consider whether there are any benefits before you assign different values for this parameter to the instances in your Oracle RAC database.

Converting an Administrator-Managed Database to a Policy-Managed Database

You can convert an administrator-managed database to a policy-managed database.

> **Note:**
>
> If the administrator-managed database is configured for a low-privileged user and you attempt to convert the database to a policy-managed database, then you must manually add a wallet (if one does not already exist) for this low privileged user, so that a Windows service for Oracle Database can be created.

To convert an administrator-managed database:

1. Check the current configuration of all services and the database (if you make a mistake and need to recover, then you can know what the configuration looked like when you began), as follows:

   ```
   srvctl config database -db db_unique_name
   srvctl config service -db db_unique_name
   ```

2. Create a server pool for the policy-managed database (you must be a cluster administrator to do this), as follows:

   ```
   srvctl add srvpool -serverpool server_pool -min 0 -max n
   ```

   In the preceding command, 0 is the minimum number of servers you want in the server pool and *n* is the maximum.

   > **Note:**
   >
   > This step does not necessarily place servers in the newly-created server pool. If there are no servers in the Free pool from which the new server pool can allocate servers, for example, then you may have to use the `srvctl relocate server` command to relocate a server from another server pool once the conversion is complete.

3. Stop the database using Oracle Enterprise Manager or SRVCTL, as follows:

   ```
   srvctl stop database -db db_unique_name
   ```

4. Modify the database to be in the new server pool, as follows:

   ```
   srvctl modify database -db db_unique_name -serverpool server_pool
   ```

5. Add a service user to the wallet, as follows:

```
crsctl add wallet -type OSUSER -user user_name -passwd
```

6. Check the status of the database to confirm that it is now policy managed by repeating the commands in step 1.

See Also:

Server Control Utility Reference for more information about these commands

---

Configure Oracle Enterprise Manager to recognize the change you made in the previous procedure, as follows:

1. In order for Oracle Enterprise Manager Cloud Control to recognize the new database instances, you must change the instance name from *db_unique_name#* to *db_unique_name_#* (notice the additional underscore (_) before the number sign (#) character).

2. Rename the `orapwd` file in the `dbs/database` directory (or create a new `orapwd` file by running the `orapwd` command).

By default, there is an `orapwd` file with the instance name appended to it, such as `orapwdORCL1`. You must change the name of the file to correspond to the instance name you changed in the previous step. For example, you must change `orapwdORCL1` to `orapwdORCL_1` or create a new `orapwd` file.

You cannot directly convert a policy-managed database to an administrator-managed database. Instead, you can remove the policy-managed configuration using the `srvctl remove database` and `srvctl remove service` commands, and then register the same database as an administrator-managed database using the `srvctl add database` and `srvctl add instance` commands. Once you register the database and instance, you must use the `srvctl add service` command to add back the services as you removed them.

See Also:

"SRVCTL Command Reference" for more information about these commands

---

Services for administrator-managed databases continue to be defined by the `PREFERRED` and `AVAILABLE` definitions. For policy-managed databases, a service is defined to a database server pool and can either be *uniform* (running on all instances in the server pool) or *singleton* (running on only one instance in the server pool). If you change the management policy of the database, then you must recreate the database services to be either uniform/singleton or `PREFERRED`/`AVAILABLE`, depending upon which database management policy you choose.

See Also:

"Service Deployment Options" for more information about managing services

---

Managing Memory Pressure for Database Servers

Enterprise database servers can use all available memory due to too many open sessions or runaway workloads. Running out of memory can result in failed transactions or, in extreme cases, a restart of the server and the loss of a valuable resource for your applications. Memory Guard detects **memory pressure** on a server in real time and redirects new sessions to other servers to prevent using all available memory on the stressed server.

When Oracle Database QoS Management is enabled and managing an Oracle Clusterware server pool, Cluster Health Monitor sends a metrics stream that provides real-time information about memory resources for the cluster servers to Memory Guard. This information includes the following:

- Amount of available memory

- Amount of memory currently in use

If Memory Guard determines that a node is experiencing memory pressure, then the database services managed by Oracle Clusterware are stopped on that node, preventing new connections from being created. After the memory stress is relieved, the services on that node are restarted automatically, and the listener starts sending new connections to that server. The memory pressure can be relieved in several ways (for example, by closing existing sessions or by user intervention).

Rerouting new sessions to different servers protects the existing workloads on the memory-stressed server and enables the server to remain available. Memory Guard is a feature of Oracle RAC that manages the memory pressure for servers, adding a new resource protection capability in managing service levels for applications hosted on Oracle RAC databases.

Quiescing Oracle RAC Databases

The procedure for quiescing Oracle RAC databases is identical to quiescing a noncluster database. You use the `ALTER SYSTEM QUIESCE RESTRICTED` statement from one instance. You cannot open the database from any instance while the database is in the process of being quiesced. When all non-DBA sessions become inactive, the `ALTER SYSTEM QUIESCE RESTRICTED` statement finishes, and the database is considered as in a quiesced state. In an Oracle RAC environment, this statement affects all instances, not just the one from which the statement is issued.

To successfully issue the `ALTER SYSTEM QUIESCE RESTRICTED` statement in an Oracle RAC environment, you must have the Database Resource Manager feature activated, and it must have been activated since instance startup for all instances in the cluster database. It is through the facilities of the Database Resource Manager that non-DBA sessions are prevented from becoming active. Also, while this statement is in effect, any attempt to change the current resource plan is queued until after the system is unquiesced.

These conditions apply to Oracle RAC:

- If you issued the `ALTER SYSTEM QUIESCE RESTRICTED` statement but Oracle Database has not finished processing it, you cannot open the database.

- You cannot open the database if it is in a quiesced state.

- The `ALTER SYSTEM QUIESCE RESTRICTED` and `ALTER SYSTEM UNQUIESCE` statements affect all instances in an Oracle RAC environment, not just the instance that issues the command.

Note:

You cannot use the quiesced state to take a cold backup. This is because Oracle Database background processes may still perform updates for Oracle Database internal purposes even while the database is in quiesced state. In addition, the file headers of online data files continue to look like they are being accessed. They do not look the same as if a clean shutdown were done. You can still take online backups while the database is in a quiesced state.

See Also:

- *Oracle Database Administrator's Guide* for details on the quiesce database feature

- *Oracle Database SQL Language Reference* for more information about the `ALTER SYSTEM QUIESCE RESTRICTED` syntax

---

Administering Multiple Cluster Interconnects on Linux and UNIX Platforms

In Oracle RAC environments that run on Linux and UNIX platforms, you can use the `CLUSTER_INTERCONNECTS` initialization parameter to specify an alternative interconnect to the one Oracle Clusterware is using for the private network.

Note:

The `CLUSTER_INTERCONNECTS` initialization parameter should not be set to highly available IP (HAIP) addresses provided by Redundant Interconnect Usage. HAIP addresses are recognized automatically.

If you set multiple values for `CLUSTER_INTERCONNECTS`, then Oracle Database uses all of the network interfaces that you specify for the interconnect, providing load balancing if all of the listed interconnects remain operational. You must use identical values, including the order in which the interconnects are listed, on all instances of your database when defining multiple interconnects with this parameter.

Note:

Oracle does not recommend setting the `CLUSTER_INTERCONNECTS` initialization parameter, which overrides the default interconnect settings at the operating system level.

Instead, the best practice is to use Redundant Interconnect Usage, available with Oracle Grid Infrastructure 11*g* release 2 (11.2) for Oracle RAC and Oracle Real Application Clusters One Node 11*g* release 2 (11.2) databases, and later. For databases

that precede Oracle Database 11*g* release 2 (11.2), use operating system-based network bonding technologies to enable high availability (and load balancing) for network interface cards meant to be used as the cluster interconnect. If you want to use multiple database versions in one cluster, you can combine both techniques. Redundant Interconnect Usage will use the interfaces as presented on the operating system level, regardless of bonding. For more information regarding bonding technologies contact your operating system vendor.

> **See Also:**
>
> *Oracle Clusterware Administration and Deployment Guide* for more information about Redundant Interconnect Usage

Use Cases for Setting the CLUSTER_INTERCONNECTS Parameter

The `CLUSTER_INTERCONNECTS` initialization parameter requires an IP address. It enables you to specify multiple IP addresses, separated by colons. Oracle RAC network traffic is distributed between the specified IP addresses.

> **Note:**
>
> - Oracle does not recommend setting the `CLUSTER_INTERCONNECTS` parameter when using a policy-managed database.
>
> - Oracle recommends that all databases and Oracle Clusterware use the same interconnect network.

Typically, you set the `CLUSTER_INTERCONNECTS` parameter only in the following situations:

- The cluster is running multiple databases and you need the interconnect traffic to be separated and you do not use Redundant Interconnect Usage.

- You have a single IP address that is made highly available by the operating system, and it does not have a stable interface name (for example, the name can change when you restart).

Do not set the `CLUSTER_INTERCONNECTS` parameter for the following common configurations:

- If you want to use Redundant Interconnect Usage.

- If you have only one cluster interconnect.

- If the default cluster interconnect meets the bandwidth requirements of your Oracle RAC database, which is typically the case.

Consider the following important points when specifying the `CLUSTER_INTERCONNECTS` initialization parameter:

- The `CLUSTER_INTERCONNECTS` initialization parameter is useful only in Linux and UNIX environments where UDP IPC is enabled.

- Specify a different value for each instance of the Oracle RAC database when setting the CLUSTER_INTERCONNECTS initialization parameter in the parameter file.

- The IP addresses you specify for the different instances of the same database on different nodes must belong to network adapters that connect to the same interconnect network.

- If you specify multiple IP addresses for this parameter, then list them in the same order for all instances of the same database. For example, if the parameter for the first instance on node1 lists the IP addresses of the alt0:, fta0:, and ics0: devices in that order, then the parameter for the second instance on node2 must list the IP addresses of the equivalent network adapters in the same order.

- If an operating system error occurs while Oracle Database is writing to the interconnect that you specify with the CLUSTER_INTERCONNECTS parameter, then Oracle Database returns an error even if some other interfaces are available. This is because the communication protocols between Oracle Database and the interconnect can vary greatly depending on your platform. See your Oracle Database platform-specific documentation for more information.

> **See Also:**
>
> *Oracle Database Reference* for more information about the CLUSTER_INTERCONNECTS initialization parameter

Example

Consider setting CLUSTER_INTERCONNECTS when a single cluster interconnect cannot meet your bandwidth requirements. You may need to set this parameter in data warehouse environments with high interconnect bandwidth demands from one or more databases that cannot use Redundant Interconnect Usage.

For example, if you have two databases with high interconnect bandwidth requirements, then you can override the default interconnect provided by your operating system and nominate a different interconnect for each database using the following syntax in each server parameter file where ip*n* is an IP address in standard dot-decimal format, for example: 144.25.16.214:

```
Database One: crm1.CLUSTER_INTERCONNECTS = ip1
Database Two: ext1.CLUSTER_INTERCONNECTS = ip2
```

If you have one database with high bandwidth demands, then you can nominate multiple interconnects using the following syntax:

```
CLUSTER_INTERCONNECTS = ip1:ip2:...:ipn
```

Customizing How Oracle Clusterware Manages Oracle RAC Databases

Use these examples to minimize Oracle Clusterware control over Oracle RAC databases, which you may need to do during upgrades.

By default, Oracle Clusterware controls database restarts in Oracle RAC environments. In some cases, you may need to minimize the level of control that Oracle Clusterware has over your Oracle RAC database, for example, during database upgrades.

Note:

When using third-party clusterware, Oracle recommends that you use Oracle Clusterware to manage the Oracle RAC instances. If you set the instance to manual and start it with third-party clusterware, then do not use the third-party clusterware to monitor and restart database instances because Oracle Clusterware must do that.

To prevent Oracle Clusterware from restarting your Oracle RAC database when you restart your system, or to avoid restarting failed instances more than once, configure a management policy to define the degree of control. There are two management policies: AUTOMATIC, which is the default, and MANUAL. If the management policy is set to AUTOMATIC, the database is automatically restored to its previous running condition (started or stopped) upon restart of the database host computer. If MANUAL, the database is never automatically restarted upon restart of the database host computer. A MANUAL setting does not prevent Oracle Restart from monitoring the database while it is running and restarting it if a failure occurs.

Use SRVCTL commands to display and change the Oracle Clusterware management policies, as shown in the following examples:

Example 1: Display the Current Management Policy

Use the following command syntax to display the current management policy where *db_unique_name* is the name of the database for which you want to change management policies:

```
srvctl config database -db db_unique_name -all
```

Example 2: Change the Current Management Policy to Another Management Policy

Use the following SRVCTL command syntax to change the current management policy to either AUTOMATIC, MANUAL, or NORESTART:

```
srvctl modify database -db db_unique_name -policy [AUTOMATIC | MANUAL | NORESTART]
```

This command syntax sets the resource attribute of the database resource.

Example 3: Specify a Management Policy for a New Database

When you add a new database using the srvctl add database command, you can use the -policy parameter to specify the management policy as either AUTOMATIC, MANUAL, or NORESTART, as shown in the following example where *db_unique_name* is the name of the database:

```
srvctl add database -db db_unique_name -policy [AUTOMATIC | MANUAL | NORESTART]
    -oraclehome $ORACLE_HOME -dbname DATA
```

This command syntax places the new database under the control of Oracle Clusterware. If you do not provide a management policy option, then Oracle Database uses the default value of automatic. After you change the management policy, the Oracle Clusterware resource records the new value for the affected database.

Related Topics:

srvctl config database

srvctl modify database

srvctl add database

Advanced Oracle Enterprise Manager Administration

You can install, configure, and monitor an Oracle RAC database from a single location using Oracle Enterprise Manager Cloud Control.

This section provides advanced administration tasks that are not covered in *Oracle Database 2 Day + Real Application Clusters Guide* or in "Overview of Monitoring and Tuning Oracle RAC Databases".

See Also:

Oracle Database 2 Day + Real Application Clusters Guide for a task-oriented guide that explains how to use Oracle Enterprise Manager to perform routine Oracle RAC database administrative tasks

---

This section includes the following topics:

- Using Oracle Enterprise Manager Cloud Control to Discover Nodes and Instances

- Other Oracle Enterprise Manager Capabilities

- Administering Jobs and Alerts in Oracle RAC

Using Oracle Enterprise Manager Cloud Control to Discover Nodes and Instances

Discovering Oracle RAC database and instance targets in Oracle Enterprise Manager enables monitoring and administration.

Oracle Enterprise Manager Cloud Control enables you to use the Oracle Enterprise Manager console interface to discover Oracle RAC database and instance targets.

If the Oracle Enterprise Manager Cloud Control agents are installed on a cluster that has an Oracle RAC database, then Oracle RAC database targets are discovered at install time. You can use the console interface to discover targets if a database is created after agents are installed or if a database is not automatically discovered at agent install time.

To discover nodes and instances, use Oracle Enterprise Manager Cloud Control as follows:

1. Log in to Oracle Enterprise Manager and click the **Targets** tab.

2. Click the **Database** tab to view all of the available targets. The column labeled **Types** shows the Oracle RAC databases using the entry *Cluster Database*.

3. Add the database target by selecting the target name, then clicking **Add**. The Add Database Target: Specify Host page appears, which enables you to add databases, listeners, and Oracle ASM as monitored targets.

4. Click the flashlight icon to display the available host names, select a host, then click **Continue**. The Add Database: Specify Source page appears.

5. Either request Oracle Enterprise Manager to discover only noncluster databases and listeners, or to discover all cluster databases, noncluster databases, and listeners on the cluster, then click **Continue**.

6. If this procedure did not discover your reconfigured cluster database and all of its instances, you can use the Targets Discovered on Cluster page to manually configure your cluster database and noncluster databases.

Other Oracle Enterprise Manager Capabilities

This section lists Oracle Enterprise Manager capabilities available with Oracle Enterprise Manager 12*c*.

- The Oracle Grid Infrastructure/Oracle RAC Provisioning deployment procedure provisions Oracle RAC 12*c* and Oracle Grid Infrastructure. This procedure also has a feature called *Profiles*, which enables you to record the inputs and subsequently use them for repeated deployments.

- Dynamic prerequisites for the new procedures enable Oracle Enterprise Manager, when connected to My Oracle Support (formerly Oracle*MetaLink*), to download the latest prerequisites and tools for Oracle RAC provisioning.

- The existing *One-Click Extend Cluster Database* capability now supports Oracle RAC 12*c* stack.

- The existing *Delete/Scale down Oracle Real Application Clusters* capability is certified with Oracle RAC 12*c* clusters.

- The existing *Oracle Database Provisioning* procedure now supports provisioning of single instances of Oracle Database 12*c*.

- A new deployment procedure—Oracle Grid Infrastructure Provisioning for Standalone Servers—has been introduced to provision Oracle Grid Infrastructure 12*c* for noncluster databases.

Administering Jobs and Alerts in Oracle RAC

The Cluster Database Home page shows all of the instances in the Oracle RAC database and provides an aggregate collection of several Oracle RAC-specific statistics that are collected by the **Automatic Workload Repository (AWR)** for server manageability.

You do not need to navigate to an instance-specific page to see these details. However, on the Cluster Database Home page, if an instance is down that should be operating, or if an instance has a high number of alerts, then you can drill down to the instance-specific page for each alert.

To perform specific administrative tasks as described in the remainder of this section, log in to the target Oracle RAC database, navigate to the Cluster Database Home page, and click the **Administration** tab.

This section includes the following topics:

- Administering Jobs in Oracle RAC

- Administering Alerts in Oracle RAC with Oracle Enterprise Manager

- Using Defined Blackouts in Oracle Enterprise Manager

Administering Jobs in Oracle RAC

You can administer Oracle Enterprise Manager jobs at both the database and instance levels. For example, you can create a job at the cluster database level to run on any active instance of the target Oracle RAC database. Or you can create a job at the instance level to run on the specific instance for which you created it. If there is a failure, then recurring jobs can run on a surviving instance.

Because you can create jobs at the instance level, cluster level, or cluster database level, jobs can run on any available host in the cluster database. This applies to scheduled jobs as well. Oracle Enterprise Manager also displays job activity in several categories, including, `Active`, `History`, and `Library`.

Use the Jobs tab to submit operating system scripts and SQL scripts and to examine scheduled jobs. For example, to create a backup job for a specific Oracle RAC database:

1. Click **Targets** and click the database for which you want to create the job.

2. Log in to the target database.

3. When Oracle Enterprise Manager displays the Database Home page, click **Maintenance**.

4. Complete the Enterprise Manage Job Wizard pages to create the job.

Administering Alerts in Oracle RAC with Oracle Enterprise Manager

You can use Oracle Enterprise Manager to configure Oracle RAC environment alerts. You can also configure special Oracle RAC database tests, such as global cache converts, consistent read requests, and so on.

Oracle Enterprise Manager distinguishes between database- and instance-level alerts in Oracle RAC environments. Alert thresholds for instance-level alerts, such as archive log alerts, can be set at the instance target level. This function enables you to receive alerts for the specific instance if performance exceeds your threshold. You can also configure alerts at the database level, such as setting alerts for tablespaces, to avoid receiving duplicate alerts at each instance.

See Also:

Oracle Technology Network for an example of configuring alerts in Oracle RAC and the *Oracle Database PL/SQL Packages and Types Reference* for information about using packages to configure thresholds

---

Using Defined Blackouts in Oracle Enterprise Manager

You can define blackouts (which are time periods in which database monitoring is suspended so that maintenance operations do not skew monitoring data or generate needless alerts) for all managed targets of an Oracle RAC database to prevent alerts from occurring while performing maintenance. You can define blackouts for an entire cluster database or for specific cluster database instances.

4

Administering Oracle RAC One Node

Oracle Real Application Clusters One Node (Oracle RAC One Node) is a single instance of an Oracle Real Application Clusters (Oracle RAC) database that runs on one node in a cluster. This option adds to the flexibility that Oracle offers for database consolidation. You can consolidate many databases into one cluster with minimal overhead while also providing the high availability benefits of failover protection, online rolling patch application, and rolling upgrades for the operating system and Oracle Clusterware.

This chapter includes the following topics:

- Creating an Oracle RAC One Node Database
- Converting Databases
- Online Database Relocation

Creating an Oracle RAC One Node Database

You can create Oracle RAC One Node databases by using the Database Configuration Assistant (DBCA), as with any other Oracle database (manually created scripts are also a valid alternative). Oracle RAC One Node databases may also be the result of a conversion from either a single-instance Oracle database (using `rconfig`, for example) or an Oracle RAC database. Typically, Oracle-provided tools register the Oracle RAC One Node database with Oracle Clusterware. Depending on your configuration, automatic registration of an Oracle RAC One Node database with Oracle Clusterware may not have happened. If this is the case, then follow the steps in this section to register the Oracle RAC One Node database with Oracle Clusterware.

> **Note:**
>
> Oracle recommends that you manage Oracle RAC One Node databases with Server Control Utility (SRVCTL). You can only perform certain operations (such as Online Database Relocation) using SRVCTL.

If your Oracle RAC One Node database did not register automatically with Oracle Clusterware, then use the `srvctl add database` command to add an Oracle RAC One Node database to your cluster. For example:

```
srvctl add database -c  RACONENODE [-e server_list] [-i instance_name] [-w timeout]
```

Use the `-e` option and the `-i` option when adding an administrator-managed Oracle RAC One Node database.

> **See Also:**
>
> - "srvctl add database" for usage information
>
> - "Using DBCA in Interactive Mode to Add Database Instances to Target Nodes" for more information about adding an Oracle RAC One Node database

For Oracle RAC One Node databases, you must configure at least one dynamic database service (in addition to and opposed to the default database service). When using an administrator-managed Oracle RAC One Node database, service registration is performed as with any other Oracle RAC database. When you add services to a policy-managed Oracle RAC One Node database, SRVCTL does not accept any placement information, but instead configures those services using the value of the SERVER_POOLS attribute.

> **Note:**
>
> When adding an administrator-managed Oracle RAC One Node database, you can optionally supply an instance prefix with the `-i` *instance_name* option of the `srvctl add database` command. The name of the instance will then be *prefix*_1. If you do not specify an instance prefix, then the first 12 characters of the unique name of the database becomes the prefix. The instance name changes to *prefix*_2 during an online database relocation and reverts back to *prefix*_1 during a subsequent online database relocation. The same instance name is used on failover.

Converting Databases

Using SRVCTL, you can convert an Oracle RAC database with one instance to an Oracle RAC One Node database and vice versa.

This section includes the following topics:

- Converting a Database from Oracle RAC to Oracle RAC One Node

- Converting a Database from Oracle RAC One Node to Oracle RAC

Converting a Database from Oracle RAC to Oracle RAC One Node

Before converting an Oracle RAC database to an Oracle RAC One Node database, you must first ensure that the Oracle RAC database has only one instance. If your Oracle RAC database is administrator managed and has more than one instance, then you must remove all instances except one using the `srvctl remove instance` command. If your Oracle RAC database is policy managed and has more than one instance, then you must stop all instances except one using the `srvctl stop instance` command.

If the Oracle RAC database is administrator managed, then you must change the configuration of all services to set the preferred instance to the instance that you want to keep as an Oracle RAC One Node database after conversion. If any service had a PRECONNECT TAF policy, then its TAF policy must be updated to BASIC or NONE before starting the conversion process. These services must no longer have any available instance.

If the Oracle RAC database is policy managed, then you must change the configuration of all services so that they all use the same server pool before you convert the Oracle RAC database to an Oracle RAC One Node database.

You can convert an Oracle RAC database with one instance to an Oracle RAC One Node database using the `srvctl convert database` command, as follows:

```
srvctl convert database -db db_unique_name -dbtype RACONENODE
   [-instance instance_name -timeout timeout]
-w timeout]
```

> **See Also:**
>
> - "srvctl remove instance" for more information about this command
>
> - "srvctl stop instance" for more information about this command
>
> - "srvctl convert database" for more information about this command

> **Note:**
>
> An Oracle RAC database that you want to convert to Oracle RAC One Node must either use Oracle Managed Files (to enable automatic thread allocation) or have at least two redo threads.

Converting a Database from Oracle RAC One Node to Oracle RAC

You can convert an Oracle RAC One Node database to an Oracle RAC database by logging in as the Oracle RAC One Node database owner and entering the following SRVCTL command:

```
srvctl convert database -db db_unique_name -dbtype RAC
```

If you are relocating the database you want to convert to Oracle RAC using online database relocation, or an online database relocation has failed, then you must either quit or complete the relocation before you run the `srvctl convert database` command.

After you run this command, you must create server pools for each database service, in addition to the database server pool. The values for SERVER_NAMES for the server pools used by the database services must be set to the node that you converted from an Oracle RAC One Node to an Oracle RAC node. You can use the CRSCTL utility or Oracle Enterprise Manager to create and configure the server pools.

Converting an administrator-managed Oracle RAC One Node database to an Oracle RAC database configures all database services so that the single-instance database is the preferred instance for that service. After you convert the database, you can add instances to your database by using the `srvctl add instance` command.

Converting a policy-managed Oracle RAC One Node database to an Oracle RAC database sets all database services to UNIFORM cardinality. It also results in reusing the server pool in which the database currently runs. The conversion reconfigures the database to run on all nodes in the server pool. The command does not start any additional instances but running the `srvctl start database` command starts the database on all nodes in the server pool.

> **See Also:**
>
> "srvctl convert database" for more information about this command

Online Database Relocation

You can relocate an Oracle RAC One Node database to another node while still maintaining service availability using the online database relocation feature. Only during a planned online database relocation is a second instance of an Oracle RAC One Node database created, so that any database sessions can continue while the database is relocated to a new node. You can only use online database relocation with Oracle RAC One Node databases but you cannot use online database relocation with Oracle RAC databases regardless of their management style (either administrator or policy managed).

You can use the srvctl relocate database command configure the amount of time after the relocated database starts and services are migrated, before the former instance of the database stops. This configured amount of time is not an upper bound on the amount of time taken by the entire operation, but only controls how long the relocated database waits for connections to migrate from the former instance to the new instance, before stopping the former instance.

Online database relocation occurs, as follows:

1. Start a new database instance in a different location.

2. Move all the services to the relocated instance.

3. Wait for all the connections to migrate to the relocated instance.

4. Shut down the former database instance, forcing any remaining connections to move to the relocated instance.

The online relocation timeout is the amount of time you configure to complete step 3.

If your Oracle RAC One Node database is administrator managed, then the target node to which you want to relocate the database instance must be part of the Free server pool at the moment you initiate the relocation. If you have a spare target node in the Free server pool, then this node is added to the candidate list for the Oracle RAC One Node database.

> **Note:**
>
> - When you relocate a database instance to a target node that is not currently in the candidate server list for the database, you must copy the password file, if configured, to the target node, unless you use shared password files stored in Oracle ASM.
>
>   When you use password file-based authentication for remote management of Oracle RAC One Node databases without any shared password file, you must have two password files on each node where the database can run: one named `SID_prefix_1` and the other named `SID_prefix_2`. You must recopy both of these files to all candidate nodes every time you update the password file. This is true for both policy-managed and administrator-managed databases.
>
>   Oracle recommends using Oracle Clusterware to start and stop the database, and defining users in the data dictionary for other management.
>
> - Before you relocate a database instance, you must ensure that the database service user is added to the wallet. Run `crsctl query wallet -type OSUSER -all` to check whether the database service user is in the wallet. If not, then run `crsctl add wallet -type OSUSER -user user_name -passwd` to add the database service user to the wallet.

> **See Also:**

Use the `srvctl relocate database` command to initiate relocation of an Oracle RAC One Node database. For example:

```
srvctl relocate database -d rac1 -n node7
```

> **See Also:**
>
> - *Oracle Database Administrator's Guide* for more information about creating shared password files in Oracle ASM
>
> - "srvctl relocate database" for more information about this command

5

Workload Management with Dynamic Database Services

This chapter describes how to manage workloads in Oracle Real Application Clusters (Oracle RAC) to provide high availability and scalability for your applications. This chapter includes the following topics:

- Using Oracle Services

- Service Deployment Options

- Connection Load Balancing

- Fast Application Notification

- Transaction Guard for Improving Client Failover

- Application Continuity: Automated Replay for Masking Outages

- Load Balancing Advisory

- Enabling Clients for Oracle RAC

- Distributed Transaction Processing in Oracle RAC

- Administering Services

- Global Services

- Automatic Workload Repository

- Measuring Performance by Service Using the Automatic Workload Repository

- Automatic Workload Repository Service Thresholds and Alerts

Using Oracle Services

To manage workloads or a group of applications, you can define services that you assign to a particular application or to a subset of an application's operations. You can also group work by type under services. For example, online users can use one service, while batch processing can use another and reporting can use yet another service to connect to the database.

Oracle recommends that all users who share a service have the same service level requirements. You can define specific characteristics for services and each service can represent a separate unit of work. There are many options that you can take advantage of when using services. Although you do not have to implement these options, using them helps optimize application performance.

Service Deployment Options

This section describes the following service deployment topics:

- Service Usage in an Oracle RAC Database

- Service Characteristics

- Default Service Connections

- Restricted Service Registration

Service Usage in an Oracle RAC Database

Services provide location transparency. A service name can identify multiple database instances, and an instance can belong to multiple services. Several database features use services for an Oracle RAC database:

This section includes the following topics:

- Oracle Clusterware Resources for a Service

- Database Resource Manager Consumer Group Mappings for Services

- Performance Monitoring by Service with AWR

- Parallel Operations and Services

- Oracle Streams and Oracle RAC

Oracle Clusterware Resources for a Service

Resource profiles are automatically created when you define a service. A resource profile describes how Oracle Clusterware should manage the service and which instance the service should failover to if the preferred instance stops. Resource profiles also define service dependencies for the instance and the database. Due to these dependencies, if you stop a database, then the instances and services are automatically stopped in the correct order.

Database Resource Manager Consumer Group Mappings for Services

Services are integrated with Oracle Resource Manager, which enables you to restrict the resources that users use to connect to an instance by using a service. Oracle Resource Manager enables you to map a consumer group to a service so that users who connect to an instance using that service are members of the specified consumer group. Oracle Resource Manager operates at an instance level.

Performance Monitoring by Service with AWR

The metric data generated by Automatic Workload Repository (AWR) is organized into various groups, such as event, event class, session, service, and tablespace metrics. Typically, you view the AWR data using Oracle Enterprise Manager or AWR reports.

> **See Also:**
>
> *Oracle Database Performance Tuning Guide* for more information about generating and viewing AWR reports

Parallel Operations and Services

By default, in an Oracle RAC environment, a SQL statement executed in parallel can run across all of the nodes in the cluster. For this cross-node or inter-node parallel execution to perform well, the interconnect in the Oracle RAC environment must be sized appropriately because inter-node parallel execution may result in a lot of interconnect traffic. To limit inter-node parallel execution, you can control parallel execution in an Oracle RAC environment using the PARALLEL_FORCE_LOCAL initialization parameter. By setting this parameter to TRUE, the parallel execution servers can only execute on the same Oracle RAC node where the SQL statement was started.

Services are used to limit the number of instances that participate in a parallel SQL operation. When the default database service is used, the parallel SQL operation can run on all available instances. You can create any number of services, each consisting of one or more instances. When a parallel SQL operation is started, the parallel execution servers are only spawned on instances which offer the specified service used in the initial database connection.

PARALLEL_INSTANCE_GROUP is an Oracle RAC parameter that, when used with services, lets you restrict parallel query operations to a limited number of instances.To restrict parallel query operations to a limited number of instances, set the PARALLEL_INSTANCE_GROUP initialization parameter to the name of a service. This does not affect other parallel operations such as parallel recovery or the processing of GV$ queries.

Oracle Streams and Oracle RAC

Oracle Streams takes advantage of Oracle RAC features in many ways. When Oracle Streams is configured in an Oracle RAC environment, each queue table has an owning instance. If the instance that hosts a queue table fails, another instance in the Oracle RAC database becomes the owning instance for the queue table, allowing Oracle Streams to continue operating.

Also, on an Oracle RAC database, a service is created for each buffered queue. This service always runs on the owner instance of the destination queue and follows the ownership of this queue if the ownership switches because of instance startup, instance shutdown, and so on. This service is used by queue-to-queue propagations.

> **See Also:**
>
> *Oracle Streams Concepts and Administration* for more information

Service Characteristics

When you create new services for your database, you should define the automatic workload management characteristics for each service. The characteristics of a service include:

- Service Name
- Service Edition
- Service Management Policy
- Database Role for a Service

- Instance Preference

- Server Pool Assignment

- Load Balancing Advisory Goal for Run-time Connection Load Balancing

- Connection Load Balancing Goal

- Distributed Transaction Processing

- Failing Over OCI Clients with TAF

See Also:

- "Creating Services with SRVCTL" for more information

- "SRVCTL Command Reference" for more information about using SRVCTL to create or modify services and descriptions of the various options

---

Service Name

Each service has a service name. The service name is used by clients to connect to one or more instances. The service name must be unique throughout your system.

Service Edition

Edition-based redefinition of database objects enables you to upgrade an application's objects while these objects are in use. You can set the *edition* attribute of a database service when you create it, or modify an existing service to set the edition. When you set the service edition, connections that use this service use this edition as the initial session edition. If the service does not specify the edition name, then the initial session edition is the database default edition.

You can set the service edition using SRVCTL, as follows:

```
$ srvctl modify service -db hr -s crmsrv -edition e2
```

Service Management Policy

When you use Oracle Clusterware to manage your database, you can configure startup options for each individual database service when you add the service using the `srvctl add service` command with the `-policy` parameter. If you set the management policy for a service to `AUTOMATIC` (the default), then the service starts automatically when you start the database with SRVCTL. If you set the management policy to `MANUAL`, then the service does not automatically start, and you must manually start it with SRVCTL. A `MANUAL` setting does not prevent Oracle Clusterware from monitoring the service when it is running and restarting it if a failure occurs. Before Oracle RAC 11*g* release 2 (11.2), all services worked as though they were defined with a `MANUAL` management policy.

See Also:

"srvctl add service" for more usage information

Using CRSCTL to stop and restart Oracle Clusterware is treated as a failure and the service is restarted if it was previously running.

> **Note:**
>
> When you use automatic services in an administrator-managed database, during planned database startup, services may start on the first instances to start rather than their preferred instances.

Database Role for a Service

If you configured Oracle Data Guard in your environment, then you can define a role for each service using SRVCTL with the `-l` parameter. When you specify a role for a service, Oracle Clusterware automatically starts the service only when the database role matches the role you specified for the service. Valid roles are `PRIMARY`, `PHYSICAL_STANDBY`, `LOGICAL_STANDBY`, and `SNAPSHOT_STANDBY` and you can specify more than one role for a service.

> **Note:**
>
> The service role only controls automatic startup of services. Using SRVCTL to manually start a service will succeed even if the roles do not match.

If multiple databases in the cluster offer the same service name, then Oracle RAC balances connections to that service across all such databases. This is useful for standby and active Oracle Data Guard databases, but if you want client connections to a service to be directed to a particular database, then the service name must be unique within the cluster (not offered by any other database).

> **See Also:**
>
> *Oracle Data Guard Concepts and Administration* for more information about database roles

Instance Preference

When you define a service for an administrator-managed database, you define which instances normally support that service using SRVCTL with the `-preferred` parameter. These are known as the *preferred* instances. You can also define other instances to support a service if the service's preferred instance fails using SRVCTL with the `-available` parameter. These are known as *available* instances.

When you specify preferred instances, you are specifying the number of instances on which a service normally runs. This is the *maximum cardinality* of the service. Oracle Clusterware attempts to ensure that the service always runs on the number of instances for which you have configured the service. Afterwards, due to either instance failure or planned service relocations, a service may be running on an available instance.

If an instance fails, then you cannot control to which available instance Oracle Clusterware relocates the services if there are multiple instances in the list. During a planned operation, however, you can manually direct the service to any instance in either the preferred or the available list not currently offering the service.

When a service moves to an available instance, Oracle Database does not move the service back to the preferred instance when the preferred instance restarts because:

- The service is running on the desired number of instances.

- Maintaining the service on the current instance provides a higher level of service availability.

- Not moving the service back to the initial preferred instance prevents a second outage.

You can, however, easily automate fail back to the preferred instance by using FAN callouts.

See Also:

"Tools for Administering Oracle RAC" for more information about policy-managed and administrator-managed databases

---

Server Pool Assignment

When you define services for a policy-managed database, you assign the service to a server pool in which the database is hosted using SRVCTL with the `-serverpool` parameter. You can define the service as either `UNIFORM` (running on all instances in the server pool) or `SINGLETON` (running on only one instance in the server pool) using the `-cardinality` parameter. For singleton services, Oracle RAC chooses on which instance in the server pool the service is active. If that instance fails, then the service fails over to another instance in the server pool. A service can only run in one server pool and Oracle recommends that every server pool has at least one service.

Note:

Oracle Database Quality of Service Management (Oracle Database QoS Management) manages singleton services in a server pool, if the maximum size of that server pool is one.

See Also:

"Tools for Administering Oracle RAC" for more information about policy-managed and administrator-managed databases

---

Load Balancing Advisory Goal for Run-time Connection Load Balancing

With run-time connection load balancing, applications can use load balancing advisory events to provide better service to users. Oracle JDBC, Oracle Universal Connection Pool for Java, OCI session pool, ODP.NET, and Oracle WebLogic Server Active GridLink for Oracle RAC clients are automatically integrated to take advantage of load balancing advisory events. The load balancing advisory informs the client about the current service level that an instance is providing for a service. To enable the load balancing advisory, use SRVCTL with the `-rlbgoal` parameter when creating or modifying the service.

The load balancing advisory also recommends how much of the workload should be sent to that instance. The goal determines whether connections are made to the service

based on best service quality (how efficiently a single transaction completes) or best throughput (how efficiently a complete job or long-running query completes).

Connection Load Balancing Goal

Oracle Net Services provides connection load balancing to enable you to spread user connections across all of the instances that are supporting a service. For each service, you can use SRVCTL to define the method you want the listener to use for load balancing by setting the connection load balancing goal, specified with the `-clbgoal` parameter. Connections are classified as `LONG` (such as connection pools and SQL*FORMS), which tells the listener to use session count, or `SHORT`, which tells the listener to use response-time or throughput statistics. If the load balancing advisory is enabled, then its information is used to balance connections; otherwise, CPU utilization is used to balance connections.

Distributed Transaction Processing

Oracle XA applications have unique requirements. To make it easier to use Oracle RAC with global transactions, use SRVCTL and set the distributed transaction processing parameter (`-dtp`) to `TRUE` for the service so that all tightly coupled branches of a distributed transaction processing transaction are run on the same instance. This is a requirement only if you are suspending and resuming the same Oracle XA branch, but may improve performance if separate branches use the same instance.

> **See Also:**
>
> "Distributed Transaction Processing in Oracle RAC" for more information about distributed transaction processing in Oracle RAC

Default Service Connections

Your Oracle RAC database includes an Oracle database service identified by `DB_UNIQUE_NAME`, if set, or `DB_NAME` or `PDB_NAME`, if not. This default service is always available on all instances in an Oracle RAC environment, unless an instance is in restricted mode. You cannot alter this service or its properties. Additionally, the database supports the following two internal services:

- `SYS$BACKGROUND` is used by the background processes only

- `SYS$USERS` is the default service for user sessions that are not associated with any application service

All of these services are used for internal management. You cannot stop or disable any of these internal services to do planned outages or to failover to Oracle Data Guard. Do not use these services for client connections.

> **Note:**
>
> You can explicitly manage only the services that you create. If a feature of the database creates an internal service, you cannot manage it using the information in this chapter.

Restricted Service Registration

Security is a high priority to all enterprises, and network security and controlling access to the database is a critical component of overall security endeavours. This feature allows listener registration only from local IP addresses, by default, and provides the ability to configure and dynamically update a set of IP addresses or subnets from which registration requests are allowed by the listener. Database Instance registration with a listener succeeds only when the request originates from a valid node. The network administrator can specify a list of valid nodes, excluded nodes, or disable valid node checking. The list of valid nodes explicitly lists the nodes and subnets that can register with the database. The list of excluded nodes explicitly lists the nodes that cannot register with the database. The control of dynamic registration results in increased manageability and security of Oracle RAC deployments.

By default, valid node checking for registration (VNCR) is enabled. In the default configuration, registration requests are only allowed from nodes within the cluster, because they are redirected to the private subnet, and only nodes within the cluster can access the private subnet. Non-SCAN listeners only accept registration from instances on the local node. You must manually include remote nodes or nodes outside the subnet of the SCAN listener on the list of valid nodes by using the `registration_invited_nodes_alias` parameter in the `listener.ora` file or by modifying the SCAN listener using SRVCTL, as follows:

```
$ srvctl modify scan_listener -invitednodes node_list -invitedsubnets subnet_list
```

See Also:

Oracle Database Net Services Administrator's Guide

Connection Load Balancing

Oracle Net Services provides the ability to distribute client connections across the instances in an Oracle RAC configuration. There are two types of load balancing that you can implement: client-side and server-side load balancing. Client-side load balancing distributes the connection requests across the listeners, independently at each client. With server-side load balancing, the SCAN listener directs a connection request to the best instance currently providing the service, based on the `-clbgoal` and `-rlbgoal` settings for the service.

In an Oracle RAC database, client connections should use both types of connection load balancing.

- Server-Side Load Balancing

- Generic Database Clients

- Client-Side Load Balancing

- Client-Side Connection Configuration for Older Clients

> **See Also:**
>
> *Oracle Database Net Services Administrator's Guide* for detailed information about both types of load balancing

Server-Side Load Balancing

When you create an Oracle RAC database with DBCA, it automatically:

- Configures and enables server-side load balancing

- Creates a sample client-side load balancing connection definition in the `tnsnames.ora` file on the server

The Oracle Clusterware Database Agent is responsible for managing the `LISTENER_NETWORKS` parameter.

> **Note:**
>
> Note: If you set the `REMOTE_LISTENER` parameter manually, then set this parameter to *scan_name*:*scan_port*.

FAN, Fast Connection Failover, and the load balancing advisory depend on an accurate connection load balancing configuration that includes setting the connection load balancing goal for the service. You can use a goal of either `LONG` or `SHORT` for connection load balancing. These goals have the following characteristics:

- **SHORT**: Use the `SHORT` connection load balancing method for applications that use run-time load balancing. When using connection pools that are integrated with Load Balancing Advisory, set the `CLB_GOAL` to `SHORT`. The following example modifies the service known as `oltpapp`, using SRVCTL to set the connection load balancing goal to `SHORT`:

  ```
  $ srvctl modify service -db db_unique_name -service oltpapp -clbgoal SHORT
  ```

- **LONG**: Use the `LONG` connection load balancing method if run-time load balancing is not required. This is typical for batch operations. `LONG` is the default connection load balancing goal. The following is an example of modifying a service, `batchconn`, using SRVCTL to define the connection load balancing goal for long-lived sessions:

  ```
  $ srvctl modify service -db db_unique_name -service batchconn -clbgoal LONG
  ```

Generic Database Clients

Oracle Net Services provides the ability to add the `CONNECT_TIMEOUT`, `RETRY_COUNT`, and `TRANSPORT_CONNECT_TIMEOUT` parameters to the `tnsnames.ora` connection string.

For example, when using SCAN addresses for the remote listeners at the database:

```
jdbc:oracle:thin:@(DESCRIPTION =(TRANSPORT_CONNECT_TIMEOUT=3)(CONNECT_TIMEOUT=60)
(RETRY_COUNT=3)(FAILOVER=ON)(ADDRESS_LIST =(ADDRESS=(PROTOCOL=tcp)(HOST=CLOUD-
SCANVIP.example.com)(PORT=1521))
(CONNECT_DATA=(SERVICE_NAME=GOLD)))Remote_listeners=CLOUD-SCANVIP.example.com:1521
```

For example, when using remote listeners pointing to VIPs at the database:

```
jdbc:oracle:thin:@(DESCRIPTION =
(TRANSPORT_CONNECT_TIMEOUT=3)
(CONNECT_TIMEOUT=60)(RETRY_COUNT=3)(FAILOVER=ON)
(ADDRESS_LIST=
(ADDRESS=(PROTOCOL=tcp)(HOST=CLOUD-VIP1)(PORT=1521) )
(ADDRESS=(PROTOCOL=tcp)(HOST=CLOUD-VIP2)(PORT=1521) )
(ADDRESS=(PROTOCOL=tcp)(HOST=CLOUD-VIP3)(PORT=1521) ))
(CONNECT_DATA=(SERVICE_NAME=GOLD)))
```

The value of these parameters is expressed in seconds. In the preceding examples, Oracle Net waits for 60 seconds for each full connection to receive a response, after which it assumes that a failure occurred and retries the next address in the `ADDRESS_LIST`. Oracle Net retries the address list 3 times before it returns a failure message to the client. The `TRANSPORT_CONNECT_TIMEOUT` parameter establishes the time to wait to establish a TCP connection to the database server.

For SCAN, Oracle Net Services tries all three addresses (returned by the SCAN address) before returning a failure to the client. EZConnect with SCAN includes this connection failover feature.

This behavior is called Oracle Net *connection failover*. If an error is returned from a chosen address in the list, then Oracle Net Services tries the next address in the list until it is either successful or it has exhausted all addresses in its list.

Client-Side Connection Configuration for Older Clients

In addition to client-side load balancing, Oracle Net Services include **connection failover**. If an error is returned from the chosen address in the list, Oracle Net Services tries the next address in the list until it is either successful or it has exhausted all addresses in its list. For SCAN, Oracle Net Services tries all three addresses before returning a failure to the client. EZConnect with SCAN includes this connection failover feature.

To increase availability, you can specify a timeout that specifies how long Oracle Net waits for a response from the listener before returning an error. The method of setting this timeout parameter depends on the type of client access. Oracle Net maintains these parameters for backward compatibility.

This section includes the following topics:

- JDBC-Thin Clients

- OCI Clients

JDBC-Thin Clients

You can avoid delays by setting the `oracle.net.ns.SQLnetDef.TCP_CONNTIMEOUT_STR` property, as follows:

```
Properties prop = new Properties ();
prop.put (oracle.net.ns.SQLnetDef.TCP_CONNTIMEOUT_STR,
"" + (1 * 1000)); // 1 second
dbPools[ poolIndex ].setConnectionProperties ( prop );
```

The parameter value is specified in milliseconds. Therefore, it is possible to reduce the timeout to 500Ms if the application retries connecting.

OCI Clients

For OCI clients, create a local `sqlnet.ora` file on the client side. Configure the connection timeout in this file by adding the following line:

```
sqlnet.outbound_connect_timeout = 1
```

The granularity of the timeout value for the OCI client is in seconds. The `sqlnet.ora` file affects all connections using this client.

> **Note:**
>
> Do *not* configure the connection timeout in the `sqlnet.ora` file on the server.

Client-Side Load Balancing

Client-side load balancing is defined in your client connection definition (`tnsnames.ora` file, for example) by setting the parameter `LOAD_BALANCE=ON`. When you set this parameter to `ON`, Oracle Database randomly selects an address in the address list, and connects to that node's listener. This balances client connections across the available SCAN listeners in the cluster.

If you configured SCAN for connection requests, then client-side load balancing is not relevant for those clients that support SCAN access. When clients connect using SCAN, Oracle Net automatically balances the load of client connection requests across the three IP addresses you defined for the SCAN, unless you are using EZConnect.

The SCAN listener redirects the connection request to the local listener of the instance that is least loaded (if `-clbgoal` is set to `SHORT`) and provides the requested service. When the listener receives the connection request, the listener connects the user to an instance that the listener knows provides the requested service. To see what services a listener supports, run the `lsnrctl services` command.

When clients connect using SCAN, Oracle Net automatically load balances client connection requests across the three IP addresses you defined for the SCAN, unless you are using EZConnect.

If you are using clients that do not support SCAN (for example, the client version is pre-Oracle Database 11*g* release 2 (11.2)), then, to use SCAN you must change the client `tnsnames.ora` to include the SCAN VIPs, and set `LOAD_BALANCE=ON` to balance requests across the VIPs. For example:

```
Sales.example.com=(DESCRIPTION=
  (ADDRESS_LIST=(LOAD_BALANCE=ON)(FAILOVER=ON)
    (ADDRESS=(PROTOCOL=TCP)(HOST=172.22.67.192)(PORT=1521))
    (ADDRESS=(PROTOCOL=TCP)(HOST=172.22.67.193)(PORT=1521))
    (ADDRESS=(PROTOCOL=TCP)(HOST=172.22.67.194)(PORT=1521))
    (CONNECT_DATA=(SERVICE_NAME=salesservice.example.com))
  ))
```

> **Note:**
>
> If your database is not Oracle Database 11*g* release 2 (11.2), or later, and you want to use SCAN, then you must add SCAN VIPs to the `REMOTE_LISTENER` parameter to enable correct listener cross-registration.

> **See Also:**
>
> *Oracle Database Reference* for more information about the REMOTE_LISTENER parameter

Fast Application Notification

This section provides a detailed description of FAN under the following topics:

- Application High Availability with Services and FAN

- Overview of Fast Application Notification

- Managing Unplanned Outages

- Managing Planned Outages

- Fast Application Notification High Availability Events

- Subscription to High Availability Events

- Using Fast Application Notification Callouts

> **See Also:**
>
> "Enabling Clients for Oracle RAC" for more information about specific client environments that you can use with FAN

Application High Availability with Services and FAN

Oracle Database focuses on maintaining the highest possible service availability. In Oracle RAC, services are designed to be continuously available with loads shared across one or more instances. The Oracle RAC high availability framework maintains service availability by using Oracle Clusterware and resource profiles.

The Oracle RAC high availability framework monitors the database and its services and sends event notifications using FAN. Oracle Clusterware recovers and balances services according to business rules and the service attributes.

Overview of Fast Application Notification

FAN is a high-availability notification mechanism that Oracle RAC uses to notify other processes about configuration and service level information that includes service status changes, such as UP or DOWN events. The Oracle client drivers and Oracle connection pools respond to FAN events and take immediate action. FAN UP and DOWN events can apply to instances, services, and nodes.

Oracle connection pools, for example, use FAN to receive very fast detection of failures, to balance connections following failures, and to balance connections again after the failed components are repaired. So, when a service at an instance starts, the FAN event is used immediately to route work to that resource. When a service at an instance or node fails, the FAN event is used immediately to interrupt applications to recover.

Importance of FAN

Applications can waste time in many critical ways:

- Waiting for TCP/IP timeouts when a node fails without closing sockets, and for every subsequent connection while that IP address is down.

- Attempting to connect when services are down.

- Not connecting when services resume.

- Processing the last result at the client when the server goes down.

- Attempting to execute work on sub-optimal nodes.

When a node fails without closing sockets, all sessions that are blocked in an I/O wait (read or write) wait for `tcp_keepalive`. This wait status is the typical condition for an application connected by a socket. Sessions processing the last result are even worse off, not receiving an interrupt until the next data is requested. Using FAN events eliminates applications waiting on TCP timeouts, time wasted processing the last result at the client after a failure has occurred, and time wasted executing work on slow, hung, or dead nodes.

For cluster configuration changes, the Oracle RAC high availability framework publishes a FAN event immediately when a state change occurs in the cluster. Instead of waiting for the application to time out against the database and detect a problem, applications can receive FAN events and react immediately. With FAN, in-flight transactions are immediately terminated and the client notified when the instance fails.

FAN also publishes load balancing advisory events. Applications can take advantage of the load balancing advisory FAN events to direct work requests to the instance in the cluster that is currently providing the best service quality.

You can take advantage of FAN events in the following three ways:

1. Your application can use FAN without programmatic changes if you use an integrated Oracle client. The integrated clients for FAN events include Oracle JDBC Universal Connection Pool, ODP.NET connection pool, OCI session pool, Oracle WebLogic Server Active Gridlink for Oracle RAC, and OCI and ODP.NET clients. This includes applications that use Application Continuity or Transaction Guard. The integrated Oracle clients must be Oracle Database 10*g* release 2 (10.2) or later to take advantage of the FAN high-availability events. The pooled clients can also take advantage of the load balancing advisory FAN events.

2. Applications can use FAN programmatically by using the JDBC and Oracle RAC FAN application programming interface (API) or by using callbacks with OCI and ODP.NET to subscribe to FAN events and to execute event handling actions upon the receipt of an event.

3. You can implement FAN with server-side callouts on your database tier.

If you use one of the integrated clients listed in item 1 of the preceding list, then, for `DOWN` events, the disruption to the application is minimized because the FAN-aware client terminates the sessions to the failed instance or node before they are reused. Incomplete transactions are terminated and the application user is immediately notified. Application users who request connections are directed to available instances only.

For UP events, when services and instances are started, new connections are created so that the application can immediately take advantage of the extra hardware resources or additional capacity.

FAN callouts are server-side scripts or executables that run whenever a FAN event is generated. You can design and build callouts to do many things. For example:

- Log status information

- Page DBAs or to open support tickets when resources fail to start

- Automatically start dependent external applications that must be co-located with a service

- Change resource plans or shut down services when the number of available instances for a policy-managed database decreases, for example, if nodes fail

- Automate the fail back of a service to preferred instances for administrator-managed databases, if needed

FAN events are published using Oracle Notification Service and Oracle Streams Advanced Queuing, the latter being continued for backward compatibility to previous Oracle Database releases. The publication mechanisms are automatically configured as part of your Oracle RAC installation.

Oracle Net Services listeners and Global Data Services (GDS) are integrated with FAN events, enabling the listener and GDS to immediately de-register services provided by the failed instance and to avoid erroneously sending connection requests to failed instances.

If you specify the connection load balancing goal CLB_GOAL_SHORT for the service, then the listener uses the load balancing advisory when the listener balances the connection loads. When load balancing advisory is enabled, the metrics used for the listener are finer grained.

Managing Unplanned Outages

You can assign services to one or more instances in an administrator-managed Oracle RAC database or to server pools in a policy-managed database. If Oracle RAC detects an outage, then Oracle Clusterware isolates the failed component and recovers the dependent components. For services, if the failed component is an instance, then Oracle Clusterware attempts to maintain the cardinality of the service. If the service definition allows for failover and that is necessary to maintain cardinality, then failover occurs.

FAN events can occur at various levels within the Oracle Database architecture and are published through Oracle Notification Service and Oracle Streams Advanced Queuing for backward compatibility with previous OCI clients. FAN callouts can also be written to execute on the database server in response to FAN events.

> **Note:**
>
> Oracle Database does not run Oracle RAC callouts with guaranteed ordering. Callouts are run asynchronously and they are subject to scheduling variability.

FAN is published from a surviving node when the failed node is out of service. The location and number of instances in an Oracle RAC environment that provide a service are transparent to applications. Restart and recovery are automatic, including

the restarting of the subsystems, such as the listener and the Oracle Automatic Storage Management (Oracle ASM) processes, not just the database. You can use FAN callouts to report faults to your fault management system and to initiate repair jobs.

Managing Planned Outages

For repairs, upgrades, and changes that require you to isolate one or more instances or nodes, Oracle RAC provides interfaces that relocate, disable, and enable services to minimize service disruption to application users. When you relocate a service, you indicate the service should run on another instance temporarily. If you forcibly disable a service, then the service is stopped on all database instances and is no longer available. Disabled services are not restarted automatically. When a service is stopped or relocated, FAN is published with a planned reason code, typically `reason=user`. Once you complete the operation, you can return the service to normal operation or enable the service and then restart it. When a service restarts, FAN is published with `UP` status codes.

Due to dependencies, if you manually shutdown your database, then all of your services for that database automatically stop. If you want your services to automatically start when you manually restart the database, then you must set the management policy of the service to automatic. If you want to shut down only one instance of the database, but not the service, you can use the `srvctl stop instance` command with the `-f` parameter. When you use the `-f` parameter with this command, the database services that were running on that instance are failed over to another instance if possible.

> **See Also:**
>
> "Service Management Policy"

Managing Planned Outages Without User Interruption

You may have to stop database instances for many reasons, such as upgrading the Oracle software, patching, and replacing hardware. The steps in this section result in the least impact to users and requests are not interrupted.

For planned outages, the recommended approach is to drain requests over a controlled time period from FAN-enabled Oracle connection pools. Application Continuity provides continuous service for those requests that do not complete within the allotted time.

Using any FAN-aware pool with Fast Connection Failover configured (such as OCI session pools, Universal Connection Pool, Oracle WebLogic Server Active GridLink for Oracle RAC, or ODP.NET) allows sessions to drain at **request** boundaries after receipt of the FAN planned `DOWN` event. Requests are far more important than transactions because this allows the issued work to complete.

To manage a planned outage without user interruption:

1. Use SRVCTL to relocate the service from an instance or, if you are using a UNIFORM service, then shut down the service on an instance. *Do not* use the `-force` flag with any of these commands. The connection pool automatically releases a connection at a request boundary.

2. The FAN planned `DOWN` event clears idle sessions from the connection pool immediately and marks active sessions to be released at the next check-in. These FAN actions drain the sessions from the instance without disrupting the users.

Existing connections on other instances remain usable, and new connections can be opened to these instances if needed.

3. Not all sessions, in all cases, will check their connections into the pool. It is best practice to have a timeout period after which the instance is forcibly shut down, evicting any remaining client connections.

   For those pools that are configured to use Application Continuity, Universal Connection Pool, or Oracle WebLogic Server Active GridLink for Oracle RAC, an attempt is made to recover these remaining sessions, masking the outage from users and applications.

   > **See Also:**
   >
   > "Application Continuity: Automated Replay for Masking Outages"

4. Once the upgrade, patch, or repair is complete, restart the instance and the service on the original node.

5. The FAN UP event for the service informs the connection pool that a new instance is available for use, allowing sessions to be created on this instance at next request boundaries.

Fast Application Notification High Availability Events

This section describes the information delivered in the FAN event to a callout program. Table 5-1 lists the FAN event types and Table 5-2 describes name-value pairs for the event parameters. The event type is always the first entry when you receive FAN information through a callout, as in the following example:

```
SERVICEMEMBER VERSION=1.0
   service=test.company.com database=ractest
   instance=ractest11 host=ractest1_host0343_1 status=up reason=FAILURE
   timestamp=2012-02-01 22:06:02 timezone=-07:00 db_domain=company.com
```

Note that the preceding examples displays as one line.

Table 5-1   FAN Event Types

Event Types	Notes
DATABASE, INSTANCE, NODE, SERVICE, SERVICEMEMBER, SRV_PRECONNECT, SERVICEMETRICS	• The DATABASE and INSTANCE types list the default database service as *DB_UNIQUE_NAME*. • All events except for NODE events include a db_domain field. • Events of SERVICEMETRICS type are load balancing advisory events. **See Also:** Table 5-4 for more information about load balancing events

Table 5-2   Event Parameter Name-Value Pairs and Descriptions

Parameter	Description
VERSION	Version of the event record. Used to identify release changes.

Table 5-2   (Cont.) Event Parameter Name-Value Pairs and Descriptions

Parameter	Description
database	The unique name of the database supporting the service; matches the initialization parameter value for DB_UNIQUE_NAME, which defaults to the value of the DB_NAME initialization parameter.
instance	The name of the instance that supports the service; matches the ORACLE_SID value.
host	The name of the node that supports the service or the node that has stopped; matches the node name known to Cluster Synchronization Services (CSS).

service	The service name; matches the name of the service as listed in DBA_SERVICES and is domain-qualified as appropriate. Refer to the following examples:
	
	`SERVICEMEMBER VERSION=1.0 service=swingbench database=orcl instance=orcl_2 host=rwsbj13 status=up reason=USER card=1 timestamp=2012-05-29 17:26:37 timezone=-07:00 db_domain=`
	
	`SERVICEMEMBER VERSION=1.0 service=swingbench.example.com database=orcl instance=orcl1 host=rwsbj09 status=up reason=USER card=2 timestamp=2012-07-03 17:29:28 timezone=-07:00 db_domain=example.com`
	
	`SERVICEMEMBER VERSION=1.0 service=swingbench.example.com database=orcl instance=orcl2 host=rwsbj10 status=up reason=USER card=1 timestamp=2012-07-03 17:29:18 timezone=-07:00 db_domain=example.com`

status	Values are UP, DOWN, NODEDOWN (this only applies to the NODE event type), NOT_RESTARTING, PRECONN_UP, PRECONN_DOWN, and UNKNOWN. **Notes:** • When the node is down, the status is NODEDOWN, as opposed to DOWN for other event types. • When STATUS=NODEDOWN and REASON=MEMBER_LEAVE, a node has failed and is no longer part of the cluster. • When STATUS=NODEDOWN and REASON=PUBLIC_NW_DOWN, the node is up but it is unreachable because the public network is down. • Multiple public networks are supported by Oracle Clusterware. The FAN event reflects this fact.

Table 5-2 (Cont.) Event Parameter Name-Value Pairs and Descriptions

Parameter	Description
reason	`AUTOSTART`, `BOOT`, `DEPENDENCY`, `FAILURE`, `MEMBER_LEAVE`, `PUBLIC_NW_DOWN`, `USER`.
	Notes:
	• For `DATABASE` and `SERVICE` event types, `REASON=AUTOSTART` if, when the node started, the `AUTO_START` resource attribute was set to restore, and the resource was offline before the node started.
	• For `DATABASE` and `SERVICE` event types, `REASON=BOOT` if, when the node started, the resource started because it was online before the node started.
	• For SRVCTL and Oracle Enterprise Manager operations, `REASON=USER` describes planned actions for such operations as draining work.
cardinality	The number of service members that are currently active; included in all `SERVICEMEMBER UP` events.
	Following is an example of `SERVICEMEMBER UP` event:
	`SERVICEMEMBER VERSION=1.0 service=swingbench.example.com database=orcl instance=orcl_2 host=mjkbj09 status=up reason=USER card=1 timestamp=2012-07-12 14:46:46 timezone=-07:00 db_domain=example.com`
incarnation	For `NODEDOWN` events; the new cluster incarnation. This value changes each time a member joins or leaves the cluster.
	Following is an example of a `NODEDOWN` event:
	`NODE VERSION=1.0 host=stru09 incarn=175615351 status=down reason=member_leave timestamp=27-Jul-2010 14:49:32 timezone=-07:00`
timestamp	The time according to Oracle Clusterware that an event occurs.
timezone	The time zone of Oracle Clusterware where the event occurred, given as GMT +/-hh:mm.

Some of the FAN event record parameters have values that correspond to values returned by the `SYS_CONTEXT` function using the default namespace `USERENV`, as shown in Table 5-3.

Table 5-3 FAN Parameters and Matching Session Information

FAN Parameter	Matching Session Information
`SERVICE`	`sys_context('userenv', 'service_name')`
`DATABASE_UNIQUE_NAME`	`sys_context('userenv', 'db_unique_name')`
`INSTANCE`	`sys_context('userenv', 'instance_name')`
`CLUSTER_NODE_NAME`	`sys_context('userenv', 'server_host')`

Subscription to High Availability Events

Oracle RAC uses FAN to notify applications about configuration changes and the current service level that is provided by each instance where the service is enabled. If you are using an OCI client or an ODP.NET client to receive FAN events, then you must enable the service used by that client to access the alert notification queue by using SRVCTL with the -notification parameter.

Using Fast Application Notification Callouts

FAN callouts are server-side executables that Oracle RAC executes immediately when high availability events occur. You can use FAN callouts to automate activities when events occur in a cluster configuration, such as:

- Opening fault tracking tickets

- Sending messages to pagers

- Sending e-mail

- Starting and stopping server-side applications

- Maintaining an uptime log by logging each event as it occurs

- Relocating low-priority services when high priority services come online

To use FAN callouts, place an executable in the *Grid_home*/racg/usrco directory on every node that runs Oracle Clusterware. The executable must be able to run standalone when called, with optional arguments, from another program. The following is an example of the *Grid_home*/racg/usrco/callout.sh callout:

```
#! /bin/bash
FAN_LOGFILE= [your_path_name]/admin/log/'hostname'_uptime'.log
echo $* "reported="'date' >> $FAN_LOGFILE &
```

The previous example produces output similar to the following:

```
NODE VERSION=1.0 host=sun880-2 incarn=23 status=nodedown reason=public_nw_down
timestamp=08-Oct-2012 04:02:14 timezone=-08:00 reported=Fri Oct 8 04:02:14 PDT 2012
```

The contents of a FAN event record matches the current session of the user logged on to the database, as shown in Table 5-3. The user environment (USERENV) information is also available using OCI connection handle and descriptor attributes (using OCIAttrGet()). Use this information to take actions on sessions that match the FAN event data.

See Also:

- Table 5-2 for information about the callout and event details

- *Oracle Call Interface Programmer's Guide* for more information about connection handle and descriptor attributes

---

In general, events are only posted to user callouts on the node from which the event originated. For example, if the database on node1 goes down, then the callout is

posted to `node1`, only. The only exceptions to this are node down and VIP down events—these events are posted to all nodes, regardless of from where they originated.

Transaction Guard for Improving Client Failover

Failing to recognize that the last submission has committed, or that it shall commit sometime soon, or that the last submission has not run to completion, is a problem for applications. It can cause users who resubmit or applications that use their own replay to issue duplicate requests, repeating changes that are already committed to the database, and other forms of logical corruption. Transaction Guard can be used to solve this problem.

This section includes the following topics:

- Overview of Transaction Guard

- Transaction Guard Configuration Checklist

- Configuring Services for Transaction Guard

Overview of Transaction Guard

One of the fundamental problems for recovering applications after an outage is that the commit message that is sent back to the client is not durable. If there is a communication disruption between the client and the server, then the client receives an error message indicating that the communication failed (also called a **recoverable error**). This error does not inform the application whether the submission executed any commit operations; or if a procedural call ran to completion executing all expected commits and session state changes or failed part way through; or worse yet, is still running disconnected from the client.

Transaction Guard provides a generic protocol and API for applications to use for at-most-once execution in case of planned and unplanned outages and repeated submissions. Applications use a new concept called the logical transaction ID (LTXID) to determine the outcome of the last transaction open in a database session following an outage. Without using Transaction Guard, applications that attempt to retry operations following outages can cause logical corruption by committing duplicate transactions. Application Continuity uses this feature, and it is available to other applications to determine the last outcome on a failed session.

Without Transaction Guard, if a transaction has been started and a commit has been issued, the commit message that is sent back to the client is not durable. The client is left not knowing whether the transaction committed or not. The transaction cannot be validly resubmitted if the non-transactional state is incorrect or if it is committed. In the absence of guaranteed commit and completion information, resubmission can cause transactions to be applied more than once. Continuing to process a committed but not completed transaction can result in the application using a database session that is in the wrong state.

In Oracle Database 12*c*, Transaction Guard provides a new, fully integrated tool for applications to use to achieve idempotence automatically and transparently, and in a manner that scales. Transaction Guard uses Logical Transaction ID (LTXID) to avoid submitting duplicate transactions. This is referred to as *transaction idempotence*. The LTXID persists on commit and is reused following a rollback. During normal runtime, a LTXID is automatically held in the session at both the client and server for each database transaction. At commit, the LTXID is persisted as part of committing the transaction and the next LTXID to use is returned to the client.

The following example describes how Application Continuity or a third-party application uses Transaction Guard:

1. Client receives a FAN down event or a recoverable error.

2. When you are using FAN, the FAN driver automatically aborts the dead session.

3. If it is a recoverable error, then Application Continuity uses Transaction Guard to determine the outcome of the dead session:

    a. Get the last LTXID from the dead session using the APIs provided by the client drivers: `getLTXID` for JDBC, `OCI_ATTR_GET` with LTXID for Oracle Call Interface, and `LogicalTransactionId` for ODP.NET.

    b. Obtain a new session.

    c. Call the PL/SQL procedure `GET_LTXID_OUTCOME` with the last LTXID obtained from the failed session.

    d. If the outcome of the transaction is COMMITTED and COMPLETED, then return the result of that transaction to the application.

    e. If the outcome of the transaction is COMMITTED and NOT USER_CALL_COMPLETED, then return COMMITTED and a warning that the user call was not completed.

    f. If the outcome of the transaction is not COMMITTED, then, if needed, cleanup any state left behind from the last request and resubmit the last request.

See Also:

Oracle Database Development Guide for more information about Transaction Guard

---

Transaction Guard Configuration Checklist

Before configuring services for Transaction Guard, use the following configuration checklist:

- Grant permission to the application user who will call `GET_LTXID_OUTCOME`, as follows:

```
GRANT EXECUTE ON DBMS_APP_CONT to user_name;
```

- Locate and define the transaction history table for optimal performance.

    The transaction history table (`LTXID_HIST`) is created, by default, in the SYSAUX tablespace when you create or upgrade an Oracle Database. New partitions are added when you add instances, using the storage of the last partition. If the location of transaction history table is not optimal for performance, then you can move it to another tablespace and create partitions there. For example, the following statement moves the transaction history table to a tablespace named FastPace:

```
ALTER TABLE LTXID_TRANS move partition LTXID_TRANS_1 tablespace FastPace
    storage ( initial 10G next 10G minextents 1 maxextents 121 );
```

- Set values for the `-commit_outcome` and `-retention_timeout` service parameters.

- If you are using Oracle RAC, Oracle Data Guard, or Oracle Active Data Guard, then Oracle recommends that you use FAN for fast notification of an outage.

Configuring Services for Transaction Guard

To configure services to use Transaction Guard, set the following service parameters:

- **`-commit_outcome`**: Set the `-commit_outcome` service parameter to `TRUE`. This service parameter determines whether the transaction **commit outcome** is accessible after the COMMIT has executed and an outage has occurred. While Oracle Database has always made COMMIT durable, Transaction Guard makes *the outcome* of the COMMIT durable, and is used by applications to enforce the status of the last transaction executed before an outage.

- **`-retention`**: Use the `-retention` service parameter with `-commit_outcome`. This service parameter determines the amount of time, in seconds, that the COMMIT outcome is retained. Oracle recommends that most installations use the default value.

The following SRVCTL command configures a policy-managed service named `sales` for Transaction Guard:

```
$ srvctl add service -db crm -service sales -serverpool spool_1
    -commit_outcome TRUE -retention 86400 -notification TRUE
```

The following SRVCTL command configures an administrator-managed service named `sales` for Transaction Guard:

```
$ srvctl add service -db crm -service sales -preferred crm_1,crm_2
    -available crm_3,crm_4 -commit_outcome TRUE -retention 86400
    -notification TRUE
```

You can also modify an existing service to configure it for Transaction Guard by using the `srvctl modify service` command.

See Also:

- "srvctl add service" and "srvctl modify service" for information about these commands

- "Application Continuity: Automated Replay for Masking Outages"

- *Oracle Database Development Guide* for more information about Transaction Guard and idempotence

- *Oracle Database JDBC Developer's Guide* for more information about Transaction Guard for Java

- *Oracle Call Interface Programmer's Guide* for more information about developing Oracle Call Interface applications that use Transaction Guard

---

> **Note:**
>
> Do not use database services because these services are for administration purposes and cannot be manipulated. Do not use a service name that is set to *db_name* or *db_unique_name*.

Application Continuity: Automated Replay for Masking Outages

Following planned and unplanned outages, Application Continuity attempts to mask the outage by rebuilding the database session, and resubmitting the pending work following recoverable errors that make the database session unavailable. Application Continuity does not resubmit work following call failures due to non-recoverable errors. Submission of invalid data values is an example of a non-recoverable error that would not be available for replay.

When Application Continuity is configured, an end-user **request** is executed at-most once; replay is started if the time has not exceeded the replay timeout attribute specified for the service. When replaying, Application Continuity appears to the user as a slightly delayed execution. When replay succeeds, this feature masks applications from transient outages (such as session failure, instance or node outages, network failures, and so on) and from planned outages such as repairs, configuration changes, and patch application.

> **See Also:**
>
> The following documents for more information about Application Continuity
>
> - *Oracle Database Development Guide*
>
> - *Oracle Database Concepts*
>
> - *Oracle Database JDBC Developer's Guide*

To use Application Continuity, you must configure attributes for the database services that applications use, and grant permission to keep **mutables**, if the application allows. The application may also need to be modified to borrow and return connections.

If your application uses an integrated pool, then the request boundaries correspond to borrowing (check-out) and returning (check-in) of the connections. If your application does not return connections (and a property is not exposed to unpin these connections), then the request boundaries will need to be added to the application. Request boundaries would also need to be added if an integrated pool is not being used.

> **See Also:**
>
> *Oracle Database JDBC Developer's Guide* for more information about request boundaries

Application Continuity is available with the Universal Connection Pool 12*c*, JDBC Type 4 Driver 12*c*, and Oracle WebLogic Server 12.1.2. For third party Java

applications and third party Java pools, use the Oracle Database 12*c* JDBC Replay Driver.

This section includes the following topics:

- Configuration Checklist for Application Continuity on Oracle RAC

- How Application Continuity Works for Applications

- Failing Over OCI Clients with TAF

Configuration Checklist for Application Continuity on Oracle RAC

Before using Application Continuity with an Oracle RAC database, use the following configuration checklist:

- Service Configuration Checks:

    - Use a database service (also referred to as an *application service*) to connect to the database. Do not use the default database service (DB_NAME or DB_UNIQUE_NAME), or the Oracle SID or instance name for client connections.

    - Ensure you enable the appropriate attributes for the service.

    See Also:

    "Creating Services for Application Continuity and Transaction Guard"

- Software Configuration Checks (database and middle tier):

    - Use Oracle Database 12*c*, or later.

    - Use the JDBC Universal Connection Pool (12.1) or WebLogic Active GridLink (12.1.2 or later) configured with the JDBC Replay data source, or use the JDBC Replay data source with your own JDBC connection pool.

    - Oracle recommends that you use FAN and FCF for WebLogic Active GridLink Data Source or Oracle JDBC Universal Connection Pool.

    See Also:

    Oracle Database 2 Day + Real Application Clusters Guide for information about simple FAN

    - Check the resource requirements; ensure there is sufficient CPU and memory in the middle tier and database tier.

Note:

Application Continuity can lead to a slight increase in CPU consumption on the server for managing purposes and on the client due to Java garbage collection. An increase in memory consumption is expected, too, because the replay driver needs to retain the request state. The individual increase in CPU and memory consumption differs on a case-by-case basis. CPU overhead on the database server may be reduced if the platform supports the current Intel and Sparc processors.

- After consulting the application certification, GRANT keeping **mutables** to the users who will use replay.

- Use a connection string that sets connection retries and a delay between these retries. For an example of configuring the connection properties using JDBC, see "Generic Database Clients".

- Application Code Checks (consult with the application developer):

  - Decide whether to use an optional callback for initializing connections from the application to the database. When using Oracle WebLogic Server or the Universal Connection Pool, connection labeling is recommended. If registered, a callback will execute at run time and replay.

  - Determine whether the application uses Oracle JDBC concrete classes. If it does, and if access is needed to Oracle proprietary APIs, then plan to replace these concrete classes with standard JDBC or Oracle JDBC interfaces.

See Also:

My Oracle Support Note 1364193.1 for information about these interfaces: https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=1364193.1

  - Determine whether the application uses mutable function calls `SYSDATE`, `SYSTIMESTAMP`, `SYS_GUID`, or Oracle sequences. If so, then determine whether it is correct to configure the application to keep the original values during failover.

  - Determine whether replay must be explicitly disabled for any code path in the application.

  - If the application borrows and returns connections from the Oracle WebLogic Server pool or Universal Connection Pool, then, for each request, do not change anything.

    If the application uses Oracle pools, and does not return connections between requests, then determine whether there is a property to set to return connections to the pool. If there is no property to set, or if you are using the application's own connection pool, then you can add `beginRequest` and `endRequest` boundaries.

> **See Also:**
>
> - *Oracle Database Development Guide*
> - *Oracle Database Concepts*
> - *Oracle Database JDBC Developer's Guide*
> - "Creating Services for Application Continuity and Transaction Guard"

This section includes the following topics:

- Administering Mutable Values
- Disabling Replay

Administering Mutable Values

To keep function results at replay, you must grant KEEP privileges to the user invoking the function. You impose this security restriction to ensure that it is valid for replay to save and restore function results for code that is not owned by that user.

Grant and Revoke Interface

To grant and revoke permission to keep **mutables** for Oracle Dates and SYS_GUID:

```
GRANT [KEEP DATE_TIME | KEEP SYS_GUID]...[to USER]
REVOKE [KEEP DATE_TIME | KEEP SYS_GUID]...[from USER]
```

For example, for possible Oracle E-Business Suite usage with original dates:

```
GRANT KEEP DATE_TIME, KEEP SYS_GUID to [custom user];
GRANT KEEP DATE_TIME, KEEP SYS_GUID to [apps user];
```

Granting Permission to Keep Mutables for Oracle Sequences

To grant permission to the owner of the sequence:

```
CREATE SEQUENCE [sequence object] [KEEP|NOKEEP];
ALTER SEQUENCE [sequence object] [KEEP|NOKEEP];
```

The preceding commands keep the original values of `sequence.nextval` for replaying so that keys match.

To grant and revoke permission for others using the sequence:

```
GRANT KEEP SEQUENCES...[to USER] on [sequence object];
REVOKE KEEP SEQUENCES...[from USER] on [sequence object];
```

For example, for possible Oracle E-Business Suite usage with original sequence values:

```
GRANT KEEP SEQUENCES to [apps user] on [sequence object];
GRANT KEEP SEQUENCES to [custom user] on [sequence object];
```

Rules for Grants on Mutables

- If you grant all on an object for a user, then **mutables** are excluded. Mutables require explicit grants. It is not supported to grant mutables to the users supplied or created by Oracle Database, such as SYS, AUDSYS, GSMUSER, SYSTEM.

- The DBA role includes mutable permission.

- If a user has mutables granted, then the objects inherit mutable access when the mutable functions are called (in SYS_GUID, SYSDATE and SYSTIMESTAMP).

- If keeping mutables on a sequence object is revoked, then SQL or PL/SQL commands using that object does not allow mutable collection or application for that sequence.

- If grants are revoked between run time and failover, then the mutables that were collected are not applied.

- If grants are granted between run time and failover, then mutables are not collected and so none are applied.

Disabling Replay

Some application **request**s are unsuitable for replay. If any application modules are using such designs, then the disable replay API allows disabling of replay on a per request basis. Disabling replay can be added to the callback or to mainline code. Disabling replay requires a callback or change to the request.

> **Caution:**
>
> During replay, autonomous transactions, external PL/SQL calls, and Java callouts are examples of side effects that are separate from the main transaction, and these side effects are replayed unless you specify otherwise. Examples of side effects separate from the main transaction include writing to an external table, sending email, forking sessions out of PL/SQL or Java, transferring files, and accessing external URLs. Actions such as these leave persistent side effects. PL/SQL messaging can leave persistent results behind.

If you configured Application Continuity so that a DBA can end or disconnect a session by using either the ALTER SYSTEM KILL SESSION or ALTER SYSTEM DISCONNECT SESSION statement, then Application Continuity, by default, attempts to recover the session. However, if you do not want the session to be replayed, then use the NOREPLAY keyword, as in the following examples:

```
alter system kill session 'sid, serial#, @inst' noreplay;
alter system disconnect session 'sid, serial#, @inst' noreplay
```

> **See Also:**
>
> - *Oracle Database Development Guide* for more information about side effects
>
> - *Oracle Database Development Guide* for information about disabling and disconnecting sessions.

How Application Continuity Works for Applications

Application Continuity operates on request boundaries. A **request** is a unit of work submitted by the application. It is normally executed between borrowing and returning connections for Universal Connection Pool, Oracle WebLogic Server, DRCP, and third-party connection pools. A request is marked by the beginRequest and

`endRequest` APIs. Oracle Database provides these APIs, by default, when using one of the aforementioned connections pools.

Application Continuity works for applications, as follows:

1. You either create a new database service or modify an existing database service so that the `-failovertype` parameter is set to `TRANSACTION`, and the `-commit_outcome` parameter is set to `TRUE`.

> **See Also:**
>
> "Creating Services for Application Continuity and Transaction Guard"

2. When an application opens a connection to the database using this service or borrows a connection from an Oracle connection pool using this service, the session handle carries a logical transaction ID (LTXID) for each transaction.

3. When the application submits a request, the server and replay driver collaborate to determine what states are to be captured for replay if there is an outage. If a **recoverable error** occurs (planned or unplanned), then the error message is sent back to the replay driver.

4. For JDBC-thin (type 4) applications, the replay driver receives the FAN message (down or error). Fast Connection Failover (FCF) aborts the dead session. The replay driver then:

   a. Replaces the dead physical session with a new clean session and reestablishes FAN in case a later error occurs during or after replay.

   b. Uses Transaction Guard to determine the outcome of the in-flight transaction, if one was open, to prepare for replay.

   c. Optionally, calls back using a labeling callback or reconnect callback for the initial state, if one of these is registered.

   d. Rebuilds the database session, recovering the transactional and nontransactional states, and validating at each step that the data and messages seen by the client driver are the same as those that the client may have seen and potentially made a decision on.

5. Oracle Database validates replay when replay is initiated and as replay progresses. If the database session cannot be rebuilt correctly, because the client data cannot be repositioned to the same data values and messages that the client potentially made decisions on previously, then the replay driver returns the original error to the application.

> **See Also:**
>
> - "Service Deployment Options"
>
> - "Fast Application Notification"
>
> - "Configuring JDBC-Thin Clients for Application Continuity for Java"
>
> - "Configuring OCI Clients to use Transaction Guard"

Failing Over OCI Clients with TAF

When Oracle Net Services establishes a connection to an instance, the connection remains open until the client closes the connection, the instance is shutdown, or a failure occurs. If you configure transparent application failover (TAF) for the connection, then Oracle Database replays the session at a surviving instance when an outage occurs.

TAF can restart a query after failover has completed but for other types of transactions, such as INSERT, UPDATE, or DELETE, the application must rollback the failed transaction and resubmit the transaction. You must also reexecute any session customizations, in other words, ALTER SESSION statements, after failover has occurred. However, with TAF, a connection is not moved during normal processing, even if the workload changes over time.

Services simplify the deployment of TAF. You can define a TAF policy for a service, and all connections using this service will automatically have TAF enabled. This does not require any client-side changes. The TAF setting on a service overrides any TAF setting in the client connection definition.

You can define a TAF policy for all users of a service by defining the –failovermethod and –failovertype parameters. You can further define the TAF policy by setting the number of times that a failed session attempts to reconnect to the service and how long it should wait between reconnection attempts using the –failoverretry and –failoverdelay parameters, respectively.

To define a TAF policy for a service, use SRVCTL as in the following example, where the service name is tafconn.example.com and the database name is CRM:

```
$ srvctl modify service -db crm -service tafconn.example.com -failovermethod BASIC
    -failovertype SELECT -failoverretry 10 -failoverdelay 30
```

OCI applications with TAF enabled should use FAN high availability events for fast connection failover.

> **See Also:**
>
> *Oracle Database Net Services Administrator's Guide* for more information about configuring TAF

Load Balancing Advisory

This section describes the load balancing advisory under the following topics:

- Overview of the Load Balancing Advisory

- Configuring Your Environment to Use the Load Balancing Advisory

- Load Balancing Advisory FAN Events

- Monitoring Load Balancing Advisory FAN Events

Overview of the Load Balancing Advisory

Load balancing distributes work across all of the available Oracle RAC database instances. Oracle recommends that applications use connection pools with persistent connections that span the instances that offer a particular service. When using

persistent connections, connections are created infrequently and exist for a long duration. Work comes into the system with high frequency, borrows these connections, and exists for a relatively short duration. The load balancing advisory provides advice about how to direct incoming work to the instances that provide the optimal quality of service for that work. This minimizes the need to relocate the work later.

By using the Load Balancing Advisory and run-time connection load balancing goals, feedback is built in to the system. Work is routed to provide the best service times globally, and routing responds gracefully to changing system conditions. In a steady state, the system approaches equilibrium with improved throughput across all of the Oracle RAC instances.

Standard architectures that can use the load balancing advisory include connection load balancing, transaction processing monitors, application servers, connection concentrators, hardware and software load balancers, job schedulers, batch schedulers, and message queuing systems. All of these applications can allocate work.

The load balancing advisory is deployed with key Oracle clients, such as a listener, the JDBC universal connection pool, OCI session pool, Oracle WebLogic Server Active GridLink for Oracle RAC, and the ODP.NET Connection Pools. Third-party applications can also subscribe to load balancing advisory events by using JDBC and Oracle RAC FAN API or by using callbacks with OCI.

Configuring Your Environment to Use the Load Balancing Advisory

You can configure your environment to use the load balancing advisory by defining service-level goals for each service for which you want to enable load balancing. Configuring a service-level goal enables the load balancing advisory and the publishing of FAN load balancing events for that service.

There are two types of service-level goals for run-time connection load balancing:

- SERVICE_TIME: Attempts to direct work requests to instances according to response time. Load balancing advisory data is based on elapsed time for work done in the service plus available bandwidth to the service. An example for the use of SERVICE_TIME is for workloads such as internet shopping where the rate of demand changes. The following example shows how to set the goal to SERVICE_TIME for connections using the online service:

  ```
  $ srvctl modify service -db db_unique_name -service online
      -rlbgoal SERVICE_TIME -clbgoal SHORT
  ```

- THROUGHPUT: Attempts to direct work requests according to throughput. The load balancing advisory is based on the rate that work is completed in the service plus available bandwidth to the service. An example for the use of THROUGHPUT is for workloads such as batch processes, where the next job starts when the last job completes. The following example shows how to set the goal to THROUGHPUT for connections using the sjob service:

  ```
  $ srvctl modify service -db db_unique_name -service sjob
      -rlbgoal THROUGHPUT -clbgoal LONG
  ```

Setting the run-time connection load balancing goal to NONE disables load balancing for the service. You can see the goal settings for a service in the data dictionary by querying the DBA_SERVICES, V$SERVICES, and V$ACTIVE_SERVICES views. You can also review the load balancing settings for a service using Oracle Enterprise Manager.

> **See Also:**
>
> - "Administering Services" for more information about administering services and adding goals to services
>
> - *Oracle Database 2 Day + Real Application Clusters Guide* for more information about managing services using Oracle Enterprise Manager

Load Balancing Advisory FAN Events

The load balancing advisory FAN events provide metrics for load balancing algorithms. The easiest way to take advantage of these events is to use the run-time connection load balancing feature of an Oracle integrated client such as JDBC, Universal Connection Pool (or the deprecated Implicit Connection Cache), ODP.NET Connection Pools, OCI session pools, or Oracle WebLogic Server Active GridLink for Oracle RAC. Other client applications can take advantage of FAN programatically by using the Oracle RAC FAN API to subscribe to FAN events and execute event-handling actions upon receipt. Table 5-4 describes the load balancing advisory FAN event parameters.

> **See Also:**
>
> *Oracle Database JDBC Developer's Guide* for more information about the Oracle RAC FAN API

Table 5-4   Load Balancing Advisory FAN Events

Parameter	Description
VERSION	Version of the event record. Used to identify release changes.
EVENT_TYPE	A load balancing advisory event is always of the SERVICEMETRICS event type.
SERVICE	The service name; matches the value of NAME in DBA_SERVICES.
DATABASE	The unique database supporting the service; matches the initialization parameter value for DB_UNIQUE_NAME, which defaults to the value of the initialization parameter DB_NAME.
INSTANCE	The name of the instance that supports the service; matches the ORACLE_SID value.
PERCENT	The percentage of work requests to send to this database instance.
FLAG	Indication of the service quality relative to the service goal. Valid values are GOOD, VIOLATING, NO DATA, and BLOCKED.
TIMESTAMP	The local time zone to use when ordering notification events.

> **Note:**
>
> The INSTANCE, PERCENT, and FLAG event parameters are generated for each instance offering the service. Each set of instance data is enclosed within braces ({ }).

Monitoring Load Balancing Advisory FAN Events

You can use the following query against the internal queue table for load balancing advisory FAN events to monitor load balancing advisory events generated for an instance:

```
SET PAGES 60 COLSEP '|' LINES 132 NUM 8 VERIFY OFF FEEDBACK OFF
COLUMN user_data HEADING "AQ Service Metrics" FORMAT A60 WRAP
BREAK ON service_name SKIP 1
SELECT
 TO_CHAR(enq_time, 'HH:MI:SS') Enq_time, user_data
 FROM sys.sys$service_metrics_tab
 ORDER BY 1 ;
```

The results of this query contain rows similar to the following:

```
02:56:05|SYS$RLBTYP('hr', 'VERSION=1.0 database=sales service=hr
   { {instance=sales_4 percent=38 flag=GOOD aff=TRUE}{instance=sales_1
   percent=62 flag=GOOD aff=TRUE} } timestamp=2012-07-16 07:56:05')
```

Following is an example of a load balancing advisory event for the lba_serv service offered on two instances (orcl1 and orcl2), as captured from Oracle Notification Service using the Oracle RAC FAN API:

```
Notification Type: database/event/servicemetrics/lba_serv.example.com
   VERSION=1.0 database=orcl service=lba_serv.example.com { {instance=orcl2
   percent=50 flag=UNKNOWN aff=FALSE}{instance=orcl1 percent=50 flag=UNKNOWN
   aff=FALSE} } timestamp=2012-07-06 13:19:12
```

> **Note:**
>
> The SERVICMETRICS events are not visible through the FAN callout mechanism.

Enabling Clients for Oracle RAC

Oracle has integrated FAN with many of the common client application environments that are used to connect to Oracle RAC databases. Therefore, the easiest way to use FAN is to use an integrated Oracle Client.

The following sections discuss how FAN is integrated with Oracle Clients and how to enable FAN events for the several specific client development environments:

- Overview of Oracle Integrated Clients and FAN

- Enabling JDBC-Thin Clients for Fast Connection Failover

- Enabling JDBC Clients for Run-time Connection Load Balancing

- Configuring JDBC-Thin Clients for Application Continuity for Java

- Configuring JDBC-Thin Clients for Transaction Guard

- Enabling OCI Clients for Fast Connection Failover

- Enabling OCI Clients for Run-time Connection Load Balancing

- Configuring OCI Clients to use Transaction Guard

- Enabling ODP.NET Clients to Receive FAN High Availability Events

- Enabling ODP.NET Clients to Receive FAN Load Balancing Advisory Events

- Configuring ODP.NET Clients to use Transaction Guard

Overview of Oracle Integrated Clients and FAN

The overall goals of FAN are to enable end-to-end, lights-out recovery of applications and load balancing based on real transaction performance. Applications use the FAN high availability (HA) events to achieve very fast detection of failures, balancing of connection pools following failures, and distribution of connections again when the failed components are repaired.

The FAN events carrying load balancing advice help connection pools consistently deliver connections to available instances that provide the best service. FAN HA is integrated with the JDBC-thin and OCI drivers. FAN HA and FAN load balancing are both integrated with the JDBC Universal Connection Pool (and the deprecated Implicit Connection Cache), the OCI session pools, the ODP.NET connection pool, and Oracle WebLogic Server Active GridLink for Oracle RAC.

Due to the integration with FAN, Oracle integrated clients are more aware of the current status of an Oracle RAC cluster. This prevents client connections from waiting or trying to connect to instances or services that are no longer available. When instances start, Oracle RAC uses FAN to notify the connection pool so that the connection pool can create connections to the recently started instance and take advantage of the additional resources that this instance provides.

Oracle client drivers that are integrated with FAN can:

- Remove terminated connections immediately when a service is declared DOWN at an instance, and immediately when nodes are declared DOWN

- Report errors to clients immediately when Oracle Database detects the NOT RESTARTING state, instead of making the client wait while the service repeatedly attempts to restart

Oracle connection pools that are integrated with FAN can:

- Balance connections across all of the Oracle RAC instances when a service starts; this is preferable to directing the sessions that are defined for the connection pool to the first Oracle RAC instance that supports the service

- Balance work requests at run time using load balancing advisory events

The use of client drivers or connection pools and FAN requires that you properly configure the Oracle Notification Service to deliver the FAN events to the clients. In addition, for load balancing, configure database connection load balancing across all of the instances that provide the services used by the connection pool. Oracle recommends that you configure both client-side and server-side load balancing with Oracle Net Services. If you use DBCA to create your database, then both client-side and server-side load balancing are configured by default.

See Also:

"Connection Load Balancing"

Enabling JDBC-Thin Clients for Fast Connection Failover

Enabling Fast Connection Failover (FCF) for Universal Connection Pool and Oracle WebLogic Server Active GridLink for Oracle RAC enables the use of FAN HA and load balancing advisory events. For Universal Connection Pool to use FAN, your application can use the JDBC development environment for either JDBC OCI or JDBC Thin clients. The Java Database Connectivity Oracle Call Interface (JDBC/OCI) driver connection pooling functionality is part of the JDBC-thin client. This functionality is provided by the `OracleOCIConnectionPool` class.

To enable FCF for the JDBC-thin client, call the method `setFastConnectionFailoverEnabled(true)` of the `OracleDataSource` class in the `oracle.jdbc.pool` package before making the first `getConnection()` request. When you enable FCF for the JDBC-thin client, the failover property applies to every connection in the connection pool. Enabling FCF with JDBC-thin driver or JDBC/OCI clients enables the connection pool to receive and react to all FAN events.

JDBC application developers can programmatically integrate with FAN by using a set of APIs introduced in Oracle Database 11*g* release 2 (11.2). The Oracle RAC FAN APIs enable application code to receive and respond to FAN event notifications sent by Oracle RAC in the following ways:

- Listening for Oracle RAC service down, service up, and node down events

- Listening for load balancing advisory events and responding to them

---

See Also:

- *Oracle Database JDBC Developer's Guide* for more information about using APIs, configuring the JDBC universal connection pool, and Oracle Notification Service

- *Oracle Database 2 Day + Real Application Clusters Guide* for more information about configuring JDBC-thin clients

---

Oracle Notification Service for JDBC-Thin Clients

FCF relies on Oracle Notification Service to propagate database events between the connection pool and the Oracle RAC database. At run time, the connection pool must be able to setup an Oracle Notification Service environment. Oracle Notification Service (`ons.jar`) is included as part of the Oracle Client software. Oracle Notification Service can be configured using either remote configuration or client-side Oracle Notification Service daemon configuration. Remote Oracle Notification Service subscription offers the following advantages:

- Support for an All Java mid-tier software

- An Oracle Notification Service daemon is not necessary on the client system, so you do not have to manage this process

- Simple configuration by way of a `DataSource` property

Configuring FCF for JDBC/OCI and JDBC-Thin Driver Clients

You can enable FCF for Universal Connection Pool or Implicit Connection Cache. Oracle recommends using the Universal Connection Pool for Java because the Implicit Connection Cache is deprecated. You can also use Oracle WebLogic Server Active GridLink for Oracle RAC.

This procedure explains how to enable FCF for JDBC. For JDBC/OCI clients, if you enable FCF, then do not use the method used with Oracle Database 11*g* release 2 (11.2) of enabling FAN for OCI clients (setting `notification` to TRUE on the service), and do not configure TAF, either on the client or for the service. You can also configure Application Continuity and Transaction Guard.

To enable FCF, you must first enable the Universal Connection Pool, as described in the following procedure:

1. Create the connection pool and set
    `setFastConnectionFailoverEnabled(true)`.

    The following example creates a connection pool and enables FCF. The `ucp.jar` library must be included in the classpath of an application to use this example.

    ```
    PoolDataSource pds = PoolDataSourceFactory.getPoolDataSource();
    pds.setFastConnectionFailoverEnabled(true);
    ```

2. Determine the ports to use for Oracle Notification Service remote subscriptions.

    Use the following command to view the Oracle Notification Service configuration on each node that is running Oracle Clusterware as in the following example:

    ```
    srvctl config nodeapps -onsonly
    ```

    The output of this command lists the local and remote ports configured for Oracle Notification Service.

    > **Note:**
    >
    > Oracle Notification Service configuration should have been automatically completed during the Oracle Clusterware installation.

3. Configure the remote Oracle Notification Service subscription.

    When using the Universal Connection Pool, an application calls `setONSConfiguration` for an `OracleDataSource` instance and specifies the nodes and port numbers to use. The port numbers used for each node are the same as the remote port displayed for each node in Step 2, as shown in the following example. The `ons.jar` library must be included in the classpath of an application to use this example.

    ```
    pds.setONSConfiguration("nodes=racnode1:6200,racnode2:6200");
    ```

    Applications that use remote Oracle Notification Service configuration must set the `oracle.ons.oraclehome` system property to the location of ORACLE_HOME before starting the application, for example:

    ```
    java -Doracle.ons.oraclehome=$ORACLE_HOME ...
    ```

4. Configure the connection URL.

A connection factory's connection URL must use the service name syntax when using FCF. The service name is used to map the connection pool to the service. The following example demonstrates configuring the connection URL:

```
pds.setConnectionFactoryClassName("oracle.jdbc.pool.OracleDataSource");
pds.setURL("jdbc:oracle:thin@//SCAN_name:service_name");...
```

See Also:

- *Oracle Database JDBC Developer's Guide* for more information about JDBC

- *Oracle Universal Connection Pool for JDBC Developer's Guide* for more information about configuring FCF

Enabling JDBC Clients for Run-time Connection Load Balancing

Oracle JDBC Universal Connection Pool and Oracle WebLogic Server Active GridLink for Oracle RAC leverage the load balancing functionality provided by an Oracle RAC database. Run-time connection load balancing requires the use of an Oracle JDBC driver and an Oracle RAC database.

The Universal Connection Pool and Oracle WebLogic Server Active GridLink for Oracle RAC are integrated to take advantage of Load Balancing Advisory information. Oracle introduced the Universal Connection Pool for JDBC in Oracle Database 11*g* release 11.1.0.7.0. Consequently, Oracle deprecated the existing JDBC connection pool, the Implicit Connection Cache, which was introduced in Oracle Database 10*g* release 1 for use with Oracle RAC databases. In addition to Oracle Database 12*c*, you can also use the Universal Connection Pool with Oracle Database 10*g* or Oracle Database 11*g*.

Run-time connection load balancing requires that FCF is enabled and configured properly. In addition, the Oracle RAC load balancing advisory must be configured with service-level goals for each service used by the connection pool. The connection load balancing goal should be set to SHORT, for example:

```
srvctl modify service -db db_unique_name -service service_name
   -rlbgoal SERVICE_TIME -clbgoal SHORT
```

See Also:

- "Configuring FCF for JDBC/OCI and JDBC-Thin Driver Clients"

- *Oracle Universal Connection Pool for JDBC Developer's Guide* for more information about configuring run-time connection load balancing for Universal Connection Pool

---

Configuring JDBC-Thin Clients for Application Continuity for Java

The Replay data source (`oracle.jdbc.replay.OracleDataSource`) is a JDBC-thin data source that Application Continuity requires for Java. This data source serves as the connection factory that produces new physical JDBC connections, for both Universal Connection Pool and Oracle WebLogic Server Active GridLink for Oracle RAC data sources. The JDBC replay driver maintains a history of calls during a client conversation with Oracle Database 12*c*, in collaboration with Oracle Database. Following any outage of the session caused by a loss of database service, planned or

unplanned, under the direction of the database, the JDBC replay driver attempts to rebuild the non-transactional and transactional database session states, so that the outage appears as a delayed execution.

To use Application Continuity for Java and the JDBC replay driver, you must use an Oracle Database 12*c* client and connect to an Oracle Database 12*c* database. Application Continuity for Java is supported in the following configurations:

- JDBC applications using Oracle JDBC Replay data source and using neither Universal Connection Pool or Oracle WebLogic Server Active GridLink—typical third-party, JDBC-based connection pools

- JDBC applications using Universal Connection Pool data sources—standalone or third-party application servers configured to use a Universal Connection Pool data source

- JDBC applications using only Oracle WebLogic Server Active GridLink but not Universal Connection Pool data sources—typical Oracle WebLogic Server J2EE cases

To configure JDBC-thin clients to use the JDBC Replay Driver:

1. Ensure that you are using an application that is certified for replay.

   > **See Also:**
   >
   > "Configuration Checklist for Application Continuity on Oracle RAC"

2. Use SRVCTL to create a service for use by the application, if one does not already exist. Set the `-failovertype` parameter to `TRANSACTION` and the `-commit_outcome` parameter to `TRUE` for this service.

   > **See Also:**
   >
   > "Creating Services for Application Continuity and Transaction Guard" for more information about creating services.

3. Configure the connection element using the `replayDataSource` object, as shown in the following example:

   ```
   replayDataSource rds = PoolDataSourceFactory.getreplayDataSource();
   rds.setConnnectionPoolName("replayExample");
   rds.setONSConfiguration("nodes=racnode1:4200,racnode2:4200");
   rds.setFastConnectionFailoverEnabled(true);
   rds.setConnectionFactoryClassName("oracle.jdbc.replay.OracleDataSource");

   Connection conn = replayDataSource.getConnection();
   ```

4. When connecting to the database, use a URL that can access all instances offering the service.

> **See Also:**
>
> - "Configuring FCF for JDBC/OCI and JDBC-Thin Driver Clients" for information about configuring FAN/FCF
>
> - *Oracle Database JDBC Developer's Guide* for information about configuring Transaction Guard without enabling Application Continuity
>
> - *Oracle Database Development Guide*
>
> - *Oracle Universal Connection Pool for JDBC Developer's Guide*

Configuring JDBC-Thin Clients for Transaction Guard

Transaction Guard provides a protocol and a generic tool for applications to use for at-most-once execution in case of planned and unplanned outages. Applications use the logical transaction ID to determine the outcome of the last transaction open in a database session following an outage. Without Transaction Guard, end users or applications that attempt to retry operations following outages can cause logical corruption by committing duplicate transactions or committing transactions out of order.

> **See Also:**
>
> - *Oracle Database JDBC Developer's Guide* for information about configuring Transaction Guard
>
> - *Oracle Database Development Guide* for information about using Transaction Guard
>
> - *Oracle Call Interface Programmer's Guide* for information about OCI and Transaction Guard
>
> - *Oracle Call Interface Programmer's Guide* for information about configuring Transparent Application Failover

Enabling OCI Clients for Fast Connection Failover

OCI clients can enable FCF by registering to receive notifications for Oracle RAC high availability FAN events and responding when events occur. Using FCF improves the session failover response time in OCI applications and also removes connections to nonfunctioning instances from connection and session pools. FCF can be used in OCI applications that also use TAF, OCI drivers (including your own connection pools), OCI connection pool, and OCI session pools. FAN is posted over the Oracle Notification Service for both high availability and load balancing events.

To use FCF, you must use a service with FAN enabled. FAN is published over Oracle Notification Service. Client applications can also register callbacks that are used whenever an event occurs. This reduces the time that it takes to detect a connection failure.

During DOWN event processing, OCI:

- Terminates affected connections at the client and returns an error

- Removes connections from the OCI connection pool and the OCI session pool—the session pool maps each session to a physical connection in the connection pool, and there can be multiple sessions for each connection

- Fails over the connection if you have configured TAF. If TAF is not configured, then the client only receives an error if the instance it is connected to fails.

If your application is using TAF, then you must enable the TAF properties for the service using SRVCTL or Oracle Enterprise Manager. Configure your OCI client applications to connect to an Oracle RAC database using the configured service.

> **Note:**
>
> OCI does not manage UP events.

> **See Also:**
>
> *Oracle Database Net Services Administrator's Guide* for more information about configuring TAF

Configuring FCF for OCI Clients

OCI applications must connect to an Oracle RAC instance to enable HA event notification. Furthermore, these applications must perform the following steps to configure FCF for an OCI client:

1. Configure the service for your OCI connection pool to enable FAN, connection load balancing, and run-time connection load balancing, as shown in the following example:

    ```
    $ srvctl modify service -db crm -service ociapp.example.com -notification TRUE
    ```

2. Link the application with a thread library.

3. After linking with the thread library, the applications can register a callback that is invoked whenever a FAN event occurs.

> **See Also:**
>
> *Oracle Call Interface Programmer's Guide* for more information about OCI

Enabling OCI Clients for Run-time Connection Load Balancing

As of Oracle Database 12*c*, OCI session pooling enables multiple threads of an application to use a dynamically managed set of pre-created database sessions. In connection pooling, the pool element is a connection, but in session pooling, the pool element is a session. Oracle Database continually reuses the sessions in the session pool to form nearly permanent channels to the instances, thus saving the overhead of creating and closing sessions every time applications need them.

Run-time connection load balancing is enabled by default in an Oracle Database 11*g* release 11.1, or later, client communicating with a server of Oracle Database 10*g* release 10.2, or later. For Oracle RAC environments, session pools use service metrics received from the Oracle RAC load balancing advisory[1] through Fast Application

Notification (FAN) events to balance application session requests. The work requests coming into the session pool can be distributed across the instances of Oracle RAC offering a service, using the current service performance.

Configuring OCI Clients to Receive Load Balancing Advisory FAN Events

For Oracle RAC environments, session pools use service metrics received from the Oracle RAC load balancing advisory through Fast Application Notification (FAN) events to balance application session requests. To enable your application to receive the service metrics based on the service time, ensure that you configure FAN, the load balancing advisory goal (-rlbgoal parameter), and the connection load balancing goal (-clbgoal parameter) for a service that is used by the session pool, as shown in the following example:

```
$ srvctl modify service -db crm -service ociapp.example.com -rlbgoal SERVICE_TIME
    -clbgoal SHORT -notification TRUE
```

> **See Also:**
>
> *Oracle Call Interface Programmer's Guide* for more information about OCI

Configuring OCI Clients to use Transaction Guard

OCI supports FAN messages and Transaction Guard. FAN is designed to quickly notify an OCI-based application of outages at the node, database, instance, service, and public network levels. Once notified of the failure, an application can leverage Transaction Guard to reliably determine the outcome of the last in-flight transaction.

Transaction Guard avoids the costs of ambiguous errors that lead to user frustration, customer support calls, and lost opportunities. Transaction Guard is safer and performs better, with lower overheads, than home grown solutions for a known outcome.

> **See Also:**
>
> - "Fast Application Notification"
>
> - "Enabling Clients for Oracle RAC"
>
> - *Oracle Call Interface Programmer's Guide* for more information about event notification and user-registered callbacks

Enabling ODP.NET Clients to Receive FAN High Availability Events

ODP.NET connection pools can subscribe to FAN HA notifications that indicate when nodes, services, and service members are down. After a DOWN event, Oracle Database cleans up sessions in the connection pool that go to the instance and ODP.NET proactively removes connections that are no longer valid. ODP.NET establishes additional connections to existing Oracle RAC instances if the removal of invalid

---

[1] Run-time connection load balancing is basically routing work requests to sessions in a session pool that can best serve the work. It comes into effect when selecting a session from an existing session pool. Thus, run-time connection load balancing is a very frequent activity.

connections reduces the total number of connections to below the value for the `Min Pool Size` parameter.

When connecting to Oracle Database 12*c* and later, ODP.NET uses Oracle Notification Service, rather than Advanced Queuing.

Enable Fast Connection Failover for ODP.NET connection pools by subscribing to FAN high availability events. To enable Fast Connection Failover, include "`HA Events=true`" and "`pooling=true`" (the default value) in the connection string, as shown in the following example where *user_name* is the name of the database user and *password* is the password for that user:

```
con.ConnectionString =
   "User Id=user_name;Password=password;Data Source=odpnet;" +
   "Min Pool Size=10;Connection Lifetime=120;Connection Timeout=60;" +
   "HA Events=true;Incr Pool Size=5;Decr Pool Size=2";
```

> **See Also:**
>
> *Oracle Data Provider for .NET Developer's Guide for Microsoft Windows* for more information about using FAN events in ODP.NET applications

Enabling ODP.NET Clients to Receive FAN Load Balancing Advisory Events

When connecting to Oracle Database 12*c* and later, ODP.NET uses Oracle Notification Service, rather than Advanced Queuing.

Use the following procedure to enable ODP.NET clients or applications to receive FAN load balancing advisory events:

1. Enable Oracle Notification Service notifications by using SRVCTL, and set the run-time load balancing goal, as shown in the following example:

   ```
   $ srvctl modify service -db crm -service odpapp.example.com
        -notification TRUE -clbgoal LONG -rlbgoal SERVICE_TIME
   ```

2. Ensure Oracle Notification Service (ONS) is configured for FAN events including run-time load balancing advice.

3. To take advantage of load balancing events with ODP.NET connection pools, set the load balancing attribute in the ConnectionString to `TRUE` (the default is `FALSE`). You can do this at connect time. This only works if you are using connection pools, or when the pooling attribute is set to `TRUE` which is the default.

   The following example demonstrates how to configure the ConnectionString to enable load balancing, where *user_name* is the name of the user and *password* is the password:

   ```
   con.ConnectionString =
     "User Id=user_name;Password=password;Data Source=odpapp;" +
     "Min Pool Size=10;Connection Lifetime=120;Connection Timeout=60;" +
     "Load Balancing=true;Incr Pool Size=5;Decr Pool Size=2";
   ```

   > **Note:**
   >
   > ODP.NET does not support connection redistribution when a node starts (UP events). However, if you have enabled failover on the server-side, then ODP.NET can migrate connections to newly available instances.

> **See Also:**
>
> - *Oracle Data Provider for .NET Developer's Guide for Microsoft Windows* for more information about ODP. NET
>
> - "srvctl modify service" for information about using this command

Configuring ODP.NET Clients to use Transaction Guard

ODP.NET supports FAN messages and Transaction Guard. FAN is designed to quickly notify an ODP.NET-based application of outages at the node, database, instance, service, and public network levels. Once notified of the failure, an application can leverage Transaction Guard to reliably determine the outcome of the last in-flight transaction.

Transaction Guard avoids the costs of ambiguous errors that lead to user frustration, customer support calls, and lost opportunities. Transaction Guard is safer and performs better, with lower overheads, than home grown solutions for a known outcome.

> **See Also:**
>
> - "Fast Application Notification"
>
> - "Administering Services"
>
> - "Creating Services for Application Continuity and Transaction Guard"
>
> - *Oracle Data Provider for .NET Developer's Guide for Microsoft Windows* for more information about ODP.NET applications

Distributed Transaction Processing in Oracle RAC

The X/Open Distributed Transaction Processing (DTP) architecture defines a standard architecture or interface that enables multiple application programs (APs) to share resources provided by multiple, and possibly different, resource managers (RMs). It coordinates the work between APs and RMs into global transactions.

The following sections discuss how Oracle RAC supports global (XA) transactions and DTP processing:

- Overview of XA Transactions and DTP Services

- Benefits of DTP Services and XA Affinity for XA Transactions

- Using DTP Services with Oracle RAC

- Configuring DTP Services

- Relocating DTP Services in Administrator-Managed Databases

Overview of XA Transactions and DTP Services

A global (XA) transaction can span Oracle RAC instances by default, allowing any application that uses the Oracle XA library to take full advantage of the Oracle RAC environment to enhance the availability and scalability of the application.

GTX*n* background processes support XA transactions in an Oracle RAC environment. The GLOBAL_TXN_PROCESSES initialization parameter, which is set to 1 by default, specifies the initial number of GTX*n* background processes for each Oracle RAC instance. Use the default value for this parameter clusterwide to allow distributed transactions to span multiple Oracle RAC instances. Using the default value allows the units of work performed across these Oracle RAC instances to share resources and act as a single transaction (that is, the units of work are *tightly coupled*). It also allows 2PC requests to be sent to any node in the cluster.

Before Oracle RAC 11*g* release 1 (11.1), the way to achieve tight coupling in Oracle RAC was to use distributed transaction processing (DTP) services, that is, services whose cardinality (one) ensured that all tightly-coupled branches landed on the same instance—regardless of whether load balancing was enabled. If the XA application does not use suspend and resume on the same transaction branch, and does not issue savepoints that span branches, then tightly coupled XA transactions no longer require the special type of singleton services to be deployed on Oracle RAC databases. If your application cannot determine whether a transaction branch has been suspended and resumed, then the application must continue to use DTP services or preferably use XA affinity.

XA affinity (placing all branches of the same XA transaction at the same Oracle RAC instance) is a requirement when suspending and resuming the same XA branch or if using savepoints across branches. It also provides much better performance because different transactions can be balanced. XA affinity is available with Oracle WebLogic Server Active GridLink for Oracle RAC, JDBC Universal Connection Pool, and Oracle Tuxedo.

Note:

Many transaction processing monitors continue to require either XA affinity or DTP services because the applications that use them suspend and resume the same branch or use savepoints.

See Also:

"Benefits of DTP Services and XA Affinity for XA Transactions"

An external transaction manager, such as Oracle Services for Microsoft Transaction Server (OraMTS), coordinates DTP/XA transactions. However, an internal Oracle transaction manager coordinates distributed SQL transactions. Both DTP/XA and distributed SQL transactions must use the DTP service in Oracle RAC.

> **See Also:**
>
> - *Oracle Database Development Guide* for complete information about using Oracle XA with Oracle RAC
>
> - *Oracle Database Reference* for information about the `GLOBAL_TXN_PROCESSES` initialization parameter

Benefits of DTP Services and XA Affinity for XA Transactions

To provide improved application performance with distributed transaction processing (DTP) in Oracle RAC, you may want to take advantage of DTP services or XA affinity. Using DTP services, you can direct all branches of a distributed transaction to a single instance in the cluster. To load balance across the cluster, it is better to have several groups of smaller application servers with each group directing its transactions to a single service, or set of services, than to have one or two larger application servers. DTP or XA affinity is required, if suspending and resuming the same XA branch.

In addition, connection pools at the application server tier that load balance across multiple connections to an Oracle RAC database can use XA affinity to ensure that all tightly-coupled branches of a global distributed transaction run on only one Oracle RAC instance. This is also true in distributed transaction environments using protocols such as X/Open Distributed Transaction Processing (DTP) or the Microsoft Distributed Transaction Coordinator (DTC).

To enhance the performance of distributed transactions, you can use services to manage DTP environments. By defining the DTP property of a service, the service is guaranteed to run on one instance at a time in an Oracle RAC database. All global distributed transactions performed through the DTP service are ensured to have their tightly-coupled branches running on a single Oracle RAC instance. This has the following benefits:

- The changes are available locally within one Oracle RAC instance when tightly coupled branches need information about changes made by each other

- Relocation and failover of services are fully supported for DTP

- By using more DTP services than there are Oracle RAC instances, Oracle Database can balance the load by services across all of the Oracle RAC database instances

Using DTP Services with Oracle RAC

To leverage all of the instances in a cluster, create one or more distributed transaction processing (DTP) services for each Oracle RAC instance that hosts distributed transactions. Choose one DTP service for one distributed transaction. Choose different DTP services for different distributed transactions to balance the workload among the Oracle RAC database instances. Because all of the branches of a distributed transaction are on one instance, you can leverage all of the instances to balance the load of many DTP transactions through multiple singleton services, thereby maximizing application throughput.

If you add or delete nodes from your cluster database, then you may have to identify and relocate services that you are using for DTP transactions to ensure that you maintain optimum performance levels.

> **See Also:**
>
> *Oracle Database Development Guide* for more information about transaction branch management in Oracle RAC

Configuring DTP Services

To create distributed transaction processing (DTP) services for distributed transaction processing, perform the following steps:

1. Create a singleton service using Oracle Enterprise Manager or SRVCTL.

    For an administrator-managed database, define only one instance as the preferred instance. You can have as many available instances as you want, for example:

    ```
    $ srvctl add service -db crm -service xa_01.example.com -preferred RAC01
        -available RAC02,RAC03
    ```

    For a policy-managed database, specify the server pool to use, and set the cardinality of the service to SINGLETON, for example:

    ```
    $ srvctl add service -db crm -service xa_01.example.com -serverpool dtp_pool
        -cardinality SINGLETON
    ```

2. Set the DTP parameter (-dtp) for the service to TRUE (the default value is FALSE). You can use Oracle Enterprise Manager or SRVCTL to modify the DTP property of the singleton service. The following example shows how to modify the xa_01.example.com service using SRVCTL:

    ```
    $ srvctl modify service -db crm -service xa_01.example.com -dtp TRUE
    ```

Relocating DTP Services in Administrator-Managed Databases

If the instance that provides a distributed transaction processing (DTP) service, for example XA_01, fails, then the service fails over in the same manner as any other service. Because there is a requirement that sessions exist on exactly one instance, however, the -f (force kill) option for database sessions always applies.

If services migrate to other instances, then you might have to force the relocation of the service back to the preferred instance after it is restarted to evenly re-balance the load on all of the available hardware. You can use data from the GV $ACTIVE_SERVICES view to determine whether you need to relocate the DTP service.

> **Note:**
>
> Beginning with Oracle RAC 11*g* release 1 (11.1), global transactions and XA affinity replace the need for DTP services. Most XA deployments should be using global transactions with XA affinity for improved load balancing and flexibility rather than the DTP attribute.

Automatic Workload Repository

The **Automatic Workload Repository (AWR)** collects, processes, and maintains performance statistics for the database. The gathered data can be displayed in both

reports and views. If you use services with your database, then AWR tracks **metric**s at the service level.

Metrics can be measured against a variety of units, including time, transactions, or database calls. For example, the number of database calls per second is a metric. Server generated alerts can be placed on these metrics when they exceed or fail to meet user-specified thresholds. The database or system administrator can then respond, for example, by:

- Using the Oracle Database Resource Manager to configure the service level for one service to have priorities relative to other services

- Stopping overloaded processes

- Modifying a service level requirement

- Implementing recovery scenarios in response to service quality changes

Using AWR metrics and performance alerts enables you to maintain continued service availability despite service level changes. It also enables you to measure the quality of service provided by the database services.

The AWR ensures that the Oracle Clusterware workload management framework and the database resource manager have persistent and global representations of performance data. This information helps Oracle Database schedule job classes by service and to assign priorities to consumer groups. If necessary, you can rebalance workloads manually with either Oracle Enterprise Manager or SRVCTL. You can also disconnect a series of sessions, but leave the service running.

Note:

Oracle *does not* recommend using the DBMS_SERVICE package for use with services used by an Oracle RAC database. Use SRVCTL or Oracle Enterprise Manager to create database services for Oracle RAC.

See Also:

- *Oracle Database 2 Day + Performance Tuning Guide* for more information on creating alerts for metric thresholds

- *Oracle Database Performance Tuning Guide* for more information about the Automatic Workload Repository

- *Oracle Database PL/SQL Packages and Types Reference* for details about Oracle Database packages

---

Measuring Performance by Service Using the Automatic Workload Repository

Services add a new dimension for performance tuning. With services, workloads are visible and measurable, and therefore resource consumption and wait times are attributable by application. Tuning by using "service and SQL" replaces tuning by "session and SQL" in the majority of systems where all sessions are anonymous and shared.

The AWR maintains performance statistics that include information about response time, throughput, resource consumption, and wait events for all services and work that a database performs. Oracle Database also maintains metrics, statistics, wait events, wait classes, and SQL-level traces for services. You can optionally augment these statistics by defining modules within your application to monitor certain statistics. You can also define the actions within those modules that business critical transactions should execute in response to particular statistical values.

Enable module and action monitoring using the DBMS_MONITOR PL/SQL package. For example, for connections that use the erp service, the following command enables monitoring for the exceptions pay action in the payroll module:

```
EXECUTE DBMS_MONITOR.SERV_MOD_ACT_STAT_ENABLE(SERVICE_NAME => 'ERP',
MODULE_NAME=> 'PAYROLL', ACTION_NAME => 'EXCEPTIONS PAY');
```

For connections that use the erp service, the following command enables monitoring for all actions in the payroll module:

```
EXECUTE DBMS_MONITOR.SERV_MOD_ACT_STAT_ENABLE(SERVICE_NAME => 'ERP',
MODULE_NAME=> 'PAYROLL', ACTION_NAME => NULL);
```

Use the DBA_ENABLED_AGGREGATIONS view to verify that you have enabled monitoring for application modules and actions.

Statistics aggregation and tracing by service are global in scope for Oracle RAC databases. In addition, these statistic aggregations are persistent across instance restarts and service relocations for both Oracle RAC and noncluster Oracle databases.

The service, module, and action names are visible in V$SESSION, V$ACTIVE_SESSION_HISTORY, and V$SQL views. The call times and performance statistics are visible in V$SERVICE_STATS, V$SERVICE_EVENT, V$SERVICE_WAIT_CLASS, V$SERVICEMETRIC, and V$SERVICEMETRIC_HISTORY. When you enable statistics collection for an important transaction, you can see the call speed for each service, module, and action name at each database instance using the V$SERV_MOD_ACT_STATS view.

The following sample SQL*Plus script provides service quality statistics for a five second interval. You can use these service quality statistics to monitor the quality of a service, to direct work, and to balance services across Oracle RAC instances:

```
SET PAGESIZE 60 COLSEP '|' NUMWIDTH 8 LINESIZE 132 VERIFY OFF FEEDBACK OFF
COLUMN service_name FORMAT A20 TRUNCATED HEADING 'Service'
COLUMN begin_time HEADING 'Begin Time' FORMAT A10
COLUMN end_time HEADING 'End Time' FORMAT A10
COLUMN instance_name HEADING 'Instance' FORMAT A10
COLUMN service_time HEADING 'Service Time|mSec/Call' FORMAT 999999999
COLUMN throughput HEADING 'Calls/sec'FORMAT 99.99
BREAK ON service_name SKIP 1
SELECT
    service_name
  , TO_CHAR(begin_time, 'HH:MI:SS') begin_time
  , TO_CHAR(end_time, 'HH:MI:SS') end_time
  , instance_name
  , elapsedpercall  service_time
  ,  callspersec  throughput
FROM
    gv$instance i
  , gv$active_services s
  , gv$servicemetric m
WHERE s.inst_id = m.inst_id
  AND s.name_hash = m.service_name_hash
```

```
          AND i.inst_id = m.inst_id
          AND m.group_id = 10
      ORDER BY
            service_name
        , i.inst_id
        , begin_time ;
```

Automatic Workload Repository Service Thresholds and Alerts

Service level thresholds enable you to compare actual service levels against required levels of service. This provides accountability for the delivery or the failure to deliver an agreed service level. The end goal is a predictable system that achieves service levels. There is no requirement to perform as fast as possible with minimum resource consumption; the requirement is to meet the *quality* of service.

AWR enables you to explicitly specify two performance thresholds for each service: the response time for calls (ELAPSED_TIME_PER_CALL), and the CPU time for calls (CPU_TIME_PER_CALL). The response time threshold indicates that the elapsed time for each user call for each service should not exceed a certain value, and the CPU time for calls threshold indicates that the time spent using the CPU for each call for each service should not exceed a certain value. Response time is a fundamental measure that reflects all delays and faults that might be blocking the call from running on behalf of the user. Response time can also indicate differences in node power across the nodes of an Oracle RAC database.

You must set these thresholds on each instance of an Oracle RAC database. The elapsed time and CPU time are calculated as the moving average of the elapsed, server-side call time. The AWR monitors the elapsed time and CPU time and publishes AWR alerts when the performance exceeds the thresholds. You can schedule actions using Oracle Enterprise Manager jobs for these alerts, or you can schedule actions to occur programmatically when the alert is received. You can respond to these alerts by changing the priority of a job, stopping overloaded processes, or by relocating, starting or stopping a service. This permits you to maintain service availability despite changes in demand.

This section includes the following topics:

- Example of Services and Thresholds Alerts

- Enable Service_ Module_ and Action Monitoring

Example of Services and Thresholds Alerts

In this scenario, you need to check the thresholds for the payroll service. You can use the AWR report to get this information. You should compare the results from reports run over several successive intervals during which time the system is running optimally. For example, assume that for servers accessed by a payroll application, the AWR report runs each Thursday during the peak usage times of 1:00 p.m. to 5:00 p.m. The AWR report contains the response time, or elapsed database time, and the CPU consumption time, or CPU time, for calls for each server, including the payroll service. The AWR report also provides a breakdown of the work done and the wait times that are contributing to the response times.

Using DBMS_MONITOR, you set a warning threshold for the elapsed time per call for the payroll service at 0.5 seconds (500000 microseconds). You also set a critical threshold for the elapsed time per call for the payroll service at 0.75 seconds (750000 microseconds).

In this example, thresholds are added for the payroll service as follows:

```
EXECUTE DBMS_SERVER_ALERT.SET_THRESHOLD(
METRICS_ID => DBMS_SERVER_ALERT.ELAPSED_TIME_PER_CALL
, warning_operator => DBMS_SERVER_ALERT.OPERATOR_GE
, warning_value => '500000'
, critical_operator => DBMS_SERVER_ALERT.OPERATOR_GE
, critical_value => '750000'
, observation_period => 30
, consecutive_occurrences => 5
, instance_name => NULL
, object_type => DBMS_SERVER_ALERT.OBJECT_TYPE_SERVICE
, object_name => 'payroll');
```

You can verify the threshold configuration is set on all the instances using the following SELECT statement:

```
SELECT METRICS_NAME, INSTANCE_NAME, WARNING_VALUE, CRITICAL_VALUE,
OBSERVATION_PERIOD FROM dba_thresholds ;
```

Enable Service, Module, and Action Monitoring

You can enable performance data tracing for important modules and actions within each service. The performance statistics are available in the V$SERV_MOD_ACT_STATS view. For example, you might decide to set the following:

- For the ERP service, enable monitoring for the exceptions pay action in the payroll module.

- For the ERP service, enable monitoring for the all actions in the payroll module.

- For the HOT_BATCH service, enable monitoring for all actions in the posting module.

The following commands show how to enable the module and action monitoring for the services:

```
EXECUTE DBMS_MONITOR.SERV_MOD_ACT_STAT_ENABLE(service_name => 'erp', module_name=>
 'payroll', action_name => 'exceptions pay');
EXECUTE DBMS_MONITOR.SERV_MOD_ACT_STAT_ENABLE(service_name => 'erp', module_name=>
 'payroll');
EXECUTE DBMS_MONITOR.SERV_MOD_ACT_STAT_ENABLE(service_name => 'hot_batch',
module_name =>'posting');
```

To verify monitoring is enabled for the service, module, and actions, use the following SELECT statement:

```
COLUMN AGGREGATION_TYPE FORMAT A21 TRUNCATED HEADING 'AGGREGATION'
COLUMN PRIMARY_ID FORMAT A20 TRUNCATED HEADING 'SERVICE'
COLUMN QUALIFIER_ID1 FORMAT A20 TRUNCATED HEADING 'MODULE'
COLUMN QUALIFIER_ID2 FORMAT A20 TRUNCATED HEADING 'ACTION'
SELECT * FROM DBA_ENABLED_AGGREGATIONS ;
```

The output is similar to the following:

```
AGGREGATION            SERVICE             MODULE     ACTION
-----------            -------------------  ----------  -------------
SERVICE_MODULE_ACTION  erp                 payroll    exceptions pay
SERVICE_MODULE         erp                 payroll
SERVICE_MODULE         hot_batch           posting
```

Administering Services

You can create and administer services with Oracle Enterprise Manager and the SRVCTL utility. The following sections describe how to perform service-related tasks using these tools:

This section includes the following topics:

- Overview of Service Administration

- Administering Services with Oracle Enterprise Manager

- Administering Services with SRVCTL

Note:

You can also use the DBMS_SERVICE package to create or modify services and service attributes, but SRVCTL will override any settings made using this package. The DBMS_SERVICE package is not recommended for use with services used by an Oracle RAC database, nor when Oracle Restart is used, nor when Oracle Clusterware is managing a single-instance database.

Overview of Service Administration

When you create and administer services, you are dividing the work that your database performs into manageable units. The goal of using services is to achieve optimal utilization of your database infrastructure. You can create and deploy services based on business requirements. Oracle Database can measure the performance for each service. Using the DBMS_MONITOR package, you can define both the application modules within a service and the individual actions for a module and monitor thresholds for these actions, enabling you to manage workloads to deliver capacity on demand.

When you create new services for your database, you should define the automatic workload management characteristics for each service, as described in "Service Characteristics".

See Also:

- *Oracle Database Quality of Service Management User's Guide* if you are using Oracle Database QoS Management with your Oracle cluster for details on how to configure the database services

- "Enabling Clients for Oracle RAC" for more details

In addition to creating services, you can:

- **Delete a service**. You can delete services that you created. However, you cannot delete or modify the properties of the default database service that Oracle Database created.

- **Check the status of a service**. A service can be assigned different roles among the available instances. In a complex database with many services, you may not

remember the details of every service. Therefore, you may have to check the status on an instance or service basis. For example, you may have to know the status of a service for a particular instance before you make modifications to that instance or to the Oracle home from which it runs.

- **Start or stop a service for a database or an instance**. A service must be started before it can be used for client connections to that instance. If you shut down your database, for example, by running the SRVCTL command `srvctl stop database -d db_unique_name` where *db_unique_name* is the name of the database you want to stop, then Oracle Database stops all services for that database. Depending on the service management policy, you may have to manually restart the services when you start the database. Both the `srvctl stop database` and `srvctl stop service` commands accept the `-force` option to forcibly disconnect connections. To drain sessions for planned outages do not use the `-force` option.

  > **Note:**
  >
  > If Oracle Database QoS Management is enabled for the Oracle RAC database, then the services are automatically restarted after they are stopped.

- **Map a service to a consumer group**. You can map services to Resource Manager Consumer groups to limit the amount of resources that services can use in an instance. You must create the consumer group and then map the service to the consumer group.

  > **See Also:**
  >
  > *Oracle Database PL/SQL Packages and Types Reference* for information about the `DBMS_RESOURCE_MANAGER.SET_CONSUMER_GROUP_MAPPING_PRI` procedure

- **Enable or disable a service for a database or an instance**. By default, Oracle Clusterware attempts to restart a service automatically after failures. You can prevent this behavior by disabling a service. Disabling a service is useful when you must perform database or instance maintenance, such as when you are performing an upgrade and you want to prevent connection requests from succeeding.

- **Relocate a service to a different instance**. You can move a service from one instance to another instance to re-balance workloads, for example, after adding or deleting cluster nodes.

> **Note:**
>
> - When you use services, do not set a value for the SERVICE_NAMES parameter; Oracle Database controls the setting for this parameter for the services that you create and for the default database service. The service features that this chapter describes are not directly related to the features that Oracle Database provides when you set SERVICE_NAMES. In addition, setting a value for this parameter may override some benefits of using services.
>
> - Service status information must be obtained from SRVCTL or from the service-related database views, such as dba_services.
>
> - If you specify a service using the DISPATCHERS initialization parameter, it overrides any service in the SERVICE_NAMES parameter, and cannot be managed. (For example, stopping the service with a SRVCTL command does not stop users connecting with the service.)

Administering Services with Oracle Enterprise Manager

The Cluster Managed Database Services page is the master page for beginning all tasks related to services. Access this page, as follows:

1. In Oracle Enterprise Manager, go to the Cluster Database Home page.

   > **See Also:**
   >
   > *Oracle Database 2 Day DBA* for details on logging in to Oracle Enterprise Manager.

2. From the **Availability** menu, select **Cluster Managed Database Services** to display the Cluster Managed Database Services page.

3. Enter or confirm the credentials for the Oracle RAC database and host operating system and click **Continue** to display the Cluster Managed Database Services page.

From the Cluster Managed Database Services page you can drill down to perform the following tasks:

- View a list of services for the cluster

- View the instances on which each service is currently running

- View the server pool and nodes offering the service in a policy-managed environment

- View the status for each service

- Create or edit a service

- Start or stop a service

- Enable or disable a service

- Perform instance-level tasks for a service

- Delete a service

Note:

You must have SYSDBA credentials to access a cluster database. Cluster Managed Database Services does not permit you to connect as anything other than SYSDBA.

See Also:

- Oracle Enterprise Manager online help for more information about administering services with Oracle Enterprise Manager

- *Oracle Database 2 Day + Real Application Clusters Guide* for more information about using Oracle Enterprise Manager to manage services

---

Administering Services with SRVCTL

When you create a service by using SRVCTL, you must start the service with a separate SRVCTL command. However, you may later have to manually stop or restart the service. You may also have to disable the service to prevent automatic restarts, to manually relocate the service, or obtain status information about the service. The following sections explain how to use SRVCTL to perform the following administrative tasks:

- Creating Services with SRVCTL

- Creating Services for Application Continuity and Transaction Guard

- Starting and Stopping Services with SRVCTL

- Enabling and Disabling Services with SRVCTL

- Relocating Services with SRVCTL

- Obtaining the Status of Services with SRVCTL

- Obtaining the Configuration of Services with SRVCTL

See Also:

Server Control Utility Reference for more information about SRVCTL commands that you can use to manage services, including descriptions of options

---

Creating Services with SRVCTL

To create a service with SRVCTL, use the srvctl add service command on the command line.

> **See Also:**
>
> "srvctl add service" for complete usage information for this command

Creating Services for Application Continuity and Transaction Guard

When using Application Continuity and Transaction Guard with your applications, you must configure a service. This section describes how to configure these application services depending on the functionality you plan to implement.

Creating Services for Application Continuity

To configure services for Application Continuity, when you create a service using SRVCTL, set the `-failovertype` parameter to `TRANSACTION` and `-commit_outcome` to `TRUE`. Additionally, you can set values for these other service parameters for Application Continuity and load balancing:

- `-replay_init_time`: Specifies how long, in seconds, you allow replay to start. Oracle recommends that you choose a value based on how long you will allow replay to be initiated. The default value is 300 seconds.

- `-retention`: Specifies the time (in seconds) that the commit outcome information is stored in the database. The default value is 86400 (1 day).

- `-session_state`: After a COMMIT has executed, if the state was changed in that transaction, then it is not possible to replay the transaction to reestablish that state if the session is lost. When configuring Application Continuity, the applications are categorized depending on whether the session state after the initial setup is dynamic or static, and then whether it is correct to continue past a COMMIT operation within a request.

  - **Dynamic**: (default) A session has a dynamic state if the session state changes are not fully encapsulated by the initialization, and cannot be fully captured in a callback at failover. Once the first transaction in a request commits, failover is internally disabled until the next request begins. This is the default mode that almost *all* applications should use for requests.

  - **Static**: (special—on request) A session has a static state if all session state changes, such as NLS settings and PL/SQL package state, can be repeated in an initialization callback. This setting is used only for database diagnostic applications that do not change session state. Do not specify `STATIC` if there are any non-transactional state changes in the request that cannot be reestablished by a callback. If you are unsure what state to specify, use `DYNAMIC`.

- `-failoverretry`: Number of connection retries for each connection attempt; recommended value is 30.

- `-failoverdelay`: Delay in seconds between each connection attempt; recommended value is 10.

- `-notification`: FAN is highly recommended—set this value to `TRUE` to enable FAN for OCI and ODP.Net clients.

- `-clbgoal`: For connection load balancing, use `SHORT` when using run-time load balancing.

- `-rlbgoal`: For run-time load balancing, set to `SERVICE_TIME`.

To create a service for Application Continuity for a policy-managed Oracle RAC database, use a command similar to the following, where `racdb` is the name of your Oracle RAC database, `app2` is the name of the service you are modifying, and `Svrpool1` is the name of the server pool in which the service is offered:

```
$ srvctl add service -db racdb -service app2 -serverpool Svrpool1
    -failovertype TRANSACTION -commit_outcome TRUE -replay_init_time 1800
    -retention 86400 -notification TRUE -rlbgoal SERVICE_TIME -clbgoal SHORT
    -failoverretry 30 -failoverdelay 10
```

You can use SRVCTL to modify an existing service for Application Continuity, similar to the following command, where `racdb` is the name of your Oracle RAC database, and `app1` is the name of the service you are modifying:

```
$ srvctl modify service -db racdb -service app1 -clbgoal SHORT
    -rlbgoal SERVICE_TIME -failoverretry 30 -failoverdelay 10
    -failovertype TRANSACTION -commit_outcome TRUE -replay_init_time 1800
    -retention 86400 -notification TRUE
```

Creating Services for Transaction Guard

To enable Transaction Guard, but not Application Continuity, create the service using SRVCTL and set only `-commit_outcome TRUE`.

You can use SRVCTL to modify an existing service to enable Transaction Guard, similar to the following command, where `racdb` is the name of your Oracle RAC database, and `app2` is the name of the service you are modifying:

```
$ srvctl modify service -db racdb -service app2 -commit_outcome TRUE
    -retention 86400 -notification TRUE
```

In the preceding example, the `-retention` parameter specifies how long, in seconds, to maintain the history. Additionally the `-notification` parameter is set to `TRUE`, enabling FAN events.

To use Transaction Guard, a DBA must grant permission, as follows:

```
GRANT EXECUTE ON DBMS_APP_CONT;
```

> **See Also:**
>
> *Oracle Database Development Guide* for more information about using Transaction Guard with applications

Starting and Stopping Services with SRVCTL

For applications to connect using a server, the service must be started. If you stop a service, then it is temporarily unavailable, but is still subject to automatic restart and failover.

Enter the following SRVCTL syntax at the command line to start or stop a service:

```
$ srvctl start service -db db_unique_name [-service service_name_list]
    [-instance inst_name] [-startoption start_options]
```

```
$ srvctl stop service -db db_unique_name -service service_name_list
    [-instance inst_name] [-startoption start_options]
```

Enabling and Disabling Services with SRVCTL

If you disable a service, then Oracle Clusterware does not consider the service for automatic startup, failover, or restart. You might disable a service when performing application maintenance, to ensure the service is not accidentally restarted by Oracle Clusterware until your maintenance operations are complete. To make the service available for normal operation again, you enable the service.

Use the following SRVCTL syntax from the command line to enable and disable services:

```
$ srvctl enable service -db db_unique_name -service service_name_list
    [-instance inst_name]
```

```
$ srvctl disable service -db db_unique_name -service service_name_list
    [-instance inst_name]
```

Relocating Services with SRVCTL

Run the `srvctl relocate service` command from the command line to relocate a service. You might use this command when a service has failed over to an available instance, but you want to move it back to the preferred instance after that instance is restarted.

The following command relocates the `crm` service from instance `apps1` to instance `apps3`:

```
$ srvctl relocate service -db apps -service crm -oldinst apps1 -newinst apps3
```

The following command relocates the `crm` service from `node1` to `node3` using node syntax:

```
$ srvctl relocate service -db apps -service crm -currentnode node1
    -targetnode node3
```

Obtaining the Status of Services with SRVCTL

Run the `srvctl status service` command from the command line to obtain the status of a service. For example, the following command returns the status of the services that are running on the `apps` database:

```
$ srvctl status service -db apps

Service erp is running on nodes: apps02,apps03
Service hr is running on nodes: apps02,apps03
Service sales is running on nodes: apps01,apps04
```

Obtaining the Configuration of Services with SRVCTL

Run the `srvctl config service` command from the command line to obtain the high availability configuration of a service. For example, the following command returns the configuration of the `erp` service that is running on the `apps` database:

```
$ srvctl config service -db apps -service erp

Service name: erp
Service is enabled
Server pool: pool1
Cardinality: 1
Disconnect: false
Service role: PRIMARY
```

```
Management policy: AUTOMATIC
DTP transaction: false
AQ HA notifications: true
Global: false
Commit Outcome: true
Failover type: TRANSACTION
Failover method: NONE
TAF failover retries: 30
TAF failover delay: 10
Connection Load Balancing Goal: LONG
Runtime Load Balancing Goal: SERVICE_TIME
TAF policy specification: NONE
Edition:
Pluggable database name:
Maximum lag time: ANY
SQL Translation Profile:
Retention: 86400 seconds
Replay Initiation Time: 1800 seconds
Session State Consistency: STATIC
Preferred instances: apps
Available instances:
```

See Also:

Server Control Utility Reference for information about other administrative tasks that you can perform with SRVCTL

Global Services

Oracle RAC supports database services and enables service-level workload management across instances in a single cluster. Global services provide dynamic load balancing, failover, and centralized service management for a set of replicated databases that offer common services. The set of databases may include Oracle RAC and non-clustered Oracle databases interconnected by Oracle Data Guard, Oracle GoldenGate, or any other replication technology.

When you create and use global services, the following workload management features are available:

- Ability to specify preferred and available databases for a global service

- Handling of replication lag

- Geographical affinity between clients and servers

- Connection load balancing

- Run-time load balancing

- Inter-database service failover

- Fast connection failover

- Connect-time failover

- Application Continuity

- Transaction Guard

- Backward compatibility with existing clients

> **Note:**
>
> You can manage instance placement of a global service within an Oracle RAC database with SRVCTL but you can only manage other global service attributes with GDSCTL.

> **See Also:**
>
> *Oracle Database Global Data Services Concepts and Administration Guide* for more information about configuring and using global services

Connecting to a Service: An Example

The following example illustrates how to create a service and then gives several examples of connecting to that service using different client methods.

In this example the service is enabled for run-time load balancing, as follows:

- Service Name: `HR.example.com`

  - Running on database named CRM

  - The system consists of 4 nodes

- Specifying `SERVICE_TIME` as the value for the `-rlbgoal` parameter

- SCAN address of the listener is `rws3010104-scan.example.com`

- Listener port is 1585

The service has a cardinality of two, but if needed, can be offered by any of the CRM database instances. The service configuration is as follows:

- Preferred Instances: CRM1, CRM2

- Available Instances: CRM3, CRM4

- Specifying `SHORT` as the value for the `-clbgoal` parameter

The application using this service takes advantage of Application Continuity, so you must set `-failovertype` and `-commit_outcome`. Use the default retention parameters, but set a 10 second delay between connection attempts, and up to 40 retries before failing to get a connection.

Creating the HR Service Using SRVCTL

Create the HR service using SRVCTL, as follows:

```
$ srvctl add service -db CRM -service HR.example.com -preferred CRM1,CRM2
    -available CRM3,CRM4 -clbgoal SHORT -failovertype TRANSACTION
    -commit_outcome TRUE -failoverdelay 10 -failoverretry 40
```

Start the `HR.example.com` service, as follows:

```
$ srvctl start service -db CRM -service HR.example.com
```

The service is now be available on up to two instances, and CRM1 and CRM2 are the preferred instances.

Connecting to the HR Service from a JDBC Application

The application that connects to the HR service, in this example, is a JDBC application using the JDBC Universal Connection Pool with the JDBC thin driver.

In this example, a URL is constructed specifying the thin-style service name format for the database specifier. Fast Connection Failover is enabled, and remote Oracle Notification Service is configured, where the Oracle Notification Service daemon on the cluster listens on port 6200.

```
//import packages and register the driver
import java.sql.Connection;
import java.sql.SQLException;
import java.sql.Statement;
import oracle.ucp.jdbc.PoolDataSourceFactory;
import oracle.ucp.jdbc.PoolDataSource;

PoolDataSource  pds = PoolDataSourceFactory.getPoolDataSource();

//set the connection properties on the data source.
pds.setConnectionPoolName("FCFPool");
pds.setFastConnectionFailoverEnabled(true);
pds.setONSConfiguration("nodes=rws3010104-scan.example.com:6200");
pds.setConnectionFactoryClassName("oracle.jdbc.pool.OracleDataSource");
pds.setURL("jdbc:oracle:thin:@//rws3010104-scan.example.com:1585/HR.example.com");
pds.setUser("HR");
pds.setPassword("hr");


//Override any pool properties.
pds.setInitialPoolSize(5);

//Get a database connection from the datasource.

Connection conn = pds.getConnection();

// do some work

//return connection to pool
conn.close();
conn=null
```

> **See Also:**
>
> *Oracle Universal Connection Pool for JDBC Developer's Guide* for more information about using Universal Connection Pool for JDBC

6

Configuring Recovery Manager and Archiving

This chapter explains how to configure Recovery Manager (RMAN) for use in Oracle Real Application Clusters (Oracle RAC) environments. This chapter also provides procedures for using for archiving in Oracle RAC environments and discusses online redo log and archived redo log considerations.

The topics in this chapter include:

Overview of Configuring RMAN for Oracle RAC

RMAN enables you to back up, restore, and recover data files, control files, server parameter files (SPFILEs) and archived redo log files. RMAN is included with Oracle Database and does not require separate installation. You can run RMAN from the command line or use RMAN in the Backup Manager in Oracle Enterprise Manager.

Archiving Mode in Oracle RAC

In order for redo log files to be archived, the Oracle RAC database must be in `ARCHIVELOG` mode. You can run the `ALTER DATABASE` SQL statement to change the archiving mode in Oracle RAC, because the database is mounted by the local instance but not open in any instances. You do not need to modify parameter settings to run this statement.

> **Note:**
>
> - The `ARCHIVELOG` mode is set at the database level, not the instance level. Either all instances archive or none do.
>
> - You can also change the archive log mode by using the Recovery Settings page in the **Maintenance** tab of the Oracle Enterprise Manager Oracle RAC Database Home Page.

> **See Also:**
>
> *Oracle Database Administrator's Guide* for more information about setting the archiving mode

Configuring the RMAN Snapshot Control File Location

The snapshot control file is a copy of a database control file created in an operating system-specific location by RMAN. RMAN creates the snapshot control file so that it has a consistent version of a control file to use when either resynchronizing the recovery catalog or backing up the control file.

You can specify a **cluster file system** or a raw device destination for the location of your snapshot control file. This file is shared across all nodes in the **cluster** and *must* be accessible by all nodes in the cluster. Run the following RMAN command to determine the configured location of the snapshot control file:

```
SHOW SNAPSHOT CONTROLFILE NAME;
```

You can change the configured location of the snapshot control file. For example, on Linux and UNIX systems you can specify the snapshot control file location as `$ORACLE_HOME/dbs/scf/snap_prod.cf` by entering the following at the RMAN prompt:

```
CONFIGURE SNAPSHOT CONTROLFILE NAME TO '$ORACLE_HOME/dbs/scf/snap_prod.cf';
```

This command sets the configuration for the location of the snapshot control file for every instance of your **cluster database**. Therefore, ensure that the `$ORACLE_HOME/dbs/scf` directory is shared by all nodes that perform backups.

The `CONFIGURE` command creates persistent settings across RMAN sessions. Therefore, you do not need to run this command again unless you want to change the location of the snapshot control file.

To delete a snapshot control file you must first change the snapshot control file location, then delete the file at the older location, as follows:

```
CONFIGURE SNAPSHOT CONTROLFILE NAME TO 'new_name';
DELETE COPY OF CONTROLFILE;
```

> **See Also:**
>
> *Oracle Database Backup and Recovery Reference* for more information about configuring the snapshot control file

Configuring RMAN to Automatically Backup the Control File and SPFILE

If you set `CONFIGURE CONTROLFILE AUTOBACKUP` to `ON`, then RMAN automatically creates a control file and an SPFILE backup after you run the `BACKUP` or `COPY` commands. RMAN can also automatically restore an SPFILE, if this is required to start an instance to perform recovery, because the default location for the SPFILE must be available to all nodes in your Oracle RAC database.

> **Note:**
>
> If you back up the control file using the SQL*Plus `ALTER DATABASE` command, then you must also create the control file backup on a device shared by all nodes.

These features are important in disaster recovery because RMAN can restore the control file even without a recovery catalog. RMAN can restore an autobackup of the control file even after the loss of both the recovery catalog and the current control file. You can change the default name that RMAN gives to this file with the `CONFIGURE CONTROLFILE AUTOBACKUP FORMAT` command. Note that if you specify an absolute path name in this command, then this path must exist identically on all nodes that participate in backups.

RMAN performs the control file autobackup on the first allocated channel. Therefore, when you allocate multiple channels with different parameters, especially when you allocate a channel with the `CONNECT` command, determine which channel will perform the control file autobackup. Always allocate the channel for this node first.

Besides using the RMAN control file, you can also use Oracle Enterprise Manager to use the RMAN features.

> **See Also:**
>
> *Oracle Database Backup and Recovery User's Guide* for more information about using the control file autobackup feature

Crosschecking on Multiple Oracle RAC Nodes

When crosschecking on multiple nodes (and when operating RMAN in general), configure the cluster so that all backups can be accessed by every node, regardless of which node created the backup. When the cluster is configured this way, you can allocate channels to any node in the cluster during restore or crosscheck operations.

If you cannot configure the cluster so that each node can access all backups, then during restore and crosscheck operations, you must allocate channels on multiple nodes by providing the `CONNECT` option to the `CONFIGURE CHANNEL` command, so that every backup can be accessed by at least one node. If some backups are not accessible during crosscheck because no channel was configured on the node that can access those backups, then those backups are marked `EXPIRED` in the RMAN repository after the crosscheck.

For example, you can use `CONFIGURE CHANNEL ... CONNECT` in an Oracle RAC configuration in which tape backups are created on various nodes in the cluster and

each backup is only accessible on the node on which it is created. This is described in more detail in "Configuring Channels to Use a Specific Node".

See Also:

Oracle Database Backup and Recovery User's Guide for more information about crosschecking

---

Configuring Channels for RMAN in Oracle RAC

This section describes how to configure channels for RMAN. You can configure channels to use automatic load balancing or you can specify specific channels for specific instances as described in the following topics:

- Configuring Channels to Use Automatic Load Balancing

- Configuring Channels to Use a Specific Node

Configuring Channels to Use Automatic Load Balancing

To configure channels to use automatic load balancing, use the following syntax:

```
CONFIGURE DEVICE TYPE [disk | sbt] PARALLELISM number_of_channels;
...
```

Where `number_of_channels` is the number of channels that you want to use for the operation. After you complete this one-time configuration, you can issue `BACKUP` or `RESTORE` commands.

Configuring Channels to Use a Specific Node

To configure one RMAN channel for each policy-managed Oracle RAC database instance, use the following syntax:

```
CONFIGURE CHANNEL DEVICE TYPE sbt CONNECT '@racinst_1'
CONFIGURE CHANNEL DEVICE TYPE sbt CONNECT '@racinst_2'
...
```

After this one-time configuration step, you can issue the `BACKUP` or `RESTORE` commands.

Managing Archived Redo Logs Using RMAN in Oracle RAC

When a node generates an archived redo log, Oracle Database always records the file name of the log in the control file of the target database. If you are using a recovery catalog, then RMAN also records the archived redo log filenames in the recovery catalog when a resynchronization occurs.

The archived redo log naming scheme that you use is important because when a node writes to a log with a specific file name on its file system, the file must be readable by any node that must access this archived redo log. For example, if `node1` archives a log to `/oracle/arc_dest/log_1_100_23452345.arc`, then `node2` can back up this archived redo log only if it can read `/oracle/arc_dest/log_1_100_23452345.arc` on its own file system.

The backup and recovery strategy that you choose depends on how you configure the archiving destinations for each node. Whether only one node or all nodes perform

archived redo log backups, you must ensure that all archived redo logs are backed up. If you use RMAN parallelism during recovery, then the node that performs recovery must have read access to all archived redo logs in your cluster.

Multiple nodes can restore archived logs in parallel. However, during recovery, only one node applies the archived logs. Therefore, the node that is performing the recovery must be able to access all of the archived logs that are needed for the recovery operation. By default, the database determines the optimum number of parallel threads to use during the recovery operation. You can use the PARALLEL clause in the RECOVER command to change the number of parallel threads.

Guidelines and Considerations for Archived Redo Logs

The primary consideration is to ensure that all archived redo logs can be read from every node during recovery, and, if possible, during backups. During recovery, if the archived log destinations are visible from the node that performs the recovery, then Oracle Database can successfully recover the archived log data.

Archived Redo Log File Conventions in Oracle RAC

For any archived redo log configuration, uniquely identify the archived redo logs with the LOG_ARCHIVE_FORMAT parameter. The format of this parameter is operating system-specific and it can include text strings, one or more variables, and a file name extension.

Table 6-1   Archived Redo Log File Name Format Parameters

Parameter	Description	Example
%r	Resetlogs identifier, not padded	log_1_62_23452345
%R	Resetlogs identifier, left-zero-padded	log_1_62_0023452345
%s	Log sequence number, not padded	log_251
%S	Log sequence number, left-zero-padded	log_0000000251
%t	Thread number, not padded	log_1
%T	Thread number, left-zero-padded	log_0001

All of the file name format parameters for the archive redo logs, in either upper or lowercase, are mandatory for Oracle RAC. These parameters enable Oracle Database to create unique names for archive logs across the incarnation. This requirement is in effect when the COMPATIBLE parameter is set to 10.0 or greater.

Use the %R or %r parameters to include the resetlogs identifier to avoid overwriting the logs from a previous incarnation. If you do not specify a log format, then the default is operating system-specific and includes %t, %s, and %r.

As an example, if the instance associated with redo thread number 1 sets LOG_ARCHIVE_FORMAT to log_%t_%s_%r.arc, then its archived redo log files are named:

```
log_1_1000_23435343.arc
log_1_1001_23452345.arc
log_1_1002_23452345.arc
...
```

> **See Also:**
>
> *Oracle Database Administrator's Guide* about specifying the archived redo log file name format and destination, and Oracle Database platform-specific documentation about the default log archiving format and destination

RMAN Archiving Configuration Scenarios

This section describes the archiving scenarios for an Oracle RAC database. The two configuration scenarios in this chapter describe a three-node UNIX cluster for an Oracle RAC database. For both scenarios, the LOG_ARCHIVE_FORMAT that you specify for the instance performing recovery must be the same as the format that you specified for the instances that archived the redo log files.

This section includes the following topics:

- Oracle Automatic Storage Management and Cluster File System Archiving Scheme

- Noncluster File System Local Archiving Scheme

Oracle Automatic Storage Management and Cluster File System Archiving Scheme

The preferred configuration for Oracle RAC is to use Oracle Automatic Storage Management (Oracle ASM) for a recovery area using a disk group for your recovery set that is different from the disk group used for your data files. When you use Oracle ASM, it uses an Oracle Managed Files naming format.

Alternatively, you can use a cluster file system archiving scheme. If you use a cluster file system, then each node writes to a single location on the cluster file system when archiving the redo log files. Each node can read the archived redo log files of the other nodes. For example, as shown in Figure 6-1, if Node 1 archives a redo log file to /arc_dest/log_1_100_23452345.arc on the cluster file system, then any other node in the cluster can also read this file.

> **See Also:**
>
> *Oracle Database Backup and Recovery User's Guide* for more information about managing space in the recovery area and backing up archived redo logs with RMAN

Figure 6-1   Cluster File System Archiving Scheme
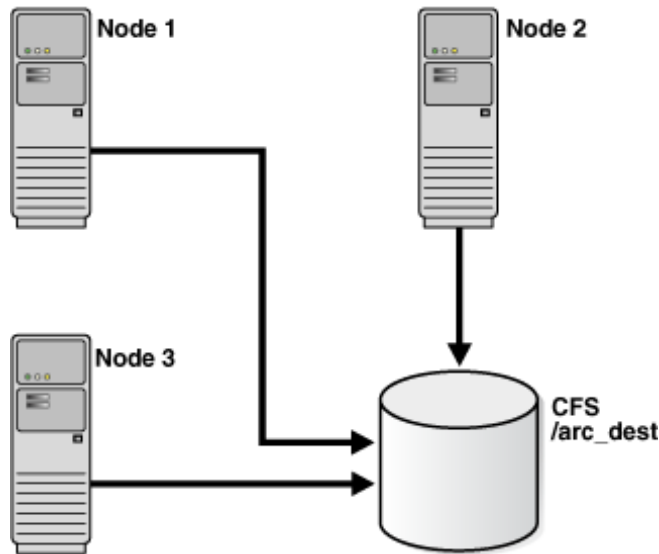
> **Note:**
>
> The archive log naming format in this example is only for a cluster file system example.

If you do not use a cluster file system, then the archived redo log files cannot be on raw devices. This is because raw devices do not enable sequential writing of consecutive archive log files.

Advantages of the Cluster File System Archiving Scheme

The advantage of this scheme is that none of the nodes uses the network to archive logs. Because the file name written by a node can be read by any node in the cluster, RMAN can back up all logs from any node in the cluster. Backup and restore scripts are simplified because each node has access to all archived redo logs.

Initialization Parameter Settings for the Cluster File System Archiving Scheme

In the cluster file system scheme, each node archives to a directory that is identified with the same name on all instances within the cluster database (`/arc_dest`, in the following example). To configure this directory, set values for the `LOG_ARCH_DEST_1` parameter, as shown in the following example:

```
*.LOG_ARCHIVE_DEST_1="LOCATION=/arc_dest"
```

The following list shows archived redo log entry examples that would appear in the RMAN catalog or in the control file based on the previous example. Note that any node can archive logs using any of the threads:

```
/arc_dest/log_1_999_23452345.arc
/arc_dest/log_1_1000_23435343.arc
/arc_dest/log_1_1001_23452345.arc <- thread 1 archived in node3
/arc_dest/log_3_1563_23452345.arc <- thread 3 archived in node2
/arc_dest/log_2_753_23452345.arc <- thread 2 archived in node1
/arc_dest/log_2_754_23452345.arc
/arc_dest/log_3_1564_23452345.arc
```

Location of Archived Logs for the Cluster File System Archiving Scheme

Because the file system is shared and because each node is writing its archived redo logs to the `/arc_dest` directory in the cluster file system, each node can read the logs written by itself and any other node.

Noncluster File System Local Archiving Scheme

When archiving locally to a noncluster file system, each node archives to a uniquely named local directory. If recovery is required, then you can configure the recovery node so that it can access directories on the other nodes remotely. For example, use NFS on Linux and UNIX computers, or mapped drives on Windows systems. Therefore, each node writes only to a local destination, but each node can also read archived redo log files in remote directories on the other nodes.

Considerations for Using Noncluster File System Local Archiving

If you use noncluster file system local archiving for media recovery, then you must configure the node that is performing recovery for remote access to the other nodes so that it can read the archived redo log files in the archive directories on the other nodes. In addition, if you are performing recovery and you do not have all of the available archive logs, then you must perform an incomplete recovery up to the first missing archived redo log sequence number. You do not have to use a specific configuration for this scheme. However, to distribute the backup processing onto multiple nodes, the easiest method is to configure channels as described in the backup scenarios in Managing Backup and Recovery.

> **Note:**
>
> Because different file systems are used in a noncluster case, the archive log directories must be unique on each node. For example, `/arc_dest_1` is only available on `node1`, `/arc_dest_2` is only directly mounted on `node2`, and so on.
>
> Then `node1` mounts `/arc_dest_2` from `node2` and `/arc_dest_3` from `node3` through NFS.

Initialization Parameter Settings for Noncluster File System Local Archiving

You can set the archiving destination values as follows in the initialization parameter file for either policy-managed or administrator-managed databases.

Set the `SID.LOG_ARCH_DEST` parameter for each instance using the SID designator, as shown in the following example:

```
sid1.LOG_ARCHIVE_DEST_1="LOCATION=/arc_dest_1"
sid2.LOG_ARCHIVE_DEST_1="LOCATION=/arc_dest_2"
sid3.LOG_ARCHIVE_DEST_1="LOCATION=/arc_dest_3"
```

For policy-managed databases, manually create a node and instance binding to ensure that `sid1` always runs on the same node, as follows:

```
$ srvctl modify database -d mydb -n node1 -i sid1
$ srvctl modify database -d mydb -n node2 -i sid2
$ srvctl modify database -d mydb -n node3 -i sid3
```

The following list shows the possible archived redo log entries in the database control file. Note that any node can read archived redo logs from any of the threads, which must happen in order for the database to recover after a failure.

```
/arc_dest_1/log_1_1000_23435343.arc
/arc_dest_2/log_1_1001_23452345.arc <- thread 1 archived in node2
/arc_dest_2/log_3_1563_23452345.arc <- thread 3 archived in node2
/arc_dest_1/log_2_753_23452345.arc <- thread 2 archived in node1
/arc_dest_2/log_2_754_23452345.arc
/arc_dest_3/log_3_1564_23452345.arc
```

Location of Archived Logs for Noncluster File System Local Archiving

As illustrated in Table 6-2, each of three nodes has a directory containing the locally archived redo logs. Additionally, if you mount directories on the other nodes remotely through NFS or mapped drives, then each node has two remote directories through which RMAN can read the archived redo log files that are archived by the remaining nodes.

Note:

The archive log destinations, similar to those shown in Table 6-2, must be different on each node so that if you mount the NFS directory on a different node, then it does not conflict with an existing archive log directory

---

Table 6-2   UNIX/NFS Location Log Examples, Noncluster File System Local Archiving

Node	Reads the archived redo log files in the directory	For logs archived by node
1	/arc_dest_1	1
1	/arc_dest_2	2 (through NFS)
1	/arc_dest_3	3 (through NFS)
2	/arc_dest_1	1 (through NFS)
2	/arc_dest_2	2
2	/arc_dest_3	3 (through NFS)
3	/arc_dest_1	1 (through NFS)
3	/arc_dest_2	2 (through NFS)
3	/arc_dest_3	3

File System Configuration for Noncluster File System Local Archiving

If you are performing recovery and a surviving instance must read all of the logs that are on disk but not yet backed up, then you should configure NFS as shown in Table 6-3.

Table 6-3   UNIX/NFS Configuration for Shared Read Local Archiving Examples

Node	Directory...	Is configured...	And mounted on...	On node...
1	/arc_dest_1	Local read/write	n/a	n/a
1	/arc_dest_2	NFS read	/arc_dest_2	2
1	/arc_dest_3	NFS read	/arc_dest_3	3
2	/arc_dest_1	NFS read	/arc_dest_1	1
2	/arc_dest_2	Local read/write	n/a	n/a
2	/arc_dest_3	NFS read	/arc_dest_3	3
3	/arc_dest_1	NFS read	/arc_dest_1	1
3	/arc_dest_2	NFS read	/arc_dest_2	2
3	/arc_dest_3	Local read/write	n/a	n/a

Note:

Windows users can achieve the same results depicted in the examples in this section by using mapped drives.

Monitoring the Archiver Processes

After your RMAN configuration is operative in your Oracle RAC environment, use the GV$ARCHIVE_PROCESSES and V$ARCHIVE_PROCESSES views to determine the status of the archiver processes. Depending on whether you query the global or local views, these views display information for all database instances, or for only the instance to which you are connected.

Note:

If you use the kill command to stop the archiver process, then the database instance will fail.

See Also:

- *Oracle Database Administrator's Guide* for more information about the archiver processes

- *Oracle Database Reference* for more information about the database views

7

Managing Backup and Recovery

This chapter explains instance recovery and how to use Recovery Manager (RMAN) to back up and restore Oracle Real Application Clusters (Oracle RAC) databases. This chapter also describes Oracle RAC instance recovery, parallel backup, recovery with SQL*Plus, and using the Fast Recovery Area in Oracle RAC. The topics in this chapter include:

- RMAN Backup Scenario for Noncluster File System Backups

- RMAN Restore Scenarios for Oracle RAC

- Instance Recovery in Oracle RAC

- Media Recovery in Oracle RAC

- Parallel Recovery in Oracle RAC

- Using a Fast Recovery Area in Oracle RAC

Note:

For restore and recovery in Oracle RAC environments, you do not have to configure the instance that performs the recovery to also be the sole instance that restores all of the data files. In Oracle RAC, data files are accessible from every node in the **cluster**, so any node can restore archived redo log files.

See Also:

Oracle Clusterware Administration and Deployment Guide for information about backing up and restoring the Oracle Clusterware components such as the Oracle Cluster Registry (OCR) and the voting disk

---

RMAN Backup Scenario for Noncluster File System Backups

In a non**cluster file system** environment, each node can back up only to a locally mounted noncluster file system directory. For example, `node1` cannot access the archived redo log files on `node2` or `node3` unless you configure the network file system for remote access. If you configure a network file system file for backups, then each node backs up its archived redo logs to a local directory.

RMAN Restore Scenarios for Oracle RAC

This section describes the following common RMAN restore scenarios:

- Restoring Backups from a Cluster File System

- Restoring Backups from a Noncluster File System

- Using RMAN or Oracle Enterprise Manager to Restore the Server Parameter File (SPFILE)

Note:

The restore and recovery procedures in a cluster file system scheme do not differ substantially from Oracle noncluster scenarios.

Restoring Backups from a Cluster File System

The scheme that this section describes assumes that you are using the "Oracle Automatic Storage Management and Cluster File System Archiving Scheme". In this scheme, assume that node3 performed the backups to a cluster file system. If node3 is available for the restore and recovery operation, and if all of the archived logs have been backed up or are on disk, then run the following commands to perform complete recovery:

```
RESTORE DATABASE;
RECOVER DATABASE;
```

If node3 performed the backups but is unavailable, then configure a media management device for one of the remaining nodes and make the backup media from node3 available to this node.

Note:

If you configured RMAN as described in "Configuring Channels to Use Automatic Load Balancing", then, to load balance the channels across nodes, note that channels cannot be load balanced before at least one instance has successfully opened the database. This means that the channels will not be load balanced across the nodes during a full database restore. To achieve load balancing of channels for RESTORE and RECOVER commands, you can temporarily reallocate channels by running commands similar to the following:

```
run {
ALLOCATE CHANNEL DEVICE TYPE sbt C1 CONNECT '@racinst_1'
ALLOCATE CHANNEL DEVICE TYPE sbt C2 CONNECT '@racinst_2'
...
}
```

Restoring Backups from a Noncluster File System

The scheme that this section describes assumes that you are using the "Noncluster File System Local Archiving Scheme". In this scheme, each node archives locally to a different directory. For example, node1 archives to /arc_dest_1, node2 archives to /arc_dest_2, and node3 archives to /arc_dest_3. You must configure a network file system file so that the recovery node can read the archiving directories on the remaining nodes.

If all nodes are available and if all archived redo logs have been backed up, then you can perform a complete restore and recovery by mounting the database and running the following commands from any node:

```
RESTORE DATABASE;
RECOVER DATABASE;
```

Because the network file system configuration enables each node read access to the redo log files on other nodes, then the recovery node can read and apply the archived redo logs located on the local and remote disks. No manual transfer of archived redo logs is required.

Using RMAN or Oracle Enterprise Manager to Restore the Server Parameter File (SPFILE)

RMAN can restore the server parameter file either to the default location or to a location that you specify.

You can also use Oracle Enterprise Manager to restore the SPFILE. From the Backup/Recovery section of the **Maintenance** tab, click **Perform Recovery**. The Perform Recovery link is context-sensitive and navigates you to the SPFILE restore only when the database is closed.

Instance Recovery in Oracle RAC

Instance failure occurs when software or hardware problems disable an instance. After instance failure, Oracle Database automatically uses the online redo logs to perform recovery as described in this section.

This section includes the following topics:

- Single Node Failure in Oracle RAC

- Multiple-Node Failures in Oracle RAC

- Using RMAN to Create Backups in Oracle RAC

- Node Affinity Awareness of Fast Connections

- Deleting Archived Redo Logs after a Successful Backup

- Autolocation for Backup and Restore Commands

Single Node Failure in Oracle RAC

Instance recovery in Oracle RAC does not include the recovery of applications that were running on the failed instance. Oracle Clusterware restarts the instance automatically.

Applications that were running on a node before it failed continue running by using failure recognition and recovery. This provides consistent and uninterrupted service if hardware or software fails. When one instance performs recovery for another instance, the surviving instance reads online redo logs generated by the failed instance and uses that information to ensure that committed transactions are recorded in the database. Thus, data from committed transactions is not lost. The instance performing recovery rolls back transactions that were active at the time of the failure and releases resources used by those transactions.

> **Note:**
>
> All online redo logs must be accessible for instance recovery. Therefore, Oracle recommends that you mirror your online redo logs.

Multiple-Node Failures in Oracle RAC

When multiple node failures occur, if one instance survives, then Oracle RAC performs instance recovery for any other instances that fail. If all instances of an Oracle RAC database fail, then Oracle Database automatically recovers the instances the next time one instance opens the database. The instance performing recovery can mount the database in either **cluster database** or exclusive mode from any node of an Oracle RAC database. This recovery procedure is the same for Oracle Database running in shared mode as it is for Oracle Database running in exclusive mode, except that one instance performs instance recovery for all of the failed instances.

Using RMAN to Create Backups in Oracle RAC

Oracle Database provides RMAN for backing up and restoring the database. RMAN enables you to back up, restore, and recover data files, control files, SPFILEs, and archived redo logs. RMAN is included with the Oracle Database server and it is installed by default. You can run RMAN from the command line or you can use it from the Backup Manager in Oracle Enterprise Manager. In addition, RMAN is the recommended backup and recovery tool if you are using Oracle Automatic Storage Management (Oracle ASM). The procedures for using RMAN in Oracle RAC environments do not differ substantially from those for Oracle noncluster environments.

> **See Also:**
>
> *Oracle Database Backup and Recovery User's Guide* for more information about noncluster RMAN backup procedures

Channel Connections to Cluster Instances with RMAN

Channel connections to the instances are determined using the connect string defined by channel configurations. For example, in the following configuration, three channels are allocated using dbauser/pwd@*service_name*. If you configure the SQL Net service name with load balancing turned on, then the channels are allocated at a node as decided by the load balancing algorithm.

```
CONFIGURE DEVICE TYPE sbt PARALLELISM 3;
CONFIGURE DEFAULT DEVICE TYPE TO sbt;
CONFIGURE CHANNEL DEVICE TYPE SBT CONNECT 'dbauser/pwd@service_name'
```

However, if the service name used in the connect string is not for load balancing, then you can control at which instance the channels are allocated using separate connect strings for each channel configuration, as follows:

```
CONFIGURE DEVICE TYPE sbt PARALLELISM 3;
CONFIGURE CHANNEL 1.. CONNECT 'dbauser/pwd@mydb_1';
CONFIGURE CHANNEL 2.. CONNECT 'dbauser/pwd@mydb_2';
CONFIGURE CHANNEL 3.. CONNECT 'dbauser/pwd@mydb_3';
```

In the previous example, it is assumed that `mydb_1`, `mydb_2` and `mydb_3` are SQL*Net service names that connect to pre-defined nodes in your Oracle RAC environment. Alternatively, you can also use manually allocated channels to backup your database files. For example, the following command backs up the SPFILE, controlfile, data files and archived redo logs:

```
RUN
{
    ALLOCATE CHANNEL CH1 CONNECT 'dbauser/pwd@mydb_1';
    ALLOCATE CHANNEL CH2 CONNECT 'dbauser/pwd@mydb_2';
    ALLOCATE CHANNEL CH3 CONNECT 'dbauser/pwd@mydb_3';
    BACKUP DATABASE PLUS ARCHIVED LOG;
}
```

During a backup operation, if at least one channel allocated has access to the archived log, then RMAN automatically schedules the backup of the specific log on that channel. Because the control file, SPFILE, and data files are accessible by any channel, the backup operation of these files is distributed across the allocated channels.

For a local archiving scheme, there must be at least one channel allocated to all of the nodes that write to their local archived logs. For a cluster file system archiving scheme, if every node writes the archived logs in the same cluster file system, then the backup operation of the archived logs is distributed across the allocated channels.

During a backup, the instances to which the channels connect must be either all mounted or all open. For example, if the instance on `node1` has the database mounted while the instances on `node2` and `node3` have the database open, then the backup fails.

See Also:

Oracle Database Backup and Recovery Reference for more information about the `CONNECT` clause of the `CONFIGURE CHANNEL` statement

---

Node Affinity Awareness of Fast Connections

In some cluster database configurations, some nodes of the cluster have faster access to certain data files than to other data files. RMAN automatically detects this situation, which is known as node affinity awareness. When deciding which channel to use to back up a particular data file, RMAN gives preference to the nodes with faster access to the data files that you want to back up. For example, if you have a three-node cluster, and if `node1` has faster read/write access to data files 7, 8, and 9 than the other nodes, then `node1` has greater node affinity to those files than `node2` and `node3`.

Deleting Archived Redo Logs after a Successful Backup

If you have configured the automatic channels as defined in section "Channel Connections to Cluster Instances with RMAN", then you can use the following example to delete the archived logs that you backed up *n* times. The device type can be `DISK` or `SBT`:

```
DELETE ARCHIVELOG ALL BACKED UP n TIMES TO DEVICE TYPE device_type;
```

During a delete operation, if at least one channel allocated has access to the archived log, then RMAN automatically schedules the deletion of the specific log on that channel. For a local archiving scheme, there must be at least one channel allocated that can delete an archived log. For a cluster file system archiving scheme, if every node

writes to the archived logs on the same cluster file system, then the archived log can be deleted by any allocated channel.

If you have not configured automatic channels, then you can manually allocate the maintenance channels as follows and delete the archived logs.

```
ALLOCATE CHANNEL FOR MAINTENANCE DEVICE TYPE DISK CONNECT 'SYS/oracle@node1';
ALLOCATE CHANNEL FOR MAINTENANCE DEVICE TYPE DISK CONNECT 'SYS/oracle@node2';
ALLOCATE CHANNEL FOR MAINTENANCE DEVICE TYPE DISK CONNECT 'SYS/oracle@node3';
DELETE ARCHIVELOG ALL BACKED UP n TIMES TO DEVICE TYPE device_type;
```

Autolocation for Backup and Restore Commands

RMAN automatically performs autolocation of all files that it must back up or restore. If you use the noncluster file system local archiving scheme, then a node can only read the archived redo logs that were generated by an instance on that node. RMAN never attempts to back up archived redo logs on a channel it cannot read.

During a restore operation, RMAN automatically performs the autolocation of backups. A channel connected to a specific node only attempts to restore files that were backed up to the node. For example, assume that log sequence 1001 is backed up to the drive attached to node1, while log 1002 is backed up to the drive attached to node2. If you then allocate channels that connect to each node, then the channel connected to node1 can restore log 1001 (but not 1002), and the channel connected to node2 can restore log 1002 (but not 1001).

Media Recovery in Oracle RAC

Media recovery must be user-initiated through a client application, whereas instance recovery is automatically performed by the database. In these situations, use RMAN to restore backups of the data files and then recover the database. The procedures for RMAN media recovery in Oracle RAC environments do not differ substantially from the media recovery procedures for noncluster environments.

The node that performs the recovery must be able to restore all of the required data files. That node must also be able to either read all of the required archived redo logs on disk or be able to restore them from backups.

When recovering a database with encrypted tablespaces (for example after a SHUTDOWN ABORT or a catastrophic error that brings down the database instance), you must open the Oracle Wallet after database mount and before you open the database, so the recovery process can decrypt data blocks and redo.

Parallel Recovery in Oracle RAC

Oracle Database automatically selects the optimum degree of parallelism for instance, crash, and media recovery. Oracle Database applies archived redo logs using an optimal number of parallel processes based on the availability of CPUs. You can use parallel instance recovery and parallel media recovery in Oracle RAC databases as described under the following topics:

- Parallel Recovery with RMAN

- Disabling Parallel Recovery

> **See Also:**
>
> *Oracle Database Backup and Recovery User's Guide* for more information on these topics

Parallel Recovery with RMAN

With RMAN's `RESTORE` and `RECOVER` commands, Oracle Database automatically makes parallel the following three stages of recovery:

Restoring Data Files

When restoring data files, the number of channels you allocate in the RMAN recover script effectively sets the parallelism that RMAN uses. For example, if you allocate five channels, you can have up to five parallel streams restoring data files.

Applying Incremental Backups

Similarly, when you are applying incremental backups, the number of channels you allocate determines the potential parallelism.

Applying Archived Redo Logs

With RMAN, the application of archived redo logs is performed in parallel. Oracle Database automatically selects the optimum degree of parallelism based on available CPU resources.

Disabling Parallel Recovery

You can override parallel recovery using the procedures under the following topics:

* Disabling Instance and Crash Recovery Parallelism

* Disabling Media Recovery Parallelism

Disabling Instance and Crash Recovery Parallelism

To disable parallel instance and crash recovery on a system with multiple CPUs, set the `RECOVERY_PARALLELISM` parameter in the database initialization parameter file, SPFILE, to `0` or `1`.

Disabling Media Recovery Parallelism

Use the `NOPARALLEL` clause of the RMAN `RECOVER` command or the `ALTER DATABASE RECOVER` statement to force Oracle Database to use non-parallel media recovery.

Using a Fast Recovery Area in Oracle RAC

To use a fast recovery area in Oracle RAC, you must place it on an Oracle ASM disk group, a Cluster File System, or on a shared directory that is configured through a network file system file for each Oracle RAC instance. In other words, the fast recovery area must be shared among all of the instances of an Oracle RAC database. In addition, set the parameter `DB_RECOVERY_FILE_DEST` to the same value on all instances.

Oracle Enterprise Manager enables you to set up a fast recovery area. To use this feature:

1. From the Cluster Database home page, click the **Maintenance** tab.

2. Under the Backup/Recovery options list, click **Configure Recovery Settings**.

3. Specify your requirements in the Fast Recovery Area section of the page.

4. Click **Help** on this page for more information.

See Also:

Oracle Database Backup and Recovery User's Guide for information about setting up and configuring the fast recovery area

---

8

Cloning Oracle RAC to Nodes in a New Cluster

This chapter describes how to clone Oracle Real Application Clusters (Oracle RAC) database homes on Linux and UNIX systems to nodes in a new cluster. To extend Oracle RAC to nodes in an existing cluster, see Using Cloning to Extend Oracle RAC to Nodes in the Same Cluster.

This chapter describes a noninteractive cloning technique that you implement with scripts. The cloning techniques described in this chapter are best suited for performing multiple simultaneous cluster installations. Creating the scripts is a manual process and can be error prone. If you only have one cluster to install, then you should use the traditional automated and interactive installation methods, such as Oracle Universal Installer, or the Provisioning Pack feature of Oracle Enterprise Manager.

Note:

Cloning is not a replacement for Oracle Enterprise Manager cloning that is a part of the Provisioning Pack. During Oracle Enterprise Manager cloning, the provisioning process interactively asks you the details about the Oracle home (such as the location to which you want to deploy the clone, the name of the Oracle Database home, a list of the nodes in the cluster, and so on).

The Provisioning Pack feature of Oracle Enterprise Manager Cloud Control provides a framework to make it easy for you to automate the provisioning of new nodes and clusters. For data centers with many Oracle RAC clusters, the investment in creating a cloning procedure to easily provision new clusters and new nodes to existing clusters is worth the effort.

This chapter includes the following topics:

- Introduction to Cloning Oracle RAC

- Preparing to Clone Oracle RAC

- Deploying Oracle RAC Clone to Nodes in a Cluster

- Locating and Viewing Log Files Generated During Cloning

Introduction to Cloning Oracle RAC

Cloning is the process of copying an existing Oracle RAC installation to a different location and updating the copied bits to work in the new environment. The changes made by one-off patches applied on the source Oracle home, would also be present after the clone operation. The source and the destination path (host to be cloned) need not be the same.

Some situations in which cloning is useful are:

- Cloning provides a way to prepare an Oracle home once and deploy it to many hosts simultaneously. You can complete the installation silently, as a noninteractive process. You do not need to use a graphical user interface (GUI) console and you can perform cloning from a Secure Shell (SSH) terminal session, if required.

- Cloning enables you to create an installation (copy of a production, test, or development installation) with all patches applied to it in a single step. Once you have performed the base installation and applied all patch sets and patches on the source system, the clone performs all of these individual steps as a single procedure. This is in contrast to going through the installation process to perform the separate steps to install, configure, and patch the installation on each node in the cluster.

- Installing Oracle RAC by cloning is a very quick process. For example, cloning an Oracle home to a new cluster of more than two nodes requires a few minutes to install the Oracle base software, plus a few minutes more for each node (approximately the amount of time it takes to run the `root.sh` script).

The cloned installation behaves the same as the source installation. For example, the cloned Oracle home can be removed using Oracle Universal Installer or patched using OPatch. You can also use the cloned Oracle home as the source for another cloning operation. You can create a cloned copy of a test, development, or production installation by using the command-line cloning scripts. The default cloning procedure is adequate for most usage cases. However, you can also customize various aspects of cloning, for example, to specify custom port assignments, or to preserve custom settings.

The cloning process works by copying all of the files from the source Oracle home to the destination Oracle home. Thus, any files used by the source instance that are located outside the source Oracle home's directory structure are not copied to the destination location.

The size of the binaries at the source and the destination may differ because these are relinked as part of the clone operation and the operating system patch levels may also differ between these two locations. Additionally, the number of files in the cloned home would increase because several files copied from the source, specifically those being instantiated, are backed up as part of the clone operation.

Preparing to Clone Oracle RAC

Use this overview to understand the procedures you use to clone Oracle RAC.

In the preparation phase, you create a copy of an Oracle home that you then use to perform the cloning procedure on one or more nodes. You also install Oracle Clusterware.

Install Oracle RAC

Use the detailed instructions in *Oracle Real Application Clusters Installation Guide* for your platform for your platform to install the Oracle RAC software and patches:

1. Install Oracle RAC and choose the **Software only** installation option.

2. Patch the release to the required level (for example, 12.1.0.n).

3. Apply one-off patches, if necessary.

> **See Also:**
>
> *Oracle Real Application Clusters Installation Guide*

Create a backup of the source home

Create a copy of the Oracle RAC home. Use this file to copy the Oracle RAC home to each node in the cluster.

When creating the backup (tar) file, the best practice is to include the release number in the name of the file. For example:

```
# cd /opt/oracle/product/12c/db_1
# tar -zcvf /pathname/db1120.tgz .
```

> **See Also:**
>
> "Deploying Oracle RAC Clone to Nodes in a Cluster"

Install and start Oracle Clusterware

Before you can use cloning to create an Oracle RAC home, you must first install and start Oracle Clusterware on the node or nodes to which you want to copy a cloned Oracle RAC home. In other words, you configure an Oracle RAC home that you cloned from a source cluster onto the nodes in a target cluster in the same order that you installed the Oracle Clusterware and Oracle RAC software components on the original nodes.

> **See Also:**
>
> *Oracle Clusterware Administration and Deployment Guide* for information about cloning Oracle Clusterware homes to create new clusters, and starting Oracle Clusterware by issuing the `crsctl start crs` command

Deploying Oracle RAC Clone to Nodes in a Cluster

After you complete the prerequisite tasks described in "Preparing to Clone Oracle RAC", you can deploy cloned Oracle homes.

Deploy the Oracle RAC database home to a cluster, as follows:

1. Perform any Oracle RAC preinstallation tasks, as described in your platform-specific Oracle RAC installation guide, to prepare the new cluster nodes, such things as:

   - Specify the kernel parameters.

   - Ensure Oracle Clusterware is active.

   - Ensure that Oracle ASM is active and that at least one Oracle ASM disk group exists and is mounted.

2. Deploy the Oracle RAC database software, as follows:

   a. Copy the clone of the Oracle home to all nodes. For example:

```
[root@node1 root]# mkdir -p /opt/oracle/product/12c/db
[root@node1 root]# cd /opt/oracle/product/12c/db
[root@node1 db]# tar -zxvf /path_name/db1120.tgz
```

When providing the home location and *path_name*, the home location can be in the same directory path or in a different directory path from the source home that you used to create the tar.

b. If either the `oracle` user or the `oinstall` group, or both is different between the source and destination nodes, then change the ownership of the Oracle Inventory files, as follows:

```
[root@node1]# chown -R oracle:oinstall /opt/oracle/product/12c/db
```

When you run the preceding command on the Oracle RAC home, it clears `setuid` and `setgid` information from the Oracle binary.

Note:

You can perform this step at the same time you perform Step 3 and Step 4 to run the `clone.pl` and `$ORACLE_HOME/root.sh` scripts on each cluster node.

---

3. Run the `clone.pl` script on each node, which performs the main Oracle RAC cloning tasks, as follows:

a. Supply the environment variables and cloning parameters in the `start.sh` script, as described in Table 8-2 and Table 8-3. Because the `clone.pl` script is sensitive to the parameters being passed to it, you must be accurate in your use of brackets, single quotation marks, and double quotation marks.

b. Run the script as `oracle` or the user that owns the Oracle RAC software.

The following table lists and describes the `clone.pl` script parameters:

Table 8-1 clone.pl Script Parameters

Parameter	Description
ORACLE_HOME=*Oracle_home*	The complete path to the Oracle home you want to clone. If you specify an invalid path, then the script exits. This parameter is required.
ORACLE_BASE=*ORACLE_BASE*	The complete path to the Oracle base you want to clone. If you specify an invalid path, then the script exits. This parameter is required.
ORACLE_HOME_NAME= *Oracle_home_name* | -defaultHomeName	The Oracle home name of the home you want to clone. Optionally, you can specify the -defaultHomeName flag. This parameter is optional.
ORACLE_HOME_USER=*Oracle_home_user*	The OracleHomeUser for Windows services. This parameter is applicable to Windows only and is optional.

Table 8-1   (Cont.) clone.pl Script Parameters

Parameter	Description
OSDBA_GROUP=*group_name*	Specify the operating system group you want to use as the OSDBA privileged group. This parameter is optional.
OSOPER_GROUP=*group_name*	Specify the operating system group you want to use as the OSOPER privileged group. This parameter is optional.
OSASM_GROUP=*group_name*	Specify the operating system group you want to use as the OSASM privileged group. This parameter is optional.
OSBACKUPDBA_GROUP=*group_name*	Specify the operating system group you want to use as the OSBACKUPDBA privileged group. This parameter is optional.
OSDGDBA_GROUP=*group_name*	Specify the operating system group you want to use as the OSDGDBA privileged group. This parameter is optional.
OSKMDBA_GROUP=*group_name*	Specify the operating system group you want to use as the OSKMDBA privileged group. This parameter is optional.
-debug	Specify this option to run the clone.pl script in debug mode
-help	Specify this option to obtain help for the clone.pl script.

See Also:

Oracle Real Application Clusters Installation Guide for your platform for more information about the operating system groups listed in the preceding table

The following example shows an excerpt from the start.sh script that calls the clone.pl script:

```
ORACLE_BASE=/opt/oracle
ORACLE_HOME=/opt/oracle/product/12c/db
cd $ORACLE_HOME/clone
THISNODE='host_name'

E01=ORACLE_HOME=/opt/oracle/product/12c/db
E02=ORACLE_HOME_NAME=OraDBRAC
E03=ORACLE_BASE=/opt/oracle
C01="-O CLUSTER_NODES={node1,node2}"
C02="-O LOCAL_NODE=$THISNODE"

perl $ORACLE_HOME/clone/bin/clone.pl $E01 $E02 $E03 $C01 $C02
```

The following table lists and describes the environment variables E01, E02, and E03 that are shown in bold typeface in the preceding example:

Table 8-2   Environment Variables Passed to the clone.pl Script

Symbol	Variable	Description
E01	`ORACLE_HOME`	The location of the Oracle RAC database home. This directory location must exist and must be owned by the Oracle operating system group: `oinstall`.
E02	`ORACLE_HOME_NAM E`	The name of the Oracle home for the Oracle RAC database. This is stored in the Oracle Inventory.
E03	`ORACLE_BASE`	The location of the Oracle Base directory.

The following table lists and describes the cloning parameters C01 and C02, that are shown in bold typeface in the preceding example:

Table 8-3   Cloning Parameters Passed to the clone.pl Script.

Variable	Name	Parameter	Description
C01	Cluster Nodes	`CLUSTER_NODE S`	Lists the nodes in the cluster.
C02	Local Node	`LOCAL_NODE`	The name of the local node.

The following example shows an excerpt from the `start.bat` script that the user must create that calls the `clone.pl` script:

```
set ORACLE_home=C:\oracle\product\12c\db1
cd %ORACLE_home%\clone\bin
set THISNODE=%hostname%
set E01=ORACLE_HOME=%ORACLE_home%
set E02=ORACLE_HOME_NAME=OraDBRAC
set E03=ORACLE_BASE=Oracle_Base
set C01="CLUSTER_NODES={node1,node2}"
set C02="-O LOCAL_NODE=%THISNODE%"
perl clone.pl %E01% %E02% %E03% %C01% %C02%
```

4.   **Note:**

This step applies to Linux and UNIX installations, only.

Run the `$ORACLE_HOME/root.sh` as the `root` operating system user as soon as the `clone.pl` procedure completes on the node.

```
[root@node1 root]# /opt/oracle/product/12c/db/root.sh -silent
```

Note that you can run the script on each node simultaneously:

```
[root@node2 root]# /opt/oracle/product/12c/db/root.sh -silent
```

Ensure the script has completed on each node before proceeding to the next step.

> **5.**   **Note:**
>
> You need only run DBCA on one node in the cluster to create Oracle RAC instances on all nodes.

This step shows how to run DBCA in silent mode and provide response file input to create the Oracle RAC instances.

The following example creates an Oracle RAC database named ERI on each node, creates database instances on each node, registers the instances in OCR, creates the database files in the Oracle ASM disk group called DATA, and creates sample schemas. It also sets the SYS, SYSTEM, SYSMAN and DBSNMP passwords to *password*, which is the password for each account:

```
[oracle@node1 oracle]$ export ORACLE_HOME=/opt/oracle/product/12c/db
[oracle@node1 oracle]$ cd $ORACLE_HOME/bin/
[oracle@node1 bin]$./dbca -silent -createDatabase -templateName
General_Purpose.dbc \
-gdbName ERI -sid ERI \
-sysPassword password -systemPassword password \
-sysmanPassword password -dbsnmpPassword password \
-emConfiguration LOCAL \
-storageType ASM -diskGroupName DATA \
-datafileJarLocation $ORACLE_HOME/assistants/dbca/templates \
-nodelist node1,node2 -characterset WE8ISO8859P1 \
-obfuscatedPasswords false -sampleSchema true
```

> **See Also:**
>
> *Oracle Database 2 Day DBA* for information about using DBCA to create and configure a database

Locating and Viewing Log Files Generated During Cloning

The cloning script runs multiple tools, each of which may generate its own log files. After the clone.pl script finishes running, you can view log files to obtain more information about the cloning process.

The following log files that are generated during cloning are the key log files of interest for diagnostic purposes:

- *Central_Inventory*/logs/cloneAction*timestamp*.log

  Contains a detailed log of the actions that occur during the Oracle Universal Installer part of the cloning.

- *Central_Inventory*/logs/oraInstall*timestamp*.err

  Contains information about errors that occur when Oracle Universal Installer is running.

- *Central_Inventory*/logs/oraInstall*timestamp*.out

  Contains other miscellaneous messages generated by Oracle Universal Installer.

- $ORACLE_HOME/clone/logs/clone*timestamp*.log

Contains a detailed log of the actions that occur before cloning and during the cloning operations.

- `$ORACLE_HOME/clone/logs/error`*`timestamp`*`.log`

  Contains information about errors that occur before cloning and during cloning operations.

Table 8-4 describes how to find the location of the Oracle inventory directory.

Table 8-4   Finding the Location of the Oracle Inventory Directory

Type of System...	Location of the Oracle Inventory Directory
All UNIX computers except Linux and IBM AIX	`/var/opt/oracle/oraInst.loc`
IBM AIX and Linux	`/etc/oraInst.loc` file.
Windows	`C:\Program Files\Oracle\Inventory`

9

Using Cloning to Extend Oracle RAC to Nodes in the Same Cluster

This chapter provides information about using cloning to extend Oracle Real Application Clusters (Oracle RAC) to nodes in an existing cluster. To add Oracle RAC to nodes in a new cluster, see Cloning Oracle RAC to Nodes in a New Cluster.

This chapter contains the following topics:

- About Adding Nodes Using Cloning in Oracle RAC Environments

- Cloning Local Oracle Homes on Linux and UNIX Systems

- Cloning Shared Oracle Homes on Linux and UNIX Systems

- Cloning Oracle Homes on Windows Systems

See Also:

- "Introduction to Cloning Oracle RAC" for an overview of cloning and a discussion about the benefits of cloning

- *Oracle Clusterware Administration and Deployment Guide* for information about adding nodes to your cluster

---

About Adding Nodes Using Cloning in Oracle RAC Environments

The cloning procedures assume that you have successfully installed and configured an Oracle RAC environment to which you want to add nodes and instances. To add nodes to an Oracle RAC environment using cloning, first extend the Oracle Clusterware configuration, then extend the Oracle Database software with Oracle RAC, and then add the listeners and instances by running the Oracle assistants

The cloning script runs multiple tools, each of which may generate its own log files. After the `clone.pl` script finishes running, you can view log files to obtain more information about the cloning process. See "Locating and Viewing Log Files Generated During Cloning" for more information.

Cloning Local Oracle Homes on Linux and UNIX Systems

This section explains how to add nodes to existing Oracle RAC environments by cloning a local (non-shared) Oracle home in Linux and UNIX system environments.

Complete the following steps to clone Oracle Database with Oracle RAC software:

1. Follow the steps in the "Preparing to Clone Oracle RAC" to create a copy of an Oracle home that you then use to perform the cloning procedure on one or more nodes.

2. Use the `tar` utility to create an archive of the Oracle home on the existing node and copy it to the new node. If the location of the Oracle home on the source node is `$ORACLE_HOME`, then you must use this same directory as the destination location on the new node.

3. On the new node, configure the environment variables `ORACLE_HOME` and `ORACLE_BASE`. Then go to the `$ORACLE_HOME/clone/bin` directory and run the following command where `existing_node` is the name of the node that you are cloning, `new_node2` and `new_node3` are the names of the new nodes, and `Oracle_home_name` is the name of the Oracle home:

   ```
   perl clone.pl -O 'CLUSTER_NODES={existing_node,new_node2,new_node3}'
   -O LOCAL_NODE=new_node2 ORACLE_BASE=$ORACLE_BASE ORACLE_HOME=$ORACLE_HOME
   ORACLE_HOME_NAME=Oracle_home_name -O -noConfig
   ```

4. Run the following command to run the configuration assistants to configure Oracle RAC on the new nodes:

   ```
   $ORACLE_HOME/cfgtoollogs/configToolFailedCommands
   ```

   This script contains all commands that failed, were skipped, or were canceled during the installation. You can use this script to run the database configuration assistants outside of Oracle Universal Installer. Note that before you run the script you should check the script to see if any passwords within it need to be updated.

5. Run the following command on the existing node from the `$ORACLE_HOME/oui/bin` directory to update the inventory in the Oracle Database home with Oracle RAC, specified by `Oracle_home`, where `existing_node` is the name of the original node that you are cloning and `new_node2` and `new_node3` are the names of the new nodes:

   ```
   ./runInstaller -updateNodeList ORACLE_HOME=$ORACLE_HOME -O "CLUSTER_
   NODES={existing_node,new_node2,new_node3}"
   ```

6. On each new node, go to the `$ORACLE_HOME` directory and run the following command:

   ```
   ./root.sh
   ```

7. From the node that you cloned, run Database Configuration Assistant (DBCA) to add Oracle RAC database instances on the new nodes.

Cloning Shared Oracle Homes on Linux and UNIX Systems

This section explains how to add nodes to existing Oracle RAC environments by cloning a shared Oracle home in Linux and UNIX system environments.

Complete the following steps to clone Oracle Database with Oracle RAC software:

1. Follow the steps in the "Preparing to Clone Oracle RAC" to create a copy of an Oracle home that you then use to perform the cloning procedure on one or more nodes.

2. On the new node, configure the environment variables `ORACLE_HOME` and `ORACLE_BASE`. Then go to the `$ORACLE_HOME/clone/bin` directory and run the following command where `existing_node` is the name of the node that you

are cloning, *new_node2* and *new_node3* are the names of the new nodes, *Oracle_home_name* is the name of the Oracle home, and the `-cfs` option indicates the Oracle home is shared:

```
perl clone.pl -O 'CLUSTER_NODES={existing_node,new_node2,new_node3}'
-O LOCAL_NODE=new_node2 ORACLE_BASE=$ORACLE_BASE ORACLE_HOME=$ORACLE_HOME
 ORACLE_HOME_NAME=Oracle_home_name [-cfs -noConfig]
```

> **Note:**
>
> In the preceding command:
>
> - Use the `-cfs` and `-noConfig` options for a shared Oracle Database home with Oracle RAC.
>
> - The value for the `ORACLE_HOME_NAME` parameter must be that of the node you are cloning.

3. Run the following command on the existing node from the `$ORACLE_HOME/oui/bin` directory to update the inventory in the Oracle Database home with Oracle RAC, specified by *Oracle_home*, where *existing_node* is the name of the original node that you are cloning and *new_node2* and *new_node3* are the names of the new nodes:

```
./runInstaller -updateNodeList ORACLE_HOME=$ORACLE_HOME "CLUSTER_
NODES={existing_node,new_node2,new_node3}"
```

4. On each new node, go to the `$ORACLE_HOME` directory and run the following command:

```
./root.sh
```

5. From the node that you cloned, run Database Configuration Assistant (DBCA) to add Oracle RAC database instances to the new nodes.

Cloning Oracle Homes on Windows Systems

This section explains how to add nodes to existing Oracle RAC environments by cloning a shared or local Oracle home in Windows system environments.

Complete the following steps to clone Oracle Database with Oracle RAC software:

1. If you have a local Oracle home, then use the ZIP utility to create an archive of the Oracle Database home with Oracle RAC on the existing node and copy it to the new node. Otherwise, proceed to the next step.

   Extract the Oracle Database with Oracle RAC home files from the ZIP file on the new node in the same directory in which the Oracle Database home with Oracle RAC resided on the existing node. For example, assume that the location of the destination Oracle RAC home on the new node is `%ORACLE_HOME%`.

2. On the new node, go to the `%ORACLE_HOME%\clone\bin` directory and run the following command where *Oracle_Home* is the Oracle Database home, *Oracle_Home_Name* is the name of the Oracle Database home, *Oracle_Base* is the Oracle base directory, *user_name* is the name of the Oracle home user (a non-Administrator user) for the Oracle home being cloned, *existing_node* is the name of the existing node, and *new_node* is the name of the new node:

```
perl clone.pl ORACLE_HOME=Oracle_Home ORACLE_BASE=Oracle_Base
ORACLE_HOME_NAME=Oracle_Home_Name ORACLE_HOME_USER=user_name
-O 'CLUSTER_NODES={existing_node,new_node}'
-O LOCAL_NODE=new_node
```

If you have a shared Oracle Database home with Oracle RAC, then append the -cfs option to the command to indicate that the Oracle home is shared, as shown in the following example:

```
perl clone.pl ORACLE_HOME=Oracle_Home ORACLE_BASE=Oracle_Base
ORACLE_HOME_NAME=Oracle_Home_Name ORACLE_HOME_USER=user_name
-O 'CLUSTER_NODES={existing_node,new_node}' -O LOCAL_NODE=new_node
[-cfs -noConfig]
```

> **Note:**
>
> - The ORACLE_HOME_USER is required *only* if you are cloning a secured Oracle home.
>
> - Use the -cfs and -noConfig options for a shared Oracle Database home with Oracle RAC.
>
> - The value for the ORACLE_HOME_NAME parameter must be that of the node you are cloning. To obtain the ORACLE_HOME_NAME, look in the registry on the node you cloning for the ORACLE_HOME_NAME parameter key under HKEY_LOCAL_MACHINE\SOFTWARE\oracle \KEY_OraCRs12c_home1.

3. On the existing node, from the %ORACLE_HOME%\oui\bin directory run the following command to update the inventory in the Oracle Database home with Oracle RAC, specified by Oracle_home, where existing_node is the name of the existing node, and new_node is the name of the new node:

```
setup.exe -updateNodeList ORACLE_HOME=Oracle_home "CLUSTER_NODES=
{existing_node,new_node}" LOCAL_NODE=existing_node
```

4. From the node that you cloned, run DBCA to add Oracle RAC database instances to the new nodes.

10

Adding and Deleting Oracle RAC from Nodes on Linux and UNIX Systems

This chapter describes how to extend an existing Oracle Real Application Clusters (Oracle RAC) home to other nodes and instances in the cluster, and delete Oracle RAC from nodes and instances in the cluster. This chapter provides instructions for Linux and UNIX systems.

If your goal is to clone an existing Oracle RAC home to create multiple new Oracle RAC installations across the cluster, then use the cloning procedures that are described in Cloning Oracle RAC to Nodes in a New Cluster.

The topics in this chapter include the following:

- Adding Oracle RAC to Nodes with Oracle Clusterware Installed

- Deleting Oracle RAC from a Cluster Node

> **Note:**
>
> - Ensure that you have a current backup of Oracle Cluster Registry (OCR) before adding or deleting Oracle RAC by running the `ocrconfig -showbackup` command.
>
> - The phrase "target node" as used in this chapter refers to the node to which you plan to extend the Oracle RAC environment.

> **See Also:**
>
> - *Oracle Database 2 Day + Real Application Clusters Guide* for additional information about configuring a new Oracle RAC cluster or scaling up an existing Oracle RAC cluster
>
> - Adding and Deleting Oracle RAC from Nodes on Windows Systems

Adding Oracle RAC to Nodes with Oracle Clusterware Installed

Before beginning this procedure, ensure that your existing nodes have the correct path to the *Grid_home* and that the `$ORACLE_HOME` environment variable is set to the Oracle RAC home.

> **See Also:**
>
> *Oracle Clusterware Administration and Deployment Guide* for information about
> extending the Oracle Clusterware home to new nodes in a cluster

- If you are using a local (non-shared) Oracle home, then you must extend the
  Oracle RAC database home that is on an existing node (`node1` in this procedure)
  to a target node (`node3` in this procedure).

  Navigate to the *Oracle_home*/addnode directory on `node1` and run the
  addnode.sh script.

  If you want to perform a silent installation, run the addnode.sh script using the
  following syntax:

  ```
  $ ./addnode.sh -silent "CLUSTER_NEW_NODES={node3}"
  ```

- If you have a shared Oracle home that is shared using Oracle Automatic Storage
  Management Cluster File System (Oracle ACFS), then do the following to extend
  the Oracle database home to `node3`:

  1. Start the Oracle ACFS resource on the new node by running the following
     command as `root` from the *Grid_home*/bin directory:

     ```
     # srvctl start filesystem -device volume_device [-node node_name]
     ```

     > **Note:**
     >
     > Make sure the Oracle ACFS resources, including Oracle ACFS registry
     > resource and Oracle ACFS file system resource where the Oracle home is
     > located, are online on the newly added node.

  2. Run the following command as the user that installed Oracle RAC from the
     Oracle_home/oui/bin directory on the node you are adding to add the
     Oracle RAC database home:

     ```
     $ ./runInstaller -attachHome ORACLE_HOME="ORACLE_HOME"
     "CLUSTER_NODES={node3}"
     LOCAL_NODE="node3" ORACLE_HOME_NAME="home_name" -cfs
     ```

  3. Navigate to the *Oracle_home*/addnode directory on `node1` and run the
     addnode.sh script as the user that installed Oracle RAC using the following
     syntax:

     ```
     $ ./addnode.sh -noCopy "CLUSTER_NEW_NODES={node3}"
     ```

     > **Note:**
     >
     > Use the –noCopy option because the Oracle home on the destination node is
     > already fully populated with software.

- If you have a shared Oracle home on a shared file system that *is not* Oracle ACFS,
  then you must first create a mount point for the Oracle RAC database home on the
  target node, mount and attach the Oracle RAC database home, and update the
  Oracle Inventory, as follows:

1. Run the `srvctl config database -db` *db_name* command on an existing node in the cluster to obtain the mount point information.

2. Run the following command as `root` on `node3` to create the mount point:

   ```
   # mkdir -p mount_point_path
   ```

3. Mount the file system that hosts the Oracle RAC database home.

4. Run the following command as the user that installed Oracle RAC from the *Oracle_home*/oui/bin directory on the node you are adding to add the Oracle RAC database home:

   ```
   $ ./runInstaller -attachHome ORACLE_HOME="ORACLE_HOME" "CLUSTER
   _NODES={local_node_name}" LOCAL_NODE="node_name"
   ORACLE_HOME_NAME="home_name"
   ```

5. Update the Oracle Inventory as the user that installed Oracle RAC, as follows:

   ```
   $ ./runInstaller -updateNodeList ORACLE_HOME=mount_point_path
   "CLUSTER_NODES={node_list}"
   ```

   In the preceding command, *node_list* refers to a list of all nodes where the Oracle RAC database home is installed, including the node you are adding.

Run the *Oracle_home*/root.sh script on `node3` as `root`.

> **Note:**
>
> Oracle recommends that you back up the OCR after you complete the node addition process.

You can now add an Oracle RAC database instance to the target node using either of the procedures in the following sections.

- Adding Policy-Managed Oracle RAC Database Instances to Target Nodes

- Adding Administrator-Managed Oracle RAC Database Instances to Target Nodes

Adding Policy-Managed Oracle RAC Database Instances to Target Nodes

You must manually add undo and redo logs, unless you store your policy-managed database on Oracle Automatic Storage Management (Oracle ASM) and **Oracle Managed Files** is enabled.

If there is space in a server pool to add a node and the database has been started at least once, then Oracle Clusterware adds the Oracle RAC database instance to the newly added node and no further action is necessary.

> **Note:**
>
> The database must have been started at least once before you can add the database instance to the newly added node.

If there is no space in any server pool, then the newly added node moves into the Free server pool. Use the `srvctl modify srvpool` command to increase the cardinality of a server pool to accommodate the newly added node, after which the node moves

out of the Free server pool and into the modified server pool, and Oracle Clusterware adds the Oracle RAC database instance to the node.

Adding Administrator-Managed Oracle RAC Database Instances to Target Nodes

> **Note:**
>
> The procedures in this section only apply to administrator-managed databases. Policy-managed databases use nodes when the nodes are available in the databases' server pool.

You can use either Oracle Enterprise Manager or DBCA to add Oracle RAC database instances to the target nodes. To add a database instance to a target node with Oracle Enterprise Manager, see the *Oracle Database 2 Day + Real Application Clusters Guide* for complete information.

This section describes using DBCA to add Oracle RAC database instances.

These tools guide you through the following tasks:

- Creating a new database instance on each target node

- Creating and configuring high availability components

- Creating the Oracle Net configuration for a non-default listener from the Oracle home

- Starting the new instance

- Creating and starting services if you entered services information on the Services Configuration page

After adding the instances to the target nodes, you should perform any necessary service configuration procedures, as described in Workload Management with Dynamic Database Services.

Using DBCA in Interactive Mode to Add Database Instances to Target Nodes

To add a database instance to a target node with DBCA in interactive mode, perform the following steps:

1. Ensure that your existing nodes have the $ORACLE_HOME environment variable set to the Oracle RAC home.

2. Start DBCA by entering dbca at the system prompt from the *Oracle_home*/bin directory.

    DBCA performs certain CVU checks while running. However, you can also run CVU from the command line to perform various verifications.

    > **See Also:**
    >
    > *Oracle Clusterware Administration and Deployment Guide* for more information about CVU

DBCA displays the Welcome page for Oracle RAC. Click **Help** on any DBCA page for additional information.

3. Select **Instance Management**, click **Next**, and DBCA displays the Instance Management page.

4. Select **Add Instance** and click **Next**. DBCA displays the List of Cluster Databases page that shows the databases and their current status, such as `ACTIVE` or `INACTIVE`.

5. From the List of Cluster Databases page, select the active Oracle RAC database to which you want to add an instance. Click **Next** and DBCA displays the List of Cluster Database Instances page showing the names of the existing instances for the Oracle RAC database that you selected.

6. Click **Next** to add a new instance and DBCA displays the Adding an Instance page.

7. On the Adding an Instance page, enter the instance name in the field at the top of this page if the instance name that DBCA provides does not match your existing instance naming scheme.

8. Review the information on the Summary dialog and click **OK** or click **Cancel** to end the instance addition operation. DBCA displays a progress dialog showing DBCA performing the instance addition operation.

9. After you terminate your DBCA session, run the following command to verify the administrative privileges on the target node and obtain detailed information about these privileges where *nodelist* consists of the names of the nodes on which you added database instances:

    ```
    cluvfy comp admprv -o db_config -d Oracle_home -n nodelist [-verbose]
    ```

10. Perform any necessary service configuration procedures, as described in Workload Management with Dynamic Database Services.

Using DBCA in Silent Mode to Add Database Instances to Target Nodes

You can use DBCA in silent mode to add instances to nodes on which you have extended an Oracle Clusterware home and an Oracle Database home. Before you run the `dbca` command, ensure that you have set the `ORACLE_HOME` environment variable correctly on the existing nodes. Run DBCA, supplying values for the variables described in Table 10-1,as follows:

```
dbca -silent -nodeList node_name -gdbName gdb_name
[-instanceName instance_name -sysDBAUserName sysdba -sysDBAPassword password]
```

Table 10-1   Variables in the DBCA Silent Mode Syntax

Variable	Description
node_name	The node on which you want to add (or delete) the instance.
gdb_name	Global database name.
instance_name	Name of the instance. Provide an instance name only if you want to override the Oracle naming convention for Oracle RAC instance names.
sysdba	Name of the Oracle user with `SYSDBA` privileges.

Table 10-1   (Cont.) Variables in the DBCA Silent Mode Syntax

Variable	Description
password	Password for the SYSDBA user.

Perform any necessary service configuration procedures, as described in Workload Management with Dynamic Database Services.

Deleting Oracle RAC from a Cluster Node

To remove Oracle RAC from a cluster node, you must delete the database instance and the Oracle RAC software before removing the node from the cluster.

> **Note:**
>
> If there are no database instances on the node you want to delete, then proceed to "Removing Oracle RAC".

This section includes the following procedures to delete nodes from clusters in an Oracle RAC environment:

- Deleting Instances from Oracle RAC Databases
- Removing Oracle RAC
- Deleting Nodes from the Cluster

Deleting Instances from Oracle RAC Databases

The procedures for deleting database instances are different for policy-managed and administrator-managed databases. Deleting a policy-managed database instance involves reducing the number of servers in the server pool in which the database instance resides. Deleting an administrator-managed database instance involves using DBCA to delete the database instance.

To delete a policy-managed database, reduce the number of servers in the server pool in which a database instance resides by relocating the server on which the database instance resides to another server pool. This effectively removes the instance without having to remove the Oracle RAC software from the node or the node from the cluster.

For example, you can delete a policy-managed database by running the following commands on any node in the cluster:

```
$ srvctl stop instance -d db_unique_name -n node_name
$ srvctl relocate server -n node_name -g Free
```

The first command stops the database instance on a particular node and the second command moves the node out of its current server pool and into the Free server pool.

> **See Also:**
>
> "Removing Oracle RAC" for information about removing the Oracle RAC software from a node

Deleting Instances from Administrator-Managed Databases

> **Note:**
>
> Before deleting an instance from an Oracle RAC database using SRVCTL to do the following:
>
> - If you have services configured, then relocate the services
>
> - Modify the services so that each service can run on one of the remaining instances
>
> - Ensure that the instance to be removed from an administrator-managed database is neither a preferred nor an available instance of any service

> **See Also:**
>
> "Administering Services with SRVCTL"

The procedure in this section explains how to use DBCA in interactive mode to delete an instance from an Oracle RAC database.

> **See Also:**
>
> *Oracle Database 2 Day + Real Application Clusters Guide* for information about how to delete a database instance from a target node with Oracle Enterprise Manager

Using DBCA in Interactive Mode to Delete Instances from Nodes

To delete an instance using DBCA in interactive mode, perform the following steps:

1. Start DBCA.

   Start DBCA on a node *other than* the node that hosts the instance that you want to delete. The database and the instance that you plan to delete should be running during this step.

2. On the DBCA Operations page, select **Instance Management** and click **Next**. DBCA displays the Instance Management page.

3. On the DBCA Instance Management page, select the instance to be deleted, select **Delete Instance,** and click **Next**.

4. On the List of Cluster Databases page, select the Oracle RAC database from which to delete the instance, as follows:

   a. On the List of Cluster Database Instances page, DBCA displays the instances that are associated with the Oracle RAC database that you selected and the status of each instance. Select the cluster database from which you will delete the instance.

   b. Click **OK** on the Confirmation dialog to proceed to delete the instance.

DBCA displays a progress dialog showing that DBCA is deleting the instance. During this operation, DBCA removes the instance and the instance's Oracle Net configuration.

Click **No** and exit DBCA or click **Yes** to perform another operation. If you click **Yes**, then DBCA displays the Operations page.

5. Verify that the dropped instance's redo thread has been removed by using SQL*Plus on an existing node to query the GV$LOG view. If the redo thread is not disabled, then disable the thread. For example:

```
SQL> ALTER DATABASE DISABLE THREAD 2;
```

6. Verify that the instance has been removed from OCR by running the following command, where *db_unique_name* is the database unique name for your Oracle RAC database:

```
srvctl config database -d db_unique_name
```

7. If you are deleting more than one node, then repeat these steps to delete the instances from all the nodes that you are going to delete.

Using DBCA in Silent Mode to Delete Instances from Nodes

You can use DBCA in silent mode to delete a database instance from a node.

Run the following command, where the variables are the same as those shown in Table 10-1 for the DBCA command to add an instance. Provide a node name only if you are deleting an instance from a node other than the one on where DBCA is running as shown in the following example where *password* is the password:

```
dbca -silent -deleteInstance [-nodeList node_name] -gdbName gdb_name
-instanceName instance_name [-sysDBAUserName sysdba -sysDBAPassword password]
```

At this point, you have accomplished the following:

- Deregistered the selected instance from its associated Oracle Net Services listeners

- Deleted the selected database instance from the instance's configured node

- Removed the Oracle Net configuration

- Deleted the Oracle Flexible Architecture directory structure from the instance's configured node.

Removing Oracle RAC

This procedure removes Oracle RAC software from the node you are deleting from the cluster and updates inventories on the remaining nodes.

1. If there is a listener in the Oracle RAC home on the node you are deleting, then you must disable and stop it before deleting the Oracle RAC software. Run the following commands on any node in the cluster, specifying the name of the listener and the name of the node you are deleting:

```
$ srvctl disable listener -l listener_name -n name_of_node_to_delete
$ srvctl stop listener -l listener_name -n name_of_node_to_delete
```

2. Run the following command from $ORACLE_HOME/oui/bin on the node that you are deleting to update the inventory on that node:

```
$ ./runInstaller -updateNodeList ORACLE_HOME=Oracle_home_location
"CLUSTER_NODES={name_of_node_to_delete}" -local
```

> **Note:**
>
> If you have a shared Oracle RAC home, then append the `-cfs` option to the preceding command and provide a complete path to the location of the cluster file system.

3. Deinstall the Oracle home—only if the Oracle home is *not* shared—from the node that you are deleting by running the following command from the `Oracle_home\deinstall` directory:

```
deinstall -local
```

> **Caution:**
>
> If the Oracle home is shared, then *do not* run this command because it will remove the shared software. Proceed to the next step, instead.

4. Run the following command from the `$ORACLE_HOME/oui/bin` directory on any one of the remaining nodes in the cluster to update the inventories of those nodes, specifying a comma-delimited list of remaining node names and the name of the local node:

```
$ ./runInstaller -updateNodeList ORACLE_HOME=Oracle_home_location
"CLUSTER_NODES={remaining_node_list}" LOCAL_NODE=local_node_name
```

> **Note:**
>
> - Because all nodes may not have database software installed on an Oracle Flex Cluster, `remaining_node_list` must list only those nodes with installed database software homes.
>
> - If you have a shared Oracle RAC home, then append the `-cfs` option to the command example in this step and provide a complete path to the location of the cluster file system.

Deleting Nodes from the Cluster

After you delete the database instance and the Oracle RAC software, you can begin the process of deleting the node from the cluster. You accomplish this by running scripts on the node you want to delete to remove the Oracle Clusterware installation and then you run scripts on the remaining nodes to update the node list.

> **See Also:**
>
> *Oracle Clusterware Administration and Deployment Guide* for information about deleting nodes from the cluster

11

Adding and Deleting Oracle RAC from Nodes on Windows Systems

This chapter describes how to extend an existing Oracle Real Application Clusters (Oracle RAC) home to other nodes and instances in the cluster, and delete Oracle RAC from nodes and instances in the cluster. This chapter provides instructions for Windows systems.

Note:

In this chapter, the entries for *Grid_home* refer to the full path name for the Oracle Grid Infrastructure home, and the entries for *Oracle_home* refer to substitutes for environment variables for the Oracle home with Oracle RAC.

If your goal is to clone an existing Oracle RAC home to create multiple new Oracle RAC installations across the cluster, then use the cloning procedures that are described in Cloning Oracle RAC to Nodes in a New Cluster.

This chapter includes the following topics:

- Adding Oracle RAC to Nodes with Oracle Clusterware Installed

- Deleting Oracle RAC from a Cluster Node

Note:

- Ensure that you have a current backup of Oracle Cluster Registry (OCR) before adding or deleting Oracle RAC by running the `ocrconfig - showbackup` command.

- For all of the add node and delete node procedures, temporary directories such as `%TEMP%` or `C:\Temp` *should not be* shared directories. If your temporary directories are shared, then set your temporary environment variable, such as `%TEMP%`, to a location on a local node. In addition, use a directory path that exists on all of the nodes.

See Also:

Oracle Database 2 Day + Real Application Clusters Guide for additional information about configuring a new Oracle RAC cluster or adding to and deleting nodes from an existing Oracle RAC cluster.

Adding Oracle RAC to Nodes with Oracle Clusterware Installed

Before beginning this procedure, ensure that your existing nodes have the correct path to the `Grid_home` and that the `Oracle_home` environment variables are set correctly.

> **See Also:**
>
> *Oracle Clusterware Administration and Deployment Guide* for information about adding nodes to a cluster

To add Oracle RAC database instances to nodes that already have Oracle Clusterware installed, you must extend the Oracle RAC home that is on an existing node (`node1` in this procedure) of the cluster to the target nodes.

1. Navigate to the `Oracle_home\addnode` directory on `node1` and run the `addnode.bat` script using the following syntax, where `node2` is the name of the node you are adding:

    ```
    addnode.bat "CLUSTER_NEW_NODES={node2}"
    ```

    To run this command in silent mode:

    ```
    addNode.bat -silent "CLUSTER_NEW_NODES={node2}"
    ```

    For the Oracle home directory you use, if an Oracle home user was specified when the Oracle Database software was installed, then OUI requires the password for the Oracle home user. OUI checks the wallet (stored in the OCR) for the user and extracts the password from there. If the user information is not contained in the wallet, then the `addnode.bat` script generates an error unless you specify the `–promptPasswd` flag on the command line.

2. If you store your policy-managed database on Oracle Automatic Storage Management (Oracle ASM), **Oracle Managed Files** is enabled, and if there is space in a server pool for `node2`, then `crsd` adds the Oracle RAC database instance to `node2` and no further action is necessary. If Oracle Managed Files is not enabled, then you must manually add undo and redo logs.

    If there is no space in a server pool, then `node2` moves into the Free server pool. Use the `srvctl modify srvpool` command to increase the cardinality of the server pool to accommodate `node2`, after which time `node2` moves out of the Free server pool and into the modified server pool, and `crsd` adds the Oracle RAC database instance to `node2`.

3. If you have an administrator-managed database, then add a new instance on `node2` as described in "Adding Administrator-Managed Oracle RAC Database Instances to Target Nodes".

If you have a shared Oracle home that is shared using Oracle Automatic Storage Management Cluster File System (Oracle ACFS), then do the following to extend the Oracle database home to `node2`:

1. Start the Oracle ACFS resource on the new node by running the following command as `root` from the `Grid_home\bin` directory:

    ```
    srvctl start filesystem -d volume_device_name [-n node_name]
    ```

> **Note:**
>
> Make sure the Oracle ACFS resources, including Oracle ACFS registry resource and Oracle ACFS file system resource where the Oracle home is located, are online on the newly added node.

2. Run the following command as the user that installed Oracle RAC from the `Oracle_home\oui\bin` directory on the node you are adding to add the Oracle RAC database home:

```
setup.exe -attachHome ORACLE_HOME="ORACLE_HOME" LOCAL_NODE="node2"
ORACLE_HOME_NAME="home_name" -cfs
```

3. Navigate to the `Oracle_home\addnode` directory on `node1` and run the `addnode.bat` script as the user that installed Oracle RAC using the following syntax:

```
addnode.bat -noCopy "CLUSTER_NEW_NODES={node2}"
```

> **Note:**
>
> Use the `-noCopy` option because the Oracle home on the destination node is already fully populated with software.

If you have a shared Oracle home on a shared file system that *is not* Oracle ACFS, then you must first create a mount point for the Oracle RAC database home on the target node, mount and attach the Oracle RAC database home, and update the Oracle Inventory, as follows:

1. Run the `srvctl config database -db db_name` command on an existing node in the cluster to obtain the mount point information.

2. Mount the file system that hosts the Oracle RAC database home.

3. Run the following command as the user that installed Oracle RAC from the `Oracle_home\oui\bin` directory on the node you are adding to add the Oracle RAC database home:

```
setup.exe -attachHome ORACLE_HOME="ORACLE_HOME" "CLUSTER
_NODES={local_node_name}" LOCAL_NODE="node_name" ORACLE_HOME_NAME="home_name"
```

4. Update the Oracle Inventory as the user that installed Oracle RAC, as follows:

```
setup.exe -updateNodeList ORACLE_HOME=mount_point_path
"CLUSTER_NODES={node_list}"
```

In the preceding command, `node_list` refers to a list of all nodes where the Oracle RAC database home is installed, including the node you are adding.

> **Note:**
>
> Oracle recommends that you back up your voting disk and Oracle Cluster Registry (OCR) files after you complete the node addition process.

Adding Administrator-Managed Oracle RAC Database Instances to Target Nodes

You can use either Oracle Enterprise Manager or DBCA to add Oracle RAC database instances to the target nodes. To add a database instance to a target node with Oracle Enterprise Manager, see the *Oracle Database 2 Day + Real Application Clusters Guide* for complete information.

This section describes using DBCA to add Oracle RAC database instances.

These tools guide you through the following tasks:

- Creating a new database instance on each target node

- Creating and configuring high availability components

- Creating the Oracle Net configuration for a non-default listener from the Oracle home

- Starting the new instance

- Creating and starting services if you entered services information on the Services Configuration page

After adding the instances to the target nodes, you should perform any necessary service configuration procedures, as described in Workload Management with Dynamic Database Services.

Using DBCA in Interactive Mode to Add Database Instances to Target Nodes

To add a database instance to a target node using DBCA in interactive mode, perform the following steps:

1. Ensure that your existing nodes have the Oracle home environment variable set correctly.

2. Start DBCA by entering `dbca` at the system prompt from the `Oracle_home\bin` directory on an existing node.

    DBCA performs certain CVU checks while running. However, you can also run CVU from the command line to perform various verifications.

    > **See Also:**
    >
    > *Oracle Clusterware Administration and Deployment Guide* for more information about CVU

3. On the Database Operations page, select **Instance Management**, click **Next**, and DBCA displays the Instance Management page.

4. Select **Add Instance** and click **Next**. DBCA displays the List of Cluster Databases page that shows the databases and their current status, such as `ACTIVE` or `INACTIVE`.

5. From the List of Cluster Databases page, select the active Oracle RAC database to which you want to add an instance. Click **Next** and DBCA displays the List of Cluster Database Instances page showing the names of the existing instances for the Oracle RAC database that you selected.

6. Click **Next** to add a new instance and DBCA displays the Adding an Instance page.

7. On the Adding an Instance page, enter the instance name in the field at the top of this page if the instance name that DBCA provides does not match your existing instance naming scheme. Then select the new node name from the list.

Note:

If you installed the Oracle home with the Oracle Home User option, then DBCA prompts you for that password on this page.

---

8. Review the information on the Summary Page and click **Finish** to initiate instance addition operation. DBCA displays a progress dialog showing DBCA performing the instance addition operation.

Creating the OraMTS Service for Microsoft Transaction Server

Oracle Services for Microsoft Transaction Server (OraMTS) permit Oracle databases to be used as resource managers in Microsoft application-coordinated transactions. OraMTS acts as a proxy for the Oracle database to the Microsoft Distributed Transaction Coordinator (MSDTC). As a result, OraMTS provides client-side connection pooling and allows client components that leverage Oracle to participate in promotable and distributed transactions. In addition, OraMTS can operate with Oracle databases running on any operating system, given that the services themselves are run on Windows.

On releases earlier than Oracle Database 12*c*, the OraMTS service was created as part of a software-only installation. Starting with Oracle Database 12*c*, you must use a configuration tool to create this service.

Create the OraMTS service after adding a node or performing a software-only installation for Oracle RAC, as follows:

1. Open a command window.

2. Change directories to `%ORACLE_HOME%\bin`.

3. Run the OraMTSCtl utility to create the OraMTS Service, where *host_name* is a list of nodes on which the service should be created:

   ```
   C:\..bin> oramtsctl.exe -new -host host_name
   ```

See Also:

Oracle Services for Microsoft Transaction Server Developer's Guide for Microsoft Windows for more information about OraMTS, which allows Oracle databases to be used as resource managers in distributed transactions

---

Using DBCA in Silent Mode to Add Database Instances to Target Nodes

You can use DBCA in silent mode to add instances to nodes on which you have extended an Oracle Clusterware home and an Oracle Database home. Use the following syntax:

```
dbca -silent -addInstance -nodeList node_name -gdbName gdb_name
[-instanceName instance_name -sysDBAUserName sysdba -sysDBAPassword password]
```

> **See Also:**
>
> Table 10-1 for definitions of the variables used in the preceding syntax

Perform any necessary service configuration procedures, as described in Workload Management with Dynamic Database Services.

Deleting Oracle RAC from a Cluster Node

To remove Oracle RAC from a cluster node, you must delete the database instance and the Oracle RAC software before removing the node from the cluster.

> **Note:**
>
> If there are no database instances on the node you want to delete, then proceed to "Removing Oracle RAC".

This section includes the following procedures to delete nodes from clusters in an Oracle RAC environment:

- Deleting Instances from Oracle RAC Databases

- Removing Oracle RAC

- Deleting Nodes from the Cluster

Deleting Instances from Oracle RAC Databases

The procedures for deleting instances are different for policy-managed and administrator-managed databases. Deleting a policy-managed database instance involves reducing the size of the server pool in which the database instance resides. Deleting an administrator-managed database instance involves using DBCA to delete the database instance.

Deleting Policy-Managed Databases

To delete a policy-managed database, decrease the size of the server pool in which a database instance resides. This effectively removes the instance without having to remove the Oracle RAC software from the node or the node from the cluster.

For example, you can delete a policy-managed database by running the following commands on any node in the cluster:

```
srvctl stop instance -db db_unique_name -node node_name
srvctl relocate server -servers "server_name_list" -serverpool Free
```

The first command stops on the instance on a particular node and the second command moves the list of servers out of their current server pool and into the Free server pool.

> **See Also:**
>
> "Removing Oracle RAC" for information about removing the Oracle RAC software from a node

Deleting Instances from Administrator-Managed Databases

> **Note:**
>
> Before deleting an instance from an Oracle RAC database using SRVCTL, do the following:
>
> - If you have services configured, then relocate the services
>
> - Modify the services so that each service can run on one of the remaining instances
>
> - Ensure that the instance to be removed from an administrator-managed database is neither a preferred nor an available instance of any service

> **See Also:**
>
> "Administering Services with SRVCTL"

The procedure in this section explains how to use DBCA in interactive mode to delete an instance from an Oracle RAC database.

> **See Also:**
>
> *Oracle Database 2 Day + Real Application Clusters Guide* for information about how to delete a database instance from a target node with Oracle Enterprise Manager

Using DBCA in Interactive Mode to Delete Instances from Nodes

To delete an instance using DBCA in interactive mode, perform the following steps:

1. Verify there is a current backup of OCR.

    Run the `ocrconfig -showbackup` command to ensure there is a valid backup.

2. Start DBCA.

    Start DBCA on a node *other than* the node that hosts the instance that you want to delete. The database and the instance that you plan to delete should continue to be started and running during this step.

3. On the DBCA Operations page, select **Instance Management**, click **Next**, and DBCA displays the Instance Management page.

4. On the Instance Management page, select **Delete Instance**, click **Next**, and DBCA displays the List of Cluster Databases page.

5. Select an Oracle RAC database from which to delete an instance. Click **Next** and DBCA displays the List of Cluster Database Instances page. The List of Cluster Database Instances page shows the instances that are associated with the Oracle RAC database that you selected and the status of each instance.

6. On the List of Cluster Databases page, select the Oracle RAC database from which to delete the instance, as follows:

    a. On the List of Cluster Database Instances page, DBCA displays the instances that are associated with the Oracle RAC database that you selected and the status of each instance. Select the cluster database from which you will delete the instance. Click **Finish**.

    b. Click **OK** on the Confirmation dialog to proceed to delete the instance.

    c. Click **OK** on the next Confirmation dialog to delete the instance and related Optimal Flexible Architecture (OFA) directory structure.

    DBCA displays a progress dialog showing that DBCA is deleting the instance. During this operation, DBCA removes the instance and the instance's Oracle Net configuration.

    Click **No** and exit DBCA or click **Yes** to perform another operation. If you click **Yes**, then DBCA displays the Operations page.

7. Verify that the dropped instance's redo thread has been removed using SQL*Plus to query the V$LOG view from an existing instance. If the redo thread is not disabled, then disable the thread. For example:

```
SQL> ALTER DATABASE DISABLE THREAD 2;
```

8. Verify that the instance has been removed from OCR by running the following command, where *db_unique_name* is the name of the database:

```
srvctl config database -db db_unique_name
```

9. If you are deleting more than one node, then repeat these steps to delete the instances from all the nodes that you are going to delete.

Using DBCA in Silent Mode to Delete Instances from Nodes

You can use DBCA in silent mode to delete a database instance from a node.

Run the following command, where the variables are the same as those shown in Table 10-1 for the DBCA command to add an instance. Provide a node name only if you are deleting an instance from a node other than the one on where DBCA is running as shown in the following example where *password* is the SYSDBA password:

```
dbca -silent -deleteInstance [-nodeList node_name] -gdbName gdb_name
-instanceName instance_name [-sysDBAUserName sysdba] [-sysDBAPassword password]
```

At this point, you have accomplished the following:

- Deregistered the selected instance from its associated Oracle Net Services listeners

- Deleted the selected database instance from the instance's configured node

- Removed the Oracle Net configuration

- Deleted the Oracle Flexible Architecture directory structure from the instance's configured node.

Removing Oracle RAC

This procedure removes the Oracle RAC software from the node you are deleting from the cluster and updates inventories on the remaining nodes.

1. If there is a listener in the Oracle RAC home on the node you are deleting, then you must disable and stop it before deleting the Oracle RAC software. Run the following commands on any node in the cluster, specifying the name of the listener and the name of the node you are deleting:

   ```
   srvctl disable listener -listener listener_name -node name_of_node_to_delete
   srvctl stop listener -listener listener_name -node name_of_node_to_delete
   ```

2. Run the following command from `%ORACLE_HOME%\oui\bin` on the node that you are deleting to update the inventory on that node:

   ```
   setup.exe -updateNodeList ORACLE_HOME=Oracle_home_location
   "CLUSTER_NODES={name_of_node_to_delete}" -local
   ```

3. Deinstall the Oracle home from the node that you are deleting by running the following command from the `Oracle_home\deinstall` directory:

   ```
   deinstall -local
   ```

4. Run the following command from the `Oracle_home\oui\bin` directory on any one of the remaining nodes in the cluster to update the inventories of those nodes, specifying a comma-delimited list of remaining node names:

   ```
   setup.exe -updateNodeList ORACLE_HOME=Oracle_home_location
   "CLUSTER_NODES={remaining_node_list}"
   ```

If you have a shared Oracle RAC home, then append the `-cfs` option to the command example in this step and provide a complete path location for the cluster file system.

Deleting Nodes from the Cluster

After you delete the instance, you can begin the process of deleting the node from the cluster. You accomplish this by running scripts on the node you want to delete to remove the Oracle Clusterware installation and you run scripts on the remaining nodes to update the node list.

See Also:

- *Oracle Clusterware Administration and Deployment Guide* for information about deleting nodes from the cluster

- *Oracle Grid Infrastructure Installation Guide for Microsoft Windows x64 (64-Bit)* for more information about removing Oracle Clusterware and Oracle ASM

12

Design and Deployment Techniques

This chapter briefly describes database design and deployment techniques for Oracle Real Application Clusters (Oracle RAC) environments. It also describes considerations for high availability and provides general guidelines for various Oracle RAC deployments.

This chapter includes the following topics:

- Deploying Oracle RAC for High Availability

- General Design Considerations for Oracle RAC

- General Database Deployment Topics for Oracle RAC

Deploying Oracle RAC for High Availability

Many customers implement Oracle RAC to provide high availability for their Oracle Database applications. For true high availability, you must make the entire infrastructure of the application highly available. This requires detailed planning to ensure there are no single points of failure throughout the infrastructure. Even though Oracle RAC makes your database highly available, if a critical application becomes unavailable, then your business can be negatively affected. For example, if you choose to use the Lightweight Directory Access Protocol (LDAP) for authentication, then you must make the LDAP server highly available. If the database is up but the users cannot connect to the database because the LDAP server is not accessible, then the entire system appears to be down to your users.

This section includes the following topics:

- About Designing a High Availability System

- Best Practices for Deploying Oracle RAC in a High Availability Environment

- Consolidating Multiple Applications in a Database or Multiple Databases in a Cluster

- Scalability of Oracle RAC

About Designing a High Availability System

For mission critical systems, you must be able to perform failover and recovery, and your environment must be resilient to all types of failures. To reach these goals, start by defining service level requirements for your business. The requirements should include definitions of maximum transaction response time and recovery expectations for failures within the datacenter (such as for node failure) or for disaster recovery (if the entire data center fails). Typically, the service level objective is a target response time for work, regardless of failures. Determine the recovery time for each redundant component. Even though you may have hardware components that are running in an

active/active mode, do not assume that if one component fails the other hardware components can remain operational while the faulty components are being repaired. Also, when components are running in active/passive mode, perform regular tests to validate the failover time. For example, recovery times for storage channels can take minutes. Ensure that the outage times are within your business' service level agreements, and where they are not, work with the hardware vendor to tune the configuration and settings.

When deploying mission critical systems, the testing should include functional testing, destructive testing, and performance testing. Destructive testing includes the injection of various faults in the system to test the recovery and to make sure it satisfies the service level requirements. Destructive testing also allows the creation of operational procedures for the production system.

To help you design and implement a mission critical or highly available system, Oracle provides a range of solutions for every organization regardless of size. Small workgroups and global enterprises alike are able to extend the reach of their critical business applications. With Oracle and the Internet, applications and their data are now reliably accessible everywhere, at any time. The Oracle Maximum Availability Architecture (MAA) is the Oracle best practices blueprint that is based on proven Oracle high availability technologies and recommendations. The goal of the MAA is to remove the complexity in designing an optimal high availability architecture.

> **See Also:**
>
> - *Oracle Database High Availability Overview*
>
> - Oracle Maximum Availability Architecture (MAA) Web site at
>
>   http://www.oracle.com/technology/deploy/availability/htdocs/
>   maa.htm

Best Practices for Deploying Oracle RAC in a High Availability Environment

Applications can take advantage of many Oracle Database, Oracle Clusterware, and Oracle RAC features and capabilities to minimize or mask any failure in the Oracle RAC environment. For example, you can:

- Remove TCP/IP timeout waits by using the VIP address to connect to the database.

- Create detailed operational procedures and ensure you have the appropriate support contracts in place to match defined service levels for all components in the infrastructure.

- Take advantage of the Oracle RAC Automatic Workload Management features such as connect time failover, Fast Connection Failover, Fast Application Notification, and the Load Balancing Advisory.

> **See Also:**
>
> Workload Management with Dynamic Database Services for more details.

- Place voting disks on separate volume groups to mitigate outages due to slow I/O throughput. To survive the failure of x voting devices, configure $2x + 1$ mirrors.

- Use Oracle Database Quality of Service Management (Oracle Database QoS Management) to monitor your system and detect performance bottlenecks.

- Place OCR with I/O service times in the order of 2 milliseconds (ms) or less.

- Tune database recovery using the `FAST_START_MTTR_TARGET` initialization parameter.

- Use Oracle Automatic Storage Management (Oracle ASM) to manage database storage.

- Ensure that strong change control procedures are in place.

- Check the surrounding infrastructure for high availability and resiliency, such as LDAP, NIS, and DNS. These entities affect the availability of your Oracle RAC database. If possible, perform a local backup procedure routinely.

- Use Oracle Enterprise Manager to administer your entire Oracle RAC environment, not just the Oracle RAC database. Use Oracle Enterprise Manager to create and modify services, and to start and stop the cluster database instances and the cluster database.

  > **See Also:**
  >
  > *Oracle Database 2 Day + Real Application Clusters Guide* for more information about using Oracle Enterprise Manager in an Oracle RAC environment

- Use Recovery Manager (RMAN) to back up, restore, and recover data files, control files, server parameter files (SPFILEs) and archived redo log files. You can use RMAN with a media manager to back up files to external storage. You can also configure parallelism when backing up or recovering Oracle RAC databases. In Oracle RAC, RMAN channels can be dynamically allocated across all of the Oracle RAC instances. Channel failover enables failed operations on one node to continue on another node. You can start RMAN from Oracle Enterprise Manager Backup Manager or from the command line.

  > **See Also:**
  >
  > Configuring Recovery Manager and Archiving for more information about using RMAN

- If you use sequence numbers, then always use `CACHE` with the `NOORDER` option for optimal performance in sequence number generation. With the `CACHE` option, however, you may have gaps in the sequence numbers. If your environment cannot tolerate sequence number gaps, then use the `NOCACHE` option or consider pre-generating the sequence numbers. If your application requires sequence number ordering but can tolerate gaps, then use `CACHE` and `ORDER` to cache and order sequence numbers in Oracle RAC. If your application requires ordered sequence numbers without gaps, then use `NOCACHE` and `ORDER`. The `NOCACHE` and `ORDER` combination has the most negative effect on performance compared to other caching and ordering combinations.

> **Note:**
>
> If your environment cannot tolerate sequence number gaps, then consider pre-generating the sequence numbers or use the `ORDER` and `CACHE` options.

- If you use indexes, then consider alternatives, such as reverse key indexes to optimize index performance. Reverse key indexes are especially helpful if you have frequent inserts to one side of an index, such as indexes that are based on insert date.

Consolidating Multiple Applications in a Database or Multiple Databases in a Cluster

Many people want to consolidate multiple applications in a single database or consolidate multiple databases in a single cluster. Oracle Clusterware and Oracle RAC support both types of consolidation.

Creating a cluster with a single pool of storage managed by Oracle ASM provides the infrastructure to manage multiple databases whether they are single instance databases or Oracle RAC databases.

Managing Capacity During Consolidation

With Oracle RAC databases, you can adjust the number of instances and which nodes run instances for a given database, based on workload requirements. Features such as cluster-managed services allow you to manage multiple workloads on a single database or across multiple databases.

It is important to properly manage the capacity in the cluster when adding work. The processes that manage the cluster—including processes both from Oracle Clusterware and the database—must be able to obtain CPU resources in a timely fashion and must be given higher priority in the system. Oracle Database Quality of Service Management (Oracle Database QoS Management) can assist consolidating multiple applications in a cluster or database by dynamically allocating CPU resources to meet performance objectives. You can also use cluster configuration policies to manage resources at the cluster level.

> **See Also:**
>
> *Oracle Database Quality of Service Management User's Guide* for more information

Managing the Global Cache Service Processes During Consolidation

Oracle recommends that the number of real time Global Cache Service Processes (LMS*n*) on a server is less than or equal to the number of processors. (Note that this is the number of recognized CPUs that includes cores. For example, a dual-core CPU is considered to be two CPUs.) It is important that you load test your system when adding instances on a node to ensure that you have enough capacity to support the workload.

If you are consolidating many small databases into a cluster, you may want to reduce the number of LMS*n* created by the Oracle RAC instance. By default, Oracle Database calculates the number of processes based on the number of CPUs it finds on the server. This calculation may result in more LMS*n* processes than is needed for the Oracle RAC instance. One LMS process may be sufficient for up to 4 CPUs.To reduce the number of LMS*n* processes, set the `GC_SERVER_PROCESSES` initialization parameter

minimally to a value of 1. Add a process for every four CPUs needed by the application. In general, it is better to have few busy LMS*n* processes. Oracle Database calculates the number of processes when the instance is started, and you must restart the instance to change the value.

Using a Database Cloud for Consolidation

A database cloud is a set of databases integrated by the Global Data Services framework into a single virtual server that offers one or more global services, while ensuring high performance, availability and optimal utilization of resources. Global Data Services manages these virtualized resources with minimum administration overhead, and allows the database cloud to quickly scale to handle additional client requests. The databases that constitute a cloud can be globally distributed, and clients can connect to the database cloud by simply specifying a service name, without needing to know anything about the components and topology of the cloud.

A database cloud can be comprised of multiple database pools. A **database pool** is a set of databases within a database cloud that provide a unique set of global services and belong to a certain administrative domain. Partitioning of cloud databases into multiple pools simplifies service management and provides higher security by allowing each pool to be administered by a different administrator. A database cloud can span multiple geographic regions. A **region** is a logical boundary that contains database clients and servers that are considered to be close to each other. Usually a region corresponds to a data center, but multiple data centers can be in the same region if the network latencies between them satisfy the service-level agreements of the applications accessing these data centers.

Global services enable you to integrate locally and globally distributed, loosely coupled, heterogeneous databases into a scalable and highly available private database cloud. This database cloud can be shared by clients around the globe. Using a private database cloud provides optimal utilization of available resources and simplifies the provisioning of database services.

> **See Also:**
>
> *Oracle Database Global Data Services Concepts and Administration Guide* for more information about global services

Scalability of Oracle RAC

Oracle RAC provides concurrent, transactionally consistent access to a single copy of the data from multiple systems. It provides scalability beyond the capacity of a single server. If your application scales transparently on symmetric multiprocessing (SMP) servers, then it is realistic to expect the application to scale well on Oracle RAC, without the need to make changes to the application code.

Traditionally, when a database server runs out of capacity, it is replaced with a new larger server. As servers grow in capacity, they become more expensive. However, for Oracle RAC databases, you have alternatives for increasing the capacity:

- You can migrate applications that traditionally run on large SMP servers to run on clusters of small servers.

- You can maintain the investment in the current hardware and add a new server to the cluster (or create or add a new cluster) to increase the capacity.

Adding servers to a cluster with Oracle Clusterware and Oracle RAC does not require an outage. As soon as the new instance is started, the application can take advantage of the extra capacity.

All servers in the cluster must run the same operating system and same version of Oracle Database but the servers do not have to have the same capacity. With Oracle RAC, you can build a cluster that fits your needs, whether the cluster is made up of servers where each server is a two-CPU commodity server or clusters where the servers have 32 or 64 CPUs in each server. The Oracle parallel execution feature allows a single SQL statement to be divided up into multiple processes, where each process completes a subset of work. In an Oracle RAC environment, you can define the parallel processes to run only on the instance where the user is connected or to run across multiple instances in the cluster.

> **See Also:**
>
> - Cloning Oracle RAC to Nodes in a New Cluster
>
> - Using Cloning to Extend Oracle RAC to Nodes in the Same Cluster
>
> - Adding and Deleting Oracle RAC from Nodes on Linux and UNIX Systems
>
> - Adding and Deleting Oracle RAC from Nodes on Windows Systems

General Design Considerations for Oracle RAC

This section briefly describes database design and deployment techniques for Oracle RAC environments. It also describes considerations for high availability and provides general guidelines for various Oracle RAC deployments.

Consider performing the following steps during the design and development of applications that you are deploying on an Oracle RAC database:

1. Tune the design and the application

2. Tune the memory and I/O

3. Tune contention

4. Tune the operating system

> **Note:**
>
> If an application does not scale on an SMP system, then moving the application to an Oracle RAC database cannot improve performance.

Consider using hash partitioning for insert-intensive online transaction processing (OLTP) applications. Hash partitioning:

- Reduces contention on concurrent inserts into a single database structure

- Affects sequence-based indexes when indexes are locally partitioned with a table and tables are partitioned on sequence-based keys

- Is transparent to the application

If you use hash partitioning for tables and indexes for OLTP environments, then you can greatly improve performance in your Oracle RAC database. Note that you cannot use index range scans on an index with hash partitioning.

General Database Deployment Topics for Oracle RAC

This section describes considerations when deploying Oracle RAC databases. Oracle RAC database performance is not compromised if you do not employ these techniques. If you have an effective noncluster design, then your application will run well on an Oracle RAC database.

This section includes the following topics:

- Tablespace Use in Oracle RAC

- Object Creation and Performance in Oracle RAC

- Node Addition and Deletion and the SYSAUX Tablespace in Oracle RAC

- Distributed Transactions and Oracle RAC

- Deploying OLTP Applications in Oracle RAC

- Flexible Implementation with Cache Fusion

- Deploying Data Warehouse Applications with Oracle RAC

- Data Security Considerations in Oracle RAC

Tablespace Use in Oracle RAC

In addition to using locally managed tablespaces, you can further simplify space administration by using automatic segment space management (ASSM) and automatic undo management.

ASSM distributes instance workloads among each instance's subset of blocks for inserts. This improves Oracle RAC performance because it minimizes block transfers. To deploy automatic undo management in an Oracle RAC environment, each instance must have its own undo tablespace.

Object Creation and Performance in Oracle RAC

As a general rule, only use DDL statements for maintenance tasks and avoid executing DDL statements during peak system operation periods. In most systems, the amount of new object creation and other DDL statements should be limited. Just as in noncluster Oracle databases, excessive object creation and deletion can increase performance overhead.

Node Addition and Deletion and the SYSAUX Tablespace in Oracle RAC

If you add nodes to your Oracle RAC database environment, then you may need to increase the size of the SYSAUX tablespace. Conversely, if you remove nodes from your **cluster database**, then you may be able to reduce the size of your SYSAUX tablespace.

> **See Also:**
>
> Your platform-specific Oracle RAC installation guide for guidelines about sizing the SYSAUX tablespace for multiple instances

Distributed Transactions and Oracle RAC

If you are running XA Transactions in an Oracle RAC environment and the performance is poor, direct all branches of a tightly coupled distributed transaction to the same instance by creating multiple Oracle Distributed Transaction Processing (DTP) services, with one or more on each Oracle RAC instance.

Each DTP service is a singleton service that is available on one and only one Oracle RAC instance. All access to the database server for distributed transaction processing must be done by way of the DTP services. Ensure that all of the branches of a single global distributed transaction use the same DTP service. In other words, a network connection descriptor, such as a TNS name, a JDBC URL, and so on, must use a DTP service to support distributed transaction processing.

> **See Also:**
>
> - "Distributed Transaction Processing in Oracle RAC" for more details about enabling services and distributed transactions
>
> - *Oracle Database Development Guide* for more information about distributed transactions

Deploying OLTP Applications in Oracle RAC

Cache Fusion makes Oracle RAC databases the optimal deployment servers for online transaction processing (OLTP) applications. This is because these types of applications require:

- High availability if there are failures

- Scalability to accommodate increased system demands

- Load balancing according to demand fluctuations

The high availability features of Oracle Database and Oracle RAC can re-distribute and load balance workloads to surviving instances without interrupting processing. Oracle RAC also provides excellent scalability so that if you add or replace a node, then Oracle Database re-masters resources and re-distributes processing loads.

Flexible Implementation with Cache Fusion

To accommodate the frequently changing workloads of online transaction processing systems, Oracle RAC remains flexible and dynamic despite changes in system load and system availability. Oracle RAC addresses a wide range of service levels that, for example, fluctuate due to:

- Varying user demands

- Peak scalability issues like trading storms (bursts of high volumes of transactions)

- Varying availability of system resources

Deploying Data Warehouse Applications with Oracle RAC

This section discusses how to deploy data warehouse systems in Oracle RAC environments by briefly describing the data warehouse features available in shared disk architectures.

This section includes the following topics:

- Speed-Up for Data Warehouse Applications on Oracle RAC

- Parallel Execution in Data Warehouse Systems and Oracle RAC

Speed-Up for Data Warehouse Applications on Oracle RAC

Oracle RAC is ideal for data warehouse applications because it augments the noncluster benefits of Oracle Database. Oracle RAC does this by maximizing the processing available on all of the nodes that belong to an Oracle RAC database to provide speed-up for data warehouse systems.

The query optimizer considers parallel execution when determining the optimal execution plans. The default cost model for the query optimizer is **CPU+I/O** and the cost unit is **time**. In Oracle RAC, the query optimizer dynamically computes intelligent defaults for parallelism based on the number of processors in the nodes of the cluster. An evaluation of the costs of alternative access paths, table scans versus indexed access, for example, takes into account the degree of parallelism available for the operation. This results in Oracle Database selecting the execution plans that are optimized for your Oracle RAC configuration.

Parallel Execution in Data Warehouse Systems and Oracle RAC

Parallel execution uses multiple processes to run SQL statements on one or more CPUs and is available on both noncluster Oracle databases and Oracle RAC databases.

Oracle RAC takes full advantage of parallel execution by distributing parallel processing across all available instances. The number of processes that can participate in parallel operations depends on the degree of parallelism assigned to each table or index.

> **See Also:**
>
> - *Oracle Database Performance Tuning Guide* for more information about the query optimizer
>
> - *Oracle Database Concepts* for more information about parallel execution

Data Security Considerations in Oracle RAC

This section describes the following two Oracle RAC security considerations:

- Transparent Data Encryption and Keystores

- Windows Firewall Considerations

Transparent Data Encryption and Keystores

Oracle Database enables Oracle RAC nodes to share the **keystore** (wallet). This eliminates the need to manually copy and synchronize the keystore across all nodes. Oracle recommends that you create the keystore on a shared file system. This allows all instances to access the same shared keystore.

Oracle RAC uses keystores in the following ways:

1. Any keystore operation, like opening or closing the keystore, performed on any one Oracle RAC instance is applicable for all other Oracle RAC instances. This means that when you open and close the keystore for one instance, then it opens and closes the keystore for all Oracle RAC instances.

2. When using a shared file system, ensure that the `ENCRYPTION_WALLET_LOCATION` parameter for all Oracle RAC instances points to the same shared keystore location. The security administrator must also ensure security of the shared keystore by assigning appropriate directory permissions.

   > **Note:**
   >
   > If Oracle Automatic Storage Management Cluster File System (Oracle ACFS) is available for your operating system, then Oracle recommends that you store the keystore in Oracle ACFS. If you do not have Oracle ACFS in Oracle ASM, then use the Oracle ASM Configuration Assistant (ASMCA) to create it. You must add the mount point to the `sqlnet.ora` file in each instance, as follows:
   >
   > ```
   > ENCRYPTION_WALLET_LOCATION=
   >   (SOURCE = (METHOD = FILE)
   >     (METHOD_DATA =
   >       (DIRECTORY = /opt/oracle/acfsmounts/data_keystore)))
   > ```
   >
   > This file system is mounted automatically when the instances start. Opening and closing the keystore, and commands to set or rekey and rotate the TDE master encryption key, are synchronized between all nodes.

3. A master key rekey performed on one instance is applicable for all instances. When a new Oracle RAC node comes up, it is aware of the current keystore open or close status.

4. Do not issue any keystore `ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN` or `CLOSE` SQL statements while setting up or changing the master key.

Deployments where shared storage does not exist for the keystore require that each Oracle RAC node maintain a local keystore. After you create and provision a keystore on a single node, you must copy the keystore and make it available to all of the other nodes, as follows:

- For systems using Transparent Data Encryption with encrypted keystores, you can use any standard file transport protocol, though Oracle recommends using a secured file transport.

- For systems using Transparent Data Encryption with auto-login keystores, file transport through a secured channel is recommended.

To specify the directory in which the keystore must reside, set the or `ENCRYPTION_WALLET_LOCATION` parameter in the `sqlnet.ora` file. The local

copies of the keystore need not be synchronized for the duration of Transparent Data Encryption usage until the server key is re-keyed though the `ADMINISTER KEY MANAGEMENT SET KEY` SQL statement. Each time you issue the `ADMINISTER KEY MANAGEMENT SET KEY` statement on a database instance, you must again copy the keystore residing on that node and make it available to all of the other nodes. Then, you must close and reopen the keystore on each of the nodes. To avoid unnecessary administrative overhead, reserve re-keying for exceptional cases where you believe that the server master key may have been compromised and that not re-keying it could cause a serious security problem.

> **See Also:**
>
> *Oracle Database Advanced Security Guide* for more information about creating and provisioning a keystore

Windows Firewall Considerations

By default, all installations of Windows Server 2003 Service Pack 1 and higher enable the Windows Firewall to block virtually all TCP network ports to incoming connections. As a result, any Oracle products that listen for incoming connections on a TCP port will not receive any of those connection requests, and the clients making those connections will report errors.

Depending upon which Oracle products you install and how they are used, you may need to perform additional Windows post-installation configuration tasks so that the Firewall products are functional on Windows Server 2003.

> **See Also:**
>
> *Oracle Real Application Clusters Installation Guide for Microsoft Windows x64 (64-Bit)* for more information about Oracle RAC executables requiring Windows Firewall exceptions

13

Monitoring Performance

This chapter describes how to monitor and tune Oracle Real Application Clusters (Oracle RAC) performance.

This chapter includes the following topics:

- Overview of Monitoring and Tuning Oracle RAC Databases
- Verifying the Interconnect Settings for Oracle RAC
- Influencing Interconnect Processing
- Performance Views in Oracle RAC
- Creating Oracle RAC Data Dictionary Views with CATCLUST.SQL
- Oracle RAC Performance Statistics
- Automatic Workload Repository in Oracle RAC Environments
- Active Session History Reports for Oracle RAC
- Monitoring Oracle RAC Statistics and Wait Events

Overview of Monitoring and Tuning Oracle RAC Databases

This section includes the following topics:

- Monitoring Oracle RAC and Oracle Clusterware
- Tuning Oracle RAC Databases

See Also:

- *Oracle Database 2 Day + Real Application Clusters Guide*
- Oracle Enterprise Manager online help
- *Oracle Database 2 Day DBA* for more information about basic database tuning
- *Oracle Database 2 Day + Performance Tuning Guide* for more information about general performance tuning
- *Oracle Clusterware Administration and Deployment Guide* for more information about diagnosing problems for Oracle Clusterware components

---

Monitoring Oracle RAC and Oracle Clusterware

Using Oracle Enterprise Manager is the preferred method for monitoring Oracle RAC and Oracle Clusterware. Oracle Enterprise Manager is an Oracle Web-based integrated management solution for monitoring and administering your computing environment. From any location where you can access a web browser, you can manage Oracle RAC databases, application servers, host computers, and Web applications, in addition to related hardware and software. For example, you can monitor your Oracle RAC database performance from your office, home, or a remote site, if you have access to a Web browser.

Oracle Enterprise Manager Cloud Control is cluster-aware and provides a central console to manage your cluster database. From the Cluster Database Home page, you can do all of the following:

- View the overall system status, such as the number of nodes in the cluster and their current status. This high-level view capability means that you do not have to access each individual database instance for details if you just want to see inclusive, aggregated information.

- View alert messages aggregated across all the instances with lists for the source of each alert message. An **alert message** is an indicator that signifies that a particular metric condition has been encountered. A **metric** is a unit of measurement used to report the system's conditions.

- Review issues that are affecting the entire cluster and those issues that are affecting individual instances.

- Monitor cluster cache coherency statistics to help you identify processing trends and optimize performance for your Oracle RAC environment. Cache coherency statistics measure how well the data in caches on multiple instances is synchronized. If the data caches are completely synchronized with each other, then reading a memory location from the cache on any instance will return the most recent data written to that location from any cache on any instance.

Oracle Enterprise Manager accumulates data over specified periods of time, called collection-based data. Oracle Enterprise Manager also provides current data, called real-time data.

Oracle Database 2 Day + Real Application Clusters Guide provides complete information about monitoring performance with Oracle Enterprise Manager, including:

- Automatic Database Diagnostic Monitor and Oracle RAC Performance

- The Cluster Database Home Page

- The Interconnects Page

- The Cluster Database Performance Page

The Cluster Database Home Page

You can use Oracle Enterprise Manager with a client browser to monitor the status of both Oracle Clusterware and the Oracle RAC environments. Monitoring can include such things as:

- Notification if there are any VIP relocations

- Status of the Oracle Clusterware on each node of the cluster using information obtained through the Cluster Verification Utility (`cluvfy`)

- Notification if node applications (`nodeapps`) start or stop

- Notification of issues in the Oracle Clusterware alert log for OCR, voting disk issues (if any), and node evictions

The Cluster Database Home page is similar to a noncluster Database Home page. However, on the Cluster Database Home page, Oracle Enterprise Manager displays the system state and availability. This includes a summary about alert messages and job activity, and links to all the database and Oracle Automatic Storage Management (Oracle ASM) instances. For example, you can track problems with services on the cluster including when a service is not running on all of the preferred instances or when a service response time threshold is not being met.

The Interconnects Page

You can use the Oracle Enterprise Manager Interconnects page to monitor the Oracle Clusterware environment. The Interconnects page shows the public and private interfaces on the cluster and the load contributed by database instances on the interconnect, including:

- Overall throughput across the private interconnect

- Notification if a database instance is using public interface due to misconfiguration

- Throughput and errors (if any) on the interconnect

- Throughput contributed by individual instances on the interconnect

All of this information is also available as collections that have a historic view, which is useful with cluster cache coherency, such as when diagnosing problems related to cluster wait events. You can access the Interconnects page by clicking the Interconnect tab on the Cluster Database home page or clicking the Interconnect Alerts link under Diagnostic Findings on the Oracle RAC database home page.

The Cluster Database Performance Page

The Oracle Enterprise Manager Cluster Database Performance page provides a quick glimpse of the performance statistics for a database. Statistics are rolled up across all the instances in the cluster database in charts. Using the links next to the charts, you can get more specific information and perform any of the following tasks:

- Identify the causes of performance issues.

- Decide whether resources need to be added or redistributed.

- Tune your SQL plan and schema for better optimization.

- Resolve performance issues

The charts on the Cluster Database Performance page include the following:

- **Chart for Cluster Host Load Average**: The Cluster Host Load Average chart in the Cluster Database Performance page shows potential problems that are outside the database. The chart shows maximum, average, and minimum load values for available nodes in the cluster for the previous hour.

- **Chart for Global Cache Block Access Latency**: Each cluster database instance has its own buffer cache in its System Global Area (SGA). Using Cache Fusion, Oracle RAC environments logically combine each instance's buffer cache to enable the database instances to process data as if the data resided on a logically combined, single cache.

- **Chart for Average Active Sessions**: The Average Active Sessions chart in the Cluster Database Performance page shows potential problems inside the database. Categories, called wait classes, show how much of the database is using a resource, such as CPU or disk I/O. Comparing CPU time to wait time helps to determine how much of the response time is consumed with useful work rather than waiting for resources that are potentially held by other processes.

- **Chart for Database Throughput**: The Database Throughput charts summarize any resource contention that appears in the Average Active Sessions chart, and also show how much work the database is performing on behalf of the users or applications. The Per Second view shows the number of transactions compared to the number of logons, and the amount of physical reads compared to the redo size per second. The Per Transaction view shows the amount of physical reads compared to the redo size per transaction. Logons is the number of users that are logged on to the database.

In addition, the **Top Activity** drill down menu on the Cluster Database Performance page enables you to see the activity by wait events, services, and instances. Plus, you can see the details about SQL/sessions by going to a prior point in time by moving the slider on the chart.

The **Cluster Database Performance** page provides a quick glimpse of the performance statistics for an Oracle RAC database. Statistics are rolled up across all of the instances in the cluster database so that users can identify performance issues without going through all the instances. To help triage the performance issues related to services, Oracle Enterprise Manager aggregates the activity data at the following levels:

- Aggregate by waits

  All the activity data is presented in 12 categories: CPU, Scheduler, User I/O, System I/O, Concurrency, Application, Commit, Configuration, Administrative, Network, Cluster and Other. The data presented is rolled up from all of the running instances.

- Aggregate by services

  All the activity data is rolled up for each service. When the activity data is presented in this way, it is easy to identify which service is most active, and needs more analysis.

- Aggregate by instances

  As a similar effort, the activity data is rolled up for each instance, if services are not the interested ones.

The aggregates are provided on the pages where the activity data is presented including: Database Performance Page, Top Activity Page, Wait Details Page and Service Details Page.

See Also:

Oracle Database 2 Day + Real Application Clusters Guide

Tuning Oracle RAC Databases

All noncluster tuning practices for Oracle Database apply to Oracle RAC databases. Therefore, implement the noncluster tuning methodologies described in the *Oracle Database 2 Day + Performance Tuning Guide* and the *Oracle Database Performance Tuning Guide*.

Verifying the Interconnect Settings for Oracle RAC

The interconnect and internode communication protocols can affect **Cache Fusion** performance. In addition, the interconnect bandwidth, its latency, and the efficiency of the IPC protocol determine the speed with which Cache Fusion processes block transfers.

To verify the interconnect settings of the Oracle RAC database instance to which you are connected, query the V$CLUSTER_INTERCONNECTS and V$CONFIGURED_INTERCONNECTS views. For example:

Example 13-1   Verify Interconnect Settings with V$CLUSTER_INTERCONNECTS

```
SQL> SELECT * FROM V$CLUSTER_INTERCONNECTS;

NAME            IP_ADDRESS      IS_PUBLIC SOURCE
--------------- --------------- ---       -------------------------------
eth2            10.137.20.181   NO        Oracle Cluster Repository
```

> **Note:**
>
> You can query the GV$CLUSTER_INTERCONNECTS view to display the entries for all of the instances in the cluster.

Example 13-2   Verify Interconnect Settings with V$CONFIGURED_INTERCONNECTS

```
SQL> SELECT * FROM V$CONFIGURED_INTERCONNECTS;

NAME            IP_ADDRESS      IS_PUBLIC  SOURCE
--------------- --------------- ---        -------------------------------
eth2            10.137.20.181   NO         Oracle Cluster Repository
eth0            10.137.8.225    YES        Oracle Cluster Repository
```

Influencing Interconnect Processing

Once your interconnect is operative, you cannot significantly influence its performance. However, you can influence an interconnect protocol's efficiency by adjusting the interprocess communication (IPC) buffer sizes.

The Oracle Cluster Registry (OCR) stores your system's interconnect information. Use the Oracle Interface Configuration (OIFCFG) command-line utility oifcfg getif command or the OCRDUMP utility to identify the interconnect that you are using. You can then change the interconnect that you are using by running an OIFCFG command.

> **See Also:**
>
> - Your vendor-specific interconnect documentation for more information about adjusting IPC buffer sizes
>
> - *Oracle Clusterware Administration and Deployment Guide* for more information about enabling and using the OIFCFG and OCRDUMP utilities

Although you *rarely* need to set the CLUSTER_INTERCONNECTS parameter, you can use it to assign a private network IP address or NIC as in the following example:

```
CLUSTER_INTERCONNECTS=10.0.0.1
```

If you are using an operating system-specific vendor IPC protocol, then the trace information may not reveal the IP address.

> **Note:**
>
> You can also use OIFCFG command to assign private network or private IP addresses.

> **See Also:**
>
> *Oracle Database Reference* for more information about the CLUSTER_INTERCONNECTS parameter

Performance Views in Oracle RAC

Each instance has a set of instance-specific views, which are prefixed with V$. You can also query global dynamic performance views to retrieve performance information from all of the qualified instances. Global dynamic performance view names are prefixed with GV$.

Querying a GV$ view retrieves the V$ view information from all qualified instances. In addition to the V$ information, each GV$ view contains an extra column named INST_ID of data type NUMBER. The INST_ID column displays the instance number from which the associated V$ view information was obtained.

You can use the INST_ID column as a filter to retrieve V$ information from a subset of available instances. For example, the following query retrieves the information from the V$LOCK view for instances 2 and 5:

```
SQL> SELECT * FROM GV$LOCK WHERE INST_ID = 2 OR INST_ID = 5;
```

> **See Also:**
>
> *Oracle Database Reference* for complete descriptions of GV$ views

Creating Oracle RAC Data Dictionary Views with CATCLUST.SQL

If you did not create your Oracle RAC database with the Database Configuration Assistant (DBCA), then you must run the CATCLUST.SQL script to create views and tables related to Oracle RAC. You must have SYSDBA privileges to run this script.

See Also:

Oracle Real Application Clusters Installation Guide for your platform for more information about creating your Oracle RAC database

---

Oracle RAC Performance Statistics

Oracle RAC statistics appear as message request counters or as timed statistics. Message request counters include statistics showing the number of certain types of block mode conversions. Timed statistics reveal the total or average time waited for read and write I/O for particular types of operations.

Automatic Workload Repository in Oracle RAC Environments

You can use **Automatic Workload Repository (AWR)** to monitor performance statistics related to Oracle RAC databases. AWR automatically generates snapshots of the performance data once every hour and collects the statistics in the workload repository. In Oracle RAC environments, each AWR snapshot captures data from all active instances in the cluster. The data for each snapshot set is captured from the same point in time. AWR stores the snapshot data for all instances in the same table and the data is identified by an instance qualifier. For example, the BUFFER_BUSY_WAIT statistic shows the number of buffer waits on each instance. AWR does not store data that is aggregated from across the entire cluster. In other words, the data is stored for each individual instance.

Using the Automatic Database Diagnostic Monitor (ADDM), you can analyze the information collected by AWR for possible performance problems with Oracle Database. ADDM presents performance data from a cluster-wide perspective, thus enabling you to analyze performance on a global basis. In an Oracle RAC environment, ADDM can analyze performance using data collected from all instances and present it at different levels of granularity, including:

- Analysis for the entire cluster

- Analysis for a specific database instance

- Analysis for a subset of database instances

To perform these analyses, you can run the ADDM Advisor in ADDM for Oracle RAC mode to perform an analysis of the entire cluster; in Local ADDM mode to analyze the performance of an individual instance; or in Partial ADDM mode to analyze a subset of instances. Activate ADDM analysis using the advisor framework through Advisor Central in Oracle Enterprise Manager, or through the DBMS_ADVISOR and DBMS_ADDM PL/SQL packages.

- *Oracle Database Performance Tuning Guide* for information about AWR and ADDM

- *Oracle Database 2 Day + Real Application Clusters Guide* for more information about how to access and analyze global and local ADDM data using Oracle Enterprise Manager

- *Oracle Database PL/SQL Packages and Types Reference* for more information about the DBMS_ADVISOR and DBMS_ADDM packages

Active Session History Reports for Oracle RAC

This section describes Active Session History (ASH) reports for Oracle RAC under the following topics:

- Overview of ASH Reports for Oracle RAC

- ASH Report for Oracle RAC: Top Cluster Events

- ASH Report for Oracle RAC: Top Remote Instance

See Also:

Oracle Database Performance Tuning Guide for more information about ASH reports

Overview of ASH Reports for Oracle RAC

ASH is an integral part of the Oracle Database self-management framework and is useful for diagnosing performance problems in Oracle RAC environments. ASH report statistics provide details about Oracle Database session activity. Oracle Database records information about active sessions for all active Oracle RAC instances and stores this data in the System Global Area (SGA). Any session that is connected to the database and using CPU is considered an active session. The exception to this is sessions that are waiting for an event that belongs to the idle wait class.

ASH reports present a manageable set of data by capturing only information about active sessions. The amount of the data is directly related to the work being performed, rather than the number of sessions allowed on the system.

ASH statistics that are gathered over a specified duration can be put into ASH reports. Each ASH report is divided into multiple sections to help you identify short-lived performance problems that do not appear in the ADDM analysis. Two ASH report sections that are specific to Oracle RAC are Top Cluster Events and Top Remote Instance as described in the next two sections.

ASH Report for Oracle RAC: Top Cluster Events

The ASH report Top Cluster Events section is part of the Top Events report that is specific to Oracle RAC. The Top Cluster Events report lists events that account for the highest percentage of session activity in the cluster wait class event along with the

instance number of the affected instances. You can use this information to identify which events and instances caused a high percentage of cluster wait events.

ASH Report for Oracle RAC: Top Remote Instance

The ASH report Top Remote Instance section is part of the Top Load Profile report that is specific to Oracle RAC. The Top Remote Instance report shows cluster wait events along with the instance numbers of the instances that accounted for the highest percentages of session activity. You can use this information to identify the instance that caused the extended cluster wait period.

Monitoring Oracle RAC Statistics and Wait Events

This section explains wait events and statistics specific to Oracle RAC and how to interpret them when assessing performance data generated by the Automatic Workload Repository (AWR), Statspack, or by ad-hoc queries of the dynamic performance views.

This section includes the following topics:

- Oracle RAC Statistics and Events in AWR and Statspack Reports

- Oracle RAC Wait Events

- Monitoring Performance by Analyzing GCS and GES Statistics

- Analyzing Cache Fusion Transfer Impact Using GCS Statistics

- Analyzing Response Times Based on Wait Events

See Also:

- *Oracle Database Performance Tuning Guide* for more information about wait event analysis

- *Oracle Streams Concepts and Administration* for information about the spdoc.txt file and the Statspack utility

---

Oracle RAC Statistics and Events in AWR and Statspack Reports

The statistics snapshots generated by AWR and Statspack can be evaluated by producing reports displaying summary data such as load and cluster profiles based on regular statistics and wait events gathered on each instance.

Most of the relevant data is summarized on the Oracle RAC Statistics Page. This information includes:

- Global cache load profile

- Global cache efficiency percentages—workload characteristics

- Global cache and Enqueue Service (GES)—messaging statistics

Additional Oracle RAC sections appear later in the report:

- Global enqueue statistics

- Global CR statistics

- Global `CURRENT` served statistics

- Global cache transfer statistics.

Oracle RAC Wait Events

Analyzing and interpreting what causes sessions to wait is an important method to determine where time is spent. In Oracle RAC, the wait time is attributed to an event which reflects the exact outcome of a request. For example, when a session on an instance is looking for a block in the global cache, it does not know whether it will receive the data cached by another instance or whether it will receive a message to read from disk. The wait events for the global cache convey precise information and waiting for global cache blocks or messages is:

- Summarized in a broader category called Cluster Wait Class

- Temporarily represented by a placeholder event which is active while waiting for a block, for example:

  - `gc current block request`

  - `gc cr block request`

- Attributed to precise events when the outcome of the request is known, for example:

  - `gc current block 3-way`

  - `gc current block busy`

  - `gc cr block grant 2-way`

In summary, the wait events for Oracle RAC convey information valuable for performance analysis. They are used in Automatic Database Diagnostic Monitor (ADDM) to enable precise diagnostics of the effect of cache fusion.

Monitoring Performance by Analyzing GCS and GES Statistics

To determine the amount of work and cost related to inter-instance messaging and contention, examine block transfer rates, remote requests made by each transaction, the number and time waited for global cache events as described under the following headings:

- Analyzing the Effect of Cache Fusion in Oracle RAC

- Analyzing Performance Using GCS and GES Statistics

Analyzing the Effect of Cache Fusion in Oracle RAC

The effect of accessing blocks in the global cache and maintaining coherency is represented by:

- The Global Cache Service (GCS) statistics for `current` and `cr` blocks, for example, `gc current blocks received`, `gc cr blocks received`, and so on

- The GCS wait events, for `gc current block 3-way`, `gc cr grant 2-way`, and so on

The response time for cache fusion transfers is determined by the messaging and processing times imposed by the physical interconnect components, the IPC protocol and the GCS protocol. It is not affected by disk I/O factors other than occasional log writes. The cache fusion protocol does not require I/O to data files to guarantee **cache coherency** and Oracle RAC inherently does not cause any more I/O to disk than a nonclustered instance.

Analyzing Performance Using GCS and GES Statistics

This section describes how to monitor GCS performance by identifying data blocks and objects which are frequently used (*hot*) by all instances. High concurrency on certain blocks may be identified by GCS wait events and times.

The `gc current block busy` wait event indicates that the access to cached data blocks was delayed because they were busy either in the remote or the local cache. This could be caused by any of the following:

- The blocks were pinned

- The blocks were held up by sessions

- The blocks were delayed by a log write on a remote instance

- A session on the same instance was already accessing a block which was in transition between instances and the current session needed to wait behind it (for example, `gc current block busy`)

Use the `V$SESSION_WAIT` view to identify objects and data blocks with contention. The GCS wait events contain the file and block number for a block request in p1 and p2, respectively.

An additional segment statistic, `gc buffer busy`, has been added to quickly determine the busy objects without having to query the `V$SESSION_WAIT` view mentioned earlier.

The AWR infrastructure provides a view of active session history which can also be used to trace recent wait events and their arguments. It is therefore useful for hot block analysis. Most of the reporting facilities used by AWR and Statspack contain the object statistics and cluster wait class category, so that sampling of the views mentioned earlier is largely unnecessary.

> **Note:**
>
> Oracle recommends using ADDM and AWR. However, Statspack is available for backward compatibility. Statspack provides reporting only. You must run Statspack at level 7 to collect statistics related to block contention and segment block waits.

It is advisable to run ADDM on the snapshot data collected by the AWR infrastructure to obtain an overall evaluation of the impact of the global cache. The advisory will also identify the busy objects and SQL highest cluster wait time.

Analyzing Cache Fusion Transfer Impact Using GCS Statistics

This section describes how to monitor GCS performance by identifying objects read and modified frequently and the service times imposed by the remote access. Waiting for blocks to arrive may constitute a significant portion of the response time, in the same way that reading from disk could increase the block access delays, only that cache fusion transfers are usually faster than disk access latencies.

The following wait events indicate that the remotely cached blocks were shipped to the local instance without having been busy, pinned or requiring a log flush:

- `gc current block 2-way`

- `gc current block 3-way`

- `gc cr block 2-way`

- `gc cr block 3-way`

The object statistics for `gc current blocks received` and `gc cr blocks received` enable quick identification of the indexes and tables which are shared by the active instances. As mentioned earlier, creating an ADDM analysis will usually point you to the SQL statements and database objects that could be impacted by inter-instance contention.

Any increases in the average wait times for the events mentioned in the preceding list could be caused by the following occurrences:

- High load: CPU shortages, long run queues, scheduling delays

- Misconfiguration: using public instead of private interconnect for message and block traffic

If the average wait times are acceptable and no interconnect or load issues can be diagnosed, then the accumulated time waited can usually be attributed to a few SQL statements which need to be tuned to minimize the number of blocks accessed.

The column `CLUSTER_WAIT_TIME` in `V$SQLAREA` represents the wait time incurred by individual SQL statements for global cache events and will identify the SQL which may need to be tuned.

Analyzing Response Times Based on Wait Events

Most global cache wait events that show a high total time as reported in the AWR and Statspack reports or in the dynamic performance views are normal and may present themselves as the top database time consumers without actually indicating a problem. This section describes frequent wait events that you should be aware of when interpreting performance data.

If user response times increase and a high proportion of time waited is for global cache, then you should determine the cause. Most reports include a breakdown of events sorted by percentage of the total time.

It is useful to start with an ADDM report, which analyzes the routinely collected performance statistics with respect to their impact, and points to the objects and SQL contributing most to the time waited, and then moves on to the more detailed reports produced by AWR and Statspack.

Wait events for Oracle RAC include the following categories:

- Block-Related Wait Events

- Message-Related Wait Events

- Contention-Related Wait Events

- Load-Related Wait Events

Block-Related Wait Events

The main wait events for block-related waits are:

- `gc current block 2-way`

- `gc current block 3-way`

- `gc cr block 2-way`

- `gc cr block 3-way`

The block-related wait event statistics indicate that a block was received as either the result of a 2-way or a 3-way message, that is, the block was sent from either the resource master requiring 1 message and 1 transfer, or was forwarded to a third node from which it was sent, requiring 2 messages and 1 block transfer.

Message-Related Wait Events

The main wait events for message-related waits are:

- `gc current grant 2-way`

- `gc cr grant 2-way`

The message-related wait event statistics indicate that no block was received because it was not cached in any instance. Instead a global grant was given, enabling the requesting instance to read the block from disk or modify it.

If the time consumed by these events is high, then it may be assumed that the frequently used SQL causes a lot of disk I/O (in the event of the `cr grant`) or that the workload inserts a lot of data and needs to find and format new blocks frequently (in the event of the current grant).

Contention-Related Wait Events

The main wait events for contention-related waits are:

- `gc current block busy`

- `gc cr block busy`

- `gc buffer busy acquire/release`

The contention-related wait event statistics indicate that a block was received which was pinned by a session on another node, was deferred because a change had not yet been flushed to disk or because of high concurrency, and therefore could not be shipped immediately. A buffer may also be busy locally when a session has already initiated a cache fusion operation and is waiting for its completion when another session on the same node is trying to read or modify the same data. High service times for blocks exchanged in the global cache may exacerbate the contention, which can be caused by frequent concurrent read and write accesses to the same data.

The `gc current block busy` and `gc cr block busy` wait events indicate that the local instance that is making the request did not immediately receive a current or consistent read block. The term *busy* in these events' names indicates that the sending of the block was delayed on a remote instance. For example, a block cannot be shipped immediately if Oracle Database has not yet written the redo for the block's changes to a log file.

In comparison to `block busy` wait events, a `gc buffer busy` event indicates that Oracle Database cannot immediately grant access to data that is stored in the local buffer cache. This is because a global operation on the buffer is pending and the operation has not yet completed. In other words, the buffer is busy and all other processes that are attempting to access the local buffer must wait to complete.

The existence of `gc buffer busy` events also means that there is block contention that is resulting in multiple requests for access to the local block. Oracle Database must queue these requests. The length of time that Oracle Database needs to process the queue depends on the remaining service time for the block. The service time is affected by the processing time that any network latency adds, the processing time on the remote and local instances, and the length of the wait queue.

The average wait time and the total wait time should be considered when being alerted to performance issues where these particular waits have a high impact. Usually, either interconnect or load issues or SQL execution against a large shared working set can be found to be the root cause.

Load-Related Wait Events

The main wait events for load-related waits are:

* `gc current block congested`

* `gc cr block congested`

The load-related wait events indicate that a delay in processing has occurred in the GCS, which is usually caused by high load, CPU saturation and would have to be solved by additional CPUs, load-balancing, off loading processing to different times or a new cluster node.For the events mentioned, the wait time encompasses the entire round trip from the time a session starts to wait after initiating a block request until the block arrives.

14

Converting Single-Instance Oracle Databases to Oracle RAC and Oracle RAC One Node

This chapter describes the procedures for converting from Oracle Database single-instance databases to Oracle Real Application Clusters (Oracle RAC) and Oracle RAC One Node databases.

The procedures in this chapter assume that your original single-instance database and the target Oracle RAC database are using the same release and running on the same platform. If you are upgrading from an earlier version of Oracle RAC to Oracle RAC 12*c*, then use Oracle Database Upgrade Assistant (DBUA).

This chapter includes the following topics:

- Administrative Issues for Converting Databases to Oracle RAC

- Converting to Oracle RAC and Oracle RAC One Node Using DBCA

- Preparing to Convert with rconfig and Oracle Enterprise Manager

- Converting Databases to Oracle RAC Using rconfig

- Example of rconfig XML Input Files for ConvertToRAC

- Postconversion Steps

> **Note:**
>
> You must use clustered Oracle Automatic Storage Management (Oracle ASM) instances for Oracle RAC databases.

> **See Also:**
>
> *Oracle Database Licensing Information* for compliance requirements that may apply to the license you have purchased

Administrative Issues for Converting Databases to Oracle RAC

Note the following administrative considerations before converting single-instance databases to Oracle RAC:

- Backup procedures should be available before converting from a single-instance Oracle Database to Oracle RAC. This includes taking a backup of your existing

database before converting to Oracle RAC and being prepared to backup your Oracle RAC database immediately following the conversion.

- For archiving with Oracle RAC environments, the archive file format requires a thread number.

- The archived logs from all instances of an Oracle RAC database are required for media recovery. Because of this requirement, if you archive to a file and you do not use a cluster file system, or some other means to provide shared file systems, then you require a method of accessing the archive logs from all nodes on which the cluster database has instances.

- By default, all database files are migrated to Oracle Managed Files. This feature simplifies tablespace creation, ensures data file location consistency and compliance with Oracle Flexible Architecture rules, and reduces human error with data file management.

Converting to Oracle RAC and Oracle RAC One Node Using DBCA

You can use Database Configuration Assistant (DBCA) to convert from single-instance Oracle databases to Oracle RAC or Oracle RAC One Node databases. DBCA automates the configuration of the control file attributes, creates the undo tablespaces and the redo logs, and creates the initialization parameter file entries for cluster-enabled environments. DBCA also configures Oracle Net Services, Oracle Clusterware resources, and the configuration for Oracle RAC database management using Oracle Enterprise Manager or the Server Control utility (SRVCTL).

Before you use DBCA to convert a single-instance database to an Oracle RAC or an Oracle RAC One Node database, ensure that your system meets the following conditions:

- Your system uses supported hardware and operating system software. Your system is configured properly to support an Oracle RAC database.

- The nodes have access to shared storage; for example, either Oracle Cluster File System or Oracle ASM is available and accessible from all nodes. On Linux on POWER systems, ensure that GPFS is available and accessible from all nodes.

- Your applications have no design characteristics that preclude their use with cluster database processing.

If your platform supports a cluster file system, then you can use it for Oracle RAC. You can also convert to Oracle RAC and use a non-shared file system. In either case, Oracle strongly recommends that you use Oracle Universal Installer to install Oracle Database 12*c*, which sets up the Oracle home and inventory in an identical location on each of the selected nodes in your cluster.

This section includes the following topics:

- Converting Oracle Database Installations to Oracle RAC Using DBCA

- Converting Single Instance on a Cluster to Oracle RAC One Node Using DBCA

- Converting Single Instance on a Cluster to Oracle RAC Using DBCA

> **See Also:**
>
> "Converting Databases" for information about converting Oracle RAC and Oracle RAC One Node databases

Converting Oracle Database Installations to Oracle RAC Using DBCA

To convert from a single-instance Oracle Database that is on a non-clustered computer to Oracle RAC, perform the procedures described in the following sections, and in the order shown:

- Use DBCA to Create an Image of the Single-Instance Database
- Complete the Oracle Clusterware Installation
- Validate the Cluster
- Copy the Preconfigured Database Image
- Install Oracle Database 12c Software with Oracle RAC

Use DBCA to Create an Image of the Single-Instance Database

Use DBCA to create a preconfigured image of your single-instance database by using the following procedure:

1. Navigate to the `bin` directory in `$ORACLE_HOME`, and start DBCA.

2. At the Welcome page, click **Next**.

3. On the Operations page, select **Manage Templates**, and click **Next**.

4. On the Template Management page, select **Create a database template** and **From an existing database (structure as well as data)**, then click **Next**.

5. On the Source Database page, select the database name in the Database instance list, and click **Next**.

6. On the Template Properties page, enter a name for your template in the Name field. Oracle recommends that you use the database name.

   By default, the template files are generated in the directory `$ORACLE_HOME/assistants/dbca/templates`. You can enter a description of the file in the Description field, and change the template file location in the Template data file field.

   When you have completed the entries, click **Next**.

7. On the Location of Database Related Files page, select **Maintain the file locations**, so that you can restore the database to the current directory structure, and click **Finish**.

DBCA generates two files: a database structure file (`template_name.dbc`), and a database preconfigured image file (`template_name.dfb`).

Complete the Oracle Clusterware Installation

Complete the installation of Oracle Clusterware, as described in *Oracle Grid Infrastructure Installation Guide* for your platform.

Validate the Cluster

Validate the cluster configuration using Cluster Verification Utility (CVU).

See Also:

Oracle Clusterware Administration and Deployment Guide for more information about CVU

Copy the Preconfigured Database Image

Copy the preconfigured database image. This includes copying the database structure `*.dbc` file and the database preconfigured image `*.dfb` file that DBCA created in "Use DBCA to Create an Image of the Single-Instance Database" to a temporary location on the node in the cluster from which you plan to run DBCA.

Install Oracle Database 12*c* Software with Oracle RAC

1. Run Oracle Universal Installer to install an Oracle Database with Oracle RAC.

2. Select **Cluster Installation Mode** on the Specify Hardware Cluster Installation page of Oracle Universal Installer, and select the nodes to include in your Oracle RAC database.

3. On the Oracle Universal Installer Database Configuration Types page, select the **Advanced** installation type.

   After installing the Oracle Database software, Oracle Universal Installer runs postinstallation configuration tools, such as Net Configuration Assistant (NETCA), DBCA, and so on.

4. On the DBCA Template Selection page, use the template that you copied to a temporary location in "Copy the Preconfigured Database Image" Use the browse option to select the template location.

   Select the option that you want to deploy. Your choices are the following: Oracle RAC database; Oracle RAC One Node database; or Oracle single-instance database.

5. After creating the Oracle RAC database, DBCA displays the Password Management page on which you must change the passwords for database users who have SYSDBA and SYSOPER privileges. When DBCA exits, the conversion process is complete.

Converting Single Instance on a Cluster to Oracle RAC One Node Using DBCA

Use DBCA to convert a single-instance Oracle Database to Oracle RAC One Node by using the following procedure:

1. Change directory to `$ORACLE_HOME/bin`.

2. Start DBCA:

   ```
   $ dbca
   ```

3. From the Welcome window, select **Oracle RAC One Node** database.

4. Use the template that you selected to deploy in "Install Oracle Database 12c Software with Oracle RAC"

Converting Single Instance on a Cluster to Oracle RAC Using DBCA

There are three scenarios in which a single-instance database can exist on a cluster node:

- Scenario 1: The Oracle home for the single-instance database was installed on a cluster node and has Oracle RAC enabled. The steps to perform a conversion to Oracle RAC in this scenario are described in "Single-Instance Database on a Cluster Running from an Oracle RAC-Enabled Home"

- Scenario 2: The Oracle home for the single-instance database was installed on a cluster node, but the Oracle RAC feature is disabled for this Oracle home. Refer to "Single-Instance Database on a Cluster Running from an Oracle RAC-Disabled Home" for instructions on how to convert the database to Oracle RAC in this environment.

- Scenario 3: The Oracle home for the single-instance database was installed on only the local node in a cluster. This happens when you select the Local Installation option on the Oracle Universal Installer Specify Hardware Cluster Installation page during the Oracle Database 12*c* installation. The procedure to convert the single-instance database to Oracle RAC in this scenario is the same as that described in "Converting Oracle Database Installations to Oracle RAC Using DBCA" with one exception. In the last step, "Install Oracle Database 12c Software with Oracle RAC" ensure that you select a new Oracle home, not the one from which the single-instance database is running.

Single-Instance Database on a Cluster Running from an Oracle RAC-Enabled Home

Perform the following procedures to convert a single-instance database on a cluster node running from an Oracle home that has the Oracle RAC option enabled.

1. Use DBCA to create a preconfigured image of your single-instance database as described in "Use DBCA to Create an Image of the Single-Instance Database" To perform the conversion manually, shut down the single-instance database.

2. Add nodes to your cluster. Ensure that all nodes can access the shared storage used by Oracle Clusterware and Oracle RAC.

> **See Also:**
>
> *Oracle Clusterware Administration and Deployment Guide* for information about adding and connecting nodes to a cluster

3. From the existing Oracle home, extend this home to the new nodes.

> **See Also:**
>
> "Adding Oracle RAC to Nodes with Oracle Clusterware Installed" for information about extending the Oracle home to nodes

4. From a newly added node, configure the listeners on the additional nodes using NETCA. Choose the same port number and protocol that you used on the existing

node. If NETCA displays the existing node in the node list page, then do not select this node, because the listener is already configured on it.

5. Convert the database using one of the following procedures:

   - Automated Conversion Procedure Using DBCA

   - Manual Conversion Procedure

Automated Conversion Procedure Using DBCA

If you used DBCA to create a preconfigured image of your single-instance database as described in "Use DBCA to Create an Image of the Single-Instance Database" then perform the following steps to complete the conversion to an Oracle RAC database:

1. Start DBCA from the initial node. Select the names of the nodes to include as part of your cluster database. On the Template Selection page, select the preconfigured template that you created. Enter the database name and respond to the remaining DBCA prompts.

2. Specify the shared storage location for the Oracle Database data files. To use raw devices for the data files, you cannot use DBCA. Instead, follow the steps documented in "Manual Conversion Procedure"

After creating the Oracle RAC database, DBCA displays the Password Management page on which you must change the passwords for the database users who have SYSDBA and SYSOPER privileges. When DBCA exits, the conversion process is complete.

Manual Conversion Procedure

If you did not use DBCA to create a preconfigured image of your single-instance database as described in "Use DBCA to Create an Image of the Single-Instance Database", then perform the following steps to complete the conversion:

1. Create the Optimal Flexible Architecture directory structure on each of the nodes that you have added.

2. Re-create the control files by running the `CREATE CONTROLFILE` SQL statement with the `REUSE` keyword and specify `MAXINSTANCES` and `MAXLOGFILES`, and so on, as needed for your Oracle RAC configuration. The `MAXINSTANCES` recommended default is 32.

3. Shut down the database instance.

4. If your single-instance database was using an SPFILE, then create a temporary parameter file (PFILE) from the SPFILE using the following SQL statement:

   ```
   CREATE PFILE='pfile_name' from spfile='spfile_name'
   ```

5. Set the `CLUSTER_DATABASE` parameter to `TRUE`, and set the `INSTANCE_NUMBER` parameter to a unique value for each instance, using the `sid.parameter=value` syntax.

   If you optimized memory usage on your single-instance database, then adjust the size of the system global area (SGA) to avoid swapping and paging when you convert to Oracle RAC. You should make this adjustment because Oracle RAC requires about 350 bytes for each buffer to accommodate the Global Cache Service (GCS). For example, if you have 10,000 buffers, then Oracle RAC requires

approximately 350 multiplied by 10,000 bytes more memory. Therefore, adjust the size of the SGA by changing the `DB_CACHE_SIZE` and `DB_nK_CACHE_SIZE` parameters accordingly.

6. Start the database instance using the PFILE created in Step 4.

7. If your single-instance database was using automatic undo management, then create an undo tablespace for each additional instance using the `CREATE UNDO TABLESPACE` SQL statement.

8. Create redo threads that have at least two redo logs for each additional instance. Enable the new redo threads by using an `ALTER DATABASE` SQL statement. Then, shut down the database instance.

9. Copy the Oracle password file from the initial node, or from the node on which you are working, to the corresponding location on the additional nodes on which the cluster database will have an instance. Replace the `ORACLE_SID` name in each password file appropriately for each additional instance.

10. Set the `REMOTE_LISTENER` parameter to the single client access name (SCAN) and port.

11. Configure the net service entries for the database and instances, and address entries for the `LOCAL_LISTENER` for each instance and for the `REMOTE_LISTENER` in the `tnsnames.ora` file, and copy the `tnsnames.ora` file to all nodes.

12. Create the SPFILE from the PFILE.

See Also:

Oracle Real Application Clusters Installation Guide for your platform for information about migrating to the SPFILE

---

13. Create the `$ORACLE_HOME/dbs/init`*sid*`.ora` file that contains the following entry, where *spfile_path_name* is the complete path name of the SPFILE:

```
spfile='spfile_path_name'
```

14. On the local node, use SQL*Plus to run `catclust.sql`. This script creates the dictionary views needed for Oracle RAC databases. For example:

```
SQL> start ?/rdbms/admin/catclust.sql
```

15. Add the configuration for the Oracle RAC or Oracle RAC One Node database and its instance-to-node mapping using SRVCTL.

    a. To add the configuration of an Oracle RAC database, use the following commands:

```
srvctl add database -d db_name -o Oracle_home -p spfile_path_name
srvctl add instance -d db_name -i inst1_name -n node1_name
srvctl add instance -d db_name -i inst2_name -n node2_name
...
```

    b. To add the configuration of an Oracle RAC One Node database, use the following command:

```
srvctl add database -d db_name -c RACONENODE -o Oracle_home
-p spfile_path_name
```

16. Start the Oracle RAC or Oracle RAC One Node database using SRVCTL:

```
srvctl start database -d db_name
```

After starting the database with SRVCTL, your conversion process is complete. You can run the following SQL statement to see the status of all the instances in your Oracle RAC database:

```
SQL> SELECT * FROM v$active_instances;
```

Single-Instance Database on a Cluster Running from an Oracle RAC-Disabled Home

You can create a single-instance database on a cluster running from an Oracle home with the Oracle RAC option disabled. To create an Oracle home on a cluster with Oracle RAC disabled, you can select **local** and **non-cluster** on the Node Selection Page of Oracle Universal Installer when installing the Oracle Database software. You can also performed a one-node cluster (with Oracle RAC) installation, but later disable the Oracle RAC option.

Perform the following procedures to convert this type of single-instance database to an Oracle RAC or Oracle RAC One Node database:

1. Use DBCA to create a preconfigured image of your single-instance database as described in "Use DBCA to Create an Image of the Single-Instance Database" To perform the conversion manually, shut down the single-instance database.

2. Change the directory to the lib subdirectory in the rdbms directory under the Oracle home.

3. Relink the oracle binary by running the following commands:

```
make -f ins_rdbms.mk rac_on
make -f ins_rdbms.mk ioracle
```

4. Continue with step 2 in "Single-Instance Database on a Cluster Running from an Oracle RAC-Enabled Home".

Preparing to Convert with rconfig and Oracle Enterprise Manager

You can use rconfig, or Oracle Enterprise Manager to assist with converting a single-instance database installation to an Oracle RAC database. The first of these, rconfig, is a command-line utility. Oracle Enterprise Manager Cloud Control database administration option, Convert to Cluster Database, provides a GUI-based conversion tool. The following sections describe how to use these conversion tools:

- Prerequisites for Converting to Oracle RAC Databases

- Configuration Changes During Oracle RAC Conversion Using rconfig

- Converting Databases to Oracle RAC Using rconfig or Oracle Enterprise Manager

- Converting Databases to Oracle RAC Using Oracle Enterprise Manager

> **Note:**
>
> Before you start the conversion, back up your existing database; you should take a backup of your database before starting any major change.

Prerequisites for Converting to Oracle RAC Databases

Before you convert a single-instance database to an Oracle RAC database, ensure that the following conditions are met for each cluster node that you intend to make an Oracle RAC database node:

- Oracle Clusterware 12*c* is installed, configured, and running.

- Oracle RAC 12*c* software is installed.

- The Oracle software has the Oracle RAC option enabled.

- Shared storage, either Oracle Cluster File System or Oracle ASM, is available and accessible from all nodes.

- User equivalence exists for the `oracle` account, or the user account used to install the Oracle software.

- If you intend to use Oracle Enterprise Manager, then the Oracle Management Agent on each node is configured and running, and is configured with cluster and host information.

- You have backed up your existing database.

> **Note:**
>
> You must use clustered Oracle ASM instances for Oracle RAC databases.

Configuration Changes During Oracle RAC Conversion Using rconfig

The following changes occur when you convert a single-instance database to Oracle RAC using the `rconfig` utility:

- During the conversion, `rconfig` places the target Oracle RAC database into archive log mode, and enables archiving for the database. If you do not plan to use archive log space, then you can disable archive logging after the conversion has completed.

- For the **Shared Storage Type** value, if you enter **CFS**, and you use a cluster file system for your single-instance database storage, then `rconfig` converts the environment to use Oracle Managed Files for database storage, and places the data files in a subdirectory located under the shared storage location.

- During the conversion, `rconfig` moves database files to a specified shared location, and configures them using Oracle Managed Files.

  To avoid using Oracle Managed Files with your converted database, the single-instance database files must be located in a shared file system, and you must indicate that `rconfig` should not move the files.

Converting Databases to Oracle RAC Using rconfig or Oracle Enterprise Manager

The following list describes scenarios for converting a single-instance Oracle database to an Oracle RAC database:

- Converting a single-instance Oracle Database 12*c* database to an Oracle RAC 12*c* database, running out of the *same* Oracle home and using the same data files as the single-instance database.

  In this scenario, run the `rconfig` utility from the Oracle RAC database home, or use the **Convert to RAC** option on the single-instance database target of Oracle Enterprise Manager Cloud Control.

- Converting a single-instance database that uses a release of Oracle Database earlier than Oracle Database 12*c* to an Oracle RAC 12*c* database, running out of the same Oracle home and using the same data files as the single-instance database

  In this scenario, use Oracle Universal Installer and Database Upgrade Assistant (DBUA) to update the single-instance database to Oracle Database 12*c*. Then use `rconfig` or the Oracle Enterprise Manager **Convert to RAC** option, as described in the preceding scenario.

- Converting a single-instance Oracle Database 12*c* to an Oracle RAC 12*c* database, running out of a *different* Oracle home and using the same data files as the single-instance database.

  In this scenario, run the `rconfig` utility in the target database home, or use the **Convert to RAC** option in the single-instance database target of Oracle Enterprise Manager Cloud Control. Provide the file storage location when prompted.

  > **Note:**
  >
  > If you specify Oracle home users for both the target database home and the source database home, then the Oracle home user for the target database home must be the same as the Oracle home user for the source database home.

- Converting a single-instance Oracle Database 12*c* to an Oracle RAC 12*c* database, running out of a *different* Oracle home, and where the host on which the single-instance database runs is *not* a node used by the Oracle RAC database.

  In this scenario, create a clone image of the single-instance database, and move the clone image to a host that is one of the nodes used by the Oracle RAC database. Then use `rconfig` or the Oracle Enterprise Manager **Convert to RAC** option, as described in the preceding scenario.

Converting Databases to Oracle RAC Using Oracle Enterprise Manager

You can use Oracle Enterprise Manager Cloud Control to convert a single-instance database to an Oracle RAC database. To use this feature, complete the following steps:

> **See Also:**
>
> *Oracle Database Upgrade Guide* for information about upgrading existing Oracle Database installations

1. Log in to Oracle Enterprise Manager Cloud Control. From the Home page, click the **Targets** tab.

2. On the Targets page, click the **Databases** secondary tab, and click the link in the Names column of the database to convert to Oracle RAC.

3. On the Database Instance Home page, click the **Change Database** secondary tab.

4. On the Administration page, in the Database Administration Change Database section, click **Convert to Cluster Database**.

5. Log in as the database user SYS with SYSDBA privileges to the database you want to convert, and click **Next**.

6. On the Convert to Cluster Database: Cluster Credentials page, provide a user name and password for the oracle user and password of the target database to convert. If the target database is using Oracle ASM, then also provide the SYSASM user and password, and then click **Next**.

7. On the Hosts page, select the host nodes in the cluster to be cluster members in the installed Oracle RAC database. When you have completed your selection, click **Next**.

8. On the Convert to Database: Options page, select whether you want to use the existing listener and port number, or specify a new listener and port number for the cluster. Also provide a prefix for cluster database instances on the cluster.

   When you have finished entering information, click **Next**, or click **Help** if you need assistance in deciding how to enter information.

9. On the Convert to Cluster Database: Shared Storage page, either select the option to use your existing shared storage area, or select the option to have your database files copied to a new shared storage location. Also, decide if you want to use your existing fast recovery area or copy your recovery files to a new fast recovery area using files managed by Oracle Database.

   If you use Oracle ASM, then Oracle recommends that you place the data files and the recovery files in separate failure groups. A failure group is defined by shared hardware, such as a controller shared between two disks, or two disks that are on the same spindle. If two disks share hardware that could fail, making both disks unavailable, then theses disks are said to be in the same failure group. If you do not use Oracle ASM, then Oracle recommends that the data files and the recovery files are stored in separate locations, for example, separate Oracle ASM failure groups, so that a hardware failure does not affect availability.

   When you have finished entering information, click **Next**, or click **Help** if you need assistance in deciding how to enter information.

10. On the Convert to Cluster Database: Review page, review the options you have selected. Click **Submit Job** to proceed with the conversion. To change any options you have selected, click **Back**. To cancel the conversion, click **Cancel**.

11. On the Confirmation page, click **View Job** to check the status of the conversion.

Converting Databases to Oracle RAC Using rconfig

You can use the command-line utility rconfig to convert a single-instance database to an Oracle RAC database, or to convert it to an Oracle RAC One Node database,

depending on the values you provide in the `ConvertToRAC.xml` file. To use this feature, complete the following steps:

> **See Also:**
>
> *Oracle Database Upgrade Guide* for information about upgrading existing Oracle Database installations

1. As the `oracle` user, navigate to the directory `$ORACLE_HOME/assistants/ rconfig/sampleXMLs`, and open the file `ConvertToRAC.xml` using a text editor, such as `vi`.

2. Review the `ConvertToRAC.xml` file, and modify the parameters as required for your system. The XML sample file contains comments that provide instructions for how to configure the file.

> **Caution:**
>
> Set the convert option `Convert verify="ONLY"` to perform a test conversion to ensure that a conversion can be completed successfully.

   When you have finished modifying parameters, save the file with a name of the format `filename.xml`. Make a note of the name you select.

3. Navigate to the directory `$ORACLE_HOME/bin`, and use the following command to run the command `rconfig`, where `input.xml` is the name of the XML input file you configured in Step 2:

   ```
   rconfig input.xml
   ```

   For example, if you create an input XML file called `convert.xml`, then you would use the following command

   ```
   $ ./rconfig convert.xml
   ```

> **Note:**
>
> The `Convert verify` option in the `ConvertToRAC.xml` file has three options:
>
> - `Convert verify="YES"`: rconfig performs checks to ensure that the prerequisites for single-instance to Oracle RAC conversion have been met before it starts conversion
>
> - `Convert verify="NO"`: rconfig does not perform prerequisite checks, and starts conversion
>
> - `Convert verify="ONLY"` rconfig only performs prerequisite checks; it does not start conversion after completing prerequisite checks

If performing the conversion fails, then use the following procedure to recover and reattempt the conversion.:

1. Attempt to delete the database using the DBCA delete database option.

2. Restore the source database.

3. Review the conversion log, and fix any problems reported by `rconfig` that may have caused the conversion failure. The `rconfig` log files are under the `rconfig` directory in `$ORACLE_BASE/cfgtoollogs`.

4. Reattempt the conversion.

Example of rconfig XML Input Files for ConvertToRAC

The following are two examples of an XML `ConvertToRAC` input file for the `rconfig` utility. Example 14-1 is an XML input file to convert a single-instance database with Oracle ASM to a policy-managed Oracle RAC database (using server pools) on Oracle ASM storage. Example 14-2 is an XML input file to convert a single-instance database with Oracle ASM to an administrator-managed Oracle RAC database.

Example 14-1   Example rconfig ConvertToRAC XML File for Policy-Managed Databases

```
<?xml version="1.0" encoding="UTF-8"?>
<n:RConfig xmlns:n="http://www.example.com/rconfig"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:schemaLocation="http://www.example.com/rconfig">
    <n:ConvertToRAC>
<!-- Verify does a precheck to ensure all pre-requisites are met, before the
 conversion is attempted. Allowable values are: YES|NO|ONLY -->
    <n:Convert verify="YES">
<!--Specify current OracleHome of non-rac database for SourceDBHome -->
         <n:SourceDBHome>/oracle/product/12.1.0/db_1</n:SourceDBHome>
<!--Specify OracleHome where the rac database should be configured. It can be same
 as SourceDBHome -->
         <n:TargetDBHome>/oracle/product/12.1.0/db_1</n:TargetDBHome>
<!--Specify SID of non-rac database and credential. User with sysdba role is
 required to perform conversion -->
             <n:SourceDBInfo SID="sales">
           <n:Credentials>
             <n:User>sys</n:User>
             <n:Password>oracle</n:Password>
             <n:Role>sysdba</n:Role>
           </n:Credentials>
         </n:SourceDBInfo>
<!--Specify the list of existing or new server pools which are used by the
 Policy Managed Cluster Database. -->
             <n:ServerPoolList>
               <n:ExistingServerPool name="custom"/>
               <n:NewServerPool name="newpool" cardinality="2"/>
             </n:ServerPoolList>
<!--Specify RacOneNode along with servicename to convert database to RACOne
Node -->
             <!--n:RacOneNode  servicename="salesrac1service"/-->
<!--InstancePrefix is not required for Policy Managed database. If specified, it
 will be ignored. Instance names are generated automatically based on db_unique_
name for Policy Managed dababase.-->
<!-- Listener details are no longer needed starting 11.2. Database is registered
 with default listener and SCAN listener running from Oracle Grid Infrastructure
 home. -->
<!--Specify the type of storage to be used by rac database. Allowable values are
 CFS|ASM. The non-rac database should have same storage type. ASM credentials are
 no needed for conversion. -->
```

```
            <n:SharedStorage type="ASM">
<!--Specify Database Area Location to be configured for rac database.If this field
 is left empty, current storage will be used for rac database. For CFS, this field
 will have directory path. -->
              <n:TargetDatabaseArea>+ASMDG</n:TargetDatabaseArea>
<!--Specify Fast Recovery Area to be configured for rac database. If this field is
 left empty, current recovery area of non-rac database will be configured for rac
 database. If current database is not using recovery Area, the resulting rac
 database will not have a recovery area. -->
              <n:TargetFlashRecoveryArea>+ASMDG</n:TargetFlashRecoveryArea>
            </n:SharedStorage>
          </n:Convert>
      </n:ConvertToRAC>
</n:RConfig>
```

Example 14-2   Example rconfig ConvertToRAC XML File for Administrator-Managed Databases

```
<?xml version="1.0" encoding="UTF-8"?>
<n:RConfig xmlns:n="http://www.example.com/rconfig"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:schemaLocation="http://www.example.com/rconfig rconfig.xsd">
    <n:ConvertToRAC>
<!-- Verify does a precheck to ensure all pre-requisites are met, before the
 conversion is attempted. Allowable values are: YES|NO|ONLY -->
    <n:Convert verify="YES">
<!--Specify current OracleHome of non-rac database for SourceDBHome -->
        <n:SourceDBHome>/oracle/product/12.1.0/db_1</n:SourceDBHome>
<!--Specify OracleHome where the rac database should be configured. It can be same
 as SourceDBHome -->
        <n:TargetDBHome>/oracle/product/12.1.0/db_1</n:TargetDBHome>
<!--Specify SID of non-rac database and credential. User with sysdba role is
 required to perform conversion -->
            <n:SourceDBInfo SID="sales">
          <n:Credentials>
            <n:User>sys</n:User>
            <n:Password>oracle</n:Password>
            <n:Role>sysdba</n:Role>
          </n:Credentials>
          </n:SourceDBInfo>
<!--Specify the list of nodes that should have rac instances running for the Admin
 Managed Cluster Database. LocalNode should be the first node in this nodelist.
 -->
        <n:NodeList>
          <n:Node name="node1"/>
          <n:Node name="node2"/>
        </n:NodeList>
<!--Specify RacOneNode along with servicename to convert database to RACOne
Node -->
<!--n:RacOneNode  servicename="salesrac1service"/-->
<!--Instance Prefix tag is optional starting with 11.2. If left empty, it is
 derived from db_unique_name.-->
        <n:InstancePrefix>sales</n:InstancePrefix>
<!-- Listener details are no longer needed starting 11.2. Database is registered
 with default listener and SCAN listener running from Oracle Grid Infrastructure
 home. -->
<!--Specify the type of storage to be used by rac database. Allowable values are
 CFS|ASM. The non-rac database should have same storage type. ASM credentials
are not needed for conversion. -->
        <n:SharedStorage type="ASM">
<!--Specify Database Area Location to be configured for rac database.If this field
```

```
      is left empty, current storage will be used for rac database. For CFS, this
field will have directory path. -->
            <n:TargetDatabaseArea>+ASMDG</n:TargetDatabaseArea>
<!--Specify Fast Recovery Area to be configured for rac database. If this field is
 left empty, current recovery area of non-rac database will be configured for rac
 database. If current database is not using recovery Area, the resulting rac
 database will not have a recovery area. -->
            <n:TargetFlashRecoveryArea>+ASMDG</n:TargetFlashRecoveryArea>
         </n:SharedStorage>
       </n:Convert>
    </n:ConvertToRAC>
</n:RConfig>
```

Postconversion Steps

After completing the conversion, note the following recommendations for Oracle RAC environments, as described in the following:

- Follow the recommendations for using load balancing and transparent application failover, as described in Workload Management with Dynamic Database Services

- Use locally managed tablespaces instead of dictionary managed tablespaces to reduce contention and manage sequences in Oracle RAC as described in *Oracle Database Administrator's Guide*

- Follow the guidelines for using automatic segment space management as described in *Oracle Database Administrator's Guide*

The buffer cache and shared pool capacity requirements in Oracle RAC are slightly greater than the requirements for single-instance Oracle databases. Therefore, you should increase the size of the buffer cache by about 10%, and the size of the shared pool by about 15%.

A

Server Control Utility Reference

Use the Server Control Utility (SRVCTL) to manage Oracle Real Application Clusters (Oracle RAC) configuration information. You can use SRVCTL commands to add, remove, start, stop, modify, enable, and disable a number of entities, such as databases, instances, listeners, SCAN listeners, services, Grid Naming Service (GNS), and Oracle ASM.

Some SRVCTL operations modify the configuration data stored in the Oracle Cluster Registry (OCR). SRVCTL performs other operations, such as starting and stopping instances, by sending requests to the Oracle Clusterware process (CRSD), which then starts or stops the Oracle Clusterware resources.

> **Note:**
>
> To manage Oracle ASM on Oracle Database 12*c* installations, use the SRVCTL binary in the Oracle Grid Infrastructure home for a cluster (Grid home). If you have Oracle RAC or Oracle Database installed, then you cannot use the SRVCTL binary in the database home to manage Oracle ASM.

> **See Also:**
>
> Administering Database Instances and Cluster Databases for more information about using SRVCTL to manage an Oracle RAC database

SRVCTL Usage Information

SRVCTL is installed on each node in a cluster by default. To use SRVCTL, log in to the operating system of a node and enter the SRVCTL command and its parameters in case-sensitive syntax as described in "SRVCTL Command Reference".

- Only use the version of SRVCTL that is provided with Oracle Database 12*c* on Oracle RAC databases that are created for or upgraded to Oracle Database 12*c*. The version of SRVCTL must be the same as the version of the object (listeners, Oracle ASM instances, Oracle RAC databases and their instances, and services) being managed. To ensure the versions are the same release, run SRVCTL commands from the Oracle home of the database or object you are managing.

- SRVCTL does not support concurrent executions of commands on the same object. Therefore, run only one SRVCTL command at a time for each database, service, or other object.

- When specifying a comma-delimited list as part of a SRVCTL command, there should not be any spaces between the items in the list. For example:

```
srvctl add database -serverpool "serverpool1,serverpool3"
```

When you specify a comma-delimited list in a Windows environment, you must enclose the list in double quotation marks (" "). You can enclose a comma-delimited list in double quotation marks in a Linux or UNIX environment but they will be ignored.

- If you are entering a SRVCTL command, and you want to continue the input on a new line, then you can use the operating system continuation character. In Linux, this is the backslash (\) symbol.

- A SRVCTL command that produces no output is a successful command. Not all SRVCTL commands return a message when it completes, successfully. However, if a SRVCTL command fails, then it always returns an error message.

- You can use the -eval parameter with several SRVCTL commands. This parameter, when you use it, enables you to simulate running a command without making any changes to the system. SRVCTL returns output that informs you what will happen if you run a particular command. For example, to know what might happen if you relocate a server:

```
$ srvctl relocate server -servers "rac1" -eval -serverpool pool2

Database db1
     will stop on node rac1
     will start on node rac7
     Service mySrv1
          will stop on node rac1, it will not run on any node
     Service myServ2
          will stop on node rac1
          will start on node rac6
Server rac1
     will be moved from pool myPoolX to pool pool2
```

The -eval parameter is available with the following commands:

- srvctl add database

- srvctl add service

- srvctl add srvpool

- srvctl modify database

- srvctl modify service

- srvctl modify srvpool

- srvctl relocate server

- srvctl relocate service

- srvctl remove srvpool

- srvctl start database

- srvctl start service

- srvctl stop database

- srvctl stop service

Specifying Command Parameters as Keywords Instead of Single Letters

Before Oracle Database 12*c*, the SRVCTL command-line interface used single letter parameters. However, this imposes a limit on the number of unique parameters available for use with SRVCTL commands. SRVCTL command parameters introduced in Oracle Database 12*c* are full words instead of single letters, such as –`multicastport` and –`subdomain`.

To support backward compatibility, you can use a mix of single-letter parameters and new keyword parameters. New parameters introduced with keywords can be used with single letter parameters.

> **Note:**
>
> Starting with Oracle Database 12*c*, the single letter parameters are deprecated in favor of the keyword parameters to avoid using the same letter to implement different functionality depending on the command.
>
> You can obtain the single-letter equivalents, where applicable, by adding the –`compatible` parameter after the –`help` parameter.

Specifying an Input File for SRVCTL Command Parameters

You can specify command parameters in a file rather than directly on the command line. Using a command parameters input file is useful in the following situations:

* You want to run a command with very long parameter values or a command with numerous parameters

* You want to bypass shell processing of certain special characters

To specify a command parameters input file, use the –`file` parameter with a value that is the location of the command parameters file. SRVCTL processes the command parameters from the command parameters file instead of from the command line.

Character Set and Case Sensitivity of SRVCTL Object Values

SRVCTL interacts with many different types of objects. The character set and name length limitations, and whether the object name is case sensitive, can vary between object types.

Table A-1   String Restrictions for SRVCTL Object Names

Object Type	Character Set Limitations	Case Sensitive?	Maximum Length
db_domain	Alpha-numeric characters, underscore (_), and number sign (#)		128 characters

Table A-1   (Cont.) String Restrictions for SRVCTL Object Names

Object Type	Character Set Limitations	Case Sensitive?	Maximum Length
db_unique_name	Alpha-numeric characters, underscore (_), number sign (#), and dollar sign ($); the first 8 characters must be unique because those characters are used to form instance names for policy-managed databases	No	30 characters but the first 8 characters must be unique relative to any other database in the same cluster
diskgroup_name	Naming disk groups have the same limitations as naming other database objects. **See Also:** *Oracle Database SQL Language Reference* for more information about database object naming rules	No (all names are converted to uppercase)	
instance_name	Alpha-numeric characters	Depends on the platform	15 characters
listener_name			
node_name		No	
scan_name	The first character must be an alphabetic character	No	
server_pool	Alpha-numeric characters, underscore (_), number sign (#), period (.), and dollar sign ($); the name cannot begin with a period, contain single quotation marks (' '), nor can the name be "Generic" or "Free" because those two names are reserved for the built-in server pools		250 characters
service_name			250 characters
volume_name	Alphanumeric characters; dashes (-) are not allowed and the first character must be an alphabetic character.	No	11 characters

Summary of Tasks for Which SRVCTL Is Used

Use SRVCTL to manage databases, instances, cluster databases, cluster database instances, Oracle ASM instances and disk groups, services, listeners, or other clusterware resources.

- Cluster Database Configuration Tasks

Tasks	Commands
Add, modify, and delete cluster database configuration information	srvctl add database srvctl modify database srvctl remove database
Add an instance to or delete an instance from the configuration of a cluster database	srvctl add instance srvctl remove instance
Add a service to or delete a service from the configuration of a cluster database	srvctl add service srvctl remove service
Move instances and services in a cluster database configuration and modify service configurations	srvctl relocate database srvctl relocate service srvctl modify instance srvctl modify service
Set and unset the environment for an instance or service in a cluster database configuration	srvctl modify instance srvctl modify service
Set and unset the environment for an entire cluster database in a cluster database configuration	srvctl setenv database srvctl unsetenv database

- General Cluster Database Administration Tasks

Tasks	Commands
Start and stop cluster databases	srvctl start database srvctl start home
Start and stop cluster database instances	srvctl start instance srvctl start home
Start, stop, and relocate cluster database services	srvctl start service srvctl stop service srvctl relocate service
Obtain statuses of cluster databases, cluster database instances, or cluster database services	srvctl status database srvctl status instance srvctl status home srvctl status service

- Node-Level Tasks

Tasks	Commands
Administering VIPs	srvctl add vip
	srvctl config vip
	srvctl disable vip
	srvctl enable vip
	srvctl getenv vip
	srvctl modify vip
	srvctl relocate vip
	srvctl remove vip
	srvctl setenv vip
	srvctl start vip
	srvctl status vip
	srvctl stop vip
	srvctl unsetenv vip
Administering disk groups	srvctl enable diskgroup
	srvctl disable diskgroup
	srvctl remove diskgroup
	srvctl start diskgroup
	srvctl status diskgroup
	srvctl stop diskgroup
Administering server pools	srvctl add srvpool
	srvctl config srvpool
	srvctl remove srvpool
	srvctl modify srvpool
	srvctl status srvpool
Administering node applications	srvctl add nodeapps
	srvctl disable nodeapps
	srvctl enable nodeapps
	srvctl getenv nodeapps
	srvctl modify nodeapps
	srvctl remove nodeapps
	srvctl setenv nodeapps
	srvctl unsetenv nodeapps
Administering Oracle ASM instances	srvctl enable asm
	srvctl getenv asm
	srvctl modify asm
	srvctl remove asm
	srvctl setenv asm
	srvctl start asm
	srvctl status asm
	srvctl stop asm
	srvctl unsetenv asm

Tasks	Commands
Administering Rapid Home Provisioning	srvctl add havip
	srvctl config havip
	srvctl disable havip
	srvctl enable havip
	srvctl modify havip
	srvctl relocate havip
	srvctl remove havip
	srvctl start havip
	srvctl status havip
	srvctl stop havip
Starting and stopping a group of programs that includes virtual IP addresses (VIPs), listeners, and Oracle Notification Services	srvctl start home
	srvctl status home
	srvctl stop home

See Also:

Oracle Clusterware Administration and Deployment Guide for information about administering Oracle Clusterware

Using SRVCTL Help

To see help for all SRVCTL commands, from the command line enter:

```
srvctl -help
```

To see the command syntax and a list of parameters for each SRVCTL command, from the command line enter:

```
srvctl command (or verb) object (or noun) -help
```

When you request online help for a command using `-help`, SRVCTL prints the full words for each parameter. You can obtain the single-letter equivalents, where applicable, by adding the `-compatible` parameter after the `-help` parameter. For example:

```
$ srvctl config gns -help -compatible
```

The preceding command prints usage information for the `srvctl config gns` command, listing all parameters as full words followed by their single-letter equivalents in parentheses, where applicable.

To see the SRVCTL version number enter:

```
$ srvctl -version
```

SRVCTL Privileges and Security

To use SRVCTL to change your Oracle RAC database configuration, log in to the operating system as the software owner of the home that you want to manage.

For example, if different users installed Oracle Database and the Oracle Grid Infrastructure, then log in as the database software owner (for example, ora_db) to

manage databases and log in as the Oracle Grid Infrastructure software owner (for example, `ora_asm`) to manage the Oracle ASM instances.

Users who are members of the OSDBA operating system group can start and stop the database. To stop and start an Oracle ASM instance, you must be a member of the OSASM operating system group.

To create or register objects such as listeners, Oracle Notification Services, and services, you must be logged in to the operating system as the software owner of the Oracle home. The objects you create or register for that Oracle home will run under the user account of the owner of the Oracle home. Databases run as the database installation owner of the home from which they run.

To perform `srvctl add` operations on any object, you must be logged in as the Oracle account owner of the home on which the object runs.

For some SRVCTL commands, on Linux and UNIX systems, you must be logged in as `root`, and on Windows systems, you must be logged in as a user with Administrator privileges to run them. In this appendix, those commands are preceded by the root prompt (#) in the command examples.

Additional SRVCTL Topics

- Use SRVCTL to manage Oracle-supplied resources such as listener, instances, disk groups, and networks, and CRSCTL for managing Oracle Clusterware and its resources.

    > **Note:**
    >
    > Oracle strongly discourages directly manipulating Oracle-supplied resources (resources whose names begin with *ora*) using CRSCTL. This could adversely impact the cluster configuration.

- Although you may be able to cancel running SRVCTL commands by pressing the Control-C keys, you may corrupt your configuration data by doing this.

    You are strongly advised *not* to attempt to terminate SRVCTL in this manner.

Deprecated SRVCTL Subprograms or Commands

A number of SRVCTL commands and parameters have been deprecated in this release:

- Single Character Parameters for all SRVCTL Commands
- Miscellaneous SRVCTL Commands and Parameters

Single Character Parameters for all SRVCTL Commands

Single-character parameters have been deprecated in Oracle Database 12*c*. Use the full keyword for each parameter instead. To support older tools and scripts that still use single-character parameters, the current version of SRVCTL supports both single-character parameters and full keyword parameters.

The command reference in this appendix shows the keywords for each SRVCTL command. Table A-2 lists the deprecated single-character parameters.

Table A-2   Deprecated Single-Character Parameters for SRVCTL Commands

Single Letter	Long Form	Values	Description	Related Commands
A	address	{*VIP_name* | *IP*}/*netmask*/ [*if1*[|*if2*...]]	VIP address specification for node applications	Node applications, VIP, network, Listener, SCAN VIP, and SCAN listener commands
a	all		All resources of that kind	srvctl config database Common
a	diskgroup	*diskgroup_list*	Comma-delimited list of Oracle ASM disk groups	Database, instance, Oracle ASM, disk group, and file system commands
a	detail		Print detailed configuration information	Common
a	available	*available_list*	A comma-delimited list of available instances	Service and server pool commands
a	abort		Abort failed online relocation	Relocate database
a	viponly		Display VIP configuration	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
B	rlbgoal	{NONE| SERVICE_TIME| THROUGHPUT}	The runtime load balancing goal of a service	Service and server pool commands
c	currentnode	*current_node*	Node name from which to relocate the service	Service and server pool commands
c	cardinality	{UNIFORM| SINGLETON}	Whether the service should run on every active server in the server pool (UNIFORM) or just one server (SINGLETON)	Service and server pool commands
c	dbtype	*type*	Type of database: Oracle RAC One Node, Oracle RAC, or single instance	Database, instance, Oracle ASM, disk group, and file system commands

Table A-2   (Cont.) Deprecated Single-Character Parameters for SRVCTL Commands

Single Letter	Long Form	Values	Description	Related Commands
d	db or database	*db_unique_name*	Database unique name	Common
d	device	*volume_device*	Volume device path	Database, instance, Oracle ASM, disk group, and file system commands
d	domain		Display subdomain served by GNS	OC4J, home, CVU, and GNS commands
e	emport	*em_port_number*	Local listen port for Oracle Enterprise Manager	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
e	failovertype	{NONE|SESSION BASIC| TRANSACTION}	The failover type for a service	Service and server pool commands
e	server	*server_list*	Candidate server list for Oracle RAC One Node database	Database, instance, Oracle ASM, disk group, and file system commands
f	force		Force remove	Common
g	diskgroup	*diskgroup_name*	Disk group name	File system, Diskgroup commands
g	gsdonly		Display GSD configuration	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
g	serverpool	*server_pool_name*	A server pool name	Service and server pool commands
		server_pool_list	Comma-delimited list of database server pool names	Database, instance, Oracle ASM, disk group, and file system commands
h	help			Common
i	importance	*number*	A number that represents the importance of the server pool	Service and server pool commands

Table A-2   (Cont.) Deprecated Single-Character Parameters for SRVCTL Commands

Single Letter	Long Form	Values	Description	Related Commands
i	instance	*instance_name* *instance_list*	Instance name prefix for administrator-managed Oracle RAC One Node database A comma-delimited list of instance names	Database, instance, Oracle ASM, disk group, and file system commands
I	ip	*ip_address*	VIP address on which GNS is to listen	OC4J, home, CVU, and GNS commands
i	oldinst	*instance_name*	The old instance name	Service and server pool commands
i	scannumber	*scan_ordinal_number*	Ordinal number of the IP address for the SCAN	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
i	vip	*vip_name* or "*vip_name_list*"	VIP names	Node applications, GNS, VIP, network, listener, SCAN VIP, and SCAN listener commands
j	acfspath	*acfs_path_list*	Comma-delimited list of Oracle ACFS paths where the dependency on the database will be set	Database, instance, Oracle ASM, disk group, and file system commands
j	clbgoal	{SHORT|LONG}	The connection load balancing goal for a service	Service and server pool commands
k	netnum	*network_number*	The network number	Service and server pool commands Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands OC4J, home, CVU, and GNS commands
l	list		List all records in GNS	OC4J, home, CVU, and GNS commands
l	listener	*listener_name*	The name of a listener	ASM commands

Table A-2   (Cont.) Deprecated Single-Character Parameters for SRVCTL Commands

Single Letter	Long Form	Values	Description	Related Commands
l	loglevel	*log_level*	Specify the level (0-6) of logging that GNS should run with	OC4J, home, CVU, and GNS commands
l	min	*number*	The minimum size of the server pool	Service and server pool commands
l	onslocalport	*port_number*	Oracle Notification Service listening port for local client connections	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
l	role	*service_role*	Comma-delimited list of server roles within double quotation marks (`""`), where each role is one of `PRIMARY`, `PHYSICAL_STANDBY`, `LOGICAL_STANDBY`, or `SNAPSHOT_STANDBY`	Service and server pool commands
m	domain	domain_name	The domain for the database	Database, instance, Oracle ASM, disk group, and file system commands
m	failovermethod	{NONE|BASIC}	The failover method of a service	Service and server pool commands
m	multicastpost		The port on which the GNS daemon is listening for multicast requests	OC4J, home, CVU, and GNS commands
m	path	*mountpoint_path*	Mountpoint path	Database, instance, Oracle ASM, disk group, and file system commands
n	name		Advertise a name through GNS using the given address	OC4J, home, CVU, and GNS commands
n	node	*node_name*	The name of a specific node	Common
n	nodes	*node_list*	A comma-delimited list of node names	File system commands

Table A-2   (Cont.) Deprecated Single-Character Parameters for SRVCTL Commands

Single Letter	Long Form	Values	Description	Related Commands
n	dbname	*database_name*	The database name (DB_NAME), if different from the unique name specified by the -db parameter	Database, instance, Oracle ASM, disk group, and file system commands
n	scanname	scan_name	Fully-qualified SCAN name (includes the domain)	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
n	servers	*server_list*	A comma-delimited list of candidate server names	Service and server pool commands
n	targetnode	*node_name*	Node name to which to relocate the service	Service and server pool commands
o	oraclehome	*oracle_home*	*$ORACLE_HOME* path	Database commands
p	endpoints	[TCP:]*port _number*[/IPC: *key*][/NMP:*pipe _name*][/TCPS: *s_port*][/SDP: *port*]	SCAN listener endpoints	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
p	port		The port which the GNS daemon uses to communicate with the DNS server	OC4J, home, CVU, and GNS commands
p	rmiport	*port_number*	OC4J RMI port number	OC4J, home, CVU, and GNS commands
P	tafpolicy	{NONE|BASIC}	TAF policy specification	Service and server pool commands
p	spfile	*spfile_location*	Server parameter file path	Database, instance, Oracle ASM, disk group, and file system commands
q	notificatio n	{TRUE|FALSE}	Whether FAN is enabled for OCI connections	Service commands

Table A-2   (Cont.) Deprecated Single-Character Parameters for SRVCTL Commands

Single Letter	Long Form	Values	Description	Related Commands
q	query		Query GNS for the records belonging to a name	OC4J, home, CVU, and GNS commands
r	preferred	*preferred_list*	A comma-delimited list of preferred instances	Service and server pool commands
r	onsremoteport	*port_number*	Oracle Notification Service listening port for connections from remote hosts	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
r	relocate		Relocate the VIP	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
r	revert		Remove target node of failed online relocation request from the candidate server list of administrator-managed Oracle RAC One Node database	Relocate database
r	role	*role_type*	Role of the standby database: PRIMARY, PHYSICAL_STANDBY, LOGICAL_STANDBY, or SNAPSHOT_STANDBY	Database, instance, Oracle ASM, disk group, and file system commands
s	onsonly		Display Oracle Notification Service daemon configuration	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
s	skip		Skip checking the ports	Listener, SCAN, and SCAN listener.
s	statfile	*file_name*	The file path of the *state_file* created by a previously executed srvctl stop home command	OC4J, home, CVU, and GNS commands
s	status		Display the status of GNS	OC4J, home, CVU, and GNS commands

Table A-2   (Cont.) Deprecated Single-Character Parameters for SRVCTL Commands

Single Letter	Long Form	Values	Description	Related Commands
S	subnet	*subnet*/*net_mask*/[*if1*[|*if2*...]]	Network address specification for a network	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
s	service	*service_name* *service_name*_list	The name of a service A comma-delimited list of service names	Service and server pool commands
s	startoption	*start_options*	Startup options for the database (mount, open, read only)	Database, instance, Oracle ASM, disk group, and file system commands
t	checkinterval	*time_interval*	Interval in minutes between checks	OC4J, home, CVU, and GNS commands
t	edition	*edition_name*	The initial session edition of a service	Service and server pool commands
t	envs	"*name_list*"	A list of environment variables	Common
t	namevals	"*name=value*,..."	Names and values of environment variables	Common
T	nameval	"*name=value*"	Name and value of a single environment variable	Common
t	update	*instance_name*	The new instance name	Service and server pool commands
t	remoteservers	*host_name*[:*port_number*][,*host_name*[:*port_number*]...]	List of remote host name and port number pairs for Oracle Notification Service daemons outside this cluster	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
t	stopoption	*stop_options*	Stop options for the database (NORMAL, TRANSACTIONAL, IMMEDIATE, or ABORT)	Database, instance, Oracle ASM, disk group, and file system commands

Table A-2   (Cont.) Deprecated Single-Character Parameters for SRVCTL Commands

Single Letter	Long Form	Values	Description	Related Commands
t	toversion	*target_version*	Version to which you are downgrading	Database, instance, Oracle ASM, disk group, and file system commands
u	max	*number*	Maximum size of the server pool	Service and server pool commands
u	nettype	*network_type*	The network server type, which can be STATIC, DHCP, or MIXED	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
u	newinst		Add a new instance to the service configuration	Service commands
u	update		Update SCAN listeners to match the number of SCAN VIPs	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
u	user	*oracle_user*	Oracle user or other authorized user to mount and unmount file systems	Database, instance, Oracle ASM, disk group, and file system commands
v	verbose		Verbose output	Common
v	volume	*volume_name*	Name of a volume	Database, instance, Oracle ASM, disk group, and file system commands
V	versions			Common
w	failoverdelay	*number*	Failover delay	Service and server pool commands
w	nettype	*network_type*	The network server type, which can be STATIC, DHCP, or MIXED	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
w	timeout	*timeout*	Online relocation timeout in minutes	Database, instance, Oracle ASM, disk group, and file system commands

Table A-2   (Cont.) Deprecated Single-Character Parameters for SRVCTL Commands

Single Letter	Long Form	Values	Description	Related Commands
x	dtp	{TRUE | FALSE}	Whether to enable distributed transaction processing	Service and server pool commands
x	node	*node_name*	Node name (use this parameter only with noncluster databases)	Common
y	noprompt		Suppress the confirmation prompt	Common
y	policy	{AUTOMATIC | MANUAL}	Management policy for the resource	Database, instance, Oracle ASM, disk group, file system, service and server pool commands
z	failoverretry	*number*	Number of failover retries	Service and server pool commands
z	rmdepondisk		To remove a database's dependency upon disk groups	Database, instance, Oracle ASM, disk group, and file system commands

Miscellaneous SRVCTL Commands and Parameters

The following command parameters have been deprecated in this release:

Table A-3   Deprecated Commands and Parameters for SRVCTL

Command	Deprecated Parameters
srvctl modify asm	-node *node_name*
srvctl modify instance	-z Instead, use the -node option with the value set to " "
srvctl modify gns	[-ip *ip_address*] [-advertise *host_name* -address *address*] [-delete *host_name* -address *address*][-createalias *name* -alias *alias*] [-deletealias *alias*] Use the srvctl update gns command instead.

SRVCTL Command Reference

SRVCTL commands, object names, and parameters are case sensitive. Database, instance, listener, and service names are case insensitive and case preserving. You

cannot create listener names that differ only in case, such as LISTENER and listener. SRVCTL uses the following command syntax:

```
srvctl command object [parameters]
```

In SRVCTL syntax:

- *command* is a verb such as `start`, `stop`, or `remove`

- *object* (also known as a noun) is the target or object on which SRVCTL performs the command, such as database or instance. You can also use object abbreviations.

- *parameters* extend the use of a preceding command combination to include additional parameters for the command. For example, the `-instances` parameter indicates that a comma-delimited list of preferred instance names follows; the `-instance` parameter only permits one value and not a list of names. Do not use spaces between the items in a comma-delimited list.

> **Note:**
>
> If specifying a comma-delimited list in Windows, then you must enclose the list within double quotation marks (`" "`).

Table A-4   Summary of SRVCTL Commands

Command	Description
add	Adds node applications, databases, database instances, highly available virtual IPs (HAVIPs), listeners, single client access names (SCANs), server pools, services, or virtual IPs (VIPs).
config	Lists the configuration for node applications, database, HAVIPs, Oracle ASM instance, or services.
srvctl convert database	Converts a database either to or from an Oracle Real Application Clusters One Node (Oracle RAC One Node) database.
disable	Disables the database, database instance, or service.
srvctl downgrade database	Downgrades the database configuration after you manually downgrade the database.
enable	Enables the database, database instance, GNS, Oracle ASM instance, or service.
srvctl export gns	Exports data that a GNS instance uses to a file either for backup or so that you can move the instance to a new server cluster.
getenv	Displays the environment variable in the configuration for the node applications, database, VIP, listener or Oracle ASM.
srvctl import gns	Imports GNS information from a file.

Table A-4   (Cont.) Summary of SRVCTL Commands

Command	Description
modify	Modifies the node applications, database, database instance, GNS, HAVIPs, or service configuration.
predict	Evaluates the consequences of resource failure.
relocate	Relocates Oracle Flex ASM instances, GNS, HAVIPs, OC4J, SCANs, Oracle RAC One Node databases, servers, and VIPs from one node to another.
remove	Removes the node applications, database, database instance, GNS, Oracle ASM instance, or service.
setenv	Sets the environment variable in the configuration for the node applications, database, VIP, listener or Oracle ASM.
start	Starts the node applications, database, database instance, GNS, HAVIPs, Oracle ASM instance, or service.
status	Displays the status of the node applications, database, database instance, GNS, HAVIPs, NFS exports, Oracle ASM instance, or service.
stop	Stops the node applications, database, database instance, GNS, HAVIPs, Oracle ASM instance, or service.
unsetenv	Unsets the environment variable in the configuration for the node applications, database, VIP, listener or Oracle ASM.
update	Modifies a GNS instance.
srvctl upgrade database	Upgrades the configuration to the version of the software on which it is running.

Table A-5 lists the keywords that can be used for the *object* portion of SRVCTL commands. You can use either the full name or the abbreviation for each object keyword. The **Purpose** column describes the object and the actions that can be performed on that object.

Table A-5   Object Keywords and Abbreviations

Object	Keyword	Purpose
Oracle Automatic Storage Management instance	asm	To add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, and remove Oracle ASM instances.

Table A-5   (Cont.) Object Keywords and Abbreviations

Object	Keyword	Purpose
Cluster Verification Utility	cvu	To add, modify, list the configuration of, enable, disable, start, stop, relocate, obtain the status of, and remove CVU resources.
Database	database	To add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, and obtain the status of databases, and also to upgrade, downgrade, and remove database configuration information about databases.
Disk group	diskgroup	To add, modify, list the configuration of, enable, disable, start, stop, obtain the status of, and remove Oracle ASM disk groups
File system	filesystem	To add, modify, list the configuration of, enable, disable, stop, start, obtain the status of, and remove disk devices for Oracle Automatic Storage Management Cluster File System (Oracle ACFS) and generic file systems.
Grid Naming Service (GNS)	gns	To add, modify, list the configuration of, enable, disable, stop, start, obtain the status of, relocate, remove a GNS daemon.
High Availability VIP	havip	To add, modify, list the configuration of, enable, disable, stop, start, obtain the status of, relocate, and remove highly available VIPs (HAVIPs) (used for highly available NFS exports)
Home directory (for patching)	home	To start, stop, and obtain the status of all clusterware resources related to a Home directory.f
Instance	instance inst	To add, modify, enable, disable, start, stop, obtain the status of, and remove database instances.
Oracle ASM IOServer	ioserver	To add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, relocate, and remove Oracle ASM IOServers
Listener	listener lsnr	To add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, and remove listeners

Table A-5   (Cont.) Object Keywords and Abbreviations

Object	Keyword	Purpose
Management database	`mgmtdb`	To add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, relocate, and remove the database used by Cluster Health Monitor (CHM)
Management listener	`mgmtlsnr`	To add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, and remove the listener used by CHM
Network	`network`	To add, modify, list the configuration of, and remove a non-default Network **Note:** The node applications object, and the config and modify commands also manage the default network.
Node applications	`nodeapps`	To add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, and remove node applications
Oracle Grid Foundation OC4J container	`oc4j`	To add, modify, list the configuration of, enable, disable, start, stop, relocate, obtain the status of, and remove OC4J instances
Oracle Notification Service	`ons`	To add, configure, enable, start, obtain the status of, stop, disable, and remove Oracle Notification Service instances only for Oracle Restart
Single client access name (SCAN)	`scan`	To add, list the configuration of, modify, enable, disable, start, stop, relocate, obtain the status of, and remove SCAN VIPs
SCAN listener	`scan_listener`	To add, list the configuration of, modify, enable, disable, start, stop, relocate, obtain the status of, and remove SCAN listeners
Server	`server`	To obtain the status of and relocate a server in a different server pool.

Server pool	`serverpool`
	
	`(srvpool)`
		To add, modify, list the configuration of, obtain the status of, and remove server pools

Service	`service`	To add, modify, list the configuration of, enable, disable, start, stop, obtain the status of, relocate, and remove services

Table A-5   (Cont.) Object Keywords and Abbreviations

Object	Keyword	Purpose
Virtual IP	`vip`	To add, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, and remove a VIP
Oracle ACFS volume	`volume`	To list the configuration of, enable, disable, start, stop, obtain the status of, and remove an Oracle ACFS volume

add

The `srvctl add` command adds the configuration and the Oracle Clusterware applications to OCR for the cluster database, named instances, named services, or for the named nodes. To perform `srvctl add` operations, you must be logged in as the database administrator and be the Oracle account owner on Linux and UNIX systems, or you must be logged on as a user with Administrator privileges on Windows systems.

When adding an instance, the name that you specify with `-instance` must match the `ORACLE_SID` parameter. The database name given with `-database db_unique_name` must match the `DB_UNIQUE_NAME` initialization parameter setting. If `DB_UNIQUE_NAME` is unspecified, then match the `DB_NAME` initialization parameter setting. The default setting for `DB_UNIQUE_NAME` uses the setting for `DB_NAME`. Also, the domain name given with `-domain db_domain` must match the `DB_DOMAIN` setting.

Table A-6   srvctl add Command Summary

Command	Description
srvctl add cvu	Adds the Cluster Verification Utility resource to an Oracle Clusterware configuration
srvctl add database	Adds a database and configuration
srvctl add havip	Adds a highly available VIP to a cluster.
srvctl add instance	Adds one or more instance and configuration
srvctl add listener	Adds a listener to the node
srvctl add network	Adds a DHCP or static network
srvctl add nodeapps	Adds node applications
srvctl add oc4j	Adds OC4J instances
srvctl add ons	Adds Oracle Notification Service daemons
srvctl add scan	Adds SCAN VIPs
srvctl add scan_listener	Adds SCAN listeners
srvctl add service	Adds services

Table A-6 (Cont.) srvctl add Command Summary

Command	Description
srvctl add srvpool	Adds a server pool to a cluster
srvctl add vip	Adds a VIP to a node

srvctl add cvu

Adds the **Cluster Verification Utility (CVU)** to an Oracle Clusterware configuration.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl add cvu command with the following syntax:

```
srvctl add cvu [-checkinterval time_in_minutes]
```

Table A-7 srvctl add cvu Parameters

Parameter	Description
-checkinterval time_in_minutes	By default, CVU runs once every 6 hours and verifies the health of cluster. The shortest interval you can specify for CVU check is 10 minutes.

Usage Notes

- Run this command while logged in as the Grid home user.

Examples

To add CVU with a check interval of 6 hours, use the following command:

```
$ srvctl add cvu -checkinterval 360
```

srvctl add database

Adds a database configuration to Oracle Clusterware.

Syntax and Parameters

Use the srvctl add database command with the following syntax:

```
srvctl add database -db db_unique_name [-eval]
    -oraclehome oracle_home [-node node_name] [-domain domain_name]
    [-spfile spfile] [-pwfile password_file_path]
    [-dbtype {RACONENODE | RAC | SINGLE} [-server "server_list"]
    [-instance instance_name] [-timeout timeout]]
    [-role {PRIMARY | PHYSICAL_STANDBY | LOGICAL_STANDBY | SNAPSHOT_STANDBY"]
    [-startoption start_options] [-stopoption stop_options] [-dbname db_name]
    [-acfspath "acfs_path_list"] [-policy {AUTOMATIC | MANUAL | NORESTART}]
```

```
[-serverpool "server_pool_list" [-pqpool "pq_pool_list"]]
[-diskgroup "disk_group_list"] [-verbose]
```

Table A-8   srvctl add database Parameters

Parameter	Description
-db db_unique_name	Unique name for the database.
-eval	Use this parameter to hypothetically evaluate the impact of the command on the system. **Note:** You can only use this parameter with a policy-managed database.
-oraclehome oracle_home	The path for the Oracle database home directory.
-node node_name	Node name on which you want to register a noncluster, or single instance, Oracle database. **Note:** This parameter can be used only with Oracle Clusterware and can be used with the -serverpool parameter to create a policy-managed, noncluster database.
-domain db_domain	The domain for the database **Note:** You must use this parameter if you set the DB_DOMAIN initialization parameter for the database.
-spfile spfile	The path name of the database server parameter file.
-pwfile password_file_path	Enter the full path to the location of the password file.
-dbtype {RACONENODE | RAC | SINGLE}	The type of database you are adding: Oracle RAC One Node, Oracle RAC, or single instance. The default is RAC unless you specify the -node node_name parameter, and the -type parameter defaults to SINGLE.
-server server_list	List candidate servers for Oracle RAC One Node databases. **Notes:** You can use this parameter only with administrator-managed Oracle RAC One Node databases. If your Oracle RAC One Node database is policy managed, you *cannot* use this parameter.
-instance instance_name	Instance name prefix for Oracle RAC One Node databases. The default value for this parameter is the first 12 characters of the global unique name of the database. **Notes:** You can use this parameter only with administrator-managed Oracle RAC One Node databases. If your Oracle RAC One Node database is policy managed, you *cannot* use this parameter.

Table A-8   (Cont.) srvctl add database Parameters

Parameter	Description
`-timeout timeout`	Online database relocation timeout, in minutes, for Oracle RAC One Node databases. The default is `30`.
`-role {PRIMARY | PHYSICAL_STANDBY | LOGICAL_STANDBY | SNAPSHOT_STANDBY}`	The role of the database in an Oracle Data Guard configuration. The default is `PRIMARY`. **See Also:** *Oracle Data Guard Concepts and Administration* for more information about database roles
`-startoption start_options`	Startup options for the database, such as `OPEN`, `MOUNT`, and `NOMOUNT`. The default value is `OPEN`. **Notes:** • For multi-word startup options, such as `read only` and `read write`, separate the words with a space and enclose in double quotation marks (`""`). For example, `"read only"`. • When performing a switchover in an Oracle Data Guard configuration, the `-startoption` for a standby database that becomes a primary database is always set to `OPEN` after the switchover. **See Also:** *SQL*Plus User's Guide and Reference* for more information about startup options
`-stoption stop_options`	Stop options for the database, such as `NORMAL`, `TRANSACTIONAL`, `IMMEDIATE`, and `ABORT` **See Also:** *SQL*Plus User's Guide and Reference* for more information about shutdown options
`-dbname db_name`	The name of the database, if it is different from the unique name given by the `-db` parameter
`-acfspath "acfs_path_list"`	A single Oracle ACFS path or a comma-delimited list of Oracle ACFS paths enclosed in double quotation marks (`""`) where the database's dependency is set. Use this parameter to create dependencies on Oracle ACFS file systems other than `ORACLE_HOME`, such as for when the database uses `ORACLE_BASE` on a file system that is different from the `ORACLE_HOME` file system.

Table A-8   (Cont.) srvctl add database Parameters

Parameter	Description
`-policy {AUTOMATIC | MANUAL | NORESTART}`	Management policy for the database. • `AUTOMATIC` (default): The database is automatically restored to its previous running condition (started or stopped) upon restart of the database host computer. • `MANUAL`: The database is never automatically restarted upon restart of the database host computer. A `MANUAL` setting does not prevent Oracle Clusterware from monitoring the database while it is running and restarting it if a failure occurs. • `NORESTART`: Similar to the `MANUAL` setting, the database is never automatically restarted upon restart of the database host computer. A `NORESTART` setting, however, never restarts the database even if a failure occurs.
`-serverpool "server_pool_list" [-pqpool "pq_pool_list"]]`	Comma-delimited list of server pool names used to control database placement. If you do not specify this parameter, then it defaults to the Generic server pool. You can optionally also specify a comma-delimited list of parallel query server pool names to be used by the database. **Notes:** • This parameter can only be used with Oracle Clusterware. You can use this parameter with the `-node` parameter but the server pool must have `MAX_SIZE=1` and exactly one configured server (the one you specify in `-node`). • After you add server pools, you can assign services to them using the srvctl add service command.
`-diskgroup "disk_group_list"`	Comma-delimited list of Oracle ASM disk groups if database uses Oracle ASM storage

Examples

An example of this command to add a policy-managed Oracle RAC database is:

```
srvctl add database -db crm -oraclehome /u01/oracle/product/12c/mydb
    -domain example.com -spfile +diskgroup1/crm/spfilecrm.ora
    -role PHYSICAL_STANDBY -startoption MOUNT -dbtype RAC -dbname crm_psd
    -policy MANUAL -serverpool "svrpool1,svrpool2" -diskgroup "dgrp1,dgrp2"
```

An example of this command to add an administrator-managed database is:

```
srvctl add database -db crm -oraclehome /u01/oracle/product/12c/mydb
-domain example.com
```

srvctl add havip

Adds highly available VIPs (HAVIPs) (used for highly available NFS exports) to a cluster.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl add havip` command with the following syntax:

```
srvctl add havip -id id -address {host_name | ip_address}
    [-netnum network_number] [-description text] [-skip] [-homenode node_name]
```

Table A-9   srvctl add havip Parameters

Parameter	Description
`-id havip_name`	Specify a unique ID for the HAVIP resource.
`-address {host_name |ip_address}`	Specify either a host name or an IPv4 IP address for the HAVIP you are going to create. **Note:** You must configure the IPv4 IP address with a non-DHCP, non-round robin DNS address.
`-netnum network_number`	Optionally, you can specify a network resource upon which the HAVIP depends. The default value for this parameter is 1.
`-description text`	Optionally, you can specify a text description of the HAVIP.
`-skip`	Optionally, you can specify this parameter to skip the reachability check for the IP address.
`-homenode node_name`	Optionally, you can specify a preferred node or an empty string to clear the home node.

Usage Notes

- You must run this command as `root` user on Linux and UNIX platforms.

- The address for the HAVIP must be in the same subnet as the network on which it is created.

- Oracle does not support using IPv6 addresses.

Example

An example of this command is:

```
# srvctl add havip -id myhavip -address 192.0.2.1 -netnum 2 -description
    "My HA VIP" -skip -homenode myNode2
```

srvctl add instance

Adds a configuration for an instance to your cluster database configuration.

You can only use this command for administrator-managed databases. If you have a policy-managed database, then use the srvctl modify srvpool command to add an

instance to increase either the maximum size, minimum size, or both, of the server pool used by the database.

> **Note:**
>
> - This command increments the CARDINALITY resource attribute.
>
> - This command is only available with Oracle Clusterware and Oracle RAC.
>
> - If you attempt to use this command on an Oracle RAC One Node database, then the command returns an error stating you must convert the database to Oracle RAC.

Syntax and Parameters

Use the srvctl add instance command with the following syntax:

```
srvctl add instance -db db_unique_name -instance instance_name
     -node node_name [-force]
```

Table A-10   srvctl add instance Parameters

Parameter	Description
-db db_unique_name	The unique name of the database you are adding the instance to
-instance instance_name	The name of the instance you are adding
-node node_name	The name of the node on which you are creating the instance
-force	Optionally, you can force the add operation, even though some resources will be stopped.

Examples

Examples of this command are:

```
$ srvctl add instance -db crm -instance crm01 -node gm01
$ srvctl add instance -db crm -instance crm02 -node gm02
$ srvctl add instance -db crm -instance crm03 -node gm03
```

srvctl add listener

Adds a listener to every node in a cluster.

Syntax and Parameters

Use the srvctl add listener command with one of the following syntax models:

To create an Oracle Database listener:

```
srvctl add listener [-listener listener_name] [-netnum network_number]
  [-oraclehome Oracle_home] -user user_name [-endpoints "[TCP:]port_list[/IPC:key]
  [/NMP:pipe_name][/TCPS:s_port][/SDP:port]"] [-skip]
```

To create an Oracle ASM listener:

```
srvctl add listener [-listener listener_name] -asmlistener [-subnet subnet]
 [-endpoints "[TCP:]port_list[/IPC:key][/NMP:pipe_name][/TCPS:s_port][/SDP:port]"]
 [-skip]
```

To create a Leaf listener:

```
srvctl add listener [-listener listener_name] -leaflistener [-subnet subnet]
 [-endpoints "[TCP:]port_list[/IPC:key][/NMP:pipe_name][/TCPS:s_port][/SDP:port]"]
 [-skip]
```

To create a SCAN listener, use the srvctl add scan_listener command. To create a management listener (for the CHM repository), use the srvctl add mgmtlsnr command.

Table A-11   srvctl add listener Parameters

Parameter	Description
-listener listener_name	Specify a listener name. This parameter is optional.
	If you do not specify this parameter, then the name of the listener defaults to LISTENER for a database listener, LISTENER_ASM for an Oracle ASM listener, and LISTENER_LEAF for a Leaf Node listener.
-netnum network_number	The optional network number from which VIPs are obtained. If not specified, the VIPs are obtained from the same default network from which the nodeapps VIP is obtained.
	Note: Use this parameter when you add an Oracle Database listener.
-oraclehome oracle_home	Specify an Oracle home for the cluster database. If you do not include this parameter, then SRVCTL uses the Grid home by default.
	Note: Use this parameter when you add an Oracle Database listener.

Table A-11   (Cont.) srvctl add listener Parameters

Parameter	Description
`-user user_name`	Use this parameter to set the user that will run the listener to a less privileged user. Oracle recommends using this parameter to increase security. **Notes:** • You must be logged in as `root` to run this command and specify the `-user` parameter. • Use this parameter when you add an Oracle Database listener. • When you use the `-user` parameter, ensure the following: The listener log directory in `ORACLE_BASE` and the `Grid_home`/network/admin/`user_name` directory must both exist on each node before you can use this parameter. Additionally, `user_name` must have read, write, and execute permission in the directory. The `ORACLE_HOME`/network/admin/`endpoints_listener.ora` file exists and `user_name` has read and write permission on it. The `$ORACLE_BASE/diag/tnslsnr/``host_name`/`lower_case_listener_name` directory exists and `user_name` has read, write, and execute permission on it.

`-endpoints "[TCP:]port_list[/`
`IPC:key]`
`[/NMP:pipe_name][/TCPS:s_port]`
`[/SDP:port]`
	Protocol specifications for the listener. `port_list` is a comma-delimited list of TCP ports or listener endpoints.
	If you do not specify the `-endpoints` parameter for an Oracle Database listener, then SRVCTL searches for a free port between 1521 and 1540.
	Note: You can modify this attribute using Online Resource Attribute Modification.

`-skip`	Indicates you want to skip the checking of ports.
`-asmlistener`	Specifies the listener type as an Oracle ASM listener. If you do not specify the `-listener` parameter, then the name of the Oracle ASM listener defaults to `LISTENER_ASM`. **Note:** You can only use this parameter with Oracle Clusterware.
`-leaflistener`	Specifies the listener type is a Leaf Node listener. If you do not specify the `-listener` parameter, then the name of the Leaf Node listener defaults to `LISTENER_LEAF`. **Note:** You can only use this parameter with Oracle Clusterware.

Table A-11   (Cont.) srvctl add listener Parameters

Parameter	Description
-subnet *subnet*	Specifies the subnet to use for an Oracle ASM or Leaf Node listener. **Note:** You can only use this parameter with Oracle Clusterware.

Usage Notes

- You must run this command as `root` user on Linux and UNIX platforms when you specify the `-user` parameter.

Example

The following command adds a listener named `listener112` that is listening on ports 1341, 1342, and 1345 and runs from the Oracle home directory on every node in the cluster:

```
$ srvctl add listener -listener listener112 -endpoints "1341,1342,1345"
-oraclehome /u01/app/oracle/product/12.1.0/db1
```

srvctl add mgmtlsnr

Adds a management listener resource (for CHM) to the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl add mgmtlsnr` command with the following syntax:

```
srvctl add mgmtlsnr [-endpoints "[TCP:]port_list[/IPC:key][/NMP:pipe_name]
 [/TCPS:s_port][/SDP:port]"] [-skip]
```

Table A-12   srvctl add mgmtlsnr Parameters

Parameter	Description
-endpoints "[TCP:]*port_list* [/IPC:*key*][/NMP:*pipe_name*] [/TCPS:*s_port*][/SDP:*port*]	Protocol specifications for the listener. *port_list* is a comma-delimited list of TCP ports or listener endpoints. **Note:** You can modify this attribute using Online Resource Attribute Modification.
-skip	Indicates you want to skip the checking of ports.

Example

The following command adds a management listener that is listening on port 1341 to the cluster:

```
$ srvctl add mgmtlsnr -endpoints "TCP:1341"
```

srvctl add network

Adds a static or dynamic network. If your server connects to more than one network, then you can use this command to configure an additional network interface for Oracle RAC, allowing you to create VIPs on multiple public networks. You can also use the LISTENER_NETWORKS database initialization parameter to control client redirects to the appropriate network

---

Note:

- This command is only available with Oracle Clusterware.

- Oracle only supports DHCP-assigned networks for the default network, not for subsequent networks.

Syntax and Parameters

Use the srvctl add network command with the following syntax:

```
srvctl add network [-netnum net_number] -subnet subnet/netmask[/if1[|if2|...]]
   [-nettype {static | dhcp | autoconfig | mixed}]
   [-leaf] [-pingtarget "ping_target_list"] [-verbose]
```

Table A-13   srvctl add network Parameters

Parameter	Description
-netnum net_number	Specify a network number. The default is 1.
-subnet subnet/netmask [/if1[|if2|...]]	Defines a subnet. If you do not specify any interface names, then the network uses any interface on the given subnet.
	For IPv6, netmask is a prefix length, such as 64.
-nettype {static|dhcp|autoconfig| mixed}	Specify the network type: static, dhcp, autoconfig, or mixed.
	If you specify static for the network type, then you must provide the virtual IP address using the srvctl add vip command.
	If you specify dhcp for the network type, then the VIP agent obtains the IP address from a DHCP server.
	If you specify autoconfig for the network type, then the VIP agent generates a stateless IPv6 address for the network. You can only use autoconfig for IPv6 networks. If the subnet/netmask specification is not for an IPv6 address, then SRVCTL returns an error.
	If you specify mixed for the network type, then the VIP resource uses both a static IP address and an IP address obtained dynamically, either from a DHCP server for IPv4 or using stateless autoconfiguration for IPv6.

Table A-13   (Cont.) srvctl add network Parameters

Parameter	Description
-leaf	Specify the -leaf parameter when you want to create an application VIP network resource that runs only on Leaf Nodes and is used by application VIPs.
[-pingtarget "*ping_target_list*"]	Specify a comma-delimited list of IP addresses or host names to ping.
-verbose	Verbose output.

Usage Notes

- On Linux and UNIX systems, you must be logged in as root and on Windows, you must be logged in as a user with Administrator privileges to run this command.

Example

An example of this command is:

```
# srvctl add network -netnum 3 -subnet 192.168.3.0/255.255.255.0
```

srvctl add nodeapps

Adds a node application configuration to the specified node.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl add nodeapps command with one the following syntax models, specifying either a specific node and VIP or a specific subnet and netmask:

```
srvctl add nodeapps {-node node_name -address {vip_name | ip_address}/netmask[/if1[|
if2|..]] [-skip]}
    [-emport em_port] [-onslocalport ons_local_port]
    [-onsremoteport ons_remote_port] [-onshostport hostname_port_list]
    [-remoteservers hostname_port_list [-verbose]


srvctl add nodeapps -subnet subnet/netmask[/if1[|if2|...]] [-emport em_port]
    [-onslocalport ons_local_port] [-onsremoteport ons_remote_port]
    [-onshostport hostname_port_list] [-remoteservers hostname_port_list]
    [-verbose]
```

Table A-14   srvctl add nodeapps Parameters

Parameter	Description
`-node node_name`	The name of the node on which you want to create the node application. Node name is optional and unnecessary if you run the command on the local node.
`-address {vip_name	ip_addr} /netmask[/if1 [|if2|...]]`	This specification creates a traditional VIP node application on the specified node. **Note:** This parameter must be used for upgrade configurations and new, non-DHCP configurations.
`-skip`	Specify this parameter to skip checking the reachability of the VIP address.
`-subnet subnet/netmask [/if1[|if2 |...]]`	Creates a DHCP subnet. If you do not specify any interface names, then the VIPs use any interface on the given subnet.
`-emport em_port`	Local port on which Oracle Enterprise Manager listens. The default port is 2016.
`-onslocalport ons_local_port`	The Oracle Notification Service daemon listener port on its node. If you do not specify this value, the Oracle Notification Service daemon listener port defaults to 6100. **Note:** The local port and remote port must each be unique.
`-onsremoteport ons_remote_port`	The port number for remote Oracle Notification Service daemon connections. If you do not specify a port number, the default value of 6200 is used for the Oracle Notification Service remote port. **Note:** The local port and remote port must each be unique.
`-onshostport host_port_list`	A list of `host[:port]` pairs of remote hosts that are part of the Oracle Notification Service network but are not part of the Oracle Clusterware cluster **Note:** If `port` is not specified for a remote host, then `ons_remote_port` is used.
`-remoteservers host_port_list`	A list of `host[:port]` pairs for Oracle Notification Service daemons on servers that are not in the cluster.
`-verbose`	Verbose output

Usage Notes

- On Linux and UNIX systems, you must be logged in as `root` and on Windows, you must be logged in as a user with Administrator privileges to run this command.

Example

An example of this command is:

```
# srvctl add nodeapps -node crmnode1 -address 1.2.3.4/255.255.255.0
```

srvctl add oc4j

Adds an OC4J instance to all the nodes in the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

```
srvctl add oc4j [-verbose]
```

This command has only one parameter, `-verbose`, which displays verbose output.

Example

An example of this command is:

```
srvctl add oc4j
```

srvctl add ons

Adds an Oracle Notification Service daemon to an Oracle Restart configuration.

> **Note:**
>
> This command is only available with Oracle Restart.

Syntax and parameters

```
srvctl add ons [-l ons_local_port] [-r ons_remote_port]
    [-t host[:port][,host[:port]][...]] [-v]
```

Table A-15   srvctl add ons Parameters

Parameter	Description
`-l ons_local_port`	The Oracle Notification Service daemon listening port for local client connections **Note:** The local port and remote port must each be unique.

Table A-15   (Cont.) srvctl add ons Parameters

Parameter	Description
`-r ons_remote_port`	The Oracle Notification Service daemon listening port for connections from remote hosts **Note:** The local port and remote port must each be unique.
`-t host[:port] [,host[:port]][...]`	A list of comma-delimited `host:port` pairs of remote hosts that are part of the Oracle Notification Service network but are not part of the Oracle Clusterware cluster **Note:** If `port` is not specified for a remote host, then `ons_remote_port` is used.
`-v`	Verbose output

Example

An example of this command is:

```
$ srvctl add ons -l 6200
```

srvctl add scan

Adds Oracle Clusterware resources for the given SCAN. This command creates the same number of SCAN VIP resources as the number of IP addresses that SCAN resolves to, or 3 when `network_number` identifies a **dynamic network** and Oracle GNS configuration. For static networks, the addresses to which the SCAN resolves in DNS must match the address type of the subnet. For an IPv4 network, the SCAN must resolve to IPv4 addresses.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl add scan` command with the following syntax:

```
srvctl add scan -scanname scan_name [-netnum network_number]
```

Table A-16   srvctl add scan Parameters

Parameter	Description
`-scanname scan_name`	A fully-qualified host name, which includes the domain name. If the network is dynamic, then you do not have to use fully-qualified host name but, if you choose to do so, then the domain must be the GNS subdomain. **Note:** You can modify this attribute using Online Resource Attribute Modification.

Table A-16 (Cont.) srvctl add scan Parameters

Parameter	Description
-netnum *network_number*	The optional network number from which SCAN VIPs are obtained. If you do not specify this parameter, then the SCAN VIPs are obtained from the same default network from which the `nodeapps` VIP is obtained.

Example

An example of this command is:

```
# srvctl add scan -scanname scan.mycluster.example.com
```

srvctl add scan_listener

Adds Oracle Clusterware resources for the SCAN listeners. The number of SCAN listener resources created is the number of SCAN VIP resources.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl add scan_listener` command with the following syntax:

```
srvctl add scan_listener [-listener lsnr_name_prefix] [-skip] [-netnum
network_number]
 [-enpoints "[TCP:]port_list[/IPC:key][/NMP:pipe_name][/TCPS:s_port][/SDP:port]"]
 [-invitednodes node_list] [-invitedsubnets subnet_list]
```

Table A-17 srvctl add scan_listener Parameters

Parameter	Description
-listener *lsnr_name_prefix*	The SCAN listener name prefix.
-skip	Use this parameter to skip checking of the ports.
-netnum *network_number*	The optional network number from which SCAN VIPs are obtained. If you do not specify this parameter, then the SCAN VIPs are obtained from the same default network from which the `nodeapps` VIP is obtained.
-endpoints "[TCP:]*port_list*[/IPC:*key*] [/NMP:*pipe_name*][/TCPS:*s_port*] [/SDP:*port*]"	Protocol specifications for the listener. *port_list* is a comma-delimited list of TCP ports or listener endpoints. If this parameter is not specified, then the default TCP port of 1521 is used. **Note:** You can modify this attribute using Online Resource Attribute Modification.

Table A-17   (Cont.) srvctl add scan_listener Parameters

Parameter	Description
-invitednodes *node_list*	Use this parameter to specify a comma-delimited list of host names from outside the cluster that are allowed to register with the SCAN listener.
-invitedsubnets *subnet_list*	Use this parameter to specify a comma-delimited list of subnets from outside the cluster that are allowed to register with the SCAN listener.

Example

An example of this command is:

```
# srvctl add scan_listener -listener myscanlistener
```

srvctl add service

Adds services to a database and assigns them to instances. If you have multiple instances of a cluster database on the same node, then always use only one instance on that node for all of the services that node manages.

> **Note:**
>
> The srvctl add service command does not accept placement parameters for Oracle RAC One Node databases.

Syntax and Parameters

Use the srvctl add service command with one of the following syntax models:

To add a service to a policy-managed database:

```
srvctl add service -db db_unique_name -service service_name [-eval]
   -serverpool server_pool [-cardinality {UNIFORM | SINGLETON}]
   [-edition edition_name] [-netnum network_number]
   [-role "[PRIMARY][,PHYSICAL_STANDBY][,LOGICAL_STANDBY][,SNAPSHOT_STANDBY]"
   [-policy {AUTOMATIC | MANUAL}] [-notification {TRUE | FALSE}]
   [-clbgoal {SHORT | LONG}] [-failovertype {NONE|SESSION|SELECT|TRANSACTION}]
   [-rlbgoal {NONE | SERVICE_TIME | THROUGHPUT}] [-dtp {TRUE | FALSE}]
   [-failovermethod {NONE | BASIC}] [-failoverretry failover_retries]
   [-failoverdelay failover_delay] [-pdb pluggable_database]
   [-sql_translation_profile sql_translation_profile]
   [-global {TRUE | FALSE}] [-maxlag max_lag_time] [-commit_outcome {TRUE|FALSE}]
   [-retention retention_time] [-replay_init_time replay_initiation_time]
   [-session_state {STATIC | DYNAMIC}] [-pqservice pq_service]
   [-pqpool pq_pool_list] [-force]
```

To add a service to an administrator-managed database:

```
srvctl add service -database db_unique_name -service service_name [-eval]
   -preferred preferred_list -available available_list] [-netnum network_number]
   [-tafpolicy {BASIC | NONE | PRECONNECT}] [-edition edition_name]
   [-role "[PRIMARY][,PHYSICAL_STANDBY][,LOGICAL_STANDBY][,SNAPSHOT_STANDBY]"
```

```
[-policy {AUTOMATIC | MANUAL}] [-notification {TRUE | FALSE}]
[-clbgoal {SHORT | LONG}] [-failovertype {NONE|SESSION|SELECT|TRANSACTION}]
[-rlbgoal {NONE | SERVICE_TIME | THROUGHPUT}] [-dtp {TRUE | FALSE}]
[-failovermethod {NONE | BASIC}] [-failoverretry failover_retries]
[-failoverdelay failover_delay] [-pdb pluggable_database]
[-sql_translation_profile sql_translation_profile]
[-global {TRUE | FALSE}] [-maxlag max_lag_time] [-commit_outcome {TRUE|FALSE}]
[-retention retention_time] [-replay_init_time replay_initiation_time]
[-session_state {STATIC|DYNAMIC}] [-pqservice pq_service]
[-pqpool pq_pool_list] [-force] [-verbose]
```

To update the preferred and available lists of an existing service:

```
srvctl add service -db db_unique_name -service service_name
  -update {-prefered preferred_list | -available available_list} [-force]
  [-verbose]
```

Table A-18 lists and describes all the `srvctl add service` parameters and whether they can be used when adding a service to either an Oracle RAC database or noncluster database.

Table A-18   srvctl add service Parameters

Parameter	Description
-db db_unique_name	Unique name for the database.
-service service_name	The service_name.service_domain should be unique within the cluster unless you want to spread connections across multiple databases that offer the same service. If you do not specify the service domain as part of the service name (such as sales.example.com), then the DB_DOMAIN database attribute is appended to the service name. **Note:** The -service parameter has a 4 kilobyte (KB) limit for its value. Therefore, the total length of the names of all services assigned to an instance cannot exceed 4 KB.
-eval	Use this parameter to hypothetically evaluate the impact of the command on the system. **Note:** You can only use this parameter with a policy-managed service.

Table A-18   (Cont.) srvctl add service Parameters

Parameter	Description
`-edition edition_name`	The initial session edition of the service. When an edition is specified for a service, all subsequent connections that specify the service use this edition as the initial session edition. However, if a session connection specifies a different edition, then the edition specified in the session connection is used for the initial session edition. SRVCTL does not validate the specified edition name. During connection, the connect user must have USE privilege on the specified edition. If the edition does not exist or if the connect user does not have USE privilege on the specified edition, then an error is raised.
`-sql_translation_profile profile_name`	Use this parameter to specify a **SQL translation profile** for a service that you are adding after you have migrated applications from a non-Oracle database to an Oracle database. This parameter corresponds to the SQL translation profile parameter in the DBMS_SERVICE service attribute. **Notes:** • Before using the SQL translation feature, you must migrate all server-side application objects and data to the Oracle database. • Use the srvctl config service command to display the SQL translation profile. **See Also:** *Oracle Database Migration Guide* for more information about SQL translation
`-preferred preferred_list`	A list of preferred instances on which the service runs when the database is administrator managed. The list of preferred instances must be mutually exclusive with the list of available instances. **Note:** This parameter can be used only with Oracle RAC and only for administrator-managed databases.
`-available available_list`	A list of available instances to which the service fails over when the database is administrator managed. The list of available instances must be mutually exclusive with the list of preferred instances. **Note:** This parameter can be used only with Oracle RAC and only for administrator-managed databases.

Table A-18 (Cont.) srvctl add service Parameters

Parameter	Description
`-serverpool server_pool`	The name of a server pool used when the database is policy managed. **Note:** This parameter can be used only with Oracle RAC and only for policy-managed databases.
`-cardinality {UNIFORM | SINGLETON}`	The cardinality of the service, either `UNIFORM` (offered on all instances in the server pool) or `SINGLETON` (runs on only one instance at a time). **Notes:** • This parameter can be used only with Oracle RAC and only for policy-managed databases. • For policy-managed Oracle RAC One Node databases, all services must be `SINGLETON`.
`-netnum network_number`	Use this parameter to determine on which network this service is offered. The service is configured to depend on VIPs from the specified network. **Note:** This parameter can be used only with Oracle RAC and Oracle RAC One Node database configurations.
`-tafpolicy {BASIC | NONE` `  | PRECONNECT}`	TAF policy specification (for administrator-managed databases only). **Note:** You can only use `PRECONNECT` when you specify the `-preferred` and `-available` parameters.

`-role "[PRIMARY][,PHYSICAL_STANDBY]`
`[,LOGICAL_STANDBY]`
`[,SNAPSHOT_STANDBY]"`
	The service role. You can specify one or more roles in a comma-delimited list.
	
	Use this option to indicate that the service should only be automatically started upon database open when the Oracle Data Guard database role matches one of the specified service roles.
	
	Using SRVCTL to manually start a service is not affected by the service role.
	
	Note: The `-role` parameter is only used at database startup and by the Oracle Data Guard Broker. All manual service startup must specify the name of the service to be started by the user.
	
	See Also:
	• *Oracle Data Guard Broker* if you specify `-role`
	• *Oracle Data Guard Concepts and Administration* for more information about database roles

Table A-18   (Cont.) srvctl add service Parameters

Parameter	Description
-policy {AUTOMATIC | MANUAL}	Service management policy.
	If AUTOMATIC (the default), then the service is automatically started upon restart of the database, either by a planned restart (with SRVCTL) or after a failure. Automatic restart is also subject to the service role, however (the -role parameter).
	If MANUAL, then the service is never automatically restarted upon planned restart of the database (with SRVCTL). A MANUAL setting does not prevent Oracle Clusterware from monitoring the service when it is running and restarting it if a failure occurs.
	Note: Using CRSCTL to stop and start the Oracle Clusterware restarts the service in the same way that a failure does.
-notification {TRUE | FALSE}	Enable Fast Application Notification (FAN) for OCI connections.
-dtp {TRUE | FALSE}	Indicates whether Distributed Transaction Processing should be enabled for this service. This service will either be a singleton service in a policy-managed database or a preferred service on a single node in an administrator-managed database.
	Note: This parameter can be used only with Oracle RAC.
-clbgoal {SHORT | LONG}	Connection Load Balancing Goal. Use a value of SHORT for this parameter for run-time load balancing, or if using an integrated connection pool. Use a value of LONG for this parameter for long running connections, such as batch jobs, that you want balanced by the number of sessions per node for the service.
-rlbgoal {NONE | SERVICE_TIME | THROUGHPUT}	Runtime Load Balancing Goal (for the Load Balancing Advisory). Set this parameter to SERVICE_TIME to balance connections by response time. Set this parameter to THROUGHPUT to balance connections by throughput.
-failovertype {NONE | SESSION | SELECT | TRANSACTION}	Failover type.
	To enable Application Continuity for Java, set this parameter to TRANSACTION. To enable TAF for OCI, set this parameter to SELECT or SESSION.
	Note: If you set -failovertype to TRANSACTION, then you must set -commit_outcome to TRUE.

Table A-18   (Cont.) srvctl add service Parameters

Parameter	Description
`-failovermethod {NONE	BASIC}`	TAF failover method (for backward compatibility only). If the failover type (`-failovertype`) is set to a value other than `NONE`, then you should choose `BASIC` for this parameter. **Note:** This parameter can be used only with Oracle RAC.
`-failoverretry failover_retries`	For Application Continuity and TAF, this parameter determines the number of attempts to connect after an incident.
`-failoverdelay failover_delay`	For Application Continuity and TAF, this parameter specifies the time delay (in seconds) between reconnect attempts per incident at failover.
`-pdb pluggable_database`	The name of the pluggable database (PDB). **Note:** You can specify a PDB property when you create or modify a service. The PDB property associates the service with the specified PDB. You can view the PDB property for a service by querying the ALL_SERVICES data dictionary view or, when using the SRVCTL utility, by running the srvctl config service command. When create or modify a service with the specified PDB, SRVCTL does not check if the PDB exists.Before running this command, you must ensure that the PDB exists.
`-global {TRUE	FALSE}`	Indicates whether this is a Global Data Services service. **Note**: This parameter can only be used with Global Data Services.
`-maxlag maximum_lag_time`	Maximum replication lag time in seconds. Must be a non-negative integer. The default value is `ANY`.
`-commit_outcome {TRUE	FALSE}`	Enable Transaction Guard; when set to `TRUE`, the commit outcome for a transaction is accessible after the transaction's session fails due to a recoverable outage.
`-retention retention_time`	If `-commit_outcome` is set to `TRUE`, then this parameter determines the amount of time (in seconds) that the commit outcome is retained in the database.

Table A-18   (Cont.) srvctl add service Parameters

Parameter	Description
-replay_init_time *replay_initialization_time*	For Application Continuity, this parameter specifies the difference between the time, in seconds, of original execution of the first operation of a request and the time that the replay is ready to start after a successful reconnect. Application Continuity will not replay after the specified amount of time has passed. This parameter is intended to avoid the unintentional execution of a transaction when a system is recovered after a long period. The default is 5 minutes (300). The maximum value is 24 hours (86400).If the -failover_type parameter is not set to TRANSACTION, then you cannot use this parameter.
-session_state {STATIC | DYNAMIC}	For Application Continuity, this parameter specifies whether the session state that is not transactional is changed by the application. Oracle recommends a setting of DYNAMIC for most applications.
	Note: This parameter is considered only if -failovertype is set to TRANSACTION for Application Continuity. It describes how non-transactional is changed during a request. Examples of session state are NLS settings, optimizer preferences, event settings, PL/SQL global variables, temporary tables, advanced queues, LOBs, and result cache. If non-transactional values change after the request starts, then use the default, DYNAMIC. Most applications should use DYNAMIC mode. If you are unsure, then use DYNAMIC mode.
-pqservice *pq_service*	A comma-delimited list of parallel query service names.
-pqpool *pq_pool_list*	A comma-delimited list of parallel query server pool names
-update {-preferred *new_preferred_instance* | -available *new_available_instance*}	Add a new preferred or available instance to the service configuration. -preferred specifies the name of the instance to add to the list of preferred instances for the service. -available specifies the name of the instance to add to the list of available instances for the service.
-verbose	Display verbose output.
-force	Force the add operation even though a listener is not configured for a network.

Examples

Use this example syntax to add the `gl.example.com` service to the `my_rac` database with Fast Application Notification enabled for OCI connections, a failover method of `BASIC`, a Connection Load Balancing Goal of `LONG`, a failover type of `SELECT`, and 180 failover retries with a failover delay of 5 seconds:

```
srvctl add service -db my_rac -service gl.example.com -notification TRUE \
-failovermethod BASIC -failovertype SELECT -failoverretry 180 -failoverdelay 5 \
-clbgoal LONG
```

Use this example syntax to add a named service to a database with preferred instances and available instances and enabled for TAF:

```
srvctl add service -db crm -service sales -preferred crm01,crm02 -available crm03\
 -tafpolicy BASIC
```

srvctl add srvpool

Adds a server pool that is configured to host Oracle databases to a cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl add srvpool` command with the following syntax:

```
srvctl add srvpool -serverpool server_pool_name [-eval]
    [-importance importance] [-min min_size] [-max max_size]
    [-servers "node_list" | -category server_category] [-force] [-verbose]
```

Table A-19   srvctl add srvpool Parameters

Parameter	Description
-serverpool *server_pool_name*	The name of the server pool.
-eval	Use this parameter to hypothetically evaluate the impact of the command on the system.
-importance *importance*	The importance of the server pool (default value is 0).
-min *min_size*	The minimum size of the server pool (default value is 0).
-max *max_size*	The maximum size of the server pool. The default value is –1, which indicates that the size is unlimited.

Table A-19   (Cont.) srvctl add srvpool Parameters

Parameter	Description
-servers "*node_list*"	A comma-delimited list of candidate node names enclosed in double quotation marks (" "). The server pool will only include nodes on the candidate list, but not all nodes on the candidate list will necessarily be in the server pool. **Note:** In Oracle Database 12*c*, servers are assigned to server pools according to the value of the -category parameter.
-category *server_category*	The category of servers to use for the server pool, or " " for the empty category value. You can use hub for ora.hub.category, leaf for ora.leaf.category, or you can define your own category. The default is ora.hub.category.
-force	Add the server pool, even if it requires stopping resources in other server pools.
-verbose	Display verbose output.

Usage Notes

SRVCTL prepends "ora." to the name of the server pool.

Example

An example of this command is:

```
$ srvctl add srvpool -serverpool SP1 -importance 1 -min 3 -max 7
```

srvctl add vip

Adds a VIP to a node.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl add vip command with the following syntax:

```
srvctl add vip -node node_name -address {VIP_name|ip}/netmask[/if1[|if2|...]]
      -netnum network_number [-skip] [-verbose]
```

Table A-20   srvctl add vip Parameters

Parameter	Description
-node *node_name*	The name of the node on which you are adding the VIP.
-address {*VIP_name*|*ip*}/*netmask* [/*if1*[|*if2*|...]]	This specification creates a traditional VIP node application on the specified node. You can specify one *VIP_name* or address, along with an IPv4 netmask or IPv6 prefix length.
-netnum *network_number*	The network number from which VIPs are obtained. The default network number is 1.
-skip	Specify this parameter to skip checking the reachability of the VIP address.
-verbose	Verbose output

Note:

You cannot have multiple VIPs on the same net number (subnet or interface pair) on the same node.

Example

An example of this command is:

```
# srvctl add network -netnum 2 -subnet 192.168.16.0/255.255.255.0
# srvctl add vip -node node7 -address 192.168.16.17/255.255.255.0 -netnum 2
```

The first command creates a network number, 2, and the second command adds a VIP to this network. You can specify the network number after the -netnum parameter in other SRVCTL commands.

config

The srvctl config command displays the configuration stored in the Oracle Clusterware resource attributes.

Note:

If you disabled an object for which you are trying to obtain configuration information using the srvctl disable *object* -n *node_name* command, then remember that using the srvctl disable *object* -n *node_name* command on a per-node basis in your cluster to disable an object throughout the cluster *is different from* globally disabling an object using the srvctl disable *object* command without the -n parameter. In the former case, the srvctl config *object* command may report that the object is still enabled.

> **See Also:**
>
> "disable" for more information about the srvctl disable command

Table A-21   srvctl config Summary

Command	Description
srvctl config cvu	Displays the time interval between CVU checks
srvctl config database	Displays the configuration information of the cluster database
srvctl config havip	Displays configuration information for highly available VIP resources
srvctl config listener	Displays a list of configured listeners that are registered with Oracle Clusterware on a given node
srvctl config network	Displays the network configuration for the cluster
srvctl config nodeapps	Displays the configuration information for the node applications
srvctl config oc4j	Displays the configuration of the OC4J instance
srvctl config ons	Displays configuration information for Oracle Notification Service
srvctl config scan	Displays the configuration information for SCAN VIPs
srvctl config scan_listener	Displays the configuration information for SCAN listeners
srvctl config service	Displays the configuration information for the services
srvctl config srvpool	Displays configuration information for a specific server pool
srvctl config vip	Displays the configuration information for the VIP
srvctl config volume	Displays the configuration information for a volume device resource

srvctl config cvu

Displays the time interval, in minutes, between CVU checks.

Syntax and Parameters

Use the srvctl config cvu command with the following syntax:

```
srvctl config cvu
```

This command has no parameters.

Examples

The `srvctl config cvu` command returns output similar to the following:

```
CVU is configured to run once every 360 minutes
```

srvctl config database

Displays the configuration for an Oracle RAC database or lists all configured databases that are registered with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl config database` command with the following syntax:

```
srvctl config database [-db db_unique_name] [-all] [-verbose]
```

Table A-22   srvctl config database Parameters

Parameter	Description
-db *db_unique_name*	Unique name for the database. If you do not specify this parameter, then the utility displays the configuration of all database resources.
-all	Print detailed configuration information.
-verbose	Display verbose output.

Example

The `srvctl config database` command returns output similar to the following:

```
$ srvctl config database -db myDB
Database unique name: myDB
Database name:
Oracle home: /scott_st2/oracle
Oracle user: scott
Spfile:
Password file:
Domain:
Start options: open
Stop options: immediate
Database role: PRIMARY
Management policy: AUTOMATIC
Server pools: myDB
Database instances:
Disk Groups:
Services: scottsvc1
Type: RACOneNode
Online relocation timeout: 30
Instance name prefix: myDB
Candidate servers: node1 node2
Database is administrator managed
```

The preceding sample output shows the database type as `RACOneNode`, an Oracle RAC One Node database. Other potential types are `RAC` and `SingleInstance`. The

Online relocation timeout, **Instance name prefix**, and **Candidate servers** fields only apply to Oracle RAC One Node databases and do not display for the other two database types. Similarly, the **Database instances** field is empty because the database type is RACOneNode.

srvctl config havip

Displays configuration information for a specific highly available VIP (HAVIP) (used for highly available NFS exports) or all HAVIP resources in a cluster.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl config havip command with the following syntax:

```
srvctl config havip [-id havip_name]
```

Table A-23   srvctl config havip Parameters

Parameter	Description
-id *havip_name*	Specify the unique name of the HAVIP resource you want to display. If you do not specify this parameter, then SRVCTL displays the configuration information for all HAVIPs known to Oracle Clusterware.

Example

This command returns output similar to the following:

```
$ srvctl config havip -id hrexports

VIP: /mjk-vm3-vip/10.149.237.196/10.149.236.0/255.255.252.0/eth0
Description: HR Exports
```

srvctl config listener

Displays configuration information of a specific listener that is registered with Oracle Clusterware.

Syntax and Parameters

Use the srvctl config listener command with the following syntax:

```
srvctl config listener [-listener listener_name | -asmlistener | -leaflistener]
   [-all]
```

Table A-24   srvctl config listener Parameters

Parameter	Description
-listener *listener_name* | -asmlistener | -leaflistener	The name of a specific listener name or the type of listener (Oracle ASM or Leaf Node).
	If you do not specify this parameter, then SRVCTL displays the configuration for the default database listener.
-all	Print detailed configuration information.

Example

This command displays output similar to the following:

```
Name: LISTENER_LEAF
Subnet: 10.100.200.195
Type: type
Owner: mjkeenan
Home: Grid_home
End points: TCP:1521
```

srvctl config network

Displays the network configuration for the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl config network command with the following syntax:

```
srvctl config network [-netnum network_number]
```

Table A-25   srvctl config network Parameters

Parameter	Description
-netnum *network_number*	Specifies the network for which configuration information should be displayed.

Example

An example of this command is:

```
$ srvctl config network -netnum 2
```

srvctl config nodeapps

Displays the VIP configuration for each node in the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl config nodeapps` command with the following syntax:

```
srvctl config nodeapps [-viponly] [-onsonly]
```

Table A-26   srvctl config nodeapps Parameters

Parameter	Description
-viponly	Displays the VIP address configuration
-onsonly	Displays the Oracle Notification Service configuration

Example

An example of this command is:

```
$ srvctl config nodeapps -viponly -onsonly
```

srvctl config oc4j

Displays configuration information for the OC4J instance.

> **Note:**
>
> This command is only available with Oracle Clusterware.

```
srvctl config oc4j
```

This command has no parameters.

Example

This command returns output similar to the following:

```
$ srvctl config oc4j

OC4J is configured to run on port number 23792
OC4J is configured to listen on HTTP port number 8888
OC4J is enabled
OC4J is individually enabled on nodes:
OC4J is individually disabled on nodes:
```

srvctl config ons

Displays configuration information for the Oracle Notification Service daemon.

> **Note:**
>
> This command is only available with Oracle Restart.

```
srvctl config ons
```

This command has no parameters.

srvctl config scan

Displays the configuration information for all SCAN VIPs, by default, or a specific SCAN VIP identified by *ordinal_number*.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl config scan` command with the following syntax:

```
srvctl config scan [-scannumber ordinal_number]
```

The only parameter available for this command is `-scannumber ordinal_number`, which identifies any one of the three SCAN VIPs, and can take a range of values from 1 to 3.

Example

This command returns output similar to the following:

```
$ srvctl config scan -scannumber 1

SCAN name: mjk12700890090-r, Network: 1
Subnet IPv4: 198.51.100.1/203.0.113.46/eth0, static
Subnet IPv6:
SCAN 1 IPv4 VIP: 198.51.100.195
SCAN VIP is enabled.
SCAN VIP is individually enabled on nodes:
SCAN VIP is individually disabled on nodes:
```

srvctl config scan_listener

Displays the configuration information for all SCAN listeners, by default, or a specific listener identified by *ordinal_number*.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl config scan_listener` command with the following syntax:

```
srvctl config scan_listener [-scannumber ordinal_number]
```

The only parameter available for this command is `-scannumber` *ordinal_number*, which identifies any one of the three SCAN VIPs, and can take a range of values from 1 to 3.

Example

This command returns output similar to the following:

```
$ srvctl config scan_listener -scannumber 1

SCAN Listener LISTENER_SCAN1 exists. Port: TCP:1529
Registration invited nodes:
Registration invited subnets:
SCAN Listener is enabled.
SCAN Listener is individually enabled on nodes:
SCAN Listener is individually disabled on nodes:
```

srvctl config service

Displays the configuration for a service.

Syntax and Parameters

Use the `srvctl config service` command with the following syntax:

```
srvctl config service -db db_unique_name [-service service_name]
    [-verbose]
```

Table A-27   srvctl config service Parameters

Parameter	Description
-db *db_unique_name*	Unique name for the database.
-service *service_name*	Service name. If this parameter is not specified, then the configuration information for all services configured for the database are displayed.
-verbose	Displays verbose output.

Examples

This command displays information similar to the following for a policy-managed database:

```
$ srvctl config service -db crm -service webapps
Service name: webapps
Service is enabled
Server pool: sales
Cardinality: SINGLETON
Disconnect: false
Service role: PRIMARY
Management policy: AUTOMATIC
DTP transaction: false
AQ HA notifications: false
Failover type: NONE
Failover method: NONE
TAF failover retries: 0
TAF failover delay: 0
```

```
Connection Load Balancing Goal: LONG
Runtime Load Balancing Goal: NONE
TAF policy specification: NONE
Service is enabled on nodes:
Service is disabled on nodes:
Edition: "my Edition"
```

This command displays information similar to the following for a administrator-managed database:

$ srvctl config service -db crm -service webapps
```
Service name: webapps
Service is enabled
Server pool: sales
Cardinality: 1
Disconnect: false
Service role: PRIMARY
Management policy: AUTOMATIC
DTP transaction: false
AQ HA notifications: false
Failover type: NONE
Failover method: NONE
TAF failover retries: 0
TAF failover delay: 0
Connection Load Balancing Goal: LONG
Runtime Load Balancing Goal: NONE
TAF policy specification: NONE
Preferred instances: crm_1
Available instances:
Edition: "my Edition"
```

Service configuration for administrator-managed Oracle RAC One Node databases displays the one instance as preferred.

Note:

The `srvctl config service` command shows exactly the string value you specified for the edition using the `srvctl add | modify service` commands. If you specified the edition in upper case, then `srvctl config service` displays upper case. If it is surrounded by double quotation marks (`""`), then the command displays the double quotation marks. Otherwise, the command displays an empty string.

---

srvctl config srvpool

Displays configuration information including name, minimum size, maximum size, importance, and a list of server names, if applicable, for a specific server pool in a cluster.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl config srvpool` command with the following syntax:

```
srvctl config srvpool [-serverpool pool_name]
```

The only parameter available for this command is -serverpool `pool_name`, which is the name of the server pool to display the configuration information for.

Example

An example of this command is:

```
$ srvctl config srvpool -serverpool dbpool
```

srvctl config vip

Displays all VIPs on all networks in the cluster except for user VIPs.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl config vip command with the following syntax:

```
srvctl config vip {-node node_name | -vip vip_name}
```

Table A-28   srvctl config vip Parameters

Parameter	Description
-node *node_name*	Specify the node name. The command returns output similar to the following: `VIP exists: ipv4, ipv6, network number 1, hosting node adc2100252`
-vip *vip_name*	Specify the VIP name. The command returns output similar to the following: `VIP exists: ipv4, ipv6, network number 1, hosting node adc2100252`

Example

An example of this command is:

```
$ srvctl config vip -node crmnode1
```

```
VIP exists: /crmnode1-vip/192.168.2.20/255.255.255.0/eth0
```

srvctl config volume

Displays the configuration for a specific volume or all volumes.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl config volume command with the following syntax:

```
srvctl config volume [-volume volume_name] [-diskgroup disk_group_name]
    [-device volume_device]
```

Table A-29   srvctl config volume Parameters

Parameter	Description
-volume *volume_name*	Specify the name of the volume for which you want to view the configuration.
-diskgroup *disk_group_name*	Specify the name of the disk group in which the volume resides for which you want to display the configuration.
-device *volume_device*	Specify the path to the volume device for which you want to display the configuration.

Usage Notes

- If you do not specify any of the optional parameters, then SRVCTL displays the configuration information for all volumes.

- If you specify only the -volume parameter, then SRVCTL displays the configuration for all volumes with that name, regardless of the diskgroup.

- If you specify only the -diskgroup parameter, then SRVCTL displays the configuration information for the volumes that reside in the disk group that you specify.

- If you specify only the -device parameter, then SRVCTL displays the configuration information for the volume matching that device specifier.

- If you specify the -diskgroup and -device parameters, then SRVCTL displays the configuration information for the volume device that resides in the disk group that you specify.

Examples

The srvctl config volume command displays information similar to the following:

```
$ srvctl config volume -device /dev/asm/volume1-123

Diskgroup Name: DG1
Volume Name   : VOL1
Volume Device : /dev/asm/volume1-123
Volume is enabled.
Volume is enabled on nodes:
Volume is disabled on nodes:
```

If you do not specify any parameters, then SRVCTL displays configuration information for all volumes, similar to the following:

```
$ srvctl config volume
```

```
Diskgroup name: DG1
Volume name: VOL1
Volume device: /dev/asm/volume1-123
Volume is enabled.
Volume is enabled on nodes:
Volume is disabled on nodes:
Diskgroup name: DG1
Volume name: VOL2
Volume device: /dev/asm/volume2-456
Volume is enabled.
Volume is enabled on nodes:
Volume is disabled on nodes:
```

srvctl convert database

The `srvctl convert database` command converts a database either to or from an Oracle RAC One Node database.

Syntax and Parameters

Use the `srvctl convert database` command with one of the following syntax models:

```
srvctl convert database -db db_unique_name -dbtype RACONENODE
        [-instance instance_name] [-timeout timeout]

srvctl convert database -db db_unique_name -dbtype RAC [-node node_name]
```

Table A-30   srvctl convert database Parameters

Parameter	Description
-db *db_unique_name*	Unique name for the database. **Note:** If you specify a noncluster database, then command returns an error instructing you to use `rconfig` to convert the noncluster database to Oracle RAC or Oracle RAC One Node.
-dbtype RACONENODE | RAC	The type of database to which you are converting, either Oracle RAC One Node or Oracle RAC. **Note:** If there is an ongoing or failed online database relocation, then the command returns an error instructing you to first complete or abort the online database relocation and then rerun the command.
-instance *instance_name*	Instance name prefix for Oracle RAC One Node databases. The default value for this parameter is the first 12 characters of the global unique name of the database. **Notes:** • You can use this parameter only when converting from an Oracle RAC database to an Oracle RAC One Node database. • In order for the converted instance to come online, you must restart the database using the `srvctl stop/start database` commands.
-timeout *timeout*	Online database relocation timeout, in minutes, for Oracle RAC One Node databases. The default is 30.

Table A-30 (Cont.) srvctl convert database Parameters

Parameter	Description
`-node node_name`	Name of the node for administrator-managed Oracle RAC database. The default is the first candidate.
	Note: If you do not specify a node name or you specify a node name where the database is not running, then the command returns an error instructing you specify the correct node.

Example

An example of this command is:

```
$ srvctl convert database -db myDB -dbtype RACONENODE -instance myDB3
```

disable

Disables a specified object (cluster database, database instance, Oracle ASM instance, or service). Use the `srvctl disable` command when you must shut down an object for maintenance. The disabled object does not automatically restart.

When you run the `disable` command, the object is disabled and unavailable to run under Oracle Clusterware for automatic startup, failover, or restart. Additionally, you cannot run the `srvctl start` command on a disabled object until you first re-enable the object. If you specify `-instance instance_name` or `-node node_name`, then SRVCTL only disables the object on the specified instance or node.

If you do not specify `-instance instance_name` or `-node node_name`, then the disable action applies to the object that you specified, globally, on all nodes in the cluster and also supersedes any per-node disable settings. Any object you disable globally is also disabled on any nodes you add to the cluster in the future.

Disabling an object globally *is different from* disabling an object, individually, on each node in the cluster. For example, if you disable a database on several nodes, individually, in a cluster, then, when you run the `srvctl start database` command, the database starts only on nodes where the database is enabled. If, however, you disable the database globally, then the `srvctl start database` command is rejected because the database is disabled, clusterwide.

Table A-31 srvctl disable Summary

Command	Description
srvctl disable cvu	Disables the Cluster Verification Utility
srvctl disable database	Disables the cluster database
srvctl disable diskgroup	Disables a disk group on a number of specified nodes
srvctl disable filesystem	Disables an Oracle ACFS volume
srvctl disable gns	Disables GNS
srvctl disable havip	Prevents an HAVIP resource from running on a number of specified nodes

Table A-31   (Cont.) srvctl disable Summary

Command	Description
srvctl disable instance	Disables an instance
srvctl disable listener	Disables a listener
srvctl disable mgmtdb	Disables the management database
srvctl disable mgmtlsnr	Disables the management listener
srvctl disable nodeapps	Disables a node application and GSD
srvctl disable oc4j	Disables OC4J instances
srvctl disable ons	Disables the Oracle Notification Service daemon
srvctl disable scan	Disables SCAN VIPs
srvctl disable scan_listener	Disables SCAN listeners
srvctl disable service	Disables a service
srvctl disable vip	Disables a VIP
srvctl disable volume	Disables a volume

srvctl disable cvu

Disable the Cluster Verification Utility (CVU) for Oracle Clusterware management, if enabled.

Syntax and Parameters

Use the `srvctl disable cvu` command with the following syntax:

```
srvctl disable cvu [-node node_name]
```

You can specify a particular node on which to disable CVU.

Table A-32   srvctl disable cvu Parameters

Parameter	Description
-node node_name	Node name

Examples

An example of this command to disable CVU on the node in the cluster named `crmnode1` is:

```
$ srvctl disable cvu -node crmnode1
```

srvctl disable database

Disables a database. If the database is a cluster database, then its instances are also disabled.

Syntax and Parameters

Use the `srvctl disable database` command with the following syntax:

```
srvctl disable database -db db_unique_name [-node node_name]
```

Table A-33   srvctl disable database Parameters

Parameter	Description
-db *database_name*	Database name
-node *node_name*	Disables the database from running on the named node. **Note:** This parameter can be used only with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl disable database -db mydb1
```

srvctl disable diskgroup

Disables a specific disk group on a number of specified nodes.

Syntax and Parameters

Use the `srvctl disable diskgroup` command with the following syntax:

```
srvctl disable diskgroup -diskgroup diskgroup_name [-node "node_list"]
```

Table A-34   srvctl disable diskgroup Parameters

Parameter	Description
-diskgroup *diskgroup_name*	The Oracle ASM disk group name.
-node "*node_list*"	Comma-delimited list of node names on which to disable the disk group This parameter is only available with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl disable diskgroup -diskgroup dgroup1 -node "mynode1,mynode2"
```

srvctl disable filesystem

Disables a specific Oracle Clusterware-managed file system volume.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl disable filesystem` command with the following syntax:

```
srvctl disable filesystem -device volume_device
```

Table A-35   srvctl disable filesystem Parameters

Parameter	Description
-device *volume_device*	Specify the path to the file system volume you want to disable.

Usage Notes

- You must run this command as `root` user on Linux and UNIX platforms.

Example

An example of this command is:

```
# srvctl disable filesystem -device /dev/asm/racvol1
```

srvctl disable gns

Disables GNS for a specific node, or all available nodes in the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl disable gns` command with the following syntax:

```
srvctl disable gns [-node node_name] [-verbose]
```

Table A-36   srvctl disable gns Parameters

Parameter	Description
-node *node_name*	Name of a node in the cluster If you do not specify this parameter, then SRVCTL disables GNS for the entire cluster.
-verbose	Display verbose output

Example

An example of this command to disable GNS on the node named `crm7` is:

```
$ srvctl disable gns -node crm7
```

srvctl disable havip

Prevents a specific highly available VIP (HAVIP) (used for highly available NFS exports) resource from running on a number of specified nodes.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl disable havip` command with the following syntax:

```
srvctl disable havip -id havip_name [-node node1,node2,...,noden]
```

Table A-37   srvctl disable havip Parameters

Parameter	Description
-id *havip_name*	Specify the unique name of the HAVIP resource you want to disable.
-node *node1,node2,...,noden*	Specify the name of a node or a comma-delimited list of node names on which you do not want the HAVIP resource to run.
	If you specify all but one node in the cluster, then the HAVIP will not fail over.

Usage Notes

- You must run this command as `root` user on Linux and UNIX platforms.

Example

An example of this command is:

```
$ srvctl disable havip -id myhavip -node myNode1,myNode3
```

srvctl disable instance

Disables an instance. If the instance that you disable with this command is the last enabled instance, then this operation also disables the database.

> **Note:**
>
> - This command is only available with Oracle Clusterware and Oracle RAC.
>
> - If you run this command on an Oracle RAC One Node database, then the command returns an error instructing you to use the `database` noun, instead.

Syntax and Parameters

Use the `srvctl disable instance` command with the following syntax:

```
srvctl disable instance -db db_unique_name -instance instance_name_list
```

Table A-38   srvctl disable instance Parameters

Parameter	Description
-db db_unique_name	Unique name for the database
-instance instance_name_list	Comma-delimited list of instance names

Example

An example of this command is:

```
$ srvctl disable instance -db crm -instance "crm1,crm3"
```

srvctl disable listener

Disables a listener resource.

Syntax and Parameters

Use the `srvctl disable listener` command with the following syntax:

```
srvctl disable listener [-listener listener_name] [-node node_name]
```

Table A-39   srvctl disable listener Parameters

Parameter	Description
-listener listener_name	Name of a listener resource. If you do not specify this parameter, the name of the listener defaults to LISTENER.
-node node_name	Name of a cluster node on which the listener you want to disable is running. This parameter is only available with Oracle Clusterware.

Example

To disable the listener named `listener_crm` on the node `node5`, use this command:

```
$ srvctl disable listener -listener listener_crm -node node5
```

srvctl disable mgmtdb

Disables the management database (CHM repository) resource on the specified node.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl disable mgmtdb` command with the following syntax:

```
srvctl disable mgmtdb [-node node_name]
```

Table A-40   srvctl disable mgmtdb Parameters

Parameter	Description
-node *node_name*	Name of a cluster node on which the management database you want to disable is running.

Example

An example of this command is:

```
$ srvctl disable mgmtdb
```

srvctl disable mgmtlsnr

Disables the management listener resource (for CHM) on the specified node.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl disable mgmtlsnr` command with the following syntax:

```
srvctl disable mgmtlsnr [-node node_name]
```

Table A-41   srvctl disable mgmtlsnr Parameters

Parameter	Description
-node *node_name*	Name of a cluster node on which the management listener you want to disable is running.

Example

An example of this command is:

```
$ srvctl disable mgmtlsnr
```

srvctl disable nodeapps

Disables node applications on all nodes in the cluster.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl disable nodeapps` command with the following syntax:

```
srvctl disable nodeapps [-gsdonly] [-adminhelper] [-verbose]
```

Table A-42   srvctl disable nodeapps Parameters

Parameter	Description
-gsdonly	Disables GSD
-adminhelper	Disable the Administrator helper only
-verbose	Verbose output

Example

An example of this command is:

```
$ srvctl disable nodeapps -gsdonly -verbose
```

srvctl disable oc4j

Disables the OC4J instance on all nodes or on a specific node.

Syntax and Parameters

Use the `srvctl disable oc4j` command with the following syntax:

```
srvctl disable oc4j [-node node_name] [-verbose]
```

Table A-43   srvctl disable oc4j Parameters

Parameter	Description
-node node_name	The name of a node in the cluster
-verbose	Verbose output

Example

An example of this command is:

```
$ srvctl disable oc4j -node crm3
```

srvctl disable ons

Disables the Oracle Notification Service daemon for Oracle Restart installations.

```
srvctl disable ons [-verbose]
```

The only parameter for this command is –verbose, which indicates that verbose output should be displayed.

srvctl disable scan

Disables all SCAN VIPs, by default, or a specific SCAN VIP identified by *ordinal_number*.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl disable scan` command with the following syntax:

```
srvctl disable scan [-scannumber ordinal_number]
```

The only parameter available for this command is -scannumber *ordinal_number*, which represents which identifies any one of the three SCAN VIPs, and can take a range of values from 1 to 3.

Example

An example of this command is:

```
$ srvctl disable scan -scannumber 1
```

srvctl disable scan_listener

Disables all SCAN listeners, by default, or a specific listener identified by *ordinal_number*.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl disable scan_listener` command with the following syntax:

```
srvctl disable scan_listener [-scannumber ordinal_number]
```

The only parameter available for this command is -scannumber *ordinal_number*, which identifies any one of the three SCAN listeners, and can take a range of values from 1 to 3.

Example

An example of this command is:

```
$ srvctl disable scan_listener -scannumber 1
```

srvctl disable service

Disables a service. Disabling an entire service affects all of the instances, disabling each one. When the entire service is already disabled, a `srvctl disable service` operation on the entire service affects all of the instances and disables them; it just returns an error. This means that you cannot always use the entire set of service operations to manipulate the service indicators for each instance.

Syntax and Parameters

Use the `srvctl disable service` command with the following syntax:

```
srvctl disable service -db db_unique_name -servics "service_name_list"
   [-instance instance_name | -node node_name]
```

If you do not specify either the `-instance instance_name` or `-node node_name` parameters, then the command disables the service on all nodes.

Table A-44   srvctl disable service Parameters

Parameter	Description
-database db_unique_name	Unique name for the database.
-service "service_name_list"	Comma-delimited list of service names, or a single service name
-instance instance_name	The name of the instance for which you want to disable the service. **Notes:** • Use this parameter with administrator-managed databases • This parameter can be used only with Oracle Clusterware and Oracle RAC
-node node_name	The name of the node on which to disable the service. **Notes:** • Use this parameter with policy-managed databases • This parameter can be used only with Oracle Clusterware and Oracle RAC

Examples

The following example globally disables two services for the CRM database:

```
$ srvctl disable service -db crm -service "crm,marketing"
```

The following example disables a service for the CRM database that is running on the CRM1 instance, resulting in the service still being available for the database, but on one less instance:

```
$ srvctl disable service -db crm -service crm -instance crm1
```

srvctl disable vip

Disables a specific VIP.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl disable vip` command with the following syntax:

```
srvctl disable vip -vip vip_name [-verbose]
```

Table A-45   srvctl disable vip Parameters

Parameter	Description
-vip *vip_name*	The VIP name
-verbose	Verbose output

Example

An example of this command is:

```
$ srvctl disable vip -vip vip1 -verbose
```

srvctl disable volume

Disables Oracle Clusterware management for a specific volume or all volumes.

This command allows a volume device to be stopped by operating on the Oracle Clusterware resource for the volume. This command does not stop volume device.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl disable volume` command with the following syntax:

```
srvctl disable volume {-volume volume_name -diskgroup disk_group_name |
    -device volume_device}
```

Table A-46   srvctl disable volume Parameters

Parameter	Description
-volume *volume_name*	Specify the name of the volume that you want to disable. This parameter is required.

Table A-46   (Cont.) srvctl disable volume Parameters

Parameter	Description
-diskgroup *disk_group_name*	Specify the name of the disk group in which the volume that you want to disable resides.
-device *volume_device*	Specify the path to the volume device that you want to disable.

Usage Notes

- You must specify a particular volume that you want to disable. You can specify a volume that resides in either a particular disk group or on a particular volume device.

Example

The following example disables a volume named VOLUME1 that resides in a disk group named DATA:

```
$ srvctl disable volume -volume VOLUME1 -diskgroup DATA
```

srvctl downgrade database

The srvctl downgrade database command downgrades the configuration of a database and its services from its current version to the specified lower version.

Syntax and Parameters

Use the srvctl downgrade database command as follows:

```
srvctl downgrade database -db db_unique_name -oraclehome Oracle_home
       -targetversion to_version
```

Table A-47   srvctl downgrade database Parameters

Parameter	Description
-db *db_unique_name*	Unique name for the database
-oraclehome *Oracle_home*	The path to the ORACLE_HOME
-targetversion *to_version*	The version to which to downgrade

enable

The srvctl enable command enables the specified object so that it can run under Oracle Clusterware for automatic startup, failover, or restart. The Oracle Clusterware application supporting the object may be up or down to use this function. The default value is enable. If the object is already enabled, then SRVCTL returns a message advising you that the object is enabled. Enabled objects can be started, and disabled objects cannot be started.

When you run the `enable` command, the object is enabled and available to run under Oracle Clusterware for automatic startup, failover, or restart. Additionally, you can run the `srvctl start` command on an enabled object. If you specify `-instance` *instance_name* or `-node` *node_name*, then SRVCTL only enables the object on the specified instance or node.

If you do not specify `-instance` *instance_name* or `-node` *node_name*, then the enable action applies to the object that you specified, globally, on all nodes in the cluster and also removes any per-node enable settings. Any object you enable globally is also enabled on any nodes you add to the cluster in the future.

Enabling an object globally *is different from* enabling an object, individually, on each node in the cluster. For example, if you enable a database on several nodes, individually, in a cluster, then, when you run the `srvctl start database` command, the database starts only on nodes where the database is enabled. If, however, you enable the database globally, then the `srvctl start database` command starts the database on all nodes on which it is configured to run.

Table A-48   srvctl enable Summary

Command	Description
srvctl enable asm	Enables an Oracle ASM instance
srvctl enable cvu	Enables the Cluster Verification Utility
srvctl enable database	Enables the database resource
srvctl enable diskgroup	Enables a specified disk group on a number of specified nodes
srvctl enable filesystem	Enables an Oracle ACFS volume
srvctl enable gns	Enables GNS
srvctl enable havip	Allows an HAVIP resource to run on a number of specified nodes
srvctl enable instance	Enables the instance
srvctl enable listener	Enables a listener
srvctl enable mgmtdb	Enables the management database for Oracle Clusterware management.
srvctl enable mgmtlsnr	Enables the management listener for Oracle Clusterware management
srvctl enable nodeapps	Enables node applications and GSD
srvctl enable oc4j	Enables OC4J instances
srvctl enable ons	Enables the Oracle Notification Service daemon
srvctl enable scan	Enables SCAN VIPs
srvctl enable scan_listener	Enables SCAN listeners
srvctl enable service	Enables a service
srvctl enable vip	Enables a VIP

Table A-48   (Cont.) srvctl enable Summary

Command	Description
srvctl enable volume	Enables a volume

srvctl enable asm

Enables an Oracle ASM instance.

> **Note:**
>
> To manage Oracle ASM on Oracle Database 12*c* installations, use the SRVCTL binary in the Oracle Grid Infrastructure home for a cluster (Grid home). If you have Oracle RAC or Oracle Database installed, then you cannot use the SRVCTL binary in the database home to manage Oracle ASM.

Syntax and Parameters

Use the `srvctl enable asm` command with the following syntax:

```
srvctl enable asm [-proxy] [-node node_name]
```

Table A-49   srvctl enable asm Parameters

Parameter	Description
-proxy	Specifies whether you are disabling a proxy Oracle ASM instance.
-node *node_name*	Node name **Note:** This parameter can be used only with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl enable asm -node crmnode1
```

srvctl enable cvu

Enable the Cluster Verification Utility (CVU) for Oracle Clusterware management if disabled.

Syntax and Parameters

Use the `srvctl enable cvu` command with the following syntax:

```
srvctl enable cvu [-node node_name]
```

You can specify a particular node on which to enable CVU.

Table A-50   srvctl enable cvu Parameters

Parameter	Description
-node *node_name*	The name of the node on which to enable CVU **Note:** This parameter can be used only with Oracle Clusterware.

Examples

An example of this command to enable CVU on the node in the cluster named crmnode1 is:

```
$ srvctl enable cvu -node crmnode1
```

srvctl enable database

Enables a cluster database and its instances.

Syntax and Parameters

Use the srvctl enable database command with the following syntax:

```
srvctl enable database -db db_unique_name [-node node_name]
```

Table A-51   srvctl enable database Parameters

Parameter	Description
-db *database_name*	Database name
-node *node_name*	The name of the node for which the database resource should be enabled **Note:** This parameter can be used only with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl enable database -db mydb1
```

srvctl enable diskgroup

Enables a specific disk group on a number of specified nodes.

Syntax and Parameters

Use the srvctl enable diskgroup command with the following syntax:

```
srvctl enable diskgroup -diskgroup diskgroup_name [-node "node_list"]
```

Table A-52   srvctl enable diskgroup Parameters

Parameter	Description
-diskgroup *diskgroup_name*	The Oracle ASM disk group name
-nodelist *node_list*	Comma-delimited list of node names on which to enable the disk group This parameter is only available with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl enable diskgroup -diskgroup diskgroup1 -node "mynode1,mynode2"
```

srvctl enable filesystem

Enables an Oracle ACFS volume or generic file system in Oracle Clusterware.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl enable filesystem command with the following syntax:

```
srvctl enable filesystem -device volume_device
```

Table A-53   srvctl enable filesystem Parameters

Parameter	Description
-device *volume_device*	Specify the path to the volume device you want to enable.

Usage Notes

- You must run this command as root user on Linux and UNIX platforms.

Example

An example of this command is:

```
# srvctl enable filesystem -device /dev/asm/racvol1
```

srvctl enable gns

Enables GNS on all nodes or a specific node.

Note:

This command can be used only with Oracle Clusterware.

Syntax and Parameters

Use the srvctl enable gns command with the following syntax:

srvctl enable gns [-node *node_name*] [-verbose]

Table A-54   srvctl enable gns Parameters

Parameter	Description
-node *node_name*	Name of the node on which to enable GNS. If this parameter is not specified, then GNS is enabled on all nodes in the cluster.
-verbose	Display verbose output

Example

An example of this command is:

$ srvctl enable gns

srvctl enable havip

Enables a specific highly available VIP (HAVIP) (used for highly available NFS exports) to run on a number of specified nodes.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl enable havip command with the following syntax:

srvctl enable havip -id *havip_name* [-node *node_name*]

Table A-55   srvctl enable havip Parameters

Parameter	Description
-id *havip_name*	Specify the unique name of the HAVIP resource you want to enable.
-node *node_name*	Specify the name of a node on which you want the HAVIP resource to be able to run.

Usage Notes

• You must run this command as root user on Linux and UNIX platforms.

Example

An example of this command is:

```
# srvctl enable havip -id myhavip -node myNode1
```

srvctl enable instance

Enables an instance for an Oracle RAC database. If you use this command to enable all instances, then the database is also enabled.

> **Note:**
>
> - This command is only available with Oracle Clusterware and Oracle RAC.
>
> - If you run this command on an Oracle RAC One Node database, then the command returns an error instructing you to use the database noun, instead.

Syntax and Parameters

Use the srvctl enable instance command with the following syntax:

```
srvctl enable instance -db db_unique_name -instance "instance_name_list"
```

Table A-56   srvctl enable instance Parameters

Parameter	Description
-db *db_unique_name*	Unique name for the database
-instance "*instance_name_list*"	Comma-delimited list of instance names.

Example

An example of this command is:

```
$ srvctl enable instance -db crm -instance "crm1,crm2"
```

srvctl enable listener

Enables a listener resource.

Syntax and Parameters

Use the srvctl enable listener command with the following syntax:

```
srvctl enable listener [-listener listener_name] [-node node_name]
```

Table A-57   srvctl enable listener Parameters

Parameter	Description
-listener *listener_name*	Name of a listener resource. If you do not specify this parameter, the name of the listener defaults to LISTENER
-node *node_name*	Name of a cluster node **Note:** This parameter can be used only with Oracle Clusterware.

Example

To enable the listener named listener_crm on the node named node5, use the following command:

```
$ srvctl enable listener -listener listener_crm -node node5
```

srvctl enable mgmtdb

Enables a management database (CHM repository) resource.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl enable mgmtdb command with the following syntax:

```
srvctl enable mgmtdb [-node node_name]
```

Table A-58   srvctl enable mgmtdb Parameters

Parameter	Description
-node *node_name*	Name of a cluster node **Note:** This parameter can be used only with Oracle Clusterware.

Example

To enable a management database on the node named node5, use the following command:

```
$ srvctl enable mgmtdb -node node5
```

srvctl enable mgmtlsnr

Enables a management database listener resource (for CHM).

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl enable mgmtlsnr` command with the following syntax:

```
srvctl enable mgmtlsnr [-node node_name]
```

Table A-59   srvctl enable mgmtlsnr Parameters

Parameter	Description
`-node node_name`	Name of a cluster node **Note:** This parameter can be used only with Oracle Clusterware.

Example

To enable a management listener on the node named `node5`, use the following command:

```
$ srvctl enable mgmtlsnr -node node5
```

srvctl enable nodeapps

Enables the node applications on all nodes in the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl enable nodeapps` command with the following syntax:

```
srvctl enable nodeapps [-gsdonly] [-adminhelper] [-verbose]
```

Table A-60   srvctl enable nodeapps Parameters

Parameter	Description
`-gsdonly`	Enable only the GSD daemon
`-adminhelper`	Enable the Administrator helper only
`-verbose`	Verbose output

Example

An example of this command is:

```
$ srvctl enable nodeapps -gsdonly -verbose
```

srvctl enable oc4j

Enables the OC4J instance on all nodes or on a specific node.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl enable oc4j` command with the following syntax:

```
srvctl enable oc4j [-node node_name] [-verbose]
```

Table A-61   srvctl enable oc4j Parameters

Parameter	Description
`-node node_name`	The name of a node in the cluster
`-verbose`	Verbose output

Example

An example of this command is:

```
$ srvctl enable oc4j -node crm3
```

srvctl enable ons

Enables the Oracle Notification Service daemon.

> **Note:**
>
> This command is only available with Oracle Restart.

Syntax and Parameters

Use the `srvctl enable ons` command with the following syntax:

```
srvctl enable ons [-verbose]
```

The only parameter for this command is `-verbose`, which indicates that verbose output should be displayed.

Example

An example of this command is:

```
$ srvctl enable ons
```

srvctl enable scan

Enables all SCAN VIPs, by default, or a specific SCAN VIP identified by its *ordinal_number*.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl enable scan` command with the following syntax:

```
srvctl enable scan [-scannumber ordinal_number]
```

The only parameter available for this command is –scannumber *ordinal_number*, which identifies one of the three SCAN VIPs, and takes a range of values from 1 to 3.

Example

An example of this command is:

```
$ srvctl enable scan -scannumber 1
```

srvctl enable scan_listener

Enables all SCAN listeners, by default, or a specific listener identified by its *ordinal_number*.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl enable scan_listener` command with the following syntax:

```
srvctl enable scan_listener [-scannumber ordinal_number]
```

The only parameter available for this command is –scannumber *ordinal_number*, which identifies one of three SCAN listeners, and takes a range of values from 1 to 3.

Example

An example of this command is:

```
$ srvctl enable scan_listener -scannumber 1
```

srvctl enable service

Enables a service for Oracle Clusterware. Enabling an entire service also affects the enabling of the service over all of the instances by enabling the service at each one. When the entire service is already enabled, an `srvctl enable service` operation does not affect all of the instances and enable them. Instead, this operation returns an error. Therefore, you cannot always use the entire set of service operations to manipulate the service indicators for each instance.

Syntax and Parameters

Use the `srvctl enable service` command with the following syntax:

```
srvctl enable service -db db_unique_name -service "service_name_list"
    [-instance instance_name | -node node_name]
```

Table A-62   srvctl enable service Parameters

Parameter	Description
-db *db_unique_name*	Unique name for the database
-service "*service_name_list*"	Comma-delimited list of service names
-instance *instance_name*	Name of the database instance where you want the service to run Use this parameter for administrator-managed databases **Note:** This parameter can be used only with Oracle Clusterware and Oracle RAC.
-node *node_name*	Name of the node where you want the service to run Use this parameter for policy-managed databases **Note:** This parameter can be used only with Oracle Clusterware and Oracle RAC.

Examples

The following example globally enables a service:

```
$ srvctl enable service -db crm -service crm
```

The following example enables a service to use a preferred instance:

```
$srvctl enable service -db crm -service crm -instance crm1
```

srvctl enable vip

Enables a specific VIP.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl enable vip command with the following syntax:

```
srvctl enable vip -vip vip_name [-verbose]
```

Table A-63   srvctl enable vip Parameters

Parameter	Description
-vip *vip_name*	The VIP name
-verbose	Verbose output

Example

The following command enables a VIP named `crm1-vip`:

```
$ srvctl enable vip -vip crm1-vip -verbose
```

srvctl enable volume

Enables Oracle Clusterware management for a specific volume or all volumes.

This command allows a volume device to be started by operating on the Oracle Clusterware resource for the volume. This command does not start the volume device, and is different from the SQL command `ALTER DISKGROUP ENABLE VOLUME` or the ASMCMD command `volenable`, because these two commands bring the volume device online, in a running state, making the volume device accessible.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl enable volume` command with the following syntax:

```
srvctl enable volume {-volume volume_name -diskgroup disk_group_name |
    -device volume_device}
```

Table A-64   srvctl enable volume Parameters

Parameter	Description
`-volume volume_name`	Specify the name of the volume that you want to enable. This parameter is required.
`-diskgroup disk_group_name`	Specify the name of the disk group in which the volume that you want to enable resides.
`-device volume_device`	Specify the path to the volume device that you want to enable.

Usage Notes

- You must specify a particular volume that you want to enable. You can specify a volume that resides in either a particular disk group or on a particular volume device.

Example

The following example enables a volume named `VOLUME1` that resides in a disk group named `DATA`:

```
$ srvctl enable volume -volume VOLUME1 -diskgroup DATA
```

srvctl export gns

The `srvctl export gns` command exports Grid Naming Service (GNS) instance data to a file that you can use when you are either moving GNS from one server cluster to another or when you are creating a client cluster

SRVCTL extracts the data from OCR. Exported data includes:

- The credentials used to authorize users, which includes the VIP address on which the server listens

- The names and DNS records kept in GNS

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl export gns` command with the following syntax:

```
srvctl export gns {-instance path_to_file | -clientdata path_to_file}
```

Table A-65   srvctl export gns Parameters

Parameter	Description
`-instance path_to_file`	Use this parameter to specify the file to which GNS instance data is written.
	Use this parameter when you are moving GNS from one server cluster to another.
`-clientdata path_to_file`	Use this parameter to specify the file to which GNS credentials data is written.
	Use this parameter when you are adding a client cluster to create a file that contains credentials that the client cluster accesses to perform name management operations on the server cluster.

Usage Notes

- You must run this command as `root` user on Linux and UNIX platforms, or as an Administrator user on Windows platforms

- You must run this command on any node in the server cluster

- GNS does not need to be running during export

- The data must be locked during the procedure to ensure consistency

Example

The following command exports the GNS credentials data to a file named `gns.txt`:

```
# srvctl export gns -clientdata /temp/gnsdata/gns.txt
```

getenv

Gets and displays values for the environment variables from the configuration file. Use SRVCTL with the `setenv`, `getenv`, and `unsetenv` verbs to administer the environment configurations for databases, instances, services, and node applications.

Table A-66   srvctl getenv Summary

Command	Description
srvctl getenv asm	Gets the Oracle ASM environment values
srvctl getenv database	Gets the database environment values
srvctl getenv listener	Gets the listener environment values
srvctl getenv mgmtdb	Gets the management database environment variables
srvctl getenv mgmtlsnr	Gets the management listener resource environment variables
srvctl getenv nodeapps	Gets the node application environment values
srvctl getenv vip	Gets the service environment values

srvctl getenv asm

Displays the values for environment variables associated with Oracle ASM.

Syntax and Parameters

Use the `srvctl getenv asm` command with the following syntax:

```
srvctl getenv asm [-envs "name_list"]
```

Table A-67   srvctl getenv asm Parameters

Parameter	Description
-envs "name_list"	Comma-delimited list of the names of environment variables.
	If this parameter is not specified, then the values of all environment variables associated with Oracle ASM are displayed.

Example

The following example displays the current values for all the environment variables used by Oracle ASM:

```
$ srvctl getenv asm
```

srvctl getenv database

Displays the values for environment variables associated with a database.

Syntax and Parameters

Use the `srvctl getenv database` command with the following syntax:

```
srvctl getenv database -db db_unique_name [-envs "name_list"]
```

Table A-68   srvctl getenv database Parameters

Parameter	Description
-db db_unique_name	Unique name for the database
-envs "name_list"	Comma-delimited list of the names of environment variables If this parameter is not specified, then the values of all environment variables associated with the database are displayed.

Example

The following example gets the environment configuration for the CRM database:

```
$ srvctl getenv database -db crm
```

srvctl getenv listener

Gets the environment variables for the specified listener.

Syntax and Parameters

Use the srvctl getenv listener command with the following syntax:

```
srvctl getenv listener [-listener listener_name] [-envs "name_list"]
```

Table A-69   srvctl getenv listener Parameters

Parameter	Description
-listener listener_name	Listener name If this parameter is not specified, the name of the listener defaults to LISTENER
-envs "name_list"	Comma-delimited list of the names of environment variables If this parameter is not specified, then the values of all environment variables associated with the listener are displayed.

Example

The following example lists all environment variables specified for the default listener:

```
$ srvctl getenv listener
```

srvctl getenv mgmtdb

Gets the environment variables for the management database (CHM repository).

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl getenv mgmtdb` command with the following syntax:

```
srvctl getenv mgmtdb [-envs "name_list"]
```

Table A-70   srvctl getenv mgmtdb Parameters

Parameter	Description
`-envs "name_list"`	Comma-delimited list of the names of environment variables
	If this parameter is not specified, then the values of all environment variables associated with the listener are displayed.

Example

The following example lists all environment variables specified for the management database:

```
$ srvctl getenv mgmtdb
```

srvctl getenv mgmtlsnr

Gets the environment variables for the management listener resource (for CHM).

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl getenv mgmtlsnr` command with the following syntax:

```
srvctl getenv mgmtlsnr [-envs "name_list"]
```

Table A-71   srvctl getenv mgmtlsnr Parameters

Parameter	Description
`-envs "name_list"`	Comma-delimited list of the names of environment variables
	If this parameter is not specified, then the values of all environment variables associated with the listener are displayed.

Example

The following example lists all environment variables specified for the management listener:

```
$ srvctl getenv mgmtlsnr
```

srvctl getenv nodeapps

Gets the environment variables for the node application configurations.

Syntax and Parameters

Use the `srvctl getenv nodeapps` command with the following syntax:

```
srvctl getenv nodeapps [-viponly] [-onsonly] [-envs "name_list"]
```

Table A-72   srvctl getenv nodeapps Parameters

Parameter	Description
-viponly	Displays the VIP address configuration
-onsonly	Displays the Oracle Notification Service configuration
-envs "name_list"	Comma-delimited list of the names of environment variables
	If this parameter is not specified, then SRVCTL displays the values of all environment variables associated with the node applications.

Example

The following example lists all environment variables for the node applications:

```
$ srvctl getenv nodeapps -viponly
```

srvctl getenv vip

Gets the environment variables for the specified VIP.

Syntax and Parameters

Use the `srvctl getenv vip` command with the following syntax:

```
srvctl getenv vip -vip vip_name [-envs "name_list"] [-verbose]
```

Table A-73   srvctl getenv vip Parameters

Parameter	Description
-vip vip_name	The name of the VIP

Table A-73   (Cont.) srvctl getenv vip Parameters

Parameter	Description
`-envs "name_list"`	Comma-delimited list of the names of environment variables
	If this parameter is not specified, then the values of all environment variables associated with the VIP are displayed.
`-verbose`	Verbose output

Example

The following example lists all environment variables for the specified VIP:

```
$ srvctl getenv vip -vip node1-vip
```

srvctl import gns

The `srvctl import gns` command imports data from a file that you create when you run the Syntax and Parameters command. Use this command when you want to locate GNS to a different server cluster.

When you import GNS data, SRVCTL stores the credentials and places the record data into OCR. If another GNS instance is running or data for another instance is encountered during the import procedure, then an error occurs.

This command also makes the cluster in which you run it the server cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl import gns` command with the following syntax:

```
srvctl import gns -instance path_to_file
```

Provide the fully-qualified file name of the file into which you exported the GNS data.

Usage Notes

- You must run this command as `root` on Linux and UNIX platforms, or as an Administrator user on Windows platforms

- You must manually copy the file into which you exported the GNS data from the location where you created it to a node in the cluster that you want to make a server cluster.

- You can only run this command on a multicluster server

Example

The following command imports the GNS data from a file named `gns.txt`:

```
# srvctl import gns -instance /temp/gnsdata/gns.txt
```

modify

Enables you to modify the instance configuration without removing and adding Oracle Clusterware resources. Using `modify` preserves the environment in the OCR configuration that would otherwise need to be reentered. The configuration description is modified in the OCR configuration, and a new Oracle Clusterware profile is generated and registered. The change takes effect when the application is next restarted.

Table A-74   srvctl modify Summary

Command	Description
srvctl modify asm	Modifies the configuration for Oracle ASM
srvctl modify cvu	Modifies the check interval of the Cluster Verification Utility
srvctl modify database	Modifies the configuration for a database
srvctl modify filesystem	Modifies the user authorized to mount and unmount the Oracle ACFS volume
srvctl modify gns	Modifies the GNS configuration
srvctl modify havip	Modifies a highly available VIP resource
srvctl modify instance	Modifies the configuration for an instance
srvctl modify listener	Modifies the listener configuration
srvctl modify mgmtdb	Modifies the management database configuration
srvctl modify mgmtlsnr	Modifies the management listener configuration
srvctl modify network	Modifies the configuration for a network
srvctl modify nodeapps	Modifies the configuration for a node application
srvctl modify oc4j	Modifies the RMI port for an OC4J instance
srvctl modify ons	Modifies the network configuration for the Oracle Notification Service daemon
srvctl modify scan	Modifies the SCAN VIP configuration to match that of a specific SCAN VIP
srvctl modify scan_listener	Updates the SCAN listener configuration to match that of the current SCAN VIP configuration
srvctl modify service	Modifies the configuration for a service
srvctl modify srvpool	Modifies a specific server pool
srvctl modify vip	Modifies the VIP to use a different IP address type

srvctl modify asm

Modify the listener used by Oracle ASM, the disk group discovery string used by Oracle ASM, or the SPFILE used by Oracle ASM for a noncluster database or a cluster database.

> **Note:**
>
> To manage Oracle ASM on Oracle Database 12*c* installations, use the SRVCTL binary in the Oracle Grid Infrastructure home for a cluster (Grid home). If you have Oracle RAC or Oracle Database installed, then you cannot use the SRVCTL binary in the database home to manage Oracle ASM.

Syntax and Parameters

Use the srvctl modify asm command with the following syntax:

```
srvctl modify asm [-listener listener_name] [-pwfile password_file_path]
  [-count {number_of_instances | ALL}] [-diskstring asm_diskstring]
  [-spfile spfile_path_name [-proxy] [-force]]
```

Table A-75   srvctl modify asm Parameters

Parameter	Description
-listener listener_name	The listener name with which Oracle ASM registers
-pwfile password_file_path	The full path to the location of the password file for Oracle ASM.
-count {number_of_instances	ALL}	Specify the number of Oracle ASM instances to create or that an Oracle ASM instance should be created on all nodes.
-diskstring asm_diskstring	The new Oracle ASM disk group discovery string
-spfile spfile_path_name	The path name of the new spfile to be used by Oracle ASM
-remote [-count number	ALL}	Indicates a remote ASM instance. You can specify the number of instances or ALL to have instances on all nodes.
-proxy	Specifies an Oracle ASM proxy instance
-force	Force modification
-rim	Specifies an ASM rim instance

Usage Notes

- The `srvctl modify asm` command with the `-listener` *listener_name* parameter can be used only with Oracle Clusterware.

- The `-diskstring` *asm_diskstring* and `-spfile` *spfile_path_name* parameters are available only with Oracle Restart.

> **See Also:**
>
> *Oracle Database Administrator's Guide* for more information about using the `srvctl modify asm` command with Oracle Restart

Example

An example of this command to modify the configuration of Oracle ASM is:

```
$ srvctl modify asm -listener lsnr1
```

srvctl modify cvu

Modifies the check interval for CVU in an Oracle Clusterware configuration.

Syntax and Parameters

Use the `srvctl modify cvu` command with the following syntax:

```
srvctl modify cvu [-checkinterval time_in_minutes]]
```

Table A-76   srvctl modify cvu Parameters

Parameter	Description
`-checkinterval` *time_in_minutes*	By default, CVU runs once every 360 minutes (six hours) and verifies the health of cluster. The shortest interval you can specify for CVU check is 10 minutes.

Examples

An example of this command to modify CVU to have a check interval of 4 hours is:

```
$ srvctl modify cvu -checkinterval 240
```

srvctl modify database

Modifies the configuration for a database.

Syntax and Parameters

Use the `srvctl modify database` command with the following syntax:

```
srvctl modify database -db db_unique_name [-eval] [-dbname db_name]
    [-instance instance_name] [-oraclehome oracle_home] [-user user_name]
    [-server server_list] [-timeout timeout] [-domain db_domain]
    [-spfile spfile] [-pwfile password_file_path]
    [-role {PRIMARY|PHYSICAL_STANDBY|LOGICAL_STANDBY|SNAPSHOT_STANDBY}]
    [-startoption start_options] [-stopoption stop_options]
    [-policy {AUTOMATIC | MANUAL | NORESTART}]
```

```
[-serverpool "server_pool_list" [-node node_name]]
[-pqpool pq_server_pools] [{-diskgroup "diskgroup_list" | -nodiskgroup}]
[-acfspath "acfs_path_list"] [-force]
```

Table A-77   srvctl modify database Parameters

Parameter	Description
-db db_unique_name	Unique name for the database.
-eval	Use this parameter to hypothetically evaluate the impact of the command on the system.
-dbname db_name	Name of the database (as specified by the DB_NAME initialization parameter), if it is different from the database unique name specified with the -database parameter.
-instance instance_name	Instance name prefix; this parameter is required for administrator-managed Oracle RAC One Node databases.
-oraclehome oracle_home	Path for the Oracle home for the database.
-user user_name	The name of the user that owns the Oracle home directory. **Note:** If you specify the -user parameter, you must run this command in privileged mode.
-server "server_list"	List candidate servers for Oracle RAC One Node databases. **Notes:** You can use this parameter only with administrator-managed Oracle RAC One Node databases. If your Oracle RAC One Node database is policy managed, you *cannot* use this parameter.
-timeout timeout	Online database relocation timeout, in minutes, for Oracle RAC One Node databases. The default is 30.
-domain db_domain	Specify a domain for the database. **Note:** If the database has the initialization parameter DB_DOMAIN set, then you must specify this parameter.
-spfile spfile	The full path to the location of the database server parameter file in an Oracle ASM disk group.
-pwfile password_file_path	The full path to the location of the password file in an Oracle ASM disk group.
-role [PRIMARY | PHYSICAL_STANDBY | LOGICAL_STANDBY | SNAPSHOT_STANDBY]	Role of the database in an Oracle Data Guard configuration role (PRIMARY, PHYSICAL_STANDBY, LOGICAL_STANDBY, or SNAPSHOT_STANDBY).

Table A-77   (Cont.) srvctl modify database Parameters

Parameter	Description
`-startoption start_options`	Startup options for the database, such as `OPEN`, `MOUNT`, or `NOMOUNT`. **Notes:** • For multi-word startup options, such as `read only` and `read write`, separate the words with a space and enclose in double quotation marks (`" "`). For example, `"read only"`. • When performing a switchover in an Oracle Data Guard configuration, the `-startoption` for a standby database that becomes a primary database is always set to `OPEN` after the switchover. **See Also:** *SQL*Plus User's Guide and Reference* for more information about startup options
`-stopoption stop_options`	Stop options for the database, such as `NORMAL`, `TRANSACTIONAL`, `IMMEDIATE`, or `ABORT`. **See Also:** *SQL*Plus User's Guide and Reference* for more information about shutdown options
`-policy [AUTOMATIC` ` | MANUAL | NORESTART`	Management policy for the database resource.
`-serverpool "server_pool_list"`	A comma-delimited list of the names of server pools to use for a policy-managed database. **Notes:** • If the database you are modifying is administrator managed, then this parameter changes it to be policy managed, in addition to changing all the database's services to run as uniform services in the specified server pool. You can specify only one server pool for converting an administrator-managed database to policy managed. • This parameter can be used only with Oracle Clusterware and Oracle RAC.
`-node node_name`	The name of the node in the specified server pool on which to run a single-instance database. This parameter can only be specified with the `-serverpool` parameter.
`-diskgroup "diskgroup_list"`	Comma-delimited list of Oracle ASM disk groups.
`-nodiskgroup`	To remove the database's dependency on Oracle ASM disk groups.

Table A-77   (Cont.) srvctl modify database Parameters

Parameter	Description
`-acfspath "acfs_path_list"`	A single Oracle ACFS path or a comma-delimited list of Oracle ACFS paths enclosed in double quotation marks (`" "`) where the database's dependency is set.
	Use this parameter to create dependencies on Oracle ACFS file systems other than `ORACLE_HOME`, such as for when the database uses `ORACLE_BASE` on a file system that is different from the `ORACLE_HOME` file system.
`-force`	Force the operation even though the some resources might be stopped.

Usage Notes

- The `srvctl modify database` command can convert administrator-managed databases to policy-managed databases. For a running administrator-managed database, if the server list is supplied, then the node where the database is running must be on that list. The instance name prefix cannot be modified after running the `srvctl add database` command.

- You cannot change the management policy from `AUTOMATIC` (using the `-policy` parameter) for Oracle RAC One Node databases. Any attempt to do so results in an error message. The same is true for the `-node` parameter, which is used to change the node on which a noncluster database runs.

- For policy-managed Oracle RAC One Node databases, you can use the `-serverpool` parameter to move an Oracle RAC One Node database between server pools but you can only specify one server pool. Specifying a list of server pools returns an error.

Examples

The following example changes the role of a database to a logical standby:

```
$ srvctl modify database -db crm -role logical_standby
```

The following example directs the `racTest` database to use the `SYSFILES`, `LOGS`, and `OLTP` Oracle ASM disk groups.

```
$ srvctl modify database -db racTest -diskgroup "SYSFILES,LOGS,OLTP"
```

srvctl modify filesystem

Modifies the configuration of the file system resource.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl modify filesystem` command with the following syntax:

```
srvctl modify filesystem -device volume_device -user user_name
    [-path mountpoint_path] [-node node_list | -serverpool
    server_pool_list] [-fsoptions options] [-description description]
    [-autostart {ALWAYS | NEVER | RESTORE}]
```

Table A-78   srvctl modify filesystem Parameters

Parameter	Description
`-device volume_device`	Specify the path to the file system volume device that you want to modify.
`-user user_name`	Name of the user that is authorized to mount and unmount the file system. If you do not specify this parameter, then SRVCTL defaults to the user running the command.
`-path mountpoint_path`	Specify the mount point path.
`-node node_list |` `-serverpool server_pool_list`	Specify a comma-delimited list of either nodes or server pools on which to mount the file system device.
`-fsoptions options`	Specify a comma-delimited list of file system mount options.
`-description description`	Specify a file system description.
`-autostart {ALWAYS | NEVER | RESTORE}`	Specify the file system resource autostart policy. ALWAYS: The file system resource always automatically starts NEVER: The file system does not automatically start RESTORE: The file system is restored to its last state. This is the default option.

Usage Notes

- You must run this command as `root` user on Linux and UNIX platforms.

- When performing online changes to the file system resource, the changes take effect only after the resource is next (re)started.

Examples

The following example changes the authorized user to `sysad` for the `RACVOL1` volume:

```
# srvctl modify filesystem -device /dev/asm/racvol1 -user sysad
```

srvctl modify gns

Modifies the IP address, domain, or other configuration parameters used by GNS.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl modify gns` command with one of the following syntax models:

```
srvctl modify gns -loglevel log_level

srvctl modify gns [-resolve name] [-verify name]
    [-parameter  name:value[,name:value...]] [-vip {vip_name | ip} [-skip]]
    [-clientdata file_name] [-verbose]
```

Table A-79   srvctl modify gns Parameters

Parameter	Description
`-loglevel log_level`	Specify the GNS diagnostic logging level (acceptable values are 1-6)
`-resolve name`	Resolve the name you specify through GNS
`-verify name`	Check to see if the specified name is advertised through GNS
`-parameter name:value [,name: value...]`	Set the value of one or more configuration parameters
`-vip vip_name	ip`	Specify a VIP name or IP address on which GNS is to listen **Note:** You can modify this attribute using Online Resource Attribute Modification.
`-skip`	Skip the reachability check of the VIP address
`-clientdata file_name`	Modifies the GNS client data with client data contained in the specified file
`-verbose`	Verbose output

Example

An example of this command is:

```
$ srvctl modify gns -vip 192.0.2.15
```

srvctl modify havip

Modifies a highly available VIP (HAVIP) (used for highly available NFS exports).

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl modify havip` command with one of the following syntax models:

```
srvctl modify havip -id havip_name [-address {host_name | ip_address}
    [-netnum network_number] [-skip]] [-description text] [-homenode node_name]
```

Table A-80   srvctl modify havip Parameters

Parameter	Description
-id *havip_name*	Specify the unique name for the HAVIP resource you want to modify.
-address {*host_name* | *ip_address*}	Specify either a host name or an IPv4 IP address for the HAVIP you want to modify. **Note:** You must configure the IPv4 IP address with a non-DHCP, non-round robin DNS address.
-netnum *network_number*	Optionally, you can change the network resource upon which the HAVIP depends. The default value for this optional parameter is 1.
-skip	Specify this parameter to skip checking the reachability of the IP address.
-description *text*	Specify a text description for the HAVIP.
-homenode *node_name*	Optionally, you can specify a preferred node or an empty string to clear the home node.

Usage Notes

- You must run this command as `root` user on Linux and UNIX platforms.

- Oracle does not support using IPv6 addresses.

Example

An example of this command is:

```
# srvctl modify havip -id myhavip -address 192.168.16.17 -netnum 2
```

srvctl modify instance

For an administrator-managed database, this command modifies the configuration for a database instance from its current node to another node. For a policy-managed

database, this command defines an instance name to use when the database runs on the specified node.

> **Note:**
>
> - You *cannot* use this command to rename or relocate a running instance.
>
> - This command is only available with Oracle Clusterware and Oracle RAC.

Syntax and Parameters

Use the `srvctl modify instance` command with the following syntax:

```
srvctl modify instance -db db_unique_name -instance instance_name
    -node node_name
```

Table A-81   srvctl modify instance Parameters

Parameter	Description
-database db_unique_name	Unique name for the database
-instance instance_name	Database instance name **Note:** If you are modifying a policy-managed database instance, then the instance name must contain an underscore (_), such as pmdb1_1.
-node node_name	Name of the node on which to run the instance. You can set the value of this parameter to " " only for a policy-managed database.

Examples

The following example to changes the configuration of an administrator-managed database, `amdb`, so that the database instance, `amdb1`, runs on the specified node, `mynode`:

```
$ srvctl modify instance -db amdb -instance amdb1 -node mynode
```

The following example causes the policy-managed database `pmdb`, when and if it runs on `mynode`, to use the instance name `pmdb1`:

```
$ srvctl modify instance -db pmdb -instance pmdb1_1 -node mynode
```

The following example removes the directive established by the previous example:

```
$ srvctl modify instance -db pmdb -instance pmdb1_1 -node ""
```

srvctl modify listener

Changes the Oracle home directory from which the listener runs, the name of the operating system user who owns Oracle home directory from which the listener runs, the listener endpoints, or the public subnet on which the listener listens, either for the default listener, or a specific listener, that is registered with Oracle Restart or with Oracle Clusterware.

If you want to change the name of a listener, then use the srvctl remove listener and srvctl add listener commands.

Syntax and Parameters

Use the `srvctl modify listener` command with the following syntax:

```
srvctl modify listener [-listener listener_name] [-oraclehome oracle_home]
 [-user user_name] [-netnum network_number]
 [-endpoints "[TCP:]port_list[/IPC:key][/NMP:pipe_name][/TCPS:s_port][/SDP:port]"]
```

Table A-82   srvctl modify listener Parameters

Parameter	Description
`-listener listener_name`	The name of the listener. If you do not specify this parameter, then SRVCTL uses the default name LISTENER.
`-oraclehome oracle_home`	When this parameter is specified, SRVCTL moves the listener to run from the specified Oracle home. **Note:** When using this parameter, the command should be run as privileged user to enable SRVCTL to update resource ownership corresponding to the new ORACLE_HOME owner
`-user user_name`	The name of the operating system user who will own the specified Oracle listener. **Note:** This parameter can be used only with Oracle Clusterware.
`-netnum network_number`	This parameter changes the public subnet on which the listener listens. **Note:** You should always have at least one listener on the default network. Do not use this parameter to change the network of the only listener that listens on the default network.
`-endpoints "[TCP:]port_list [/IPC:key][/NMP:pipe_name][/TCPS:s_port][/SDP:port]"`	Protocol specifications for the listener. `port_list` is comma-delimited list of port numbers. **Note:** You can modify this attribute using Online Resource Attribute Modification.

Example

The following example changes the TCP ports for the default listener:

```
$ srvctl modify listener -endpoints "TCP:1521,1522"
```

srvctl modify mgmtdb

Modifies the configuration for the management database (CHM repository).

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl modify mgmtdb` command with the following syntax:

```
srvctl modify mgmtdb [-pwfile password_file_path] [-spfile spfile_path]
  [-startoption start_option] [-stopoption stop_option]
  [-diskgroup "diskgroup_list"]
```

Table A-83   srvctl modify mgmtdb Parameters

Parameter	Description
`-pwfile password_file_path`	The full path to the location of the password file for the management database.
`-spfile spfile_path_name`	The path name of the new spfile to be used by the management database
`-startoption start_options`	Startup options for the management database, such as OPEN, MOUNT, or NOMOUNT **Note:** For multi-word startup options, such as `read only` and `read write`, separate the words with a space and enclose in double quotation marks (`" "`). For example, `"read only"`. **See Also:** *SQL*Plus User's Guide and Reference* for more information about startup options
`-stopoption stop_options`	Stop options for the management database, such as NORMAL, TRANSACTIONAL, IMMEDIATE, or ABORT **See Also:** *SQL*Plus User's Guide and Reference* for more information about shutdown options
`-diskgroup "diskgroup_list"`	Comma-delimited list of Oracle ASM disk groups used by the management database

Example

The following example directs the management database to use the SYSFILES Oracle ASM disk group.

```
$ srvctl modify mgmtdb -diskgroup "SYSFILES"
```

srvctl modify mgmtlsnr

Modifies the configuration for the management listener resource (for CHM).

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl modify mgmtlsnr command with the following syntax:

```
srvctl modify mgmtlsnr [-endpoints "[TCP:]port_list[/IPC:key][/NMP:pipe_name]
  [/TCPS:s_port][/SDP:port]"]
```

Table A-84   srvctl modify mgmtlsnr Parameters

Parameter	Description
-endpoints "[TCP:]port_list [/ IPC:key][/NMP:pipe_name] [/TCPS:s_port][/SDP:port]"	Protocol specifications for the management listener. port_list is comma-delimited list of port numbers. **Note:** You can modify this attribute using Online Resource Attribute Modification.

Example

The following example changes the TCP ports for the management listener:

```
$ srvctl modify mgmtlsnr -endpoints "TCP:2521,2522"
```

srvctl modify network

Modifies the subnet, network type, or IP address type for a network.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl modify network command with one of the following syntax models:

```
srvctl modify network [-netnum network_number] [-subnet subnet/netmask
    [/if1[|if2|...]]] [-nettype network_type | -iptype {ipv4 | ipv6 | both]}
    [-pingtarget "ping_target_list"] [-verbose]
```

Table A-85   srvctl modify network Parameters

Parameter	Description
-netnum network_number	Specify a network number. The default is 1.

Table A-85   (Cont.) srvctl modify network Parameters

Parameter	Description
`-subnet` *`subnet`*/*`netmask`* `[/`*`if1`*`[`|*`if2`*|`...]]`	Specifies a subnet number for the public network. The netmask and interfaces specified, if any, change those of the network you are modifying. If you specify an IPv6 subnet, then enter a prefix length, such as 64, in place of *`netmask`*. If you do not specify any interface names, then the VIPs use any interface on the given subnet.
	If you are changing the network type using the `-nettype` parameter, then you must specify either an existing IPv4 or IPv6 network using the `-subnet` parameter. Additionally, the subnet and netmask you specify in the `-subnet` parameter do not change those of the network you are modifying.
`-nettype` *`network_type`*	Specify the network type: `static`, `dhcp`, `autoconfig`, or `mixed`.
`-iptype` *`ip_type`*	The type of IP address: `ipv4`, `ipv6`, or `both`.
`-pingtarget` `"`*`ping_target_list`*`"`	Specify a comma-delimited list of IP addresses or host names to ping.
`-verbose`	Verbose output.

Usage Notes

- On Linux and UNIX systems, you must be logged in as `root` and on Windows, you must be logged in as a user with Administrator privileges to run this command.

- You can modify the IP address type for a network from IPv4 to IPv6, or from IPv6 to IPv4.

  > **See Also:**
  >
  > *Oracle Clusterware Administration and Deployment Guide* for more information

- If you specify `static` for the network type, then you must provide the virtual IP address using the srvctl add vip command.

- If you specify `dhcp` for the network type, then the VIP agent obtains the IP address from a DHCP server.

- If you specify `autoconfig` for the network type, then the VIP agent generates a stateless IPv6 address for the network. You can only use this parameter for IPv6 networks. If the subnet/netmask specification is not for an IPv6 address, then SRVCTL returns an error.

- If you change a network from static to mixed, then you must first configure GNS, so that the dynamic addresses obtained can have names registered for them.

- If you specify mixed for the network type, then the VIP resource uses both a static IP address and an IP address obtained dynamically, either DHCP or autoconfig.

- If you specify mixed_autoconfig for the network type, then the VIP resource retains the static IP configuration and either obtains an IP address from a DHCP server for an IPv4 network specification or generates a stateless auto-configured IP address for an IPv6 network specification.

Examples

The following example changes the subnet number, netmask, and interface list:

```
# srvctl modify network -subnet 192.168.2.0/255.255.255.0/eth0
```

The following example changes the second network to DHCP:

```
# srvctl modify network -netnum 2 -nettype dhcp
```

The following example adds an IPv6 subnet and netmask to the default network:

```
# srvctl modify network -subnet 2606:b400:400:18c0::/64
```

The following example removes the IPv4 configuration from a network:

```
# srvctl modify network -iptype ipv6
```

srvctl modify nodeapps

Modifies the configuration for a node application.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl modify nodeapps command with one of the following syntax models, specifying either a specific node and VIP or a specific subnet and netmask:

```
srvctl modify nodeapps {[-node node_name -address {vip_name|vip_address}/
    netmask[/if1[|if2|...]] [-skip]] [-nettype network_type] [-emport em_port]
    [-onslocalport ons_local_port] [-onsremoteport ons_remote_port]
    [-remoteservers host:[port][,host:port,...]] [-verbose]
    [-clientdata file] [-pingtarget "ping_target_list"]

srvctl modify nodeapps [-subnet subnet/netmask[/if1[|if2|...]]
    [-nettype network_type] [-emport em_port]
    [-onslocalport ons_local_port] [-onsremoteport ons_remote_port]
    [-remoteservers host:[port][,host:port,...]] [-verbose]
    [-clientdata file] [-pingtarget "ping_target_list"]
```

Table A-86   srvctl modify nodeapps Parameters

Parameter	Description
`-node node_name`	The name of the node on which the node application you want to modify resides.
`-address {vip_name|vip_address}/ netmask[/if1[|if2|...]]`	Node-level virtual IP name or address. The address specified by name or IP must match the subnet number of the default network. **Note:** This parameter must be used for upgrade configurations and new non-DHCP configurations
`-skip`	Specify this parameter to skip checking the reachability of the VIP address.
`-subnet subnet/netmask [/if1[|if2|...]]`	Specifies a subnet number for the public network. The netmask and interfaces specified, if any, change those of the default network. Additionally, if you specify the *netmask* option, then you need only specify it for the first node on each network.
`-nettype network_type`	Specifies the network server type, such as `static`, `dhcp`, or `mixed`.
`-emport em_port`	Local port on which Oracle Enterprise Manager listens. **Note:** You can modify this attribute using Online Resource Attribute Modification.
`-onslocalport ons_local_port`	Port on which the Oracle Notification Service daemon listens for local client connections. **Notes:** • The local port and remote port must each be unique. • You can modify the local port while the resource remains online, without restarting the resource.
`-onsremoteport ons_remote_port`	Port on which the Oracle Notification Service daemon listens for connections from remote hosts. **Notes:** • The local port and remote port must each be unique. • You can modify the remote port while the resource remains online, without restarting the resource.
`-remoteservers host:port, [host:port,...]`	List of `host:port` pairs of remote hosts that are part of the Oracle Notification Service network but are not part of the cluster. If you do not specify a `port` for a remote host, then the utility uses the value you specified for *ons_remote_port*.

Table A-86   (Cont.) srvctl modify nodeapps Parameters

Parameter	Description
`-clientdata file`	Specify the file with a wallet to import, or an empty string to delete a wallet used for SSL to secure Oracle Notification Service communication.
`-pingtarget "pingtarget_list"`	Specify a comma-separated list enclosed in double quotation marks (`""`) of IPs or host names to ping.
`-verbose`	Verbose output.

Example

The following example changes the nodeapps resource on `mynode1` to use the application VIP of 100.200.300.40 with a subnet mask of 255.255.255.0 on the network interface `eth0`:

```
$ srvctl modify nodeapps -node mynode1 -addr 100.200.300.40/255.255.255.0/eth0
```

srvctl modify oc4j

Modifies the RMI port for the OC4J instances.

Syntax and Parameters

Use the `srvctl modify oc4j` command with the following syntax:

```
srvctl modify oc4j -rmiport port_number [-verbose] [-force]
```

Table A-87   srvctl modify oc4j Parameters

Parameter	Description
`-rmiport port_number`	The RMI port number used by the OC4J instance **Note:** You can modify this attribute using Online Resource Attribute Modification.
`-verbose`	Verbose output
`-force`	Stops and restarts the resource to effect a change

Example

An example of this command is:

```
$ srvctl modify oc4j -rmiport 5385
```

srvctl modify ons

Modifies the ports used by the Oracle Notification Service daemon that is registered with Oracle Restart.

> **Note:**
>
> This command is available only with Oracle Restart.

Syntax and Parameters

Use the `srvctl modify ons` command with the following syntax:

```
srvctl modify ons [-onslocalprt ons_local_port] [-onsremoteport ons_remote_port]
     [-emport em_port] [-remoteservers host[:port][,host[:port]][...]] [-verbose]
```

Table A-88   srvctl modify ons Parameters

Parameter	Description
`-onslocalport ons_local_port`	The Oracle Notification Service daemon listening port for local client connections **Note:** The local port and remote port must each be unique.
`-onsremoteport ons_remote_port`	The Oracle Notification Service daemon listening port for connections from remote hosts **Note:** The local port and remote port must each be unique.
`-remoteservers host[:port] [,host[:port]][...]]`	A list of `host:port` pairs of remote hosts that are part of the Oracle Notification Service network but are not part of the Oracle Clusterware cluster **Note:** If you do not specify `port` for a remote host, then the utility uses `ons_remote_port`.
`-verbose`	Display verbose output

srvctl modify scan

Modifies the number of SCAN VIPs to match the number of IP addresses returned by looking up the `scan_name` you specify in DNS. You use this command when DNS was modified to add, change, or remove IP addresses, and now you must adjust the Oracle Clusterware resource configuration to match.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl modify scan` command with the following syntax:

```
srvctl modify scan -scanname scan_name
```

Table A-89   srvctl modify scan Parameters

Parameter	Description
`-scanname scan_name`	Identifies the SCAN name that resolves to the SCAN VIPs that you want to modify. **Note:** You can modify this attribute using Online Resource Attribute Modification.

Example

Assume your system currently has a SCAN named scan_name1, and it resolves to a single IP address in DNS. If you modify the SCAN scan_name1 in DNS to resolve to three IP addresses, then use the following command to create the additional SCAN VIP resources:

```
$ srvctl modify scan -scanname scan_name1
```

srvctl modify scan_listener

Modifies the SCAN listener to match SCAN VIP's or modifies the SCAN listener endpoints or service registration restrictions.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl modify scan_listener` command with the following syntax:

```
srvctl modify scan_listener {-update | -endpoints [TCP:]port[/IPC:key]
    [/NMP:pipe_name][/TCPS:s_port][/SDP:port] } [-invitednodes node_list]
    [-invitedsubnets subnet_list]
```

Table A-90   srvctl modify scan_listener Parameters

Parameter	Description
`-update`	Use this parameter to update SCAN listener configuration to match the current SCAN VIP configuration. This parameter adds new resources or removes existing SCAN listener resources to match the SCAN VIP resources.

`-endpoints [TCP:]port[/IPC:key]`
`[/NMP:pipe_name]`
`[/TCPS:s_port][/SDP:port]`
	Use this parameter to change SCAN listener end points.
	Note: You can modify this attribute using Online Resource Attribute Modification.

`-invitednodes node_list`	Use this parameter to specify a comma-delimited list of host names from outside the cluster that are allowed to register with the SCAN listener.

Table A-90   (Cont.) srvctl modify scan_listener Parameters

Parameter	Description
-invitedsubnets *subnet_list*	Use this parameter to specify a comma-delimited list of subnets from outside the cluster that are allowed to register with the SCAN listener.

Example

Assume your system currently has a SCAN named scan_name1, and you recently modified the DNS entry to resolve to three IP addresses instead of one. After running the srvctl modify scan command to create additional SCAN VIP resources, use the following command to create Oracle Clusterware resources for the additional two SCAN listeners to go with the two additional SCAN VIPs:

```
$ srvctl modify scan_listener -update
```

srvctl modify service

This command supports some online modifications to the service, such as:

- Moving a service member from one instance to another

- Performing online changes to service attributes from DBMS_SERVICE (for example, failover delay, runtime load balancing goal, and so on)

- Adding a new preferred or available instance

- Removing preferred or available instances for a service

> **WARNING:**
>
> Oracle recommends that you limit configuration changes to the minimum requirement and that you not perform other service operations while the online service modification is in progress.

Use one of the following forms of the srvctl modify service command with the specified syntax:

To move a service from one instance to another:

```
srvctl modify service -db db_unique_name -service service_name
     -oldinst old_instance_name -newinst new_instance_name [-force]
```

> **Note:**
>
> This form of the command is only available with Oracle Clusterware.

Table A-91   srvctl modify service Parameters for Moving a Service

Parameter	Description
-database *db_unique_name*	Unique name for the database

Table A-91   (Cont.) srvctl modify service Parameters for Moving a Service

Parameter	Description
`-service service_name`	Service name
`-oldinst old_instance_name`	Old instance name
`-newinst new_instance_name`	New instance name
`-force`	Disconnect all sessions during stop or relocate service operations

To change an available instance to a preferred instance for a service:

```
srvctl modify service -db db_unique_name -service service_name
    -available avail_inst_name -toprefer [-force]
```

> **Note:**
>
> This form of the command is only available with Oracle Clusterware and does not accept placement parameters for Oracle RAC One Node databases.

Table A-92   srvctl modify service Parameters for Changing to a Preferred Instance

Parameter	Description
`-db db_unique_name`	Unique name for the database.
`-service service_name`	Specify the name of the service you want to modify.
`-available available_inst_name`	Name of the available instance to change.
`-toprefer`	Change instance status to preferred.
`-force`	Disconnect all sessions during stop or relocate service operations. For planned operations, the user experience is best if using an Oracle Connection Pool with FAN, and not forcing disconnect. The FAN planned event causes the Oracle pool to drain the requests with no interruption to the users.

To change the available and preferred status for multiple instances:

```
srvctl modify service -db db_unique_name -service service_name
    -modifyconfig -preferred "preferred_list" [-available "available_list"]
    [-force]
```

> **Note:**
>
> This form of the command is only available with Oracle Clusterware and does not accept placement parameters for Oracle RAC One Node databases.

Table A-93   srvctl modify service Parameters for Changing Status of Multiple Instances

Parameter	Description
-db *db_unique_name*	Unique name for the database
-service *service_name*	Service name
-modifyconfig	Uses only the instances named for this service (unnamed instances already assigned to the service are removed)
-preferred "*preferred_instance_list*"	Comma-delimited list of preferred instances
-available "*available_instance_list*"	Comma-delimited list of available instances
-force	Disconnect all sessions during stop or relocate service operations. For planned operations, the user experience is best if using an Oracle Connection Pool with FAN, and not forcing disconnect. The FAN planned event causes the connection pool to drain the requests with no interruption to the users.

To modify other service attributes or to modify a service for Oracle Clusterware:

```
srvctl modify service -db db_unique_name -service service_name [-eval]
 [-serverpool pool_name] [-cardinality {UNIFORM|SINGLETON}]
 [-pqservice pqsvc_name] [-pqpool pq_pool_list]
 [-tafpolicy {BASIC|NONE}] [-edition edition_name]
 [-role "[PRIMARY][,PHYSICAL_STANDBY][,LOGICAL_STANDBY][,SNAPSHOT_STANDBY]"]
 [-notification {TRUE|FALSE}]  [-dtp {TRUE|FALSE}]
 [-clbgoal {SHORT|LONG}] [-rlbgoal {NONE|SERVICE_TIME|THROUGHPUT}]
 [-failovertype {NONE|SESSION|SELECT|TRANSACTION}] [-failovermethod {NONE|BASIC}]
 [-failoverretry failover_retries] [-failoverdelay failover_delay]
 [-policy {AUTOMATIC | MANUAL}] [-pdb pluggable_database]
 [-sql_translation_profile profile_name] [-commit_outcome {TRUE|FALSE}]
 [-retention retention_time] [-replay_init_time replay_initiation_time]
 [-session_state {STATIC|DYNAMIC}] [-global_override] [-verbose] [-force]
```

Table A-94   srvctl modify service Parameters

Parameter	Description
-db *db_unique_name*	Unique name for the database

Table A-94   (Cont.) srvctl modify service Parameters

Parameter	Description
`-service service_name`	Service name
`-eval`	Use this parameter to hypothetically evaluate the impact of the command on the system. **Note:** You can only use this parameter with a policy-managed service.
`-serverpool pool_name`	The name of a server pool used when the database is policy managed. **Note:** This parameter can be used only with Oracle RAC and only for policy-managed databases.
`-cardinality {UNIFORM	SINGLETON}`	The cardinality of the service, either UNIFORM (offered on all instances in the server pool) or SINGLETON (runs on only one instance at a time) **Note:** This parameter can be used only with Oracle Clusterware.
`-pqservice pqsvc_name`	Comma-delimited list of parallel query service names
`-pqpool pq_pool_list`	Comma-delimited list of parallel query server pool names
`-tafpolicy {BASIC|NONE}`	TAF policy specification (for administrator-managed databases only).
`-edition edition_name`	The initial session edition of the service. When an edition is specified for a service, all subsequent connections that specify the service use this edition as the initial session edition. However, if a session connection specifies a different edition, then the edition specified in the session connection is used for the initial session edition. SRVCTL does not validate the specified edition name. During connection, the connect user must have USE privilege on the specified edition. If the edition does not exist or if the connect user does not have USE privilege on the specified edition, then an error is raised.

Table A-94   (Cont.) srvctl modify service Parameters

Parameter	Description

`-role "[PRIMARY]`
`[,PHYSICAL_STANDBY]`
`[,LOGICAL_STANDBY]`
`[,SNAPSHOT_STANDBY]"`
	The database modes for which the service should be started automatically. You can specify one or more roles in a comma-delimited list.
	
	Note: The `-role` parameter is only used at database startup and by the Oracle Data Guard Broker. All manual service startup must specify the name of the service to be started by the user.
	
	See Also:
	Oracle Data Guard Broker if you specify `-role`

`-notification {TRUE	FALSE}`	Enable Fast Application Notification (FAN) for OCI connections
`-dtp {TRUE	FALSE}`	Indicates whether Distributed Transaction Processing should be enabled for this service. This ensures that the service is offered at exactly one instance at a time for XA affinity.
`-clbgoal {SHORT	LONG}`	For connection load balancing goal: set to SHORT if using runtime load balancing, set to LONG for long running connections such as batch jobs or older SQL*Forms style.
`-rlbgoal {NONE |` `SERVICE_TIME	THROUGHPUT}`	Runtime Load Balancing Goal. Set this parameter to SERVICE_TIME to balance connections by response time. Set this parameter to THROUGHPUT to balance connections by throughput.
`-failovertype {NONE |` `SESSION	SELECT |` `TRANSACTION}`	Failover type To enable Application Continuity for Java, set this parameter to TRANSACTION. To enable Transparent Application Failover (TAF) for OCI, set this parameter to SELECT or SESSION.
`-failovermethod {NONE |` `BASIC}`	TAF failover method (for backward compatibility only).
`-failoverretry` `failover_retries`	For Application Continuity and TAF, the number of attempts to connect after an incident.
`-failoverdelay` `failover_delay`	For Application Continuity and TAF, the time delay (in seconds) between reconnect attempts per incident at failover.
`-policy {AUTOMATIC	MANUAL}`	Service management policy.

Table A-94 (Cont.) srvctl modify service Parameters

Parameter	Description
`-pdb pluggable_database`	Pluggable database (PDB) name.
	Note: You can specify a PDB property when you create or modify a service. The PDB property associates the service with the specified PDB. You can view the PDB property for a service by querying the ALL_SERVICES data dictionary view or, when using the SRVCTL utility, by running the srvctl config service command.
	When create or modify a service with the specified PDB, SRVCTL does not check if the PDB exists.Before running this command, you must ensure that the PDB exists.
`-sql_translation_profile profile_name`	Use this parameter to specify a SQL translation profile for a service that you are modifying after you have migrated applications from a non-Oracle database to an Oracle database.
	If you want to set the SQL translation profile to a NULL value, then you must enter an empty string after the `-p` flag.
	Note: Before using the SQL translation feature, you must migrate all server-side application objects and data to the Oracle database.
	See Also: *Oracle Database Migration Guide* for more information about SQL translation
`-commit_outcome {TRUE|FALSE}`	Enable Transaction Guard; when set to TRUE, the commit outcome for a transaction is accessible after the transaction's session fails due to a recoverable outage.
`-retention retention_time`	For Transaction Guard (`commit_outcome` set to TRUE), this parameter determines the amount of time (in seconds) that the commit outcome is retained in the database.
`-replay_init_time replay_init_time`	For Application Continuity, this parameter specifies the time, in seconds, from when the original request started execution. Application Continuity will not replay after the specified amount of time has passed. This attribute avoids the unintentional replay of a request when a system is recovered after a long period. The default value is 300 (5 minutes).
`-session_state {STATIC|DYNAMIC}`	For Application Continuity, this parameter specifies whether the session state that is not transactional is changed by the application. Oracle recommends a value of DYNAMIC for most applications. If you are unsure which value to use or you can customize the application, then use DYNAMIC. STATIC mode is on request for database-agnostic applications, that never change non-transactional state inside a request.
`-global_override global_override`	Override value to modify the global service attributes.
	This parameter is used with the `-role`, `-policy`, `-notification`, `-failovertype`, `-failovermethod`, `-failoverdelay`, `-failoverretry` and `-edition` parameters.

Table A-94   (Cont.) srvctl modify service Parameters

Parameter	Description
-verbose	Display verbose output.
-force	Force the modify operation, stopping the service on some nodes as necessary.

Usage Notes

- When performing online changes to service attributes (for example, failover delay, Runtime Load Balancing Goal, and so on), the changes take effect only when the service is next (re)started.

- When a service configuration is modified so that a new preferred or available instance is added, the running state of the existing service is not affected. However, the newly added instances will not automatically provide the service, until a srvctl start service command is issued.

- When there are available instances for the service, and the service configuration is modified so that a preferred or available instance is removed, the running state of the service may change unpredictably:

    - The service is stopped and then removed on some instances according to the new service configuration.

    - The service may be running on some instances that are being removed from the service configuration.

    - These services will be relocated to the next *free* instance in the new service configuration.

Because of these considerations, when the online service is being modified, users may experience a brief service outage on some instances even if the instances are not being removed. Or users may experience a brief service outage on instances that are being removed from the service.

Examples

An example of moving a service member from one instance to another is:

```
$ srvctl modify service -db crm -service crm -oldinst crm1 -newinst crm2
```

An example of changing an available instance to a preferred instance is:

```
$ srvctl modify service -db crm -service crm -available crm1 -toprefer
```

The following command exchanges a preferred and available instance:

```
$ srvctl modify service -db crm -service crm -modifyconfig -preferred "crm1" \
-available "crm2"
```

srvctl modify srvpool

Modifies a server pool in a cluster. If minimum size, maximum size, and importance are numerically increased, then the CRS daemon may attempt to reassign servers to

this server pool, if by resizing other server pools have comparatively lower minimum size and importance, to satisfy new sizes of this server pool.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl modify srvpool command with the following syntax:

```
srvctl modify srvpool -serverpool pool_name [-eval] [-importance importance]
    [-min min_size] [-max max_size] [-servers "server_list"]
    [-category "server_category"] [-verbose] [-force]
```

Table A-95   srvctl modify srvpool Parameters

Parameter	Description
-serverpool *pool_name*	The name of the server pool to modify,
-eval	Use this parameter to hypothetically evaluate the impact of the command on the system.
-importance *importance*	The new importance of the server pool.
-min *min_size*	The new minimum size of the server pool. The default value is 0.
-max *max_size*	The new maximum size of the server pool. A value of -1 sets the server pool maximum size to UNLIMITED.
-servers "*server_list*"	A comma-delimited list of candidate server names. **Note:** In Oracle Database 12*c*, servers are assigned to server pools according to the value of the -category parameter.
-category "*server_category*"	Server category (or " " for empty category value).
-verbose	Display verbose output
-force	Force the operation even though the utility stops some resource(s).

Example

The following example changes the importance rank to 0, the minimum size to 2, and the maximum size to 4 for the server pool srvpool1:

```
$ srvctl modify srvpool -serverpool srvpool1 -importance 0 -min 2 -max 4
```

srvctl modify vip

Modifies IP address type but you can also use it to modify just the IP address.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl modify vip` command with the following syntax:

```
srvctl modify vip -node node_name -address {VIP_name|ip}/netmask[/if1[|if2|...]]
    [-netnum network_number] [-verbose]
```

Table A-96   srvctl modify vip Parameters

Parameter	Description
`-node node_name`	The name of the node on which you are changing the VIP.
`-address {VIP_name|ip}/ netmask [/if1[|if2|...]]`	Use this parameter to change the configuration of an existing VIP. If the VIP has an IPv4 address and the address you specify is IPv6, and the IP address type is set to `both` and the network type is set to `static`, then SRVCTL adds the IPv6 address to the existing IPv4 address of that resource. You can specify one `VIP_name` or IP address, along with an IPv4 netmask or IPv6 prefix length.
`-netnum network_number`	The optional network number from which VIPs are obtained. If you do not specify this parameter, then the VIPs are obtained from the same default network from which the `nodeapps` VIP is obtained.
`-verbose`	Verbose output

> **Note:**
>
> You cannot have multiple VIPs on the same net number (subnet or interface pair) on the same node.

Example

The following example adds an IPv4 address to a VIP, if one does not already exist. If the VIP has an IPv4 address, then it is replaced with the new network specification.

```
# srvctl modify vip -node node7 -address 192.168.16.17/255.255.255.0 -netnum 2
```

predict

The `predict` command predicts what happens when a resource fails and cannot be restarted on the same node. This command does not make any modifications to the system.

Table A-97   srvctl predict Summary

Command	Description
srvctl predict asm	Predicts the consequences of Oracle ASM failure
srvctl predict database	Predicts the consequences of database failure
srvctl predict diskgroup	Predicts the consequences of diskgroup failure
srvctl predict filesystem	Predicts the consequences of filesystem failure
srvctl predict listener	Predicts the consequences of listener failure
srvctl predict network	Predicts the consequences of network failure
srvctl predict oc4j	Predicts the consequences of OC4J failure
srvctl predict scan	Predicts the consequences of SCAN failure
srvctl predict scan_listener	Predicts the consequences of SCAN listener failure
srvctl predict service	Predicts the consequences of service failure
srvctl predict vip	Predicts the consequences of VIP failure

srvctl predict asm

Predicts the consequences of ASM failure.

Syntax and Parameters

Use the `srvctl predict asm` command with the following syntax:

```
srvctl predict asm [-node node_name] [-verbose]
```

Table A-98   srvctl predict asm Parameters

Parameter	Description
-node node_name	Node name
-verbose	Display verbose output

Examples

An example of this command to predict the consequences of an Oracle ASM failure:

```
$ srvctl predict asm -node crmnode2
```

srvctl predict database

The `srvctl predict database` command predicts what happens if the specified database fails.

Syntax and Parameters

Use the `srvctl predict database` command with the following syntax:

```
srvctl predict database -db db_unique_name [-verbose]
```

Table A-99   srvctl predict database Parameters

Parameter	Description
-db *db_unique_name*	Unique name for the database.
-verbose	Verbose output.

Example

The following example predicts what happens if the database named `racdb` fails:

```
srvctl predict database -db racdb
```

srvctl predict diskgroup

Predicts the consequences of and Oracle ASM disk group failure.

Syntax and Parameters

Use the `srvctl predict diskgroup` command with the following syntax:

```
srvctl predict diskgroup -diskgroup diskgroup_name [-verbose]
```

Table A-100   srvctl predict diskgroup Parameters

Parameter	Description
-diskgroup *diskgroup_name*	The name of the Oracle ASM disk group for which you want to evaluate failure
-verbose	Display verbose output.

Example

An example of this command is:

```
$ srvctl predict diskgroup -diskgroup data
```

srvctl predict filesystem

Predicts the consequences of file system failure.

Syntax and Parameters

Use the `srvctl predict filesystem` command with the following syntax:

```
srvctl predict filesystem -device volume_device_name [-verbose]
```

Table A-101   srvctl predict filesystem Parameters

Parameter	Description
-device *volume_device_name*	The path name of the filesystem volume device.
-verbose	Display verbose output.

Example

An example of this command is:

```
srvctl predict filesystem -device /dev/asm/volume1-123
```

srvctl predict listener

Predicts the consequences of listener failure.

Syntax and Parameters

Use the `srvctl predict listener` command with the following syntax:

```
srvctl predict listener listener_name [-verbose]
```

Table A-102   srvctl predict listener Parameters

Parameter	Description
-listener *listener_name*	Specify the name of the listener for which you want to predict the consequences of a failure.
-verbose	Display verbose output.

Example

An example of this command is:

```
$ srvctl predict listener -listener NODE3_CRMAPP_LISTENER
```

srvctl predict network

Predicts the consequences of network failure.

Syntax and Parameters

Use the `srvctl predict network` command with the following syntax:

```
srvctl predict network [-netnum network_number [-verbose]
```

Table A-103   srvctl predict network Parameters

Parameter	Description
-netnum *network_number*	Specify the network for which you want to evaluate failure. The default value is 1.
-verbose	Display verbose output.

Example

An example of this command is:

```
$ srvctl predict network -netnum 2
```

srvctl predict oc4j

Predicts the consequences of Oracle Container for Java (OC4J) failure.

Syntax and Parameters

Use the srvctl predict oc4j command with the following syntax:

```
srvctl predict oc4j [-verbose]
```

Table A-104   srvctl predict oc4j Parameters

Parameter	Description
-verbose	Display verbose output

Example

An example of this command is:

```
$ srvctl predict oc4j
```

srvctl predict scan

Predicts the consequences of SCAN failure.

Syntax and Parameters

Use the srvctl predict scan command with the following syntax:

```
srvctl predict scan -scannumber ordinal_number [-verbose]
```

Table A-105   srvctl predict scan Parameters

Parameter	Description
-scannumber *ordinal_number*	An ordinal number that identifies the SCAN VIP for which you want to simulate failure. The range of values you can specify for this parameter is 1 to 3.
-verbose	Display verbose output.

Example

An example of this command is:

```
$ srvctl predict scan -scannumber 1 -verbose
```

srvctl predict scan_listener

Predicts the consequences of SCAN listener failure.

Syntax and Parameters

Use the `srvctl predict scan_listener` command with the following syntax:

```
srvctl predict scan_listener -scannumber ordinal_number [-verbose]
```

Table A-106   srvctl predict scan_listener Parameters

Parameter	Description
`-scannumber ordinal_number`	An ordinal number that identifies the SCAN listener. The range of values you can specify for this parameter is 1 to 3.
`-verbose`	Display verbose output.

Example

An example of this command is:

```
$ srvctl predict scan_listener -scannumber 1
```

srvctl predict service

Predicts the consequences of service failure.

Syntax and Parameters

Use the `srvctl predict service` command with the following syntax:

```
srvctl predict service -db db_unique_name -service service_name [-verbose]
```

Table A-107   srvctl predict service Parameters

Parameter	Description
`-db db_unique_name`	Unique name for the database
`-service "service_name,..."`	Comma-delimited list of service names
`-verbose`	Display verbose output

Example

To evaluate the consequences of failure of the service named `crm` for the database `racdb`:

```
$ srvctl predict service -db racdb -service "crm"
```

srvctl predict vip

Predicts the consequences of VIP failure.

Syntax and Options

Use the `srvctl predict vip` command with the following syntax:

```
srvctl predict vip [-vip vip_name] [-verbose]
```

Table A-108   srvctl predict vip Parameters

Parameter	Description
-vip vip_name	Specify the name of the VIP for which you want to evaluate the consequences of failure.
-verbose	Display verbose output

Example

An example of this command is:

```
$ srvctl predict vip -vip racdb1_vip
```

relocate

The `relocate` command causes the specified object to run on a different node. The specified object must be running already.

The relocation of the object is temporary until you modify the configuration. The previously described modify command permanently changes the configuration.

Table A-109   srvctl relocate Summary

Command	Description
srvctl relocate asm	Relocate an Oracle Flex ASM instance from one node of an Oracle Flex Cluster to another
srvctl relocate cvu	Temporarily relocates the Cluster Verification Utility
srvctl relocate database	Relocates an Oracle RAC One Node database to a different node
srvctl relocate gns	Relocates GNS to a different node
srvctl relocate havip	Relocates a highly available VIP resource
srvctl relocate mgmtdb	Relocates the management database to a different node
srvctl relocate oc4j	Relocates an OC4J instance to a different node
srvctl relocate scan	Relocates a SCAN VIP from its current hosting server to another server within the cluster
srvctl relocate scan_listener	Relocates a SCAN listener from its current hosting server to another server within the cluster

Table A-109   (Cont.) srvctl relocate Summary

Command	Description
srvctl relocate server	Relocates named servers to another server pool
srvctl relocate service	Relocates the named service names from one named instance to another named instance
srvctl relocate vip	Relocates a specific VIP from one node to another node within the cluster

srvctl relocate asm

Relocates an Oracle Flex ASM instance from one Hub Node of an Oracle Flex Cluster to another.

Syntax and Parameters

Use the `srvctl relocate asm` command with the following syntax:

```
srvctl relocate asm -currentnode current_node [-targetnode target_node] [-force]
```

Table A-110   srvctl relocate asm Parameters

Parameter	Description
`-currentnode current_node`	Name of the node on which the Oracle Flex ASM instance currently resides
`-targetnode current_node`	Name of the node to which you want to relocate the Oracle Flex ASM instance
`-force`	Force the relocation of the Oracle Flex ASM instance

Usage Notes

You cannot use this command if there is only one Oracle Flex ASM instance running.

Example

An example of this command to relocate an Oracle Flex ASM instance from the current node to a node named `hubnode2` is:

```
$ srvctl relocate asm -currentnode hubnode1 -targetnode hubnode2
```

srvctl relocate cvu

Temporarily relocates the CVU to another node in a cluster.

Syntax and Parameters

Use the `srvctl relocate cvu` command with the following syntax:

```
srvctl relocate cvu [-node node_name]
```

Table A-111   srvctl relocate cvu Parameters

Parameter	Description
-node *node_name*	Node name

Example

An example of this command to temporarily relocate CVU from the local node to the cluster node named crmnode2 is:

```
$ srvctl relocate cvu -node crmnode2
```

srvctl relocate database

The srvctl relocate database command initiates the relocation of an Oracle RAC One Node database from one node to another node. This command also cleans up after a failed relocation.

The srvctl relocate database command can only be used for relocating Oracle RAC One Node databases.

Syntax and Parameters

Use the srvctl relocate database command with one of these syntax models:

```
srvctl relocate database -db db_unique_name [-node target_node]
    [-timeout timeout] [-stopoption NORMAL] [-verbose]

srvctl relocate database -db db_unique_name -abort [-revert] [-verbose]
```

Table A-112   srvctl relocate database Parameters

Parameter	Description
-db *db_unique_name*	Unique name of the database to relocate.
-node *target*	Target node to which to relocate the Oracle RAC One Node database. **Note:** You must supply this parameter if you are relocating an administrator-managed Oracle RAC One Node database.
-timeout *timeout*	Online database relocation timeout, in minutes, for Oracle RAC One Node databases. The default is 30.
-stopoption NORMAL	Use this parameter to shut down an existing database instance using SHUTDOWN NORMAL instead of the default of SHUTDOWN TRANSACTIONAL LOCAL for a primary database or SHUTDOWN IMMEDIATE for a standby database.
-abort	Abort failed online database relocation.
-revert	Removes the target node of a failed online relocation request from the candidate server list of an administrator-managed Oracle RAC One Node database.

Table A-112   (Cont.) srvctl relocate database Parameters

Parameter	Description
-verbose	Verbose output.

Usage Notes

- If the Oracle RAC One Node database you want to relocate is not running, then the command returns an error.

- If another online database relocation is active for this Oracle RAC One Node database, then the command returns an error.

- If an online database relocation for this Oracle RAC One Node database has failed and the target nodes are not the same for either relocation, then the command returns an error instructing you to abort the failed online database relocation and then initiate a new one.

- If an online database relocation for this Oracle RAC One Node database has failed and the target nodes are the same (or you do not specify the target), then the command attempts to relocate the database.

Example

The following example relocates an administrator-managed Oracle RAC One Node database named rac1 to a server called node7.

```
srvctl relocate database -db rac1 -node node7
```

srvctl relocate gns

Relocates GNS from its current hosting node to another node within the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl relocate gns command with the following syntax:

```
srvctl relocate gns [-node node_name] [-verbose]
```

Table A-113   srvctl relocate gns Parameters

Parameter	Description
-node node_name	The name of the node to which you want to move GNS
-verbose	Display verbose output.

Usage Notes

- On Linux and UNIX systems, you must be logged in as `root` and on Windows, you must be logged in as a user with Administrator privileges to run this command.

Example

An example of this command is:

```
# srvctl relocate gns -node node1
```

srvctl relocate havip

Relocates a highly available VIP (HAVIP) (used for highly available NFS exports) to another node in a cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl relocate havip` command with the following syntax:

```
srvctl relocate havip -id havip_name [-node node_name] [-force]
```

Table A-114   srvctl relocate havip Parameters

Parameter	Description
`-id havip_name`	Specify the unique name of the HAVIP resource you want to relocate.
`-node node_name`	Specify the name of the node to which you want to relocate the HAVIP resource.
`-force`	Optionally, you can force the relocation of the HAVIP resource.

Example

An example of this command is:

```
$ srvctl relocate havip -id myhavip -node node3
```

srvctl relocate mgmtdb

Relocates the management database (CHM repository) resource from one node of the cluster to another.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl relocate mgmtdb command with the following syntax:

```
srvctl relocate mgmtdb [-node node_name]
```

Specify a node to which you want to relocate the management database.

Example

An example of this command is:

```
$ srvctl relocate mgmtdb -node crsNode2
```

srvctl relocate oc4j

Relocates an OC4J instance from its current hosting node to another node within the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl relocate oc4j command with the following syntax:

```
srvctl relocate oc4j [-node node_name] [-verbose]
```

Table A-115   srvctl relocate oc4j Parameters

Parameter	Description
-node node_name	The name of the node to relocate the OC4J instance to.
-verbose	Display verbose output

Example

An example of this command is:

```
$ srvctl relocate oc4j -node crsNode01 -verbose
```

srvctl relocate scan

Relocates a specific SCAN VIP from its current hosting node to another node within the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl relocate scan command with the following syntax:

```
srvctl relocate scan -scannumber ordinal_number [-node node_name]
```

Table A-116   srvctl relocate scan Parameters

Parameter	Description
-scannumber ordinal_number	An ordinal number that identifies which SCAN VIP you want to relocate. The range of values you can specify for this parameter is 1 to 3.
-node node_name	The name of a single node. If you do not specify this parameter, then the utility chooses the node to which the SCAN VIP is relocated.

Example

An example of this command is:

```
$ srvctl relocate scan -scannumber 1 -node node1
```

srvctl relocate scan_listener

Relocates a specific SCAN listener from its current hosting node to another node within the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl relocate scan_listener command with the following syntax:

```
srvctl relocate scan_listener -scannumber ordinal_number [-node node_name]
```

Table A-117   srvctl relocate scan_listener Parameters

Parameter	Description
-scannumber ordinal_number	An ordinal number that identifies which SCAN VIP you want to relocate. The range of values you can specify for this parameter is 1 to 3.
-node node_name	The name of a single node. If you do not specify this parameter, then the utility chooses the node to which the SCAN VIP is relocated.

Example

An example of this command is:

```
$ srvctl relocate scan_listener -scannumber 3
```

srvctl relocate server

Relocates servers to a server pool in the cluster.

Syntax and Parameters

Use the `srvctl relocate server` command with the following syntax:

```
srvctl relocate server -servers "server_name_list" -serverpool pool_name
    [-eval] [-force]
```

Table A-118   srvctl relocate server Parameters

Parameter	Description
-servers "server_name_list"	A single server name or a comma-delimited list of server names enclosed in double quotation marks (" ") that you want to relocate to a different server pool.
-serverpool pool_name	The name of the server pool to which you want to move servers.
-eval	Use this parameter to hypothetically evaluate the impact of the command on the system.
-force	Use the -force parameter to force the relocation of servers even if it means stopping some resources.

Example

An example of this command is:

```
$ srvctl relocate server -servers "server1, server2" -serverpool sp3
```

srvctl relocate service

Temporarily relocates the specified service names from one specified instance to another specified instance. The `srvctl relocate service` command works on only one source instance and one target instance at a time, relocating a service from a single source instance to a single target instance.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl relocate service` command with the following syntax:

```
srvctl relocate service -db db_unique_name -service service_name {-oldinst
    old_inst_name -newinst new_inst_name | -currentnode source_node -targetnode
target_node}
    [-pq] [-force [noreplay]] [-eval] [-verbose]
```

Table A-119   srvctl relocate service Parameters

Parameter	Description
-db db_unique_name	Unique name for the database

Table A-119   (Cont.) srvctl relocate service Parameters

Parameter	Description
-service *service_name*	Service name
-oldinst *old_instance_name*	Old instance name
-newinst *new_instance_name*	New instance name **Note:** If you are using an administrator-managed, then you must use the -oldinst and -newinst parameters and the target instance must be on the preferred or available list for the service.
-currentnode *source_node*	Name of the node where the service is currently running
-targetnode *target_node*	Name of node where the service should be relocated **Note:** If you are using a policy-managed, then you must use the -currentnode and -targetnode parameters.
-pq	Performs the action on a parallel query service
-force	Disconnect all sessions during stop or relocate service operations
-noreplay	Disables session replay during disconnection
-eval	Use this parameter to hypothetically evaluate the impact of the command on the system
-verbose	Verbose output

Example

To temporarily relocate a named service member for the crm service from the database instance crm1 to the database instance crm3:

```
$ srvctl relocate service -db crm -service crm -oldinst crm1 -newinst crm3
```

srvctl relocate vip

Relocates a specific VIP from its current hosting node to another node within the cluster.

Syntax and Options

Use the srvctl relocate vip command with the following syntax:

```
srvctl relocate vip -vip vip_name [-node node_name] [-force] [-verbose]
```

Table A-120   srvctl relocate vip Parameters

Parameter	Description
`-vip vip_name`	Specify the name of the VIP you want to relocate.
`-node node_name`	Specify the name of the target node where the VIP should be relocated.
`-force`	Specify this option to force the relocation of the VIP.
`-verbose`	Display verbose output.

Example

An example of this command is:

```
$ srvctl relocate vip -vip vip1 -node node3
```

remove

Removes the configuration information for the specified target from Oracle Clusterware. Environment settings for the object are also removed. Using this command does not destroy the specified target.

Use the `remove` verb to remove the associated resource from the management of Oracle Clusterware or Oracle Restart. Depending on the noun used, you can remove databases, services, nodeapps, Oracle ASM, Oracle Notification Service, and listeners.

If you do not use the force parameter (`-force`), then Oracle Clusterware or Oracle Restart prompts you to confirm whether to proceed. If you use `-force`, then the remove operation proceeds without prompting and continues processing even when it encounters errors. Even when the Oracle Clusterware resources cannot be removed, the OCR configuration is removed, so that the object now appears not to exist, but there are still Oracle Clusterware resources. Use the force parameter (`-force`) with extreme caution because this could result in an inconsistent OCR.

To use the `remove` verb, you must first stop the node applications, database, instance, or service for which you are specifying `srvctl remove`. Oracle recommends that you perform a disable operation before using this command, but this is not required. You must stop the target object before running the `srvctl remove` command. See the stop command.

Table A-121   srvctl remove Summary

Command	Description
srvctl remove asm	Removes Oracle ASM instances
srvctl remove cvu	Removes the Cluster Verification Utility configured for the cluster
srvctl remove database	Removes a database and configuration
srvctl remove diskgroup	Removes a disk group from the Oracle Clusterware or Oracle Restart configuration

Table A-121   (Cont.) srvctl remove Summary

Command	Description
srvctl remove filesystem	Removes the configuration for an Oracle ACFS volume
srvctl remove gns	Removes GNS
srvctl remove havip	Removes a highly available VIP
srvctl remove instance	Removes instances and configurations of administrator-managed databases
srvctl remove listener	Removes the configuration of the specified listener from Oracle Clusterware or Oracle Restart
srvctl remove mgmtdb	Removes the management database from Oracle Clusterware
srvctl remove mgmtlsnr	Removes the management database listener from Oracle Clusterware
srvctl remove network	Removes a network from the cluster configuration
srvctl remove nodeapps	Removes node applications
srvctl remove oc4j	Removes the OC4J instance configuration
srvctl remove ons	Removes Oracle Notification Service instances
srvctl remove scan	Removes all Oracle Clusterware resources for all SCAN VIPs
srvctl remove scan_listener	Removes all Oracle Clusterware resources for all SCAN listeners
srvctl remove service	Removes services from the Oracle Clusterware or Oracle Restart configuration
srvctl remove srvpool	Removes a specific server pool
srvctl remove vip	Removes specific VIPs
srvctl remove volume	Removes a specific volume

srvctl remove asm

Removes the Oracle ASM resource from Oracle Clusterware management.

> **Note:**
>
> To manage Oracle ASM on Oracle Database 12*c* installations, use the SRVCTL binary in the Oracle Grid Infrastructure home for a cluster (Grid home). If you have Oracle RAC or Oracle Database installed, then you cannot use the SRVCTL binary in the database home to manage Oracle ASM.

Syntax and Parameters

Use the srvctl remove asm command with the following syntax:

```
srvctl remove asm [-proxy] [-force]
```

Table A-122   srvctl remove asm Parameters

Parameter	Description
-proxy	Specifies whether the Oracle ASM instance to remove is a proxy instance.
-force	Forcefully removes an Oracle ASM resource (ignores any dependencies)

Example

An example of this command is:

```
$ srvctl remove asm -force
```

srvctl remove cvu

Removes CVU from an Oracle Clusterware configuration.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl remove cvu command with the following syntax:

```
srvctl remove cvu [-force]
```

Use the -force parameter to remove a CVU resource that is running or starting.

Examples

An example of this command to remove CVU is:

```
$ srvctl remove cvu -force
```

srvctl remove database

Removes a database configuration.

Syntax and Parameters

Use the srvctl remove database command with the following syntax:

```
srvctl remove database -db db_unique_name [-force] [-noprompt] [-verbose]
```

Table A-123   srvctl remove database Parameters

Parameter	Description
-database *db_unique_name*	Unique name for the database
-force	Force remove (ignore dependencies)
-noprompt	Suppress prompts
-verbose	Display verbose output.

Example

An example of this command is:

```
$ srvctl remove database -db crm
```

srvctl remove diskgroup

Removes a specific Oracle ASM disk group resource from Oracle Clusterware or Oracle Restart.

Syntax and Parameters

Use the srvctl remove diskgroup command with the following syntax:

```
srvctl remove diskgroup -diskgroup diskgroup_name [-force]
```

Table A-124   srvctl remove diskgroup Parameters

Parameter	Description
-diskgroup *diskgroup_name*	The Oracle ASM disk group name.
-force	Force remove (ignore dependencies)

Example

An example of this command is:

```
$ srvctl remove diskgroup -diskgroup DG1 -force
```

srvctl remove filesystem

Removes a specific file system resource from the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl remove filesystem command with the following syntax:

```
srvctl remove filesystem -device volume_device_name [-force]
```

Table A-125   srvctl remove filesystem Parameters

Parameter	Description
-device *volume_device*	Specify the path to the file system resource device you want to remove.
-force	Ignores the resource dependencies and forcibly removes the resource from the cluster.

Usage Notes

- You must run this command as root user on Linux and UNIX platforms, or as an Administrator user on Windows platforms.

Example

An example of this command is:

```
# srvctl remove filesystem -device /dev/asm/racvol1
```

srvctl remove gns

Removes GNS from the cluster.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl remove gns command with the following syntax:

```
srvctl remove gns [-force] [-verbose]
```

Table A-126   srvctl remove gns Parameters

Parameter	Description
-force	Forcibly remove GNS from the cluster regardless of any errors that might occur.
-verbose	Display verbose output.

Example

An example of this command is:

```
$ srvctl remove gns
```

srvctl remove havip

Removes a highly available VIP (HAVIP) (used for highly available NFS exports).

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl remove havip` command with the following syntax:

```
srvctl remove havip -id havip_name [-force]
```

Table A-127   srvctl remove havip Parameters

Parameter	Description
`-id havip_name`	Specify the unique name of the HAVIP resource you want to remove.
`-force`	Forcibly removes the resource from the cluster ignoring any dependencies.

Usage Notes

- You must first stop the HAVIP resource before you attempt to remove it.

- You must run this command as `root` user on Linux and UNIX platforms.

Example

An example of this command is:

```
# srvctl remove havip -id myhavip
```

srvctl remove instance

Removes the configurations for an instance of an administrator-managed database. To remove the configurations of a policy-managed database, you must shrink the size of the server pool with the srvctl modify srvpool command.

If you use the `-force` parameter, then any services running on the instance stop. Oracle recommends that you reconfigure services to not use the instance to be removed as a preferred or available instance before removing the instance.

> **Note:**
>
> - This command is only available with Oracle Clusterware and Oracle RAC.
>
> - If you attempt to use this command on an Oracle RAC One Node database, then the command returns an error stating that cannot remove the instance except by removing the database.

Syntax and Parameters

Use the srvctl remove instance command with the following syntax:

```
srvctl remove instance -db db_unique_name -instance instance_name
    [-noprompt] [-force]
```

Table A-128   srvctl remove instance Parameters

Parameter	Description
-db db_unique_name	Unique name for the administrator-managed database.
-instance instance_name	Instance name.
-noprompt	Suppress prompts
-force	Specify this parameter to skip checking that the instance is not running, and remove it even though it is running. This parameter also skips checking that the instance has no running services using it, and causes those services to stop before the instance is removed.

Example

An example of this command is:

```
$ srvctl remove instance -db crm -instance crm01
```

srvctl remove listener

Removes the configuration of a specific listener or all listeners from Oracle Clusterware or Oracle Restart.

Syntax and Parameters

Use the srvctl remove listener command with the following syntax:

```
srvctl remove listener [-listener listener_name | -all] [-force]
```

Table A-129   srvctl remove listener Parameters

Parameter	Description
-listener listener_name	Name of the listener that you want to remove.
	If you do not specify a listener name, then the listener name defaults to LISTENER for a database listener; LISTENER_ASM for an Oracle ASM listener; or LISTENER_LEAF for a Leaf Node listener.
-all	Removes all listener configurations.

Table A-129   (Cont.) srvctl remove listener Parameters

Parameter	Description
-force	Specify this parameter to skip checking whether there are other resources that depend on this listener, such as databases, and remove the listener anyway.

Examples

The following command removes the configuration for the listener named `lsnr01`:

```
$ srvctl remove listener -listener lsnr01
```

srvctl remove mgmtdb

Removes the management database (CHM repository) from Oracle Clusterware management.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl remove mgmtdb` command with the following syntax:

```
srvctl remove mgmtdb [-force] [-noprompt] [-verbose]
```

Table A-130   srvctl remove mgmtdb Parameters

Parameter	Description
-force	Forcibly remove the management database from the cluster regardless of any errors that might occur.
-noprompt	Suppress prompts
-verbose	Display verbose output.

Usage Notes

- You must first attempt to shut down the management database before you attempt to remove it.

Example

An example of this command is:

```
$ srvctl remove mgmtdb -noprompt
```

srvctl remove mgmtlsnr

Use this command to remove the management listener resource (for CHM) from Oracle Clusterware.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl remove mgmtlsnr` command with the following syntax:

```
srvctl remove mgmtlsnr [-force]
```

Use the `-force` parameter to forcibly remove the management listener, ignoring any dependencies.

Example

An example of this command is:

```
srvctl remove mgmtlsnr
```

srvctl remove network

Removes the network configuration. You must have full administrative privileges to run this command. On Linux and UNIX systems, you must be logged in as `root` and on Windows systems, you must be logged in as a user with Administrator privileges.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl remove network` command as follows:

```
srvctl remove network {-netnum network_number | -all} [-force] [-verbose]
```

Table A-131   srvctl remove network Parameters

Parameter	Description
`-netnum network_number`	Specifies which network to remove
`-all`	Remove all networks
`-force`	Force remove the networks

Table A-131   (Cont.) srvctl remove network Parameters

Parameter	Description
-verbose	Verbose output

Example

An example of this command is:

```
# srvctl remove network -netnum 3
```

srvctl remove nodeapps

Removes the node application configuration.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl remove nodeapps command as follows:

```
srvctl remove nodeapps [-force] [-noprompt] [-verbose]
```

Table A-132   srvctl remove nodeapps Parameters

Parameter	Description
-force	Force remove
-noprompt	Suppress prompts
-verbose	Verbose output

Usage Notes

You must have full administrative privileges to run this command. On Linux and UNIX systems, you must be logged in as root and on Windows systems, you must be logged in as a user with Administrator privileges.

Example

An example of this command is:

```
# srvctl remove nodeapps
```

srvctl remove oc4j

Removes the OC4J instance from the Oracle Clusterware configuration.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl remove oc4j` command with the following syntax:

```
srvctl remove oc4j [-force] [-verbose]
```

Table A-133   srvctl remove oc4j Parameters

Parameter	Description
-force	Force remove
-verbose	Verbose output

srvctl remove ons

Removes Oracle Notification Service from the Oracle Grid Infrastructure home.

> **Note:**
>
> This command is only available with Oracle Restart.

Syntax and Parameters

Use the `srvctl remove ons` command with the following syntax:

```
srvctl remove ons [-force] [-verbose]
```

Table A-134   srvctl remove ons Parameters

Parameter	Description
-force	Force remove
-verbose	Verbose output

srvctl remove scan

Removes Oracle Clusterware resources from all SCAN VIPs.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl remove scan` command with the following syntax:

```
srvctl remove scan [-force] [-noprompt]
```

Table A-135   srvctl remove scan Parameters

Parameter	Description
-force	Removes the SCAN VIPs even though there are SCAN listeners running that are dependent on the SCAN VIPs.
-noprompt	Suppress all prompts

Usage Notes

If you use the `-force` option, then SCAN VIPs that are running are not stopped before the dependent resources are removed, which may require manual cleanup.

Example

An example of this command is:

```
$ srvctl remove scan -force
```

srvctl remove scan_listener

Removes Oracle Clusterware resources from all SCAN listeners.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl remove scan_listener` command with the following syntax:

```
srvctl remove scan_listener [-force] [-noprompt]
```

Table A-136   srvctl remove scan_listener Parameters

Parameter	Description
-force	Removes the SCAN listener without stopping the SCAN listener if it is running
-noprompt	Suppress all prompts

Example

An example of this command is:

```
$ srvctl remove scan_listener -force
```

srvctl remove service

Removes the configuration for a service.

Syntax and Parameters

Use the `srvctl remove service` command as follows:

```
srvctl remove service -db db_unique_name -service service_name
    [-instance instance_name] [-global_override]
```

Table A-137   srvctl remove service Parameters

Parameter	Description
-db *db_unique_name*	Unique name for the database
-service *service_name*	Service name
-instance *instance_name*	Instance name of an administrator-managed database. **Note:** This parameter can be used only for Oracle Clusterware.
-global_override	Override value to operate on a global service. This parameter is ignored for a non-global service.

Examples

This command removes the `sales` service from all instances of the clustered database named `crm`:

```
$ srvctl remove service -db crm -service sales
```

The following example removes the `sales` service from a specific instance of the `crm` clustered database:

```
$ srvctl remove service -db crm -service sales -instance crm02
```

srvctl remove srvpool

Removes a specific server pool. If there are databases or services that depend upon this server pool, then those resources are removed from the server pool first so that the remove server pool operation succeeds.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl remove srvpool` command with the following syntax:

```
srvctl remove srvpool -serverpool pool_name [-eval] [-verbose]
```

Table A-138   srvctl remove srvpool Parameters

Parameter	Description
-eval	Evaluates the effects of removing a server pool without making any changes to the system.
-verbose	Display verbose output.

Usage Notes

If you successfully remove the specified server pool, then the CRS daemon may assign its servers to other server pools depending upon their minimum size, maximum size, and importance. The CRS daemon may also return these servers to its Free server pool.

Example

An example of this command is:

```
$ srvctl remove srvpool -serverpool srvpool1
```

srvctl remove vip

Removes specific VIPs.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl remove vip command with the following syntax:

```
srvctl remove vip -vip "vip_name_list" [-force] [-noprompt] [-verbose]
```

Table A-139   srvctl remove vip Parameters

Parameter	Description
-vip "vip_name_list"	A comma-delimited list of VIP names surrounded by double quotation marks ("")
-force	Force remove
-noprompt	Suppress prompts
-verbose	Verbose output

Example

An example of this command is:

```
$ srvctl remove vip -vip "vip1,vip2,vip3" -force -noprompt -verbose
```

srvctl remove volume

Removes a specific volume.

> **Note:**
>
> - This command is only available with Oracle Clusterware.
>
> - The volume gets created when you create volumes in Oracle ASM.

> **See Also:**
>
> *Oracle Automatic Storage Management Administrator's Guide* for more information about creating volumes

Syntax and Parameters

Use the `srvctl remove volume` command with one of these syntax models:

```
srvctl remove volume -volume volume_name -diskgroup disk_group_name [-force]
```

```
srvctl remove volume -device volume_device [-force]
```

Table A-140   srvctl remove volume Parameters

Parameter	Description
-volume *volume_name*	Specify the name of the volume that you want to remove. This parameter is required.
-diskgroup *disk_group_name*	Specify the name of the disk group in which the volume that you want to remove resides.
-device *volume_device*	Specify the path to the file system resource in which the volume that you want to remove resides.
-force	Removes the volume even if it is running.

Usage Notes

You must specify a particular volume that you want to remove. You can specify a volume that resides in either a particular disk group or on a particular volume device.

Example

The following example removes a volume named VOLUME1 that resides in a disk group named DATA:

```
$ srvctl remove volume -volume VOLUME1 -diskgroup DATA
```

setenv

The `setenv` command sets values for the environment in the configuration file. Use `setenv` to set environment variables—items such as language or `TNS_ADMIN`—for Oracle Clusterware that you would typically set in your profile or session when you manage this database or database instance.

The unsetenv command unsets values for the environment in the configuration file.

Table A-141   srvctl setenv Summary

Command	Description
srvctl setenv asm	Administers environment configuration for Oracle ASM
srvctl setenv database	Administers cluster database environment configurations
srvctl setenv listener	Administers listener environment configurations **Note:** You cannot use this command to administer SCAN listeners.
srvctl setenv mgmtdb	Administers environment configuration for the management database
srvctl setenv mgmtlsnr	Administers environment configuration for the management listener
srvctl setenv nodeapps	Administers node application environment configurations
srvctl setenv vip	Administers VIP environment configurations

srvctl setenv asm

Administers Oracle ASM environment configurations.

Syntax and Parameters

Use the `srvctl setenv asm` command with one of these syntax models:

```
srvctl setenv asm -envs "name=val[,name=val][...]"

srvctl setenv asm -env "name=val"
```

Table A-142   srvctl setenv asm Parameters

Parameter	Description
-envs "name=val[,name=val][...]"	Comma-delimited list of name-value pairs of environment variables
-env "name=val"	Enables single environment variable to be set to a value that contains commas or other special characters

Example

The following example sets the language environment configuration for Oracle ASM:

```
$ srvctl setenv asm -env "LANG=en"
```

srvctl setenv database

Administers cluster database environment configurations.

Syntax and Parameters

Use the `srvctl setenv database` command with one of these syntax models:

```
srvctl setenv database -db db_unique_name -envs "name=val[,name=val][...]"
```

```
srvctl setenv database -db db_unique_name -env "name=val"
```

Table A-143   srvctl setenv database Parameters

Parameter	Description
`-db db_unique_name`	Unique name for the database
`-envs "name=val,..."`	Comma-delimited list of name-value pairs of environment variables
`-env "name=val"`	Enables single environment variable to be set to a value that contains commas or other special characters

Example

The following example sets the language environment configuration for a cluster database:

```
$ srvctl setenv database -db crm -env LANG=en
```

srvctl setenv listener

Administers listener environment configurations.

Syntax and Parameters

Use the `srvctl setenv listener` with one of these syntax models:

```
srvctl setenv listener [-listener listener_name] -envs "name=val[,name=val][...]"
```

```
srvctl setenv listener [-listener listener_name] -env "name=val"
```

Table A-144   srvctl setenv listener Parameters

Parameter	Description
`-listener listener_name`	Name of the listener. If you do not specify this parameter, then the listener name defaults to `LISTENER`.

Table A-144   (Cont.) srvctl setenv listener Parameters

Parameter	Description
-envs "*name=val*"	Comma-delimited list of name-value pairs of environment variables.
-env "*name=val*"	Enables single environment variable to be set to a value that contains commas or other special characters.

Example

The following example sets the language environment configuration for the default listener:

```
$ srvctl setenv listener -env "LANG=en"
```

srvctl setenv mgmtdb

Administers the environment configuration for the management database (CHM repository).

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl setenv mgmtdb` command with one of these syntax models:

```
srvctl setenv mgmtdb -envs "name=val[,name=val][...]"
```

```
srvctl setenv mgmtdb -env "name=val"
```

Table A-145   srvctl setenv mgmtdb Parameters

Parameter	Description
-envs "*name=val,...*"	Comma-delimited list of name-value pairs of environment variables
-env "*name=val*"	Enables single environment variable to be set to a value that contains commas or other special characters

Example

The following example sets the language environment configuration for the management database:

```
$ srvctl setenv mgmtdb -env LANG=en
```

srvctl setenv mgmtlsnr

Administers the environment configuration for the management listener resource (for CHM).

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl setenv mgmtlsnr` command with one of these syntax models:

```
srvctl setenv mgmtlsnr -envs "name=val[,name=val][...]"
```

```
srvctl setenv mgmtlsnr -env "name=val"
```

Table A-146   srvctl setenv mgmtlsnr Parameters

Parameter	Description
`-envs "name=val,..."`	Comma-delimited list of name-value pairs of environment variables
`-env "name=val"`	Enables single environment variable to be set to a value that contains commas or other special characters

Example

The following example sets the language environment configuration for the management listener:

```
$ srvctl setenv mgmtlsnr -env LANG=en
```

srvctl setenv nodeapps

Sets the environment variables for the node application configurations.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl setenv nodeapps` command with the following syntax model:

```
srvctl setenv nodeapps {-namevals "name=val[,name=val][...]" |
    -nameval "name=val"} [-viponly] [-gsdonly] [-onsonly] [-verbose]
```

Table A-147   srvctl setenv nodeapps Parameters

Parameter	Description
`-envs "name=val[,name=val] [...]"`	Comma-delimited list of name-value pairs of environment variables
`-env "name=val"`	Enables single environment variable to be set to a value that contains commas or other special characters

Table A-147   (Cont.) srvctl setenv nodeapps Parameters

Parameter	Description
-viponly	Modify only the VIP configuration
-gsdonly	Modify only the GSD configuration
-onsonly	Modify only the ONS daemon configuration
-verbose	Verbose output

Example

To set an environment variable for all node applications:

```
$ srvctl setenv nodeapps -env "CLASSPATH=/usr/local/jdk/jre/rt.jar" -verbose
```

srvctl setenv vip

Administers cluster VIP environment configurations.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl setenv vip` command with the following syntax:

```
srvctl setenv vip -vip vip_name {-envs "name=val[,name=val,...]" |
    -env "name=val"} [-verbose]
```

Table A-148   srvctl setenv vip Parameters

Parameter	Description
-vip vip_name	Name of the VIP
-envs "name=val,..."	Comma-delimited list of name-value pairs of environment variables
-env "name=val"	Enables single environment variable to be set to a value that contains commas or other special characters
-verbose	Verbose output

Example

The following example sets the language environment configuration for a cluster VIP:

```
$ srvctl setenv vip -vip crm1-vip -env "LANG=en"
```

start

Starts Oracle Restart or Oracle Clusterware enabled, non-running applications for the database, all or named instances, all or named service names, or node-level applications. For the `start` command, and for other operations that use a connect string, if you do not provide a connect string, SRVCTL uses `/as sysdba` to perform the operation. To run such operations, the owner of the `oracle` binary executables must be a member of the OSDBA group, and users running the commands must also be in the OSDBA group.

Table A-149   srvctl start Summary

Command	Description
srvctl start asm	Starts Oracle ASM instances
srvctl start cvu	Starts the Cluster Verification Utility
srvctl start database	Starts the cluster database and its instances
srvctl start diskgroup	Starts a specified disk group on a number of nodes
srvctl start filesystem	Starts the Oracle ACFS volume resource
srvctl start gns	Starts GNS
srvctl start havip	Starts a specific HAVIP resource
srvctl start home	Starts Oracle Clusterware-managed or Oracle Restart-managed resources in a specific Oracle home
srvctl start instance	Starts the instance
srvctl start listener	Starts the specified listener or listeners
srvctl start mgmtdb	Starts the management database
srvctl start mgmtlsnr	Starts the management listener
srvctl start nodeapps	Starts the node applications
srvctl start oc4j	Starts the OC4J instance
srvctl start ons	Starts the Oracle Notification Service daemon for Oracle Restart
srvctl start scan	Starts all SCAN VIPs
srvctl start scan_listener	Starts all SCAN listeners
srvctl start service	Starts the service
srvctl start vip	Starts a VIP
srvctl start volume	Starts an enabled volume

srvctl start asm

Starts an Oracle ASM instance.

> **Note:**
>
> To manage Oracle ASM on Oracle Database 12*c* installations, use the SRVCTL binary in the Oracle Grid Infrastructure home for a cluster (Grid home). If you have Oracle RAC or Oracle Database installed, then you cannot use the SRVCTL binary in the database home to manage Oracle ASM.

Syntax and Parameters

Use the `srvctl start asm` command with the following syntax:

```
srvctl start asm [-proxy] [-node node_name [-startoption start_options]]
```

Table A-150   srvctl start asm Parameters

Parameter	Description
`-proxy`	Start a proxy Oracle ASM instance.
`-node node_name`	Node name **Note:** This parameter can be used only with Oracle Clusterware.
`-startoption start_options`	Options to startup command, for example `OPEN`, `MOUNT`, or `NOMOUNT` **Note:** For multi-word startup options, such as `read only` and `read write`, separate the words with a space and enclose in double quotation marks (`" "`). For example, `"read only"`. **See Also:** *SQL*Plus User's Guide and Reference* for more information about startup options

Examples

An example of this command to start an Oracle ASM instance on a single node of a cluster is:

```
$ srvctl start asm -node crmnode1
```

An example to start an Oracle ASM instance on all nodes in the cluster, or for a noncluster database, is:

```
$ srvctl start asm
```

srvctl start cvu

Starts the CVU resource on one node in a cluster. If you specify a node name, then CVU starts on that node.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl start cvu` command with the following syntax:

```
srvctl start cvu [-node node_name]
```

Specify a node on which you want to start the CVU resource.

Examples

An example of this command to start CVU on a single node of a cluster is:

```
$ srvctl start cvu -node crmnode1
```

srvctl start database

Starts a cluster database and its enabled instances and all listeners on nodes with database instances. You can disable listeners that should not be started.

Syntax and Parameters

Use the `srvctl start database` command with the following syntax:

```
srvctl start database -db db_unique_name [-eval]
  [-startoption start_options] [-stopconcurrency number_of_instances] [-node
node_name]
```

Table A-151   srvctl start database Parameters

Parameter	Description
-db *db_unique_name*	Unique name for the database
-eval	Use this parameter to hypothetically evaluate the impact of the command on the system.
-startoption *start_options*	Options for startup command (for example: OPEN, MOUNT, or NOMOUNT) **Notes:** • This command parameter supports all database startup options. • For multi-word startup options, such as read only and read write, separate the words with a space and enclose in double quotation marks (" "). For example, "read only". **See Also:** *SQL*Plus User's Guide and Reference* for more information about startup options
[-startconcurrency *number_of_instances*]	Specify a number of database instances to start simultaneously, or specify 0 for an empty start concurrency value.

Table A-151   (Cont.) srvctl start database Parameters

Parameter	Description
`-node node_name`	The name of the node on which you want to start the database
	Notes:
	• This command only applies to Oracle RAC One Node databases.
	• The node you specify must be in the candidate list for an administrator-managed Oracle RAC One Node database or in the server pool for a policy-managed Oracle RAC One Node database.
	• If the database is already running on a node than the one you specify, then the command returns an error.
	• If you do not specify a node, then Oracle Clusterware chooses which node on which to start the Oracle RAC One Node database according to its policies, such as dispersion, number of resources, and order of candidate nodes.
	• If there is an active online database relocation for the Oracle RAC One Node database you are attempting to start, then both instances will already be running and the command returns an error message saying so (remember that only during an online database relocation are two instances of an Oracle RAC One Node database in existence).
	If the online database relocation failed for the Oracle RAC One Node database and you do not specify a node, then the command attempts to start both database instances.
	If the online database relocation failed for the Oracle RAC One Node database and you specify a node, then the command attempts to abort the failed relocation and start the instance on that node.

Example

An example of this command is:

```
$ srvctl start database -db crm -startoption "read only"
```

srvctl start diskgroup

Start a a specific disk group resource on a number of specified nodes.

Syntax and Parameters

Use the `srvctl start diskgroup` command with the following syntax:

```
srvctl start diskgroup -diskgroup diskgrp_name [-node "node_list"]
```

Table A-152   srvctl start diskgroup Parameters

Parameter	Description
`-diskgroup diskgrp_name`	The Oracle ASM disk group name

Table A-152   (Cont.) srvctl start diskgroup Parameters

Parameter	Description
-node "*node_list*"	Comma-delimited list of node names on which to start the disk group resource **Note:** This parameter can be used only with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl start diskgroup -diskgroup diskgroup1 -node "mynode1,mynode2"
```

srvctl start filesystem

Starts (mounts) the file system resource.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl start filesystem command with the following syntax:

```
srvctl start filesystem -device volume_device [-node node_name]
```

Table A-153   srvctl start filesystem Parameters

Parameter	Description
-device *volume_device*	The path of the file system resource device that you want to start.
-node *node_name*	The name of the node on which the file system resource should be started. If you do not specify this parameter, then the utility starts the file system resource on all the available nodes in the cluster.

Usage Notes

- You must run this command as root user on Linux and UNIX platforms, or as an Administrator user on Windows platforms, or as a configured file system user.

> **See Also:**
>
> srvctl add filesystem for more information about configuring file system users

Examples

To start a file system on all configured nodes in the cluster:

```
$ srvctl start filesystem -device /dev/asm/data_db1-68
```

To start the file system on `node1`:

```
$ srvctl start filesystem -device /dev/asm/data_db1-68 -node node1
```

srvctl start gns

Starts GNS on a specific node.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl start gns` command with the following syntax:

```
srvctl start gns [-loglevel log_level] [-node node_name] [-verbose]
```

Table A-154   srvctl start gns Parameters

Parameter	Description
`-loglevel log_level`	Specify the level of logging with which GNS should run. Log levels vary between 1 (minimal tracing) and 6 (traces everything and is time consuming).
`-node node_name`	The name of a node in the cluster where you want to start GNS.
`-verbose`	Display verbose output

Usage Notes

You can only run this command in the **server cluster**. If you attempt to run this command in a **client cluster**, then an error occurs.

Example

An example of this command to start the GNS on the cluster node named `crmnode1` is:

```
$ srvctl start gns -node crmnode1
```

srvctl start havip

Starts a specific highly available VIP (HAVIP) (used for highly available NFS exports) on a specific node.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl start havip` command with the following syntax:

```
srvctl start havip -id havip_name [-node node_name]
```

Table A-155   srvctl start havip Parameters

Parameter	Description
-id *havip_name*	Specify the unique name of the HAVIP resource you want to start.
-node *node_name*	Optionally, you can use the -node parameter to specify the name of the node on which the HAVIP resource starts.

Usage Notes

• You must run this command as `root` user on Linux and UNIX platforms.

Example

An example of this command is:

```
# srvctl start havip -id myhavip -node myNode1
```

srvctl start home

Starts all the Oracle Restart-managed or Oracle Clusterware-managed resources on the specified Oracle home.

Syntax and Parameters

Use the `srvctl start home` command with the following syntax:

```
srvctl start home -oraclehome Oracle_home -statefile state_file -node node_name
```

Table A-156   srvctl start home Parameters

Parameter	Description
-oraclehome *Oracle_home*	The path to the Oracle home for which you want to start the Oracle Restart or Oracle Clusterware-managed resources.
-statefile *state_file*	The path name where you want the state file to be written.
-node *node_name*	The name of the node where the Oracle home resides. **Note:** This parameter can be used only with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl start home -oraclehome /u01/app/oracle/product/12.1.0/db_1
    -statefile ~/state.txt -node node1
```

srvctl start instance

Starts instances in the cluster database and all listeners on nodes with database instances. You can disable listeners that should not be started.

> **Note:**
>
> - This command is only available with Oracle Clusterware and Oracle RAC.
>
> - If you run this command on an Oracle RAC One Node database, then the command returns an error instructing you to use the `database` noun, instead.

Syntax and Parameters

Use the `srvctl start instance` command with one of these syntax models:

```
srvctl start instance -db db_unique_name -node node_name
   [-instance "instance_name"] [-startoption start_options]

srvctl start instance -db db_unique_name -instance "inst_name_list"
    [-startoption start_options]
```

In Windows, you must enclose the list of comma-delimited instance names in double quotation marks (`""`).

Table A-157   srvctl start instance Parameters

Parameter	Description
`-db db_unique_name`	Unique name for the database
`-node node_name`	The name of a single node **Note:** Use this parameter for policy-managed databases.
`-instance { "instance_name" | "inst_name_list" }`	The name of a single instance or a comma-delimited list of instance names **Note:** Use this parameter for policy-managed databases.
`-startoption start_options`	Options for startup command, such as OPEN, MOUNT, or NOMOUNT) **Note:** For multi-word startup options, such as read only and read write, separate the words with a space and enclose in double quotation marks (`""`). For example, `"read only"`. **See Also:** *SQL*Plus User's Guide and Reference* for more information about startup options

Example

An example of starting an instance for a policy-managed database is:

```
$ srvctl start instance -db crm -node node2
```

An example of starting instances for an administrator-managed database is:

```
$ srvctl start instance -db crm -instance "crm2,crm3"
```

srvctl start listener

Starts the default listener on the specified *node_name*, or starts the specified listener on all nodes that are registered with Oracle Clusterware or on the given node.

Syntax and Parameters

Use the srvctl start listener command with the following syntax:

srvctl start listener [-node *node_name*] [-listener *listener_name*]

Table A-158   srvctl start listener Parameters

Parameter	Description
-node *node_name*	Specify a particular node name to start the listener on that node. **Note:** This parameter can be used only with Oracle Clusterware.
-listener *listener_name*	Specify a particular listener name. Use the srvctl config listener command to obtain the name of a listener. If you do not specify this parameter, then the listener name defaults to LISTENER for a database listener; LISTENER_ASM for an Oracle ASM listener; or LISTENER_LEAF for a Leaf Node listener.

Example

An example of this command is:

$ srvctl start listener -listener LISTENER_LEAF

srvctl start mgmtdb

Starts the management database (CHM repository) resource.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl start mgmtdb command with the following syntax:

srvctl start mgmtdb [-startoption *start_options*] [-node *node_name*]

Table A-159   srvctl start mgmtdb Parameters

Parameter	Description
`-startoption` `start_options`	Options to startup command, for example OPEN, MOUNT, or NOMOUNT
	Note: For multi-word startup options, such as `read only` and `read write`, separate the words with a space and enclose in double quotation marks (`""`). For example, `"read only"`.
	See Also: *SQL*Plus User's Guide and Reference* for more information about startup options
`-node node_name`	Node name
	Note: This parameter can be used only with Oracle Clusterware.

Examples

An example of this command to start the management database on the `crmnode1` node of the cluster is:

```
$ srvctl start mgmtdb -node crmnode1
```

srvctl start mgmtlsnr

Starts the management listener resource (for CHM).

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl start mgmtlsnr` command with the following syntax:

```
srvctl start mgmtlsnr [-node node_name]
```

Specify the node on which you want to start the management listener.

Examples

For example, to start the management listener on the `crmnode1` node:

```
$ srvctl start mgmtlsnr -node crmnode1
```

srvctl start nodeapps

Starts node-level applications on a node or all nodes in the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl start nodeapps` command with the following syntax:

```
srvctl start nodeapps [-node node_name] [-gsdonly] [-adminhelper] [-verbose]
```

Table A-160   srvctl start nodeapps Parameters

Parameter	Description
-node *node_name*	Node name If you do not specify this parameter, then the utility starts the nodeapps on all active nodes in the cluster.
-gsdonly	Starts only GSD instead of all node applications
-adminhelper	Starts only an Administrator helper instead of all node applications
-verbose	Verbose output

Example

An example of this command is:

```
$ srvctl start nodeapps
```

srvctl start oc4j

Starts the OC4J instance.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl start oc4j command with the following syntax:

```
srvctl start oc4j [-node node_name] [-verbose]
```

You can specify a particular node on which to start the OC4J instance. Specify -verbose to indicate that verbose output should be displayed.

srvctl start ons

Starts the Oracle Notification Service daemon.

> **Note:**
>
> This command is only available with Oracle Restart.

Syntax and Parameters

Use the srvctl start ons command with the following syntax:

```
srvctl start ons [-verbose]
```

There is only one parameter for this command, -verbose, which is used to indicate that verbose output should be displayed.

srvctl start scan

Starts all SCAN VIPs, by default, or a specific SCAN VIP, on all nodes or a specific node in the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl start scan command with the following syntax:

```
srvctl start scan [-scannumber ordinal_number] [-node node_name]
```

Table A-161   srvctl start scan Parameters

Parameter	Description
-scannumber *ordinal_number*	An ordinal number that identifies which SCAN VIP you want to start. The range of values you can specify for this parameter is 1 to 3.
	If you do not specify this parameter, then the utility starts all the SCAN VIPs.
-node *node_name*	The name of a single node.
	If you do not specify this parameter, then the utility starts the SCAN VIPs on all nodes in the cluster.

Example

To start the SCAN VIP identified by the ordinal number 1 on the crm1 node, use the following command:

```
$ srvctl start scan -scannumber 1 -node crm1
```

srvctl start scan_listener

Starts all SCAN listeners, by default, or a specific listener on all nodes or a specific node in the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl start scan_listener command with the following syntax:

```
srvctl start scan_listener [-node node_name] [-scannumber ordinal_number]
```

Table A-162   srvctl start scan_listener Parameters

Parameter	Description
-scannumber *ordinal_number*	An ordinal number that identifies which SCAN Listener you want to start. The range of values you can specify for this parameter is 1 to 3.
	If you do not specify this parameter, then the utility starts all the SCAN listeners.
-node *node_name*	The name of a single node.
	If you do not specify this parameter, then the utility starts the SCAN listeners on all nodes in the cluster.

Example

An example of this command is:

```
$ srvctl start scan_listener -scannumber 1
```

srvctl start service

Starts a service or multiple services on the specified instance. The srvctl start service command will fail if you attempt to start a service on an instance if that service is already running on its maximum number of instances, that is, its number of preferred instances. You may move a service or change the status of a service on an instance with the srvctl modify service and srvctl relocate service commands described later in this appendix.

> **Note:** All manual service startup must specify the name of the service to be started by the user.

> **See Also:** *Oracle Data Guard Broker*

Syntax and Parameters

Use the srvctl start service command with the following syntax:

```
srvctl start service -db db_unique_name [-eval] [-service "services_list"
  [-node node_name | -instance instance_name | -serverpool pool_name |
   -global_override] [-startoption start_options] [-verbose]
```

Table A-163   srvctl start service Parameters

Parameter	Description
-db *db_unique_name*	Unique name for the database.
-eval	Use this parameter to hypothetically evaluate the impact of the command on the system.

Table A-163   (Cont.) srvctl start service Parameters

Parameter	Description
-service "*services_list*"	Comma-delimited list of service names. If you do not include this parameter, then SRVCTL starts all of the services for the specified database.
-node *node_name*	The name of the node where the service should be started. Use this parameter for policy-managed databases. **Note:** This parameter can be used only with Oracle Clusterware.
-instance *instance_name*	The name of the instance for which the service should be started. Use this parameter for administrator-managed databases. **Note:** This parameter can be used only with Oracle Clusterware.
-serverpool *pool_name*	The name of the server pool in which to start the service. Use this parameter for policy-managed databases.
-global_override	Override value to operate on a global service. Use this parameter only with global services; this parameter is ignored if specified for a non-global service.
-startoption *start_options*	Startup options used when service startup requires starting a database instance. Options include OPEN, MOUNT, and NOMOUNT. **Note:** For multi-word startup options, such as read only and read write, separate the words with a space and enclose in double quotation marks (""). For example, "read only". **See Also:** *SQL*Plus User's Guide and Reference* for more information about startup options
-verbose	Display verbose output.

Examples

The following example starts a named service. If the instances that support these services, including available instances that the service uses for failover, are not running but are enabled, then SRVCTL starts them.

```
$ srvctl start service -db crm -service crm
```

The following example starts a named service on a specified instance:

```
$ srvctl start service -db crm -service crm -instance crm2
```

srvctl start vip

Starts a specific VIP or a VIP on a specific node.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl start vip` command with the following syntax:

```
srvctl start vip {-node node_name | -vip vip_name } [-verbose]
```

Table A-164   srvctl start vip Parameters

Parameter	Description
`-node node_name`	Node name
`-vip vip_name`	The VIP name
`-verbose`	Verbose start

Example

An example of this command is:

```
$ srvctl start vip -vip crm1-vip -verbose
```

srvctl start volume

Starts a specific, enabled volume.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl start volume` command with the following syntax:

```
srvctl start volume {-volume volume_name -diskgroup disk_group_name |
    -device volume_device} [-node node_list]
```

Table A-165   srvctl start volume Parameters

Parameter	Description
`-volume volume_name`	Specify the name of the volume that you want to start. This parameter is required.
`-diskgroup disk_group_name`	Specify the name of the disk group in which the volume that you want to start resides.

Table A-165   (Cont.) srvctl start volume Parameters

Parameter	Description
`-device volume_device`	Specify the path to the volume device that you want to start.
`-node node_list`	Specify a comma-delimited list of node names where volumes that you want to start reside.

Usage Notes

The `srvctl start volume` command does not create a volume service. Provided that the volume already exists and the volume resource is enabled, SRVCTL attempts to start it. If the volume exists but the resource is disabled, then `srvctl start volume` returns an error.

Example

The following example starts a volume named `VOLUME1` that resides in a disk group named `DATA`:

```
$ srvctl start volume -volume VOLUME1 -diskgroup DATA
```

status

Displays the current state of a named database, instances, services, disk group, listener, node application, or other resource managed by Oracle Clusterware.

If you use the `-verbose` parameter with the status command, then SRVCTL displays the `INTERNAL_STATE`, which indicates actions in progress by Oracle Clusterware, such as starting, stopping, or cleaning. If the internal state is STABLE, then the `-verbose` parameter displays nothing because this is the normal state. Additionally, the `-verbose` parameter displays `STATE_DETAILS`, which may indicate additional information provided by the resource about why it is in its current state

Table A-166   srvctl status Summary

Command	Description
srvctl status asm	Displays the status of Oracle ASM instances
srvctl status cvu	Displays the status of the Cluster Verification Utility
srvctl status database	Displays the status of a database
srvctl status diskgroup	Displays the status of a specific disk group on a number of nodes
srvctl status filesystem	Displays the status of an Oracle ACFS volume
srvctl status gns	Displays the status of GNS
srvctl status havip	Displays the status of highly available VIPs
srvctl status home	Displays the status of the resources associated with the specified Oracle home

Table A-166   (Cont.) srvctl status Summary

Command	Description
srvctl status instance	Displays the status of a instance
srvctl status listener	Displays the status of a listener resource
srvctl status mgmtdb	Displays the status of the management database
srvctl status mgmtlsnr	Displays the status of the management listener
srvctl status nodeapps	Displays the status of node applications
srvctl status oc4j	Determines which node is running the Oracle Database QoS Management server
srvctl status ons	Displays the status of Oracle Notification Service
srvctl status scan	Displays the status of SCAN VIPs
srvctl status scan_listener	Displays the status of SCAN listeners
srvctl status server	Displays the status of servers
srvctl status service	Displays the status of services
srvctl status srvpool	Displays the status of server pools
srvctl status vip	Displays the status of VIPs
srvctl status volume	Displays the status of volumes

srvctl status asm

Displays the status of an Oracle ASM instance.

> **Note:**
>
> To manage Oracle ASM on Oracle Database 12*c* installations, use the SRVCTL binary in the Oracle Grid Infrastructure home for a cluster (Grid home). If you have Oracle RAC or Oracle Database installed, then you cannot use the SRVCTL binary in the database home to manage Oracle ASM.

Syntax and Parameters

Use the `srvctl status asm` command with the following syntax:

```
srvctl status asm [-proxy] [-node node_name] [-detail] [-verbose]
```

Table A-167   srvctl status asm Parameters

Parameter	Description
-proxy	Display the status for an Oracle ASM proxy instance.

Table A-167   (Cont.) srvctl status asm Parameters

Parameter	Description
-node *node_name*	Node name. If you do not specify this parameter, the SRVCTL displays the status of all Oracle ASM instances. **Note:** This parameter can be used only with Oracle Clusterware.
-detail	Print detailed status information.
-verbose	Displays verbose output.

Example

An example of this command is:

```
$ srvctl status asm -node crmnode1 -detail
```

srvctl status cvu

Displays the current state of the CVU resource on one node in a cluster. If you specify a node name, then the command checks CVU status on that node.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl status cvu command with the following syntax:

```
srvctl status cvu [-node node_name]
```

You can optionally specify the node where the CVU resource is running, if it is not the local node.

Example

An example of this command to check the status of CVU on the node in the cluster named crmnode1 is:

```
$ srvctl status cvu -node crmnode1
```

srvctl status database

Displays the status of instances and their services, and where the instances are running.

If you run this command on an Oracle RAC One Node database, then the output shows the status of any online database relocation (active, failed, or inactive), and the source and destination nodes of the relocation.

Syntax and Parameters

Use the srvctl status database command with the following syntax:

```
srvctl status database -db db_unique_name [-force] [-verbose]
```

Table A-168   srvctl status database Parameters

Parameter	Description
-db db_unique_name	Unique name for the database
-force	Include disabled applications
-verbose	Displays verbose output.

Example

An example of this command is:

```
$ srvctl status database -db crm -verbose
```

srvctl status diskgroup

Displays the status of a specific disk group on a number of specified nodes.

Syntax and Parameters

Use the srvctl status diskgroup command with the following syntax:

```
srvctl status diskgroup -diskgroup diskgroup_name [-node "node_list"]
    [-detail] [-verbose]
```

Table A-169   srvctl status diskgroup Parameters

Parameter	Description
-diskgroup diskgroup_name	The Oracle ASM disk group name
-node node_list	Comma-delimited list of node names on which to check status of the disk group **Note:** This parameter can be used only with Oracle Clusterware.
-detail	Display detailed status information for the disk group
-verbose	Displays verbose output.

Example

An example of this command is:

```
$ srvctl status diskgroup -diskgroup dgrp1 -node "mynode1,mynode2" -detail
```

srvctl status filesystem

Displays the status of the file system resource.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl status filesystem` command with the following syntax:

```
srvctl status filesystem [-device volume_device] [-verbose]
```

Table A-170   srvctl status filesystem Parameters

Parameter	Description
-device *volume_device*	The path of the file system resource for which you want to obtain the status. If you do not specify this parameter, then SRVCTL displays the status of all file systems.
-verbose	Displays verbose output.

Examples

This command displays output similar to the following, depending on whether you specify a device name:

If you specify a device name:

```
$ srvctl status filesystem -device /dev/asm/racvol_1
ACFS file system is not mounted on node1
ACFS file system is not mounted on node2
```

If you do not specify a device name:

```
$ srvctl status filesystem
ACFS file system is not running
ACFS file system is running on node1,node3
```

Note:

In the preceding examples, the file system is Oracle ACFS. If you are using other file systems, then they will display as EXT3 or EXT4.

---

srvctl status gns

Displays the current state of GNS.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl status gns` command with the following syntax:

```
srvctl status gns [-node node_name] [-verbose]
```

Table A-171   srvctl status gns Parameters

Parameter	Description
`-node node_name`	Specify a node on which GNS is running for which you want to display the state
`-verbose`	Displays verbose output.

srvctl status havip

Displays the status of all highly available VIPs (HAVIPs) (used for highly available NFS exports) in a cluster or one particular highly available VIP.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl status havip` command with the following syntax:

```
srvctl status havip [-id havip_name]
```

Table A-172   srvctl status havip Parameters

Parameter	Description
`-id havip_name`	Specify the unique name of the HAVIP resource you want to display. If you do not specify this parameter, then SRVCTL displays the status of all HAVIPs known to Oracle Clusterware.

Example

This command returns output similar to the following:

```
$ srvctl status havip

HAVIP ora.ha1.havip is enabled
HAVIP ora.ha1.havip is not running
```

srvctl status home

Displays the status of all the Oracle Restart-managed or Oracle Clusterware-managed resources for the specified Oracle home.

Syntax and Parameters

Use the `srvctl status home` command with the following syntax:

```
srvctl status home -oraclehome Oracle_home -statefile state_file -node node_name
```

Table A-173   srvctl status home Parameters

Parameter	Description
-oraclehome *Oracle_home*	The path to the Oracle home for which you want to start the Oracle Restart or Oracle Clusterware-managed resources
-statefile *state_file*	The path name for the text file that holds the state information generated by this command.
-node *node_name*	The name of the node where the Oracle home resides. **Note:** This parameter is required and can be used only with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl status home -oraclehome /u01/app/oracle/product/12.1/dbhome_1 -statefile
 ~/state.txt -node stvm12
```

The preceding command returns output similar to the following:

```
Database cdb1 is running on node stvm12
```

srvctl status instance

Displays the status of instances.

> **Note:**
>
> This command is only available with Oracle Clusterware and Oracle RAC.

Syntax and Parameters

Use the `srvctl status instance` command with the following syntax:

```
srvctl status instance -db db_unique_name {-node node_name
     | -instance "instance_name_list"} [-force] [-verbose]
```

Table A-174   srvctl status instance Parameters

Parameter	Description
-db *db_unique_name*	Unique name for the database
-node *node_name*	Node name **Note:** Use this parameter for policy-managed databases
-instance "*inst_name_list*"	Comma-delimited list of instance names **Note:** Use this parameter for administrator-managed databases
-force	Include disabled applications
-verbose	Displays verbose output.

Example

An example of this command is:

```
$ srvctl status instance -db crm -instance "crm1,crm2" -verbose
```

srvctl status listener

Displays the status of listener resources.

Syntax and Parameters

Use the srvctl status listener command with the following syntax:

```
srvctl status listener [-listener listener_name] [-node node_name] [-verbose]
```

Table A-175   srvctl status listener Parameters

Parameter	Description
-listener *listener_name*	Name of a listener. If you do not specify this parameter, then the listener name defaults to LISTENER
-node *node_name*	Name of a cluster node. **Note:** This parameter can be used only for Oracle Clusterware.
-verbose	Displays verbose output.

Example

To display the status of the default listener on the node node2, use the following command:

```
$ srvctl status listener -node node2
```

srvctl status mgmtdb

Displays the current state of the management database (CHM repository) resource.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl status mgmtdb` command with the following syntax:

```
srvctl status mgmtdb [-verbose]
```

You can optionally specify that verbose output should be displayed.

Examples

An example of this command to check the status of the management database is:

```
$ srvctl status mgmtdb
```

srvctl status mgmtlsnr

Displays the status of the management listener resource (for CHM).

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl status mgmtlsnr` command with the following syntax:

```
srvctl status mgmtlsnr [-node node_name] [-verbose]
```

Table A-176   srvctl status mgmtlsnr Parameters

Parameter	Description
-node node_name	Name of a cluster node. **Note:** This parameter can be used only for Oracle Clusterware.
-verbose	Displays verbose output.

Example

To display the status of the management listener on the node node2, use the following command:

```
$ srvctl status mgmtlsnr -node node2
```

srvctl status nodeapps

Displays the status of node applications.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl status nodeapps` command with the following syntax:

```
srvctl status nodeapps [-node node_name]
```

You can optionally specify the node for which to display the status of the node applications.

srvctl status oc4j

Determines which node is running the Oracle Database QoS Management server.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl status oc4j` command with the following syntax:

```
srvctl status oc4j [-node node_name] [-verbose]
```

Table A-177   srvctl status oc4j Parameters

Parameter	Description
-node node_name	Specify a node on which the OC4J instance (used by Oracle Database QoS Management server) is running
-verbose	Displays verbose output.

srvctl status ons

Displays the current state of the Oracle Notification Service daemon.

> **Note:**
>
> This command is only available with Oracle Restart.

Syntax and Parameters

Use the `srvctl status ons` command with the following syntax:

```
srvctl status ons
```

srvctl status scan

Displays the status for all SCAN VIPs, by default, or a specific SCAN VIP.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl status scan` command with the following syntax:

```
srvctl status scan [-scannumber ordinal_number] [-verbose]
```

Table A-178   srvctl status scan Parameters

Parameter	Description
`-scannumber ordinal_number`	Specify an ordinal number that identifies a specific SCAN VIP. The range of values you can specify for this parameter is 1 to 3. If you do not specify this parameter, then the utility displays the status of all SCAN VIPs in the cluster.
`-verbose`	Displays verbose output.

Example

An example of this command is:

```
$ srvctl status scan -scannumber 1
```

srvctl status scan_listener

Displays the status for all SCAN listeners, by default, or a specific listener.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl status scan_listener` command with the following syntax:

```
srvctl status scan_listener [-scannumber ordinal_number] [-verbose]
```

Table A-179   srvctl status scan_listener Parameters

Parameter	Description
`-scannumber ordinal_number`	Specify an ordinal number that identifies a specific SCAN VIP. The range of values you can specify for this parameter is 1 to 3. If you do not specify this parameter, then the utility displays the status of all SCAN VIPs in the cluster.
`-verbose`	Displays verbose output.

Example

An example of this command is:

```
$ srvctl status scan_listener -scannumber 1
```

srvctl status server

Displays the current state of named servers.

Syntax and Parameters

Use the `srvctl status server` command with the following syntax:

```
srvctl status server -server "server_name_list" [-detail]
```

Table A-180   srvctl status server Parameters

Parameter	Description
`-server "server_name_list"`	Comma-delimited list of server names.
`-detail`	Print detailed status information.

Example

The following example displays the status of a named server:

```
$ srvctl status server -server "server11" -detail
```

srvctl status service

Displays the status of a service.

For Oracle RAC One Node databases, if there is an online database relocation in process, then the `srvctl status service` command displays the source and destination nodes and the status of the relocation, whether it is active or failed.

Syntax and Parameters

Use the `srvctl status service` command with the following syntax:

```
srvctl status service -db db_unique_name [-service "service_name_list"]
     [-force] [-verbose]
```

Table A-181   srvctl status service Parameters

Parameter	Description
-db *db_unique_name*	Unique name for the database
-services "*service_name_list*"	Comma-delimited list of service names. If you do not specify this parameter, then the utility lists the status of all services for the specified database.
-force	Include disabled applications
-verbose	Displays verbose output.

Example

The following example displays the status of a named service across the clustered database:

```
$ srvctl status service -db crm -service "crm" -verbose
```

srvctl status srvpool

Displays server pool names, number of servers in server pools, and, optionally, the names of the servers in the server pools.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl status srvpool command with the following syntax:

```
srvctl status srvpool [-serverpool pool_name] [-detail]
```

Table A-182   srvctl status srvpool Parameters

Parameter	Description
-serverpool *pool_name*	Name of the server pool
-detail	Print detailed status information

Usage Notes

- If you include the -detail parameter but do not specify a specific server pool with the -serverpool parameter, then the output of this command includes the names of servers that are currently assigned to each server pool.

- If you use the `-serverpool` parameter to specify a server pool, then the output for this command includes the server pool name and number of servers in the server pool (and, optionally, the server names) for the specified server pool.

Example

An example of this command is:

```
$ srvctl status srvpool -serverpool srvpool1 -detail
```

srvctl status vip

Displays status for a specific VIP or a VIP on a specific node.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl status vip` command with the following syntax:

```
srvctl status vip {-node node_name | -vip vip_name} [-verbose]
```

Table A-183   srvctl status vip Parameters

Parameter	Description
`-node node_name`	Node name
`-vip vip_name`	The VIP name
`-verbose`	Displays verbose output.

Example

An example of this command is:

```
$ srvctl status vip -vip node1-vip
```

srvctl status volume

Displays the status of a specific volume or all volumes.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl status volume` command with the following syntax:

```
srvctl status volume [-device volume_device] [-volume volume_name]
   [-diskgroup disk_group_name] [-node "node_list"]
```

Table A-184   srvctl status volume Parameters

Parameter	Description
-device *volume_device*	Specify the path to the volume device for which you want to display the status.
-volume *volume_name*	Specify the name of the volume for which you want to view the status.
-diskgroup *disk_group_name*	Specify the name of the disk group in which the volume resides for which you want to display the status.
-node "*node_list*"	Specify a comma-delimited list of node names where volumes for which you want to view the status reside.

Usage Notes

- If you do not specify any of the optional parameters, then SRVCTL displays the status for all volumes.

- If you specify only the -volume parameter, then SRVCTL displays the status for the volume that you specify.

- If you specify only the -diskgroup parameter, then SRVCTL displays the status for the volumes that reside in the disk group that you specify.

- If you specify only the -device parameter, then SRVCTL displays the status for the volume device that you specify.

- If you specify the -diskgroup and -device parameters, then SRVCTL displays the status for the volume device in the disk group that you specify.

- If you specify the -node parameter, then SRVCTL displays the status of the volumes that reside on the nodes you list.

Examples

The srvctl status volume command displays information similar to the following:

```
$ srvctl status volume -volume vol1
Volume vol1 of diskgroup diskgrp1 for device volume_device_path1 is enabled
Volume vol1 of diskgroup diskgrp1 for device volume_device_path1 is running
```

In the preceding example, SRVCTL performs a status query on all nodes because the -node parameter is not specified.

```
$ srvctl status volume
Volume vol1 of diskgroup diskgrp for device volume_device_path1 is enabled
Volume vol1 of diskgroup diskgrp for device volume_device_path1 is running
Volume vol2 of diskgroup diskgrp for device volume_device_path2 is enabled
Volume vol2 of diskgroup diskgrp for device volume_device_path2 is running
```

In the preceding example, SRVCTL displays the status of all registered volumes because the no parameter is specified.

stop

Stops the Oracle Clusterware applications for the database, all or named instances, all or named service names, listeners, or node level application resources. Only the Oracle Clusterware applications that are enabled, starting or running are stopped. Objects running outside of Oracle Clusterware are not stopped.

You should disable an object that you intend to remain stopped after you issue a srvctl stop command. See the srvctl disable command.

Note:

- If the object is stopped and is not disabled, then it can restart as the result of another planned operation. The object does *not* restart because of a failure. Oracle recommends that you disable any object that should remain stopped after you issue a stop command.

- When you use the -force parameter to stop dependent applications and the object, then those dependent applications do not restart when the object is later restarted or after the node fails. You must restart the dependent applications, manually, except for services with AUTOMATIC management policy when a database is stopped.

Table A-185   srvctl stop Summary

Command	Description
srvctl stop asm	Stops Oracle ASM instances
srvctl stop cvu	Stops the Cluster Verification Utility resource
srvctl stop database	Stops the cluster database
srvctl stop diskgroup	Stops a specific disk group on a specified number of nodes
srvctl stop filesystem	Stops the Oracle ACFS volume resource
srvctl stop gns	Stops GNS
srvctl stop havip	Stops a specific highly available VIP (used for highly available NFS exports)
srvctl stop home	Stops the resources for the specified Oracle home
srvctl stop instance	Stops the instance
srvctl stop listener	Stops the specified listener or listeners
srvctl stop mgmtdb	Stops the management database
srvctl stop mgmtlsnr	Stops the management listener
srvctl stop nodeapps	Stops the node-level applications
srvctl stop oc4j	Stops the OC4J instance

Table A-185   (Cont.) srvctl stop Summary

Command	Description
srvctl stop ons	Stops Oracle Notification Service
srvctl stop scan	Stops all SCAN VIPs
srvctl stop scan_listener	Stops all SCAN listeners
srvctl stop service	Stops the service
srvctl stop vip	Stops VIP resources
srvctl stop volume	Stops a volume device

srvctl stop asm

Stops an Oracle ASM instance.

Syntax and Parameters

Use the `srvctl stop asm` command with the following syntax:

```
srvctl stop asm [-proxy] [-node node_name]
   [-stopoption stop_options] [-force]
```

> **Note:**
>
> - To manage Oracle ASM on Oracle Database 12*c* installations, use SRVCTL in the Oracle Grid Infrastructure home for a cluster (Grid home). If you have Oracle RAC or Oracle Database installed, then you cannot use SRVCTL in the database home to manage Oracle ASM.
>
> - You cannot use this command when OCR is stored in Oracle ASM because it will not stop Oracle ASM. To stop Oracle ASM you must shut down Oracle Clusterware.

Table A-186   srvctl stop asm Parameters

Parameter	Description
-proxy	Stop an Oracle ASM proxy instance.
-node node_name	The name of the node on which to stop the Oracle ASM instance.
	If you do not specify this parameter, then the utility stops the Oracle ASM instance on every active node in the cluster.
	Note: This parameter can be used only with Oracle Clusterware.

Table A-186   (Cont.) srvctl stop asm Parameters

Parameter	Description
-stopoption *stop_options*	Options for shutdown command, such as NORMAL, TRANSACTIONAL, IMMEDIATE, or ABORT
	See Also: *SQL*Plus User's Guide and Reference* for more information about shutdown options
-force	Use this parameter to stop disk groups, file systems and databases that depend on Oracle ASM

Example

An example of this command is:

```
$ srvctl stop asm -node crmnode1 -stopoption IMMEDIATE
```

srvctl stop cvu

Stops the Cluster Verification Utility (CVU) that is either in the running or starting state.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl stop cvu command with the following syntax:

```
srvctl stop cvu [-force]
```

The -force parameter is the only parameter you can use with this command. The -force parameter forces CVU to stop.

Examples

An example of this command to stop CVU:

```
$ srvctl stop cvu
```

srvctl stop database

Stops a database, its instances, and its services. When the database later restarts, services with AUTOMATIC management start automatically but services with MANUAL management policy must be started manually.

Syntax and Parameters

Use the srvctl stop database command with the following syntax:

```
srvctl stop database -db db_unique_name [-stopoption stop_options]
  [[-stopconcurrency number_of_instances] [-eval] [-force] [-verbose]
```

Table A-187   srvctl stop database Parameters

Parameter	Description
-db *db_unique_name*	Unique name for the database.
-stopoption *stop_options*	Use this parameter to specify shutdown command options, such as NORMAL, TRANSACTIONAL, IMMEDIATE, or ABORT **See Also:** *SQL*Plus User's Guide and Reference* for more information about shutdown options
[-stopconcurrency *number_of_instances*]	Specify a number of database instances to stop simultaneously, or specify 0 for an empty stop concurrency value.
-eval	Use this parameter to hypothetically evaluate the impact of the command on the system.
-force	This parameter stops the database, its instances, its services, and any resources that depend on those services
-verbose	Display verbose output

Example

An example of this command is:

```
$ srvctl stop database -db crm -stopoption NORMAL
```

srvctl stop diskgroup

Use this command to stop a specific disk group resource on a number of specified nodes.

Syntax and Parameters

Use the srvctl stop diskgroup command with the following syntax:

```
srvctl stop diskgroup -diskgroup diskgroup_name [-node "node_list"] [-force]
```

Table A-188   srvctl stop diskgroup Parameters

Parameter	Description
-diskgroup *diskgroup_name*	The Oracle ASM disk group name
-node "*node_list*"	Comma-delimited list of node names on which to stop the disk group resource **Note:** This parameter can be used only with Oracle Clusterware.

Table A-188   (Cont.) srvctl stop diskgroup Parameters

Parameter	Description
-force	This parameter does not stop the databases that depend on the disk group you are stopping, but instead performs a forceful dismount that may cause those databases to fail.

Example

An example of this command is:

```
$ srvctl stop diskgroup -diskgroup diskgroup1 -node "mynode1,mynode2" -force
```

srvctl stop filesystem

Use this command to stop (unmount) the Oracle ACFS file system or generic file system.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl stop filesystem command with the following syntax:

```
srvctl stop filesystem -device volume_device [-node node_name] [-force]
```

Table A-189   srvctl stop filesystem Parameters

Parameter	Description
-device volume_device	Specify the path to the file system volume device to stop.
-node node_name	The name of a node. If you do not specify this parameter, then the utility stops the volume resource on all active nodes in the cluster.
-force	Stops the file system and also stops any databases or other resources that depend on this file system.

Usage Notes

- You must run this command as root user on Linux and UNIX platforms, or as an Administrator user on Windows platforms, or as a configured file system user.

> **See Also:**
>
> srvctl add filesystem for more information about configuring file system users

Example

An example of this command is:

```
$ srvctl stop filesystem -device /dev/asm/racvol_1 -force
```

srvctl stop gns

Use this command to stop GNS in the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl stop gns` command with the following syntax:

```
srvctl stop gns [-node node_name] [-verbose] [-force]
```

Table A-190   srvctl stop gns Parameters

Parameter	Description
-node *node_name*	The name of a node on which GNS is running
-verbose	Verbose output
-force	Force stop

Example

An example of this command to stop GNS on the local node is:

```
$ srvctl stop gns
```

srvctl stop havip

Stops the highly available VIPs (HAVIPs) (used for highly available NFS exports) on a specific node.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl stop havip` command with the following syntax:

```
srvctl stop havip -id havip_name [-node node_name] [-force]
```

Table A-191 srvctl stop havip Parameters

Parameter	Description
-id *havip_name*	Specify the unique name of the HAVIP resource to stop.
-node *node_name*	Optionally, you can specify the name of the node on which the HAVIP resource to stop resides.
-force	Force stop

Usage Notes

- You must run this command as root user on Linux and UNIX platforms.

Example

An example of this command is:

```
# srvctl stop havip -id myhavip -node myNode1
```

srvctl stop home

Stops all the Oracle Restart-managed or Oracle Clusterware-managed resources that run from the specified Oracle home.

Syntax and Parameters

Use the srvctl stop home command with the following syntax:

```
srvctl stop home -oraclehome Oracle_home -statefile state_file -node node_name
      [-stopoption stop_options] [-force]
```

Table A-192 srvctl stop home Parameters

Parameter	Description
-oraclehome *Oracle_home*	The path to the Oracle home for which you want to start the Oracle Restart or Oracle Clusterware-managed resources. **Note:** The path to the Oracle home you specify must be the same version as the Oracle home from which you invoke SRVCTL.
-statefile *state_file*	The path name where you want the state file to be written.
-node *node_name*	The name of the node where the Oracle home resides. **Note:** This parameter can be used only with Oracle Clusterware.
-stopoption *stop_options*	Shutdown options for the database, such as NORMAL, TRANSACTIONAL, IMMEDIATE, or ABORT **See Also:** *SQL*Plus User's Guide and Reference* for more information about shutdown options

Table A-192   (Cont.) srvctl stop home Parameters

Parameter	Description
-force	Stop the resources even if errors are reported.

Example

An example of this command is:

```
$ srvctl stop home -oraclehome /u01/app/oracle/product/12.1.0/db_1 -statefile
 ~/state.txt
```

srvctl stop instance

Stops instances and stops any services running on specified instances, unless you specify the -force parameter. If you specify -force, then the services fail over to an available instance when the instance stops.

> **Note:**
>
> - This command is only available with Oracle Clusterware and Oracle RAC.
>
> - If you run this command on an Oracle RAC One Node database, then the command returns an error instructing you to use the database noun, instead.

Syntax and Parameters

Use the srvctl stop instance command with the following syntax:

```
srvctl stop instance -db db_unique_name {-node node_name
    | -instance "instance_name_list"} [-stopoption stop_options] [-force]
```

You must enclose the list of comma-delimited instance names in double quotation marks (" ").

Table A-193   srvctl stop instance Parameters

Parameter	Description
-db db_unique_name	Unique name for the database
-node node_name	The name of a single node **Note:** Use this parameter for policy-managed databases.
-instance "inst_name_list"	Comma-delimited list of instance names enclosed in double quotation marks **Note:** Use this parameter for administrator-managed databases.

Table A-193   (Cont.) srvctl stop instance Parameters

Parameter	Description
-stopoption *stop_options*	Options for shutdown command, such as NORMAL, TRANSACTIONAL LOCAL, IMMEDIATE, or ABORT
	See Also: *SQL*Plus User's Guide and Reference* for more information about shutdown options
-force	This parameter fails the running services over to another instance.

Example

An example of stopping an instance in a policy-managed database is:

```
$ srvctl stop instance -db crm -node node1
```

An example of stopping an instance in an administrator-managed database is:

```
$ srvctl stop instance -db crm -instance "crm1"
```

srvctl stop listener

Stops the default listener or a specific listener on all nodes or the specified node.

This command can also be used to stop a listener on a noncluster database from the noncluster database home. However, SRVCTL does not accept the -node parameter when run from a noncluster database home.

Syntax and Parameters

Use the srvctl stop listener command with the following syntax:

```
srvctl stop listener [-listener listener_name] [-node node_name] [-force]
```

Table A-194   srvctl stop listener Parameters

Parameter	Description
-listener *listener_name*	The name of the listener to stop.
	If you do not specify this parameter, then the listener name defaults to LISTENER for a database listener; LISTENER_ASM for an Oracle ASM listener; or LISTENER_LEAF for a Leaf Node listener.
-node *node_name*	The name of a single node on which a particular listener runs.
	Note: This parameter can be used only with Oracle Clusterware.
-force	Force stop

Example

The following command stops the default listener LISTENER on the node mynode1:

```
$ srvctl stop listener -node mynode1
```

srvctl stop mgmtdb

Stops the management database (CHM repository) resource.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl stop mgmtdb` command with the following syntax:

```
srvctl stop mgmtdb [-stopoption stop_options] [-force]
```

Table A-195   srvctl stop mgmtdb Parameters

Parameter	Description
-stopoption *stop_options*	Use this parameter to specify `shutdown` command options, such as NORMAL, TRANSACTIONAL, IMMEDIATE, or ABORT **See Also:** *SQL*Plus User's Guide and Reference* for more information about shutdown options
-force	Stops the database and any associated services and any dependent resources

Example

An example of this command is:

```
$ srvctl stop mgmtdb -stopoption NORMAL
```

srvctl stop mgmtlsnr

Stops the management listener resource (for CHM) on all nodes or the specified node.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl stop mgmtlsnr` command with the following syntax:

```
srvctl stop mgmtlsnr [-node node_name] [-force]
```

Table A-196   srvctl stop mgmtlsnr Parameters

Parameter	Description
-node *node_name*	The name of a single node on which the management listener runs.

Table A-196   (Cont.) srvctl stop mgmtlsnr Parameters

Parameter	Description
-force	Force stop

Example

The following command stops the management listener on the node `mynode1`:

```
$ srvctl stop mgmtlsnr -node mynode1
```

srvctl stop nodeapps

Stops node-level applications on a node in the cluster.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl stop nodeapps` command with the following syntax:

```
srvctl stop nodeapps [-node node_name] [-gsdonly] [-adminhelper] [-force]
   [-relocate] [-verbose]
```

Table A-197   srvctl stop nodeapps Parameters

Parameter	Description
-node node_name	Node name If you do not specify this parameter, then the utility stops the node applications (nodeapps) on all active nodes in the cluster.
-gsdonly	Stops only the GSD instead of all nodeapps
-adminhelper	Stops only the Administrator helper instead of all nodeapps
-relocate	Relocate VIP and possibly dependent services **Note:** If you specify this parameter, then you must also specify the -node node_name parameter.
-force	Force stop
-verbose	Display verbose output

Example

An example of this command is:

```
$ srvctl stop nodeapps
```

srvctl stop oc4j

Stops the OC4J instance that is in the running or starting state.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl stop oc4j` command with the following syntax:

```
srvctl stop oc4j [-force] [-verbose]
```

Table A-198   srvctl stop oc4j Parameters

Parameter	Description
-force	Force stop of the SCAN Listener.
-verbose	Display verbose output

Example

An example of this command is:

```
$ srvctl stop oc4j -force -verbose
```

srvctl stop ons

Stops the Oracle Notification Service daemon.

Note:

This command is only available with Oracle Restart.

Syntax and Parameters

Use the `srvctl stop ons` command with the following syntax:

```
srvctl stop ons [-verbose]
```

The only parameter for this command is the `-verbose` parameter, which specifies that verbose output should be displayed.

Example

An example of this command is:

```
$ srvctl stop ons -verbose
```

srvctl stop scan

Stops all SCAN VIPs, by default, that are running or in starting state, or stops a specific SCAN VIP identified by *ordinal_number*.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl stop scan command with the following syntax:

```
srvctl stop scan [-scannumber ordinal_number] [-force]
```

Table A-199   srvctl stop scan Parameters

Parameter	Description
-scannumber *ordinal_number*	An ordinal number that identifies which SCAN VIP you want to stop. The range of values you can specify for this parameter is 1 to 3.
	If you do not specify this parameter, then the utility stops all the SCAN VIPs.
-force	Force stop of the SCAN VIP.

Example

An example of this command is:

```
$ srvctl stop scan -scannumber 1
```

srvctl stop scan_listener

Stops all SCAN listeners, by default, that are in a running or starting state, or a specific listener identified by *ordinal_number*.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl start scan_listener command with the following syntax:

```
srvctl stop scan_listener [-scannumber ordinal_number] [-force]
```

Table A-200   srvctl stop scan_listener Parameters

Parameter	Description
-scannumber *ordinal_number*	An ordinal number that identifies which SCAN listener to stop. You can specify values of 1, 2, or 3 for this parameter.
	If you do not specify this parameter, then the utility stops all the SCAN listeners.
-force	Force stop of the SCAN listener.

Example

An example of this command is:

```
$ srvctl stop scan_listener -scannumber 1
```

srvctl stop service

Stops one or more services globally across the cluster database, or on the specified instance.

Syntax and Parameters

Use the srvctl stop service command with the following syntax:

```
srvctl stop service -db db_unique_name [-service "services_list"
  [-node node_name | -instance instance_name | -serverpool pool_name |
   -global_override]] [-eval] [-force]
```

You must enclose the list of comma-delimited service names in double quotation marks (" ").

Table A-201   srvctl stop service Parameters

Parameter	Description
-db db_unique_name	Unique name for the database
-service "service_name_list"	Comma-delimited list of service names enclosed in double quotation marks (" ").
	If you do not provide a service name list, then SRVCTL stops all services on the database.
-node node_name	The name of the node on which the services should be stopped. Use this parameter for policy-managed databases.
	Note: This parameter can be used only with Oracle Clusterware.
-instance instance_name	The name of the instance for which the services should be stopped. Use this parameter for administrator-managed databases.
	Note: This parameter can be used only with Oracle Clusterware.
-serverpool pool_name	The name of the server pool that contains the service
-global_override	Override value to operate on a global service. This parameter is ignored if the service is not a global service.
-eval	Use this parameter to hypothetically evaluate the impact of the command on the system.

Table A-201   (Cont.) srvctl stop service Parameters

Parameter	Description
-force	Force SRVCTL to stop the service; this causes SRVCTL to disconnect all of the sessions transactionally, requiring the sessions using the service to reconnect and then connect to another instance.
	Note: If you do not specify the -force parameter, then sessions already connected to this service stay connected, but new sessions cannot be established to the service.

Examples

The following example stops a service for all cluster database instances:

```
$ srvctl stop service -db crm -service "crm"
```

The following example stops a service on a specified instance:

```
$ srvctl stop service -db crm -service "crm" -instance crm2
```

srvctl stop vip

Stops a specific VIP or all VIPs on a specific node, including any VIPs that were relocated due to a failover.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl stop vip command with the following syntax:

```
srvctl stop vip {-node node_name | -vip vip_name} [-force] [-relocate] [-verbose]
```

Table A-202   srvctl stop vip Parameters

Parameter	Description
-node node_name	This parameter stops all VIPs on a specific node, including failed-over VIPs
-vip vip_name	This parameter stops a specific VIP
-force	Force SRVCTL to stop the VIP
-relocate	Relocate VIP **Note:** You must use the -nodenode_name parameter with the -relocate parameter.
-verbose	Verbose output

Example

To stop all the VIPs on `mynode1`, including any failed-over VIPs, use the following command:

```
$ srvctl stop vip -node mynode1 -verbose
```

srvctl stop volume

Stops a specific, running volume.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl stop volume` command with the following syntax:

```
srvctl stop volume {-volume volume_name -diskgroup disk_group_name |
    -device volume_device} [-node "node_list"]
```

Table A-203   srvctl stop volume Parameters

Parameter	Description
`-volume volume_name`	Specify the name of the volume to stop.
`-diskgroup disk_grp_name`	Specify the name of the disk group in which the volume to stop resides.
`-device volume_device`	Specify the path to the volume device to stop.
`-node "node_list"`	Specify a comma-delimited list of node names where volumes to stop reside.

Usage Notes

The `srvctl stop volume` command attempts to stop (disable) the volume but it does not disable the resource or remove the volume from Oracle ASM.

Example

The following example stops a volume named `VOLUME1` that resides in a disk group named `DATA`:

```
$ srvctl stop volume -volume VOLUME1 -diskgroup DATA
```

unsetenv

The `unsetenv` command unsets values for the environment in the configuration file. It allows users to administer environment configuration for the objects.

Table A-204   srvctl unsetenv Command Summary

Command	Description
srvctl unsetenv asm	Unsets the value for one or more Oracle ASM environment variables
srvctl unsetenv database	Unsets the value for one or more cluster database environment variables
srvctl unsetenv listener	Unsets the value for one or more listener environment variables.
srvctl unsetenv mgmtdb	Unsets the value for the management database.
srvctl unsetenv mgmtlsnr	Unsets the value for the management listener.
srvctl unsetenv nodeapps	Unsets the value for one or more node application environment variables
srvctl unsetenv vip	Unsets the value for one or more VIP environment variables

srvctl unsetenv asm

Unsets the Oracle ASM environment configurations.

Syntax and Parameters

Use the `srvctl unsetenv asm` command as follows:

```
srvctl unsetenv asm -envs "name_list"
```

Use the `-envs` parameter to specify a comma-delimited list of the names of environment variables enclosed in double quotation marks (`" "`).

Example

The following example unsets the environment configuration for an Oracle ASM environment variable:

```
$ srvctl unsetenv asm -envs "CLASSPATH"
```

srvctl unsetenv database

Unsets the cluster database environment configurations.

Syntax and Parameters

Use the `srvctl unsetenv database` command as follows:

```
srvctl unsetenv database -db db_unique_name -envs "name_list"
```

Table A-205   srvctl unsetenv database Parameters

Parameter	Description
-db db_unique_name	Unique name for the database

Table A-205   (Cont.) srvctl unsetenv database Parameters

Parameter	Description
`-envs "name_list"`	Comma-delimited list of environment variable names enclosed in double quotation marks (`" "`)

Example

The following example unsets the environment configuration for a cluster database environment variable:

```
$ srvctl unsetenv database -db crm -envs "CLASSPATH,LANG"
```

srvctl unsetenv listener

Unsets the environment configuration for a listener.

Syntax and Parameters

Use the `srvctl unsetenv listener` command as follows:

```
srvctl unsetenv listener [-listener listener_name] -envs "name_list"
```

Table A-206   srvctl unsetenv listener Parameters

Parameter	Description
`-listener listener_name`	Name of the listener. If you do not specify this parameter, then the name of the listener defaults to `LISTENER`.
`-envs "name_list"`	A comma-delimited list of names of environment variables

Example

The following example unsets the environment variable `TNS_ADMIN` for the default listener:

```
$ srvctl unsetenv listener -envs "TNS_ADMIN"
```

srvctl unsetenv mgmtdb

Unsets the management database (CHM repository) environment configurations.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl unsetenv mgmtdb` command as follows:

```
srvctl unsetenv mgmtdb -envs "name_list"
```

Use the -envs parameter to specify a comma-delimited list of the names of environment variables enclosed in double quotation marks (" ").

Example

The following example unsets the environment configuration for a management database environment variable:

```
$ srvctl unsetenv mgmtdb -envs "LANG"
```

srvctl unsetenv mgmtlsnr

Unsets the management listener resource (for CHM) environment configurations.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl unsetenv mgmtlsnr command as follows:

```
srvctl unsetenv mgmtlsnr -envs "name_list"
```

Use the -envs parameter to specify a comma-delimited list of the names of environment variables enclosed in double quotation marks (" ").

Example

The following example unsets the environment configuration for a management listener environment variable:

```
$ srvctl unsetenv mgmtlsnr -envs "LANG"
```

srvctl unsetenv nodeapps

Unsets the environment configuration for the node applications.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the srvctl unsetenv nodeapps command as follows:

```
srvctl unsetenv nodeapps -envs "name_list" [-viponly] [-gsdonly]
   [-onsonly] [-verbose]
```

Table A-207   srvctl unsetenv nodeapps Parameters

Parameter	Description
-envs "name_list"	Comma-delimited list of the names of environment variables enclosed in double quotation marks (" ")

Table A-207   (Cont.) srvctl unsetenv nodeapps Parameters

Parameter	Description
-viponly	Modify only the VIP configuration instead of all nodeapps
-gsdonly	Modify only the GSD configuration instead of all nodeapps
-onsonly	Modify only the ONS daemon configuration instead of all nodeapps
-verbose	Display verbose output

Example

The following example unsets the environment configuration for all node applications:

```
$ srvctl unsetenv nodeapps -envs "test_var1,test_var2"
```

srvctl unsetenv vip

Unsets the environment configuration for the specified cluster VIP.

Syntax and Parameters

Use the `srvctl unsetenv vip` command as follows:

```
srvctl unsetenv vip -vip "vip_name_list" -envs "name_list" [-verbose]
```

Table A-208   srvctl unsetenv vip Parameters

Parameter	Description
-vip "vip_name_list"	Comma-delimited list of VIP names enclosed in double quotation marks (" ")
-envs "name_list"	Comma-delimited list of environment variable names enclosed in double quotation marks (" ")
-verbose	Display verbose output

Example

The following example unsets the CLASSPATH environment variable for a cluster VIP:

```
$ srvctl unsetenv vip -vip "crm2-vip" -envs "CLASSPATH"
```

update

The `srvctl update` command requests that the specified running object use the new configuration information stored in the OCR.

Table A-209   srvctl update Command Summary

Command	Description
srvctl update database	Updates the specified database to use the new listener endpoints
srvctl update gns	Updates the configuration for GNS
srvctl update listener	Updates the listener to listen on the new endpoints
srvctl update scan_listener	Update the SCAN listeners to listen on the new endpoints.

srvctl update database

Updates the specified database to use the new listener endpoints.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl update database` command with the following syntax:

```
srvctl update database -db db_unique_name
```

Use the `-db` parameter to specify the unique name of the database to update.

srvctl update gns

Use the `srvctl update gns` command to modify a Grid Naming Service (GNS) instance.

Note:

This command is only available with Oracle Clusterware.

Syntax and Parameters

Use the `srvctl update gns` command with one of these syntax models:

```
srvctl update gns -advertise name -address ip_address [-timetolive time_to_live]
  [-verbose]

srvctl update gns -delete name [-address address] [-verbose]

srvctl update gns -alias alias -name name [-timetolive time_to_live] [-verbose]

srvctl update gns -deletealias alias [-verbose]

srvctl update gns -createsrv service -target target -protocol protocol
  [-weight weight] [-priority priority] [-port port_number]
  [-timetolive time_to_live] [-instance instance_name] [-verbose]
```

```
srvctl update gns -deletesrv service_name -target target -protocol protocol
   [-verbose]

srvctl update gns -createtxt name -target target [-timetolive time_to_live]
   [-namettl name_ttl] [-verbose]

srvctl update gns -deletetxt name -target target [-verbose]

srvctl update gns -createptr name -target target [-timetolive time_to_live]
   [-namettl name_ttl] [-verbose]

srvctl update gns -deleteptr name -target target [-verbose]
```

Table A-210   srvctl update gns Parameters

Parameter	Description
-advertise *name*	Advertise a name through GNS
-address *ip_address*	The IP address for an advertised name
-timetolive *time_to_live*	Optionally, you can specify, in seconds, an amount of time for the record to live
-delete *name*	Remove the advertisement of a name from GNS
-alias *alias*	Create an alias for an advertised name
-name *name*	The advertised name associated with the alias
-deletalias *alias*	Delete an alias
-createsrv *service*	Create a service that is described by the record
-target *target*	Specify a name that is associated with the service
-protocol *protocol*	Specify which protocol is used to connect to the service
-weight *weight*	Specify the weight of the record
-priority *priority*	Specify the priority of the record, with a value ranging from 0 to 65535
-port *port_number*	Specify the port number used to connect to the service, with a value ranging from 0 to 65535

Table A-210   (Cont.) srvctl update gns Parameters

Parameter	Description
`-instance instance_name`	Instance name of service
`-deletesrv service`	Name of the service record to be deleted
`-createtxt name`	Add a text (TXT) record using the specified name
`-namettl name_ttl`	Time to live for the name, specified in seconds
`-deletetxt name`	Delete a text (TXT) record of the specified name
`-createptr name`	Add a pointer (PTR) record for the specified name
`-deleteptr name`	Delete a pointer (PTR) record for the specified name

Usage Notes

You must be logged in as the `root` user on Linux and UNIX platforms, or as an Administrator user on Windows platforms, to run this command.

Example

The following command advertises a name with GNS:

```
# srvctl update gns -advertise myname -address 192.168.1.45
```

srvctl update listener

Updates the listener to listen on the new endpoints.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Use the `srvctl update listener` command with the following syntax:

```
srvctl update listener
```

This command does not accept any additional parameters, except for `-help`.

srvctl update scan_listener

Updates the SCAN listeners to listen on the new endpoints.

> **Note:**
>
> This command is only available with Oracle Clusterware.

Use the `srvctl update scan_listener` command with the following syntax:

```
srvctl update scan_listener
```

This command does not accept any additional parameters, except for `-help`.

srvctl upgrade database

The `srvctl upgrade database` command upgrades the configuration of a database and all of its services to the version of the database home from where this command is run.

Syntax and Parameters

Use the `srvctl upgrade database` command as follows:

```
srvctl upgrade database -db db_unique_name -oraclehome Oracle_home
```

Table A-211   srvctl upgrade database Parameters

Parameter	Description
`-db db_unique_name`	Unique name for the database
`-oraclehome Oracle_home`	The path to the `ORACLE_HOME`

B

Troubleshooting Oracle RAC

This appendix explains how diagnose problems for Oracle Real Application Clusters (Oracle RAC) components using trace and log files. This section includes the following topics:

- Where to Find Files for Analyzing Errors

- Managing Diagnostic Data in Oracle RAC

- Using Instance-Specific Alert Files in Oracle RAC

- Enabling Tracing for Java-Based Tools and Utilities in Oracle RAC

- Resolving Pending Shutdown Issues

- How to Determine If Oracle RAC Instances Are Using the Private Network

- Database Fails to Start after Private NIC Failure

> **Note:**
>
> Trace and log files, similar to those generated for Oracle Database with Oracle RAC, are also available for the Oracle Clusterware components. For Oracle Clusterware, Oracle Database stores these under a unified directory log structure.
>
> See the *Oracle Clusterware Administration and Deployment Guide* for more information about troubleshooting Oracle Clusterware.

Where to Find Files for Analyzing Errors

Oracle Database records information about important events that occur in your Oracle RAC environment in trace files. The trace files for Oracle RAC are the same as those in noncluster Oracle databases. As a best practice, monitor and back up trace files regularly for all instances to preserve their content for future troubleshooting.

Information about `ORA-600` errors appear in the `alert_SID.log` file for each instance where *SID* is the instance identifier.

The alert log and all trace files for background and server processes are written to the Automatic Diagnostic Repository, the location of which you can specify with the `DIAGNOSTIC_DEST` initialization parameter. For example:

```
diagnostic_dest=/oracle/11.1/diag/rdbms/rac/RAC2/trace
```

Oracle Database creates a different trace file for each background thread. Oracle RAC background threads use trace files to record database operations and database errors. These trace logs help troubleshoot and also enable Oracle Support to more efficiently

debug **cluster database** configuration problems. The names of trace files are operating system specific, but each file usually includes the name of the process writing the file (such as LGWR and RECO). For Linux, UNIX, and Windows systems, trace files for the background processes are named `SID_process name_process identifier.trc`.

> **See Also:**
>
> *Oracle Database Administrator's Guide* and *Oracle Database 2 Day + Real Application Clusters Guide* for more information about monitoring errors and alerts in trace files

Trace files are also created for user processes if you set the `DIAGNOSTIC_DEST` initialization parameter. User process trace file names have the format `SID_ora_process_identifier/thread_identifier.trc`, where `process_identifier` is a 5-digit number indicating the process identifier (PID) on Linux and UNIX systems, and `thread_identifier` is the thread identifier on Windows systems.

Managing Diagnostic Data in Oracle RAC

Problems that span Oracle RAC instances can be the most difficult types of problems to diagnose. For example, you may need to correlate the trace files from across multiple instances, and merge the trace files. Oracle Database 11*g* includes an advanced fault diagnosability infrastructure for collecting and managing diagnostic data, and uses the Automatic Diagnostic Repository (ADR) file-based repository for storing the database diagnostic data. When you create the ADR base on a shared disk, you can place ADR homes for all instances of the same Oracle RAC database under the same ADR Base. With shared storage:

- You can use the ADRCI command-line tool to correlate diagnostics across all instances.

  ADRCI is a command-line tool that enables you to view diagnostic data in the ADR and package incident and problem information into a zip file for transmission to Oracle Support. The diagnostic data includes incident and problem descriptions, trace files, dumps, health monitor reports, alert log entries, and so on.

  > **See Also:**
  >
  > *Oracle Database Utilities* for information about using ADRCI

- You can use the Data Recovery Advisor to help diagnose and repair corrupted data blocks, corrupted or missing files, and other data failures.

  The Data Recovery Advisor is an Oracle Database infrastructure that automatically diagnoses persistent data failures, presents repair options, and repairs problems at your request.

Using Instance-Specific Alert Files in Oracle RAC

Each instance in an Oracle RAC database has one alert file. The alert file for each instance, `alert.SID.log`, contains important information about error messages and exceptions that occur during database operations. Information is appended to the alert file each time you start the instance. All process threads can write to the alert file for the instance.

The `alert_SID.log` file is in the directory specified by the `DIAGNOSTIC_DEST` initialization parameter.

Enabling Tracing for Java-Based Tools and Utilities in Oracle RAC

All Java-based tools and utilities that are available in Oracle RAC are called by executing scripts of the same name as the tool or utility. This includes the Cluster Verification Utility (CVU), Database Configuration Assistant (DBCA), the Net Configuration Assistant (NETCA), Server Control Utility (SRVCTL), and the **Global Services Daemon (GSD)**. For example to run DBCA, enter the command `dbca`.

By default, Oracle Database enables traces for DBCA and the Database Upgrade Assistant (DBUA). For the CVU, GSDCTL, and SRVCTL, you can set the `SRVM_TRACE` environment variable to `TRUE` to make Oracle Database generate traces. Oracle Database writes traces to log files. For example, Oracle Database writes traces to log files in *Oracle home*/cfgtoollogs/dbca and *Oracle home*/cfgtoollogs/ dbua for DBCA and DBUA, respectively.

Resolving Pending Shutdown Issues

In some situations a `SHUTDOWN IMMEDIATE` may be pending and Oracle Database will not quickly respond to repeated shutdown requests. This is because Oracle Clusterware may be processing a current shutdown request. In such cases, issue a `SHUTDOWN ABORT` using SQL*Plus for subsequent shutdown requests.

How to Determine If Oracle RAC Instances Are Using the Private Network

This section describes how to manually determine if Oracle RAC instances are using the private network. However, the best practice for this task is to use the Oracle Enterprise Manager Cloud Control graphical user interface (GUI) to check the interconnect. Also, see the *Oracle Database 2 Day + Real Application Clusters Guide* for more information about monitoring Oracle RAC using Oracle Enterprise Manager.

With most network protocols, you can issue the `oradebug ipc` command to see the interconnects that the database is using. For example:

```
oradebug setmypid
oradebug ipc
```

These commands dump a trace file to the location specified by the `DIAGNOSTIC_DEST` initialization parameter. The output may look similar to the following:

```
SSKGXPT 0x1a2932c flags SSKGXPT_READPENDING    info for network 0
        socket no 10    IP 172.16.193.1         UDP 43749
        sflags SSKGXPT_WRITESSKGXPT_UP  info for network 1
        socket no 0     IP 0.0.0.0      UDP 0...
```

In the example, you can see the database is using IP 172.16.193.1 with a User Datagram Protocol (UDP) protocol. Also, you can issue the `oradebug tracefile_name` command to print the trace location where the output is written.

Additionally, you can query the `V$CLUSTER_INTERCONNECTS` view to see information about the private interconnect. For example:

```
SQL> SELECT * FROM V$CLUSTER_INTERCONNECTS;

NAME   IP_ADDRESS                 IS_   SOURCE
-----  -------------------------  ---   -----------------------------
eth0   138.2.236.114              NO    Oracle Cluster Repository
```

Database Fails to Start after Private NIC Failure

In a two-node cluster that is not using grid interprocess communication (GIPC) and has only a single private network interface card (NIC), when the NIC fails on the node on which the Oracle Grid Infrastructure Management Repository database (mgmtdb) is running (node A, for example) after the other node (node B) was evicted, then when the private network is reestablished on node A, node B joins the cluster but the database instance on node A is unable to start.

To start the database instance:

1. Stop the Oracle Clusterware stack on both nodes, as follows:

   ```
   crsctl stop cluster
   ```

2. Start the Oracle Clusterware stack on both nodes, as follows:

   ```
   crsctl start cluster
   ```

3. Stop the database instance on node A, on which the NIC failed, as follows:

   ```
   srvctl stop instance -db db_unique_name -node A
   ```

4. Start the database on both nodes by running the following commands one of the nodes:

   ```
   srvctl start instance -db db_unique_name -node B
   srvctl start instance -db db_unique_name -node A
   ```

   The database instance will fail to start on node A until after the instance starts on node B.

Glossary

Automatic Workload Repository (AWR)

A built-in repository that exists in every Oracle database. At regular intervals, Oracle Database makes a snapshot of all of its vital statistics and workload information and stores them in the AWR.

administrator-managed database

A database that you specifically define on which servers it can run, and where services can run within the database.

cache coherency

The synchronization of data in multiple caches so that reading a memory location through any cache will return the most recent data written to that location through any other cache. Sometimes called cache consistency.

Cache Fusion

A diskless cache coherency mechanism in Oracle RAC that provides copies of blocks directly from a holding instance's memory cache to a requesting instance's memory cache.

cardinality

The number of database instances you want running during normal operations.

CDB

A multitenant container database (CDB) is an Oracle database that includes zero, one, or many user-created pluggable databases (**PDB**s). Every Oracle database is either a CDB or a non-CDB.

client cluster

A cluster that advertises its names with the **server cluster**.

cluster

Multiple interconnected computers or servers that appear as if they are one server to end users and applications.

cluster configuration policy

A document that contains exactly one definition for each server pool defined in the system.

cluster configuration policy set

A document that defines the names of all server pools configured in the cluster and contains one or more configuration policies.

cluster database

The generic term for a Oracle RAC database.

cluster file system

A distributed file system that is a cluster of servers that collaborate to provide high performance service to their clients. Cluster file system software deals with distributing requests to storage cluster components.

Cluster Ready Services Daemon (CRSD)

The primary Oracle Clusterware process that performs high availability recovery and management operations, such as maintaining OCR. Also manages application resources and runs as `root` user (or by a user in the `admin` group on Mac OS X-based systems) and restarts automatically upon failure.

Cluster Synchronization Services (CSS)

An Oracle Clusterware component that discovers and tracks the membership state of each node by providing a common view of membership across the cluster. CSS also monitors process health, specifically the health of the database instance. The Global Enqueue Service Monitor (LMON), a background process that monitors the health of the cluster database environment and registers and de-registers from CSS. See also, OCSSD.

Cluster Time Synchronization Service

A time synchronization mechanism that ensures that all internal clocks of all nodes in a cluster are synchronized.

Cluster Verification Utility (CVU)

A tool that verifies a wide range of Oracle RAC components such as shared storage devices, networking configurations, system requirements, Oracle Clusterware, groups, and users.

commit outcome

A message sent to the client from the Oracle database after a transaction has been committed. These messages are not durable.

database pool

A set of databases within a database cloud that provide a unique set of global services and belong to a certain administrative domain. Partitioning of cloud databases into multiple pools simplifies service management and provides higher security because each pool can be administered by a different administrator.

Distributed Transaction Processing (DTP)

The paradigm of distributed transactions, including both XA-type externally coordinated transactions, and distributed-SQL-type (database links in Oracle) internally coordinated transactions.

dynamic network

A network that uses DHCP for IPv4 or stateless autoconfiguration (autoconfig) for IPv6.

Event Manager (EVM)

The background process that publishes Oracle Clusterware events. EVM scans the designated callout directory and runs all scripts in that directory when an event occurs.

Event Manager Daemon (EVMD)

A Linux or UNIX event manager daemon that starts the `racgevt` process to manage callouts.

extended distance cluster

A cluster where the nodes in the cluster are separated by greater distances from two buildings across the street, to across a campus or across a city. For availability reasons, the data must be located at both sites, and therefore one needs to look at alternatives for mirroring the storage.

failure group

A failure group is a subset of the disks in a disk group, which could fail at the same time because they share hardware. Failure groups are used to store mirror copies of data.

Fast Application Notification (FAN)

Applications can use FAN to enable rapid failure detection, balancing of connection pools after failures, and re-balancing of connection pools when failed components are repaired. The FAN notification process uses system events that Oracle Database publishes when cluster servers become unreachable or if network interfaces fail.

Fast Connection Failover

Fast Connection Failover provides high availability to FAN integrated clients, such as clients that use JDBC, OCI, or ODP.NET. If you configure the client to use fast

connection failover, then the client automatically subscribes to FAN events and can react to database `UP` and `DOWN` events. In response, Oracle Database gives the client a connection to an active instance that provides the requested database service.

file system

A file system is a software component providing structured access to disks. File systems present objects, such as files, to application programs. Access to files is generally specified with standard API defining operating system calls such as Open/ Close and Read/Write that the application program uses for accessing files. File systems are usually provided as a component of an operating system, but may be provided as an independent software component.

forced disk write

In Oracle RAC, a particular data block can only be modified by one instance at a time. If one instance modifies a data block that another instance needs, then whether a forced disk write is required depends on the type of request submitted for the block.

General Parallel File System (GPFS)

General Parallel File System (GPFS) is a shared-disk IBM file system product that provides data access from all of the nodes in a homogenous or heterogeneous cluster.

Global Cache Service (GCS)

Process that implements Cache Fusion. It maintains the block mode for blocks in the global role. It is responsible for block transfers between instances. The Global Cache Service employs various background processes such as the Global Cache Service Processes (LMSn) and Global Enqueue Service Daemon (LMD).

Global Cache Service Processes (LMSn)

Processes that manage remote messages. Oracle RAC provides for up to 10 Global Cache Service Processes.

Global Cache Service (GCS) resources

Global resources that coordinate access to data blocks in the buffer caches of multiple Oracle RAC instances to provide cache coherency.

global database name

The full name of the database that uniquely identifies it from any other database. The global database name is of the form *database_name*.*database_domain*—for example: `OP.US.FOO.COM`

global dynamic performance views (GV$)

Dynamic performance views storing information about all open instances in an Oracle RAC cluster. (Not only the local instance.) In contrast, standard dynamic performance views (`V$`) only store information about the local instance.

Global Enqueue Service (GES)

A service that coordinates enqueues that are shared globally.

Global Enqueue Service Daemon (LMD)

The resource agent process that manages requests for resources to control access to blocks. The LMD process also handles deadlock detection and remote resource requests. Remote resource requests are requests originating from another instance.

Global Enqueue Service Monitor (LMON)

The background LMON process monitors the entire cluster to manage global resources. LMON manages instance deaths and the associated recovery for any failed instance. In particular, LMON handles the part of recovery associated with global resources. LMON-provided services are also known as Cluster Group Services.

Global Services Daemon (GSD)

A component that receives requests from SRVCTL to execute administrative job tasks, such as startup or shutdown. The command is executed locally on each node, and the results are returned to SRVCTL. GSD is installed on the nodes by default.

Grid Plug and Play Daemon (GPNPD)

This process provides access to the Grid Plug and Play profile, and coordinates updates to the profile among the nodes of the cluster to ensure that all of the nodes node have the most recent profile.

High Availability Cluster Multi-Processing (HACMP)

High Availability Cluster Multi-Processing is an IBM AIX-based high availability cluster software product. HACMP has two major components: high availability (HA) and cluster multi-processing (CMP).

high availability

Systems with redundant components that provide consistent and uninterrupted service, even if there are hardware or software failures. This involves some degree of redundancy.

instance

For an Oracle RAC database, each node in a cluster usually has one instance of the running Oracle software that references the database. When a database is started, Oracle Database allocates a memory area called the System Global Area (SGA) and starts one or more Oracle Database processes. This combination of the SGA and the Oracle Database processes is called an instance. Each instance has unique Oracle System Identifier (SID), instance name, rollback segments, and thread ID.

instance caging

A method that uses an initialization parameter to limit the number of CPUs that an instance can use simultaneously for foreground processes.

instance membership recovery

The method used by Oracle RAC guaranteeing that all cluster members are functional or active. instance membership recovery polls and arbitrates the membership. Any members that do not show a heartbeat by way of the control file or who do not respond to periodic activity inquiry messages are presumed terminated.

instance name

Represents the name of the instance and is used to uniquely identify a specific instance when clusters share common services names. The instance name is identified by the INSTANCE_NAME parameter in the instance initialization file, initsid.ora. The instance name is the same as the Oracle System Identifier (SID).

instance number

A number that associates extents of data blocks with particular instances. The instance number enables you to start an instance and ensure that it uses the extents allocated to it for inserts and updates. This will ensure that it does not use space allocated for other instances.

interconnect

The communication link between nodes.

keystore

A container that stores a Transparent Data Encryption key. In previous releases, this was referred to as a wallet.

Logical Volume Manager (LVM)

A generic term that describes Linux or UNIX subsystems for online disk storage management.

Interprocess Communication (IPC)

A high-speed operating system-dependent transport component. The IPC transfers messages between instances on different nodes. Also referred to as the interconnect.

Master Boot Record (MBR)

A program that executes when a computer starts. Typically, the MBR resides on the first sector of a local hard disk. The program begins the startup process by examining the partition table to determine which partition to use for starting the system. The MBR program then transfers control to the boot sector of the startup partition, which continues the startup process.

memory pressure

A state indicating that there is a limited amount of available memory on a server.

metric

The rate of change in a cumulative statistic.

multitenant container database

See **CDB**.

mutables

Non-deterministic functions that can change their results each time they are called. Mutable functions can cause replay to be rejected, if the function results change at replay. Consider *sequence*.nextval and SYSDATE used in key values. If a primary key is built with values from these function calls, and is used in later foreign keys or other binds, then, at replay the same function result must be returned. Application Continuity provides mutable value replacement at replay for granted Oracle function calls to provide opaque bind-variable consistency.

Network Attached Storage (NAS)

Storage that is attached to a server by way of a network.

Network Time Protocol (NTP)

An Internet standard protocol, built on top of TCP/IP, that ensures the accurate synchronization to the millisecond of the computer clock times in a network of computers.

Network Interface Card (NIC)

A card that you insert into a computer to connect the computer to a network.

node

A node is a computer system on which Oracle RAC and Oracle Clusterware software are installed.

Object Link Manager (OLM)

The Oracle interface that maps symbolic links to logical drives and displays them in the OLM graphical user interface.

OCSSD

A Linux or UNIX process that manages the Cluster Synchronization Services (CSS) daemon. Manages cluster node membership and runs as oracle user; failure of this process results in cluster restart.

Oracle Cluster File Systems

Oracle offers two cluster file systems, OCFS2 for Linux and Oracle ASM Cluster File System (Oracle ACFS). While Oracle ACFS is a proprietary file system, the source for OCFS2 for Linux is available to all under GNUs' General Public License (GPL). The two file systems are not compatible.

Oracle Cluster Registry (OCR)

The Oracle RAC configuration information repository that manages information about the cluster node list and instance-to-node mapping information. OCR also manages information about Oracle Clusterware resource profiles for customized applications.

Oracle Clusterware

This is clusterware that is provided by Oracle to manage cluster database processing including node membership, group services, global resource management, and high availability functions.

Oracle Grid Infrastructure

The software that provides the infrastructure for an enterprise grid architecture. In a cluster this software includes Oracle Clusterware and Oracle Automatic Storage Management (Oracle ASM). For a standalone server, this software includes Oracle Restart and Oracle ASM. Oracle Database combines these infrastructure products into one software installation called the *Oracle Grid Infrastructure home* (`Grid_home`).

Oracle Grid Naming Service Daemon (GNSD)

The Oracle Grid Naming Service is a gateway between the cluster mDNS and external DNS servers. The `gnsd` process performs name resolution within the cluster.

Oracle High Availability Services Daemon (OHASD)

This process anchors the lower part of the Oracle Clusterware stack, which consists of processes that facilitate cluster operations.

Oracle Interface Configuration Tool (OIFCFG)

A command-line tool for both noncluster Oracle databases and Oracle RAC databases that enables you to allocate and de-allocate network interfaces to components, direct components to use specific network interfaces, and retrieve component configuration information. The Oracle Universal Installer also uses OIFCFG to identify and display available interfaces.

Oracle Managed Files

A service that automates naming, location, creation, and deletion of database files such as control files, redo log files, data files and others, based on a few initialization parameters. You can use Oracle Managed Files on top of a traditional file system supported by the host operating system, for example, VxFS or ODM. It can simplify many aspects of the database administration by eliminating the need to devise your own policies for such details.

Oracle Notification Service

A publish and subscribe service for communicating information about all FAN events.

Oracle Universal Installer

A tool to install Oracle Clusterware, the Oracle relational database software, and the Oracle RAC software. You can also use the Oracle Universal Installer to launch the Database Configuration Assistant (DBCA).

Oracle XA

An external interface that allows global transactions to be coordinated by a transaction manager other than Oracle Database.

PDB

In a multitenant container database (**CDB**), a portable collection of schemas, schema objects, and nonschema objects that appears to an Oracle Net client as a non-CDB.

pluggable database

See **PDB**.

policy-managed database

A database that you define as a cluster resource. Management of the database is defined by how you configure the resource, including on which servers the database can run and how many instances of the database are necessary to support the expected workload.

raw device

A disk drive that does not yet have a file system set up. Raw devices are used for Oracle RAC because they enable the sharing of disks. See also **raw partition**.

raw partition

A portion of a physical disk that is accessed at the lowest possible level. A raw partition is created when an extended partition is created and logical partitions are assigned to it without any formatting. Once formatting is complete, it is called a cooked partition. See also raw device.

recoverable error

A class of errors that arise due to an external system failure, independent of the application session logic that is executing. Recoverable errors occur following planned and unplanned outages of foregrounds, networks, nodes, storage, and databases. The application receives an error code that can leave the application not knowing the status of the last operation submitted.

Recovery Manager (RMAN)

An Oracle tool that enables you to back up, copy, restore, and recover data files, control files, and archived redo logs. It is included with the Oracle server and does not require separate installation. You can run RMAN as a command line utility from the operating system (O/S) prompt or use the GUI-based Oracle Enterprise Manager Backup Manager.

region

A logical boundary that contains database clients and servers that are considered to be close to each other.

request

A unit of work submitted from the application. A request typically corresponds to the SQL and PL/SQL, and other database calls of a single web request, on a single database connection, and generally is demarcated by the calls made to check-out and check-in the database connection from a connection pool.

result cache

A result cache is an area of memory, either in the SGA or client application memory, that stores the result of a database query or query block for reuse. The cached rows are shared across statements and sessions unless they become stale.

Runtime Connection Load Balancing

Enables Oracle Database to make intelligent service connection decisions based on the connection pool that provides the optimal service for the requested application based on current workloads. The JDBC, ODP.NET, and OCI clients are integrated with the load balancing advisory; you can use any of these client environments to provide runtime connection load balancing.

scalability

The ability to add additional nodes to Oracle RAC applications and achieve markedly improved scale-up and speed-up.

Secure Shell (SSH)

A program for logging into a remote computer over a network. You can use SSH to execute commands on a remote system and to move files from one system to another. SSH uses strong authentication and secure communications over insecure channels.

Server Control Utility (SRVCTL)

Server Management (SRVM) comprises the components required to operate Oracle Enterprise Manager in Oracle RAC. The SRVM components, such as the Intelligent Agent, Global Services Daemon, and SRVCTL, enable you to manage cluster databases running in heterogeneous environments through an open client/server architecture using Oracle Enterprise Manager.

server

A computer system that has no Oracle software installed upon it.

server cluster

The cluster in which the shared GNS server is running.

server group

A logical partition of nodes in a cluster into a group that hosts applications, databases, or both. Server groups can be members of other server groups.

service level

A measure of the performance of a system.

services

Entities that you can define in Oracle RAC databases that enable you to group database workloads and route work to the optimal instances that are assigned to offer the service.

session state consistency

After a COMMIT has executed, if the state was changed in that transaction, then it is not possible to replay the transaction to reestablish that state if the session is lost. When configuring Application Continuity, the applications are categorized depending on whether the session state after the initial setup is dynamic or static, and then whether it is correct to continue past a COMMIT operation within a request.

- **Dynamic**: (default) A session has dynamic state if the session state changes are not fully encapsulated by the initialization, and cannot be fully captured in a callback at failover. Once the first transaction in a request commits, failover is internally disabled until the next request begins. This is the default mode that most applications should use for requests.

- **Static**: (special—on request) A session has a static state if all session state changes, such as NLS settings and PL/SQL package state, can be repeated in an initialization callback. This setting is used only for database diagnostic applications that do not change session state inside requests. Do not set STATIC mode if there are any non-transactional state changes in the request that cannot be reestablished by a callback. If you are unsure, use DYNAMIC mode.

shared everything

A database architecture in which all instances share access to all of the data.

single client access name (SCAN)

Oracle Database 11*g* database clients use SCAN to connect to the database. SCAN can resolve to multiple IP addresses, reflecting multiple listeners in the cluster handling public client connections.

singleton services

Services that run on only one instance at any one time. By defining the Distributed Transaction Property (DTP) property of a service, you can force the service to be a singleton service.

split brain syndrome

Where two or more instances attempt to control a cluster database. In a two-node environment, for example, one instance attempts to manage updates simultaneously while the other instance attempts to manage updates.

SQL translation profile

A SQL translation profile is a database schema object that directs how SQL statements in non-Oracle databases are translated to Oracle, and how Oracle error codes and ANSI SQLSTATES are translated into other vendors' equivalents.

system identifier (SID)

The Oracle system identifier (SID) identifies a specific instance of the running Oracle software. For an Oracle RAC database, each node within the cluster has an instance referencing the database.

transparent application failover (TAF)

A runtime failover for high-availability environments, such as Oracle RAC and Oracle RAC Guard, TAF refers to the failover and re-establishment of application-to-service connections. It enables client applications to automatically reconnect to the database if the connection fails, and optionally resume a SELECT statement that was in progress. This reconnect happens automatically from within the Oracle Call Interface library.

Virtual Internet Protocol (VIP)

An IP address assigned to multiple applications residing on a single server, multiple domain names, or multiple servers, rather than being assigned to a specific single server or network interface card (NIC).

volume manager

A volume manager is a software component that manages the mapping of the collection of the pieces of the disks into a volume.

voting disk

A file that manages information about node membership.

Index