Oracle® Real Application Clusters

Administration and Deployment Guide

11*g* Release 2 (11.2)

E10718-04

September 2009

ORACLE®

Oracle Real Application Clusters Administration and Deployment Guide, 11*g* Release 2 (11.2)

E10718-04

Copyright © 1999, 2009, Oracle and/or its affiliates. All rights reserved.

Primary Author: Richard Strohm

Contributing Authors: Troy Anthony, Lance Ashdown, Ram Avudaiappan, Prasad Bagal, Mark Bauer, Anand Beldalker, Eric Belden, Gajanan Bhat, David Brower, George Claborn, Carol Colrain, Jonathan Creighton, Rajesh Dasari, Steve Fogel, Richard Frank, GP Prabhaker Gongloor, Wei Hu, Yong Hu, Dominique Jeunot, Sameer Joshi, Roland Knapp, Raj Kumar, Ken Lee, Karen Li, Barb Lundhild, Venkat Maddali, Bill Manry, Gaurav Manglik, 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, Sudheendra Sampath, Viv Schupmann, Daniel Semler, Ara Shakian, Cathy Shea, Khethavath P. Singh, Kesavan Srinivasan, Janet Stern, Juan Tellez, Leo Tominna, Peter Wahl, Tak Wang, Douglas Williams, Mike Zampiceni, Michael Zoll

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

3 Administering Database Instances and Cluster Databases

4 Introduction to Automatic Workload Management

5 Configuring Recovery Manager and Archiving

6 Cloning Oracle RAC to Nodes in a Cluster

7 Managing Backup and Recovery

8 Using Cloning to Extend Oracle RAC to Nodes in the Same Cluster

9 Adding and Deleting Oracle RAC from Nodes on Linux and UNIX Systems

10 Design and Deployment Techniques

11 Monitoring Performance

A Server Control Utility Reference

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. 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 Oracle single-instance databases that appear in other Oracle documentation. Where necessary, this manual refers to platform-specific documentation. This Preface contains these topics:

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

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at `http://www.oracle.com/accessibility/`.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Deaf/Hard of Hearing Access to Oracle Support Services

To reach Oracle Support Services, use a telecommunications relay service (TRS) to call Oracle Support at 1.800.223.1711. An Oracle Support Services engineer will handle technical issues and provide customer support according to the Oracle service request process. Information about TRS is available at `http://www.fcc.gov/cgb/consumerfacts/trs.html`, and a list of phone numbers is available at `http://www.fcc.gov/cgb/dro/trsphonebk.html`.

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 Real Application Clusters 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 Real Application Clusters 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 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 11*g* 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.

What's New in Oracle RAC Administration and Deployment?

This section describes the new administration and deployment features for Oracle Real Application Clusters (Oracle RAC) in Oracle Database 11*g* release 2 (11.2).

Oracle Database 11*g* Release 2 (11.2) New Features in Oracle RAC

This section describes the Oracle Database 11*g* release 2 (11.2) features for Oracle RAC administration and deployment.

- **Oracle Real Application Clusters One Node (Oracle RAC One Node)**

  Oracle Database 11*g* release 2 (11.2) introduces a new option, Oracle Real Application Clusters One Node (Oracle RAC One Node). Oracle RAC One Node is a single instance of Oracle Real Application Clusters (Oracle RAC) 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.

  You can limit the CPU utilization of individual database instances within the cluster through Resource Manager Instance Caging and dynamically change this limit if needed. 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 easily upgrade your applications online to Oracle RAC. If the node that is running Oracle RAC One Node becomes overloaded, then you can migrate the instance to another node in the cluster using the Omotion online migration utility with no downtime for application users.

  Oracle RAC One Node is supported on all platforms on which Oracle RAC is certified. With Oracle RAC and Oracle RAC One Node, you can standardize your deployments across a data center while achieving the required level of scalability and high availability for your applications.

  Similar to Oracle RAC, Oracle RAC One Node will be certified on Oracle Virtual Machine (Oracle VM). Oracle VM is a no-cost, next-generation server virtualization and management solution that simplifies enterprise applications deployment, management, and support. 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.

  If your Oracle VM is sized too small, then you can migrate the Oracle RAC One Node instance to another Oracle VM node in your cluster using Omotion, and

then resize your Oracle VM. When you move the Oracle RAC One Node instance back to the newly resized Oracle VM node, you can dynamically increase any limits programmed with Resource Manger Instance Caging.

Alternatively, you can create a larger Oracle VM and use Omotion to migrate to the new Oracle VM and then dynamically resize the Oracle instance, depending on the resources available in the cluster. Since Oracle clients use the Single Client Access Name (SCAN) to connect to the database, they can locate the service independently of the node on which it is running.

See Also: Oracle Technology Network for more information at

`http://otn.oracle.com/rac`

- **Grid Plug and Play**

  Grid Plug and Play reduces per-node configuration data and the need for explicit add and delete nodes steps, where possible. This allows a system administrator to take a template system image and run it on a node to be added with no further configuration. This removes many manual operations, reduces the opportunity for errors, and encourages configurations that can be changed more easily. Removal of the per-node configuration makes the nodes easier to replace because it is not required that they contain individual states that must be managed.

  Grid Plug and Play also introduces simplified instance addition. When your databases are backed with **Oracle Managed Files (OMF)** and Oracle Automatic Storage Management (Oracle ASM), recovery threads and undo tablespaces are automatically created for an instance that you add explicitly with the `srvctl add instance` command, or implicitly when a policy-managed database brings up a new instance.

  All tools and utilities such as DBCA, NETCA, and SRVCTL have been updated to support Grid Plug and Play. Oracle Enterprise Manager, the graphical interface for managing Oracle RAC, provides management and monitoring for the Grid Plug and Play environment.

  Grid Plug and Play reduces the cost of installing, configuring, and managing database nodes by making their per-node state disposable. Nodes can easily be replaced with regenerated state.

  See Also: *Oracle Real Application Clusters Installation Guide* for more information about Grid Plug and Play

- **Policy-based cluster and capacity management**

  Oracle Clusterware allocates and reassigns capacity based on policies you define, enabling faster resource failover and dynamic capacity assignment using policy-based management.

  Policy-based cluster and capacity management allows the efficient allocation of different types of applications in the cluster. Various applications can be hosted on a shared infrastructure, being isolated with regard to their resource consumption by policies and, therefore, behave as if they were deployed in single-system environments. Policy-managed Oracle RAC databases utilize policy-based cluster management to provide the required resources for the workloads the database supports.

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

- **Role-separated management**

  Role-separated management for Oracle Clusterware allows certain administrative tasks to be delegated to different people, representing different roles in the company. It is based on the idea of a clusterware administrator, who can grant administrative tasks on a per resource basis. For example, if two databases are placed into the same cluster, the cluster administrator can manage both databases in the cluster. But, the cluster administrator can also decide to grant different administrative privileges to each DBA responsible for each one of those databases.

  Role-separated management enables multiple applications and databases to share the same cluster and hardware resources, but ensures that different administration groups do not interfere with each other.

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

- **Improved Cluster Resource Modeling**

  Oracle Clusterware can manage different types of applications and processes. You can create dependencies among the applications and processes and manage them as a single entity.

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

- **Oracle Enterprise Manager-based Oracle Clusterware resource management**

  You can use Oracle Enterprise Manager to manage Oracle Clusterware resources. You can create and configure resources in Oracle Clusterware and also monitor and manage resources after they are deployed in the cluster.

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

- **Oracle Cluster Registry performance enhancements**

  Improvements in the way Oracle Clusterware accesses Oracle Cluster Registry (OCR) speed up relocation of services when a node fails. Oracle Clusterware now supports up to five copies of OCR for improved availability of the cluster and OCR can now be stored in Oracle ASM.

  The tools to manage OCR have changed to support the new management options. Consistent storage management automation provides improved performance in Oracle Clusterware and Oracle RAC environments, and easier management of the cluster.

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

- **SRVCTL support for single-instance database**

  Server Control Utility (SRVCTL) commands have been enhanced to manage the configuration in a standalone server using Oracle Restart. The new SRVCTL functionality enables you to register a single-instance database that can be managed by Oracle Clusterware. Once registered, Oracle Clusterware can start, stop, monitor, and restart the database instance.

The new SRVCTL functionality simplifies management of Oracle Database through a consistent interface that can be used from the console or scripted. An improved management interface makes it easy to provide higher availability for single-instance databases that run on a server that is part of a cluster.

See Also:

- *Oracle Database Administrator's Guide* for more information about using SRVCTL commands on a single-instance database
- Appendix A, "Server Control Utility Reference" for a list of SRVCTL commands

- **Enhanced Cluster Verification Utility**

  New Cluster Verification Utility (CVU) functionality checks certain storage types and configurations. Also, more consideration is given to user-specific settings.

  In addition to command-line commands, these checks are done through the Oracle Universal Installer, database configuration assistant (DBCA), and Oracle Enterprise Manager. These enhancements facilitate implementation and configuration of cluster environments and provide assistance in diagnosing problems in a cluster environment, improving configuration and installation.

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

- **Oracle Enterprise Manager support for Grid Plug and Play**

  You can use Oracle Enterprise Manager:

  – To support the Grid Plug and Play environment

  – To administer dynamic configuration use

  – To manage Grid Plug and Play profiles and targets, such as hosts, clusters, and Oracle RAC databases and Oracle RAC database instances

  Additionally, Oracle Enterprise Manager supports other Oracle RAC administration tasks, including:

  – Monitoring

  – Startup

  – Shutdown

  – Backup and recovery

  – Tablespace management

  – Node addition

- **Oracle Enterprise Manager provisioning for Oracle Clusterware and Oracle RAC**

  The Oracle Enterprise Manager provisioning framework has been updated to reflect the changes to the installation and configuration of Oracle Clusterware and Oracle RAC. You can achieve easier implementation and management of a clustered database environment using the Oracle Enterprise Manager provisioning framework.

- **Patch application with Oracle Enterprise Manager Database Control**

Oracle Enterprise Manager Database Control now manages the application of patches to a single-instance database, Oracle RAC, and Oracle Clusterware. Using Oracle Enterprise Manager to apply patches simplifies software maintenance.

- **Zero downtime for patching Oracle RAC**

  Patching Oracle Clusterware and Oracle RAC can be completed without taking the entire cluster down. This also allows for out-of-place upgrades to the cluster software and Oracle Database, reducing the planned maintenance downtime required in an Oracle RAC environment.

- **Integrated support for application failover in an Oracle Data Guard configuration**

  Applications connected to a primary database transparently failover to a new primary database when Oracle Data Guard changes roles. Clients integrated with Fast Application Notification (FAN) can achieve fast failover between primary and standby databases, in addition to fast failover within the cluster. Services have an attribute with which you can associate the service with a database role, such as `PHYSICAL_STANDBY`, so that the service is only active when the database is mounted in the associated role.

  See Also:

  - *Oracle Data Guard Broker* for more information
  - Chapter 4, "Introduction to Automatic Workload Management"

- **Oracle ASM Dynamic Volume Manager**

  The Oracle ASM Dynamic Volume Manager is a kernel-loadable device driver that provides a standard device driver interface to clients, such as the Oracle Automatic Storage Management Cluster File System (Oracle ACFS). Oracle ASM Dynamic Volume Manager is the primary I/O interface for Oracle ACFS to perform I/O and build a file system using Oracle Automatic Storage Management (Oracle ASM) as a volume manager. Oracle ASM Dynamic Volume Manager is loaded upon Oracle ASM startup, is cluster aware, and communicates with Oracle ASM for extent map information, extent rebalancing, and I/O failures.

  Oracle ASM Dynamic Volume Manager provides a standard I/O interface allowing general-purpose file systems to leverage the full functionality of Oracle ASM as a volume manager. Files not directly supported by Oracle ASM, such as Oracle binaries, can now reside on ACFS on Oracle ASM volumes. This eliminates the need for third-party file systems or volume managers to host general-purpose files.

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

- **Oracle Enterprise Manager support for Oracle Automatic Storage Management Cluster File System**

  Oracle Enterprise Manager provides a comprehensive management solution that extends Oracle ASM technology to support all customer application data files, both database and non-database, and in both single-host and cluster configurations. It also enhances existing Oracle Enterprise Manager support for Oracle ASM, and adds features to support the Oracle ASM Dynamic Volume Manager (ADVM) and Oracle ASM Cluster File System (ACFS) technology.

Oracle Enterprise Manager provides a graphical user interface that makes it easier to manage the environment, whether it is a standalone server or a cluster deployment of Oracle ASM. The centralized console provides a consistent interface for managing volumes, database files, file systems, and the Oracle Database.

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

- **Oracle Automatic Storage Management Cluster File System**

  The Oracle Automatic Storage Management Cluster File System (Oracle ACFS) provides a robust, modern, general purpose file system for files beyond the Oracle database files. Oracle ACFS also provides support for files such as Oracle binaries, report files, trace files, alert logs, and other application data files. With the addition of Oracle ACFS, Oracle ASM becomes a complete storage management solution for both Oracle database and non-database files.

  Additionally, Oracle ACFS

  – Supports large files with 64-bit file and file system data structure sizes leading to exabyte-capable file and file system capacities

  – Uses extent-based storage allocation for improved performance

  – Uses a log-based metadata transaction engine for file system integrity and fast recovery

  – Can be exported to remote clients through industry standard protocols such as NFS and CIFS

  Oracle ACFS complements and leverages Oracle ASM and provides a general purpose journaling file system for storing and managing non-Oracle database files. This eliminates the need for third-party cluster file system solutions, while streamlining, automating, and simplifying all file type management in both single node and Oracle RAC and Grid computing environments.

  Oracle ACFS supports dynamic file system expansion and contraction without any downtime and is also highly available, leveraging the Oracle ASM mirroring and striping features in addition to hardware RAID functionality.

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

- **Automatic Storage Management file access control**

  This feature implements access control on Oracle ASM files on UNIX platforms to isolate itself and different database instances from each other and prevent unauthorized access. The feature includes SQL statements to grant, modify, and deny file permissions.

  This feature enables multiple database instances to store their Oracle ASM files in the same disk group and enables consolidation of multiple databases, securely, to prevent database instances from accessing or overwriting files belonging to other database instances.

See Also:

- *Oracle Database Storage Administrator's Guide* for more information
- "Oracle Automatic Storage Management in Oracle RAC" on page 2-4

- **Universal Connection Pool**

  Universal Connection Pool (UCP) is a Java connection pool that replaces the deprecated JDBC Implicit Connection Cache with Oracle Database 11*g* (11.1.0.7). UCP is integrated with Oracle RAC to provide the following benefits:

  - A single UCP can be leveraged by any Oracle component or user.
  - Eliminates redundant connection pools from several Oracle Components, such as AOL/J, ADF Business Components, and TopLink.
  - Provides consistent connection pool behavior for an Oracle component or product. For example, the connection pool sizes can be configured to provide consistent connection management behavior for an application.
  - Provides JMX interfaces for the UCP Manager, which delivers a consistent management interface to manage the connection pool.
  - UCP adapters can provide standards compliance for a specific connection type being pooled.
  - Supports connection pooling for Oracle and non-Oracle connections.
  - Supports pooling for any type of connections, including JDBC or JCA connections.

  See Also: *Oracle Database 2 Day + Real Application Clusters Guide* for more information about configuring JDBC clients

- **Expose high availability events through a Java API**

  You can access fast application notification (FAN) events with a simplified JAVA API if you are not using the Oracle connection pool features.

  See Also: "Fast Application Notification" on page 4-8 for more information about FAN events

- **SRVCTL enhancements to support Grid Plug and Play**

  This feature includes enhancements to the server control utility (SRVCTL) in conjunction with the Grid Plug and Play feature.

  See Also: Appendix A, "Server Control Utility Reference" for a list of SRVCTL commands

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

A **cluster** comprises multiple interconnected computers or servers that appear as if they are one server to end users and applications. Oracle RAC enables you to cluster an Oracle database. 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 also a required component for using Oracle RAC. In addition, Oracle Clusterware enables both single-instance Oracle databases and Oracle RAC databases to use the Oracle high-availability infrastructure. Oracle Clusterware enables you to create a clustered pool of storage to be used by any combination of single-instance and Oracle RAC databases.

Oracle Clusterware is the only clusterware that you need for most platforms on which Oracle RAC operates. You can also use clusterware from other vendors if the clusterware is certified for Oracle RAC.

> **See Also:** *Oracle Clusterware Administration and Deployment Guide* and *Oracle Grid Infrastructure Installation Guide* for more details

Single-instance 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.

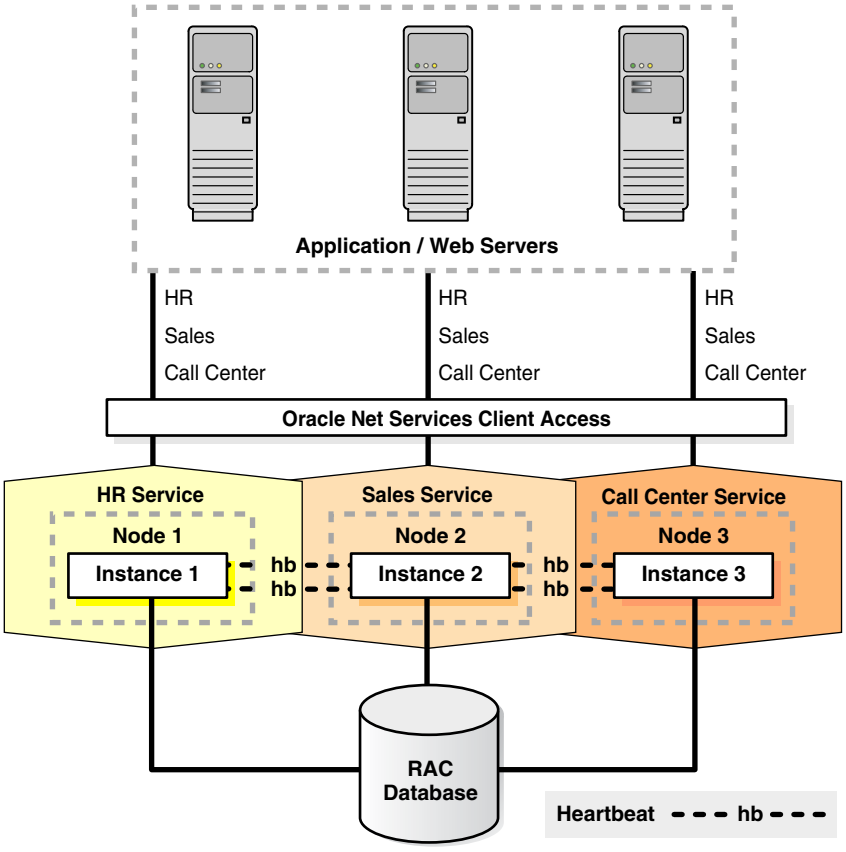
Oracle RAC databases differ architecturally from single-instance Oracle databases in that each Oracle RAC database instance also has:

- 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.

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 usually runs 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 single database cluster. In an extended cluster, the nodes in the cluster are located in two buildings that are separated by greater distances (anywhere from across the street, to across a campus, or across a city). For availability reasons, the data must be located at both sites, thus requiring the implementation of disk mirroring technology for storage.

[1] For configurations running Oracle Database 10*g* release 2 (10.2) and later releases, Oracle Clusterware supports 100 nodes in a cluster, and Oracle RAC supports 100 instances in an Oracle RAC database.

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 a 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, airplane crash, 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 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 Virtual Internet Protocol (VIP) address, the Single Client Access Name (SCAN), the SCAN listener, Oracle Notification Service (ONS), and the Oracle Net listeners. These resources are automatically started when Oracle Clusterware starts the node and automatically restarted 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 allows you to create CRS resources to manage any process running on servers in the cluster which are not already 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
- Overview of Connecting to Oracle Database Using Services and VIP Addresses
- 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,[2] 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 disk, but you must use a supported cluster-aware storage solution. Oracle Database provides the following file storage options for Oracle RAC:

- Oracle Automatic Storage Management (Oracle ASM)

  This is the recommended solution to manage your disk.

- A certified cluster file system, including OCFS2 and Oracle Cluster File System (OCFS)

  OCFS2 is available for Linux and OCFS is available for Windows platforms. However you may optionally use a third-party cluster file system or cluster-aware volume manager that is certified for Oracle RAC.

- Certified network file system (NFS) file servers

Overview of Connecting to Oracle Database Using Services and VIP Addresses

All **node**s in an Oracle RAC environment must connect to a Local Area Network (LAN) to enable users and applications to access the database. Applications should use the Oracle Database services feature to connect to an Oracle 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. Oracle Net Services enable

[2] Note that PFILE files do not need to be shared.

the load balancing of application connections across all of the instances in an Oracle RAC database.

Users can access an Oracle RAC database using a client/server configuration or through one or more middle tiers, with or without connection pooling. Users can be database administrators, developers, application users, power users, such as data miners who create their own searches, and so on.

Most public networks typically use TCP/IP, but you can use any supported hardware and software combination. Oracle RAC database instances should be accessed through the Single Client Access Name (SCAN) for the cluster.

> **See Also:** "Overview of Automatic Workload Management" on page 1-7 for more information about SCANs

The interconnect network is a private network that connects all of the servers in the cluster. The interconnect network uses a switch (or multiple switches) that only the nodes in the cluster can access. Configure User Datagram Protocol (UDP) on a Gigabit Ethernet for your cluster interconnect. On Linux and UNIX systems, you can configure Oracle Clusterware to use either the UDP or Reliable Data Socket (RDS) protocols. Windows clusters use the TCP protocol. Crossover cables are not supported for use with Oracle Clusterware interconnects.

Note: Do not use the interconnect for the private network for user communication, because **Cache Fusion** uses the private interconnect for interinstance communication.

If a node fails, then the node's 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. 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

Clients that attempt to connect to the VIP address 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.

If you use Network Attached Storage (NAS), then you are required to configure a second private network. Access to this network is typically controlled by the vendor's software. The private network uses static IP addresses.

Oracle RAC 11*g* release 2 (11.2) supports multiple public networks. Each network has its own virtual IP address and uses a unique service to access the Oracle RAC database. Each network is a resource managed by Oracle Clusterware.

You must also create a SCAN for each cluster, which 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).

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 single-instance Oracle database and additional process 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 SGA size requirements for Oracle RAC are greater than the SGA requirements for single-instance Oracle databases due to Cache Fusion.

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 manages 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 a 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.

- `RMS`*n*: Oracle RAC Management Processes (RMS*n*)

  The RMS*n* processes perform manageability tasks for Oracle RAC. Tasks accomplished by an RMS*n* 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 single-instance Oracle databases in *Oracle Database Concepts.*

Overview of Automatic Workload Management

Automatic workload management enables you to manage the distribution of workloads to provide optimal performance for users and applications. This includes providing the highest availability for database connections, rapid failure recovery, and balancing workloads optimally across the active configuration. Oracle Database with Oracle RAC includes many features that can enhance automatic workload management, such as connection load balancing, fast connection failover, the load balancing advisory, and run-time connection load balancing. Automatic workload management provides the greatest benefits to Oracle RAC environments. You can, however, take advantage of automatic workload management by using Oracle Database services in single-instance Oracle databases, especially those that use Oracle Data Guard or Oracle Streams. Automatic workload management comprises the following components:

- **High Availability Framework**: The Oracle RAC high availability framework enables Oracle Database to always maintain components in a running state. Oracle high availability implies that Oracle Clusterware monitors and restarts critical components if they stop, unless you override the restart processing. Oracle Clusterware and Oracle RAC also provide alerts to clients when configurations change, enabling clients to immediately react to the changes, enabling application developers to hide outages and reconfigurations from end users. The scope of Oracle high availability spans from the restarting of stopped Oracle Database processes in an Oracle database instance to failing over the processing of an entire instance to other available instances.

- **Single Client Access Name (SCAN)**: A single network name and IP addresses defined either in your DNS or GNS that all clients should use to access the Oracle RAC database. With SCAN, you are no longer required to modify your clients

when changes occur to the cluster configuration. SCAN also allows clients to use an Easy Connect string to provide load balancing and failover connections to the Oracle RAC database.

> **Note:** SCAN is required regardless of whether you use GNS. If you use GNS, then Oracle automatically creates the SCAN. If you do not use GNS, then you must define the SCAN in DNS.

- **Load Balancing Advisory**: This is the ability of the database to provide information to applications about the current service levels being provided by the database and its instances. Applications can take advantage of this information to direct connection requests to the instance that will provide the application request with the best service quality to complete the application's processing. Oracle Database has integrated its Java Database Connectivity (JDBC) and Oracle Data Provider for .NET (ODP.NET) connection pools to work with the load balancing information. Applications can use the integrated connection pools without programmatic changes.
- **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. Services enable you to group database workloads and route the work to the optimal instances that are assigned to process the service. Furthermore, you can use services to define the resources that Oracle Database assigns to process workloads and to monitor workload resources. Applications that you assign to services transparently acquire the defined automatic workload management characteristics, including high availability and load balancing rules. Many Oracle Database features are integrated with services, such as Resource Manager, which enables you to restrict the resources that a service can use within an instance. Some database features are also integrated with Oracle Streams, Advanced Queuing (to achieve queue location transparency), and Oracle Scheduler (to map services to specific job classes).

  In Oracle RAC databases, the service performance rules that you configure control the amount of work that Oracle Database allocates to each available instance for that service. As you extend your database by adding nodes, applications, components of applications, and so on, you can add more services.
- **Server Pools**: Server pools enable the cluster administrator to create a policy which defines how Oracle Clusterware allocates resources. An Oracle RAC policy-managed database runs in a server pool. Oracle Clusterware attempts to keep the required number of servers in the server pool and, therefore, the required number of instances of the Oracle RAC database. A server can be in only one server pool at any time. However, a database can run in multiple server pools. Cluster-managed services run in a server pool where they are defined as either UNIFORM (active on all instances in the server pool) or SINGLETON (active on only one instance in the server pool).
- **Connection Load Balancing**: Oracle Net Services provides connection load balancing for database connections. Connection load balancing occurs when the connection is created. Connections for a given service are balanced across all of the running instances that offer the service. You should define how you want connections to be balanced in the service definition. However, you must still configure Oracle Net Services. When you enable the load balancing advisory, the listener uses the load balancing advisory for connection load balancing.

See Also: Chapter 4, "Introduction to Automatic Workload Management"

Overview of Installing Oracle RAC

Install Oracle Clusterware 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. As a database administrator, after installation your tasks are to administer your Oracle RAC environment at three levels:

- Instance Administration
- Database Administration
- Cluster Administration

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

- Understanding Compatibility in Oracle RAC Environments
- Overview of Oracle RAC Installation and Database Creation
- Overview of Extending the Grid Foundation and Oracle RAC Software

Note: You must first install Oracle Clusterware before installing Oracle RAC. See *Oracle Clusterware Administration and Deployment Guide* for more information.

Understanding Compatibility in Oracle RAC Environments

To run Oracle RAC in configurations with different versions of the database in the same cluster, you must also install clusterware. For example, to run Oracle9*i* and Oracle Database 10*g* in the same cluster:

- For Oracle RAC nodes running the Oracle9*i* database, you must install an Oracle9*i* cluster:
  - For UNIX the cluster can be HACMP, Serviceguard, Sun Cluster, or Veritas SF
  - For Windows and Linux, the cluster is Oracle Cluster Manager
- If you want to install Oracle RAC running Oracle Database 10*g* or later releases, you must also install Oracle Clusterware. For 11*g* release 2 (11.2), this means you must first install the grid infrastructure for a cluster. See your platform-specific *Oracle Grid Infrastructure Installation Guide* and the *Oracle Clusterware Administration and Deployment Guide* for more information.
- If you run Oracle RAC 10*g* and Oracle RAC 11*g* in the same cluster, you must run Oracle Clusterware 11*g* (only)

Oracle requires that you install the Oracle9*i* RAC software first if you are going to run it in a cluster with Oracle RAC 10*g* or Oracle RAC 11*g*.

Note: If you are adding Oracle RAC to servers that are part of a cluster, either migrate to Oracle Clusterware or ensure the clusterware you run is supported to run with Oracle RAC on release 10*g* or later releases and ensure you have installed the correct options for the two to work together. Oracle strongly recommends that you do not run different cluster software on the same servers unless they are certified to work together.

Overview of Oracle RAC Installation and Database Creation

Once you have installed Oracle Clusterware and it is operational, run Oracle Universal Installer to install the Oracle database software with Oracle RAC components.

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

Note: If a default listener is not present in the grid infrastructure home, then DBCA returns an error instructing you to run NETCA from the grid infrastructure home to create a default listener. A prerequisite of creating a database is that a default listener must be running in the grid infrastructure home.

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

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

Oracle recommends that you select Oracle ASM during the 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.

By default, Oracle Database creates one service for your environment and the service is for the database. (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: Avoid changing host names after you complete the Oracle Clusterware installation, including adding or deleting domain qualifications. Nodes with changed host names must be deleted from the cluster and added back with the new name.

Overview of Extending the Grid Foundation and Oracle RAC Software

You can extend Oracle ASM and Oracle RAC in grid environments to additional nodes by copying cloned images of the Oracle ASM and Oracle RAC database homes to other nodes in the cluster. Oracle cloning copies images of the software to other nodes that have similar hardware and software. Cloning is best suited to a scenario where you must quickly extend your Oracle RAC environment to several nodes of the same configuration.

Oracle provides the following methods of extending Oracle Clusterware environments:

- Oracle cloning procedure using cloning scripts
- Oracle Enterprise Manager cloning
- The `addNode.sh` script and OUI cloning

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

For new installations or if you install only one Oracle RAC database, you should use the traditional automated and interactive installation methods, such as OUI, 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 Chapter 9, "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.

At a high level, Oracle cloning involves the following main tasks:

1. Clone the Oracle Clusterware home following the instructions in *Oracle Clusterware Administration and Deployment Guide*.
2. Clone the Oracle home with the Oracle RAC software.

   The process for cloning the Oracle home onto new nodes is similar to the process for cloning the Oracle Clusterware home.
3. If you have not yet created a database, then run the DBCA to create one.
4. Follow the post-cloning procedures to complete the extension of your Oracle RAC environment onto the new nodes.

See Also:

- Chapter 6, "Cloning Oracle RAC to Nodes in a Cluster"
- Chapter 9, "Adding and Deleting Oracle RAC from Nodes on Linux and UNIX Systems" for information about adding and deleting nodes and instances on Linux and UNIX Systems
- Oracle Enterprise Manager online Help system for more information about the Provisioning Pack
- *Oracle Database Net Services Administrator's Guide* for more information about NETCA

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 will achieve both business and technical objectives.

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

- Chapter 10, "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 RAC provides a new method to manage your clustered database. Traditionally, databases were administrator-managed, where a DBA managed each instance of the database by defining specific instances to run on specific nodes in the cluster. To help implement dynamic grid configurations, Oracle RAC 11*g* release 2 (11.2) introduces *policy-managed* databases, where the DBA is required only to define the *cardinality* (number of database instances required). Oracle Clusterware manages the allocation of nodes to run the instances and Oracle RAC allocates the required redo threads and undo tablespaces, as needed.

Note: Automatic allocation of the required redo threads and undo tablespaces only happens when the database uses Oracle Managed Files.

About Administrative Tools for Oracle RAC Environments

Oracle enables you to administer a cluster database as a single system image through Oracle Enterprise Manager, SQL*Plus, or through Oracle RAC command-line interfaces such as Server Control Utility (SRVCTL):

- **Oracle Enterprise Manager**: Oracle Enterprise Manager has both the Database Control and Grid Control GUI interfaces for managing both single-instance database and Oracle RAC database environments. Oracle recommends that you use Oracle Enterprise Manager to perform administrative tasks whenever feasible.
- **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 Clusterware, and Oracle ASM.
- **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 the CVU for ongoing administrative tasks, such as node addition and node deletion.
- **DBCA**: DBCA is the recommended method for creating and initially configuring Oracle RAC databases.
- **NETCA**: Configures the network for your Oracle RAC environment.

See Also:

- Chapter 3, "Administering Database Instances and Cluster Databases" for an introduction to Oracle RAC administration using Oracle Enterprise Manager, SQL*Plus, and the SRVCTL utility
- "Monitoring Oracle RAC and Oracle Clusterware" on page 11-1
- Appendix A, "Server Control Utility Reference" for SRVCTL reference information
- *Oracle Clusterware Administration and Deployment Guide* for information about Oracle Clusterware tools such as the OIFCFG tool for allocating and deallocating network interfaces, the `OCRCONFIG` command-line tool for managing OCR, and the Cluster Verification Utility (CVU)
- *Oracle Database Net Services Administrator's Guide* for more information about NETCA

About Monitoring Oracle RAC Environments

Web-based Oracle Enterprise Manager Database Control and Grid Control enable you to monitor an Oracle RAC database. The Oracle Enterprise Manager Console 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" on page 11-1 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 the Oracle Enterprise Manager Console to initiate cluster database management tasks.
- Use Oracle Enterprise Manager Grid Control to administer multiple Oracle RAC databases.
- Use the global views, or `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.
- Use the sophisticated management and monitoring features of the Oracle Database Diagnostic and Tuning packs that include the Automatic Database Diagnostic Monitor (ADDM) and 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.

- Use the Oracle Instantaneous Problem Detection Tool for the operating system (IPD/OS) to detect and analyze operating system and cluster resource related degradation and failures, such as node evictions, in your cluster.

  Note: You can download IPD/OS from:

  `http://otn.oracle.com/rac`

  View the README for installation instructions.

  See Also: The *Oracle Database Performance Tuning Guide* describes the Oracle Database automatic features for performance diagnosing and tuning, including ADDM.

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 performed well on a single-instance Oracle database, then it will perform well in an Oracle RAC environment. Many of the tuning tasks that you would perform on a single-instance Oracle database can also improve Oracle RAC database performance. This is especially true if your environment required 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 Enterprise Manager Database Control and Grid Control are Integrated with Oracle RAC

2

Administering Storage

This chapter describes storage topics, such as Oracle Automatic Storage Management (Oracle ASM), in Oracle Real Application Clusters (Oracle RAC) environments.

This chapter includes the following topics:

- Overview of Storage in Oracle RAC
- Optimal Flexible Architecture
- Data File Access in Oracle RAC
- Redo Log File Storage in Oracle RAC
- Automatic Undo Management in Oracle RAC
- Oracle Automatic Storage Management in Oracle RAC

See Also:

- *Oracle Clusterware Administration and Deployment Guide*
- *Oracle Grid Infrastructure Installation Guide*
- *Oracle Real Application Clusters Installation Guide*

Overview of Storage in Oracle RAC

All data files (including an undo tablespace for each instance) and redo log files (at least two for each instance) must reside in an Oracle ASM disk group, on a **cluster file system**, or on shared raw devices. In addition, Oracle recommends that you use one shared server parameter file (SPFILE) with instance-specific entries. Alternatively, you can use a local file system to store instance-specific parameter files (PFILEs).

Note: Database Configuration Assistant (DBCA) does not support shared raw devices for this release, nor does DBCA allow PFILES. However, you can use SQL commands to configure data files on shared raw devices.

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

See Also: For additional information about these storage features:

- *Oracle Database 2 Day DBA*
- *Oracle Database Storage Administrator's Guide*
- *Oracle Database Administrator's Guide*

If you do not use Oracle ASM, if your platform does not support a cluster file system, or if you do not want to use a cluster file system for database file storage, then create additional raw devices as described in your platform-specific Oracle RAC installation and configuration guide. However, Oracle recommends that you use Oracle ASM for database file storage, as described in "Oracle Automatic Storage Management in Oracle RAC" on page 2-4.

Notes:

- If you use raw devices, then you cannot use DBCA.
- To create an Oracle RAC database using Oracle Database Standard Edition, you must use Oracle ASM for your database storage.

Optimal Flexible Architecture

Optimal Flexible Architecture (OFA) ensures reliable installations and improves software manageability. This feature streamlines the way in which Oracle software installations are organized, thereby simplifying the on-going management of your installations and improves manageability by making default Oracle Database installs more compliant with OFA specifications.

During installation, you are prompted to specify an Oracle base (`ORACLE_BASE`) location, which is owned by the user performing the installation. You can choose an existing `ORACLE_BASE`, or choose another directory location that does not have the structure for an `ORACLE_BASE` directory.

Using the Oracle base directory path helps to facilitate the organization of Oracle installations, and helps to ensure that installations of multiple databases maintain an OFA configuration. During the installation, `ORACLE_BASE` is the only required input, as the `ORACLE_HOME` is defaulted based on the value chosen for the `ORACLE_BASE`. In addition, Oracle recommends that you set the `ORACLE_BASE` environment variable in addition to `ORACLE_HOME`, when starting databases. Note that `ORACLE_BASE` may become a required environment variable for database startup in a future release.

See Also: *Oracle Real Application Clusters Installation Guide* installation guide for more information about specifying an `ORACLE_BASE` directory

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

If you increase the cardinality of a server pool in a policy-managed database, and a new server is allocated to the server pool, then Oracle starts an instance on the new server if you have Oracle Managed Files (OMF) enabled. If the instance starts and there is no thread or redo log file available, then Oracle automatically enables a thread of redo and allocates the associated redo log files if the database uses Oracle ASM or any cluster file system.

You should create redo log groups only if you are using administrator-managed databases. For policy-managed databases, increase the cardinality and when the instance starts, if you are using Oracle Managed Files and Oracle ASM, then Oracle automatically allocates the thread, redo, and undo.

For administrator-managed databases, each instance has its own online redo log groups. Create these redo log groups and establish group members, as described in the *Oracle Database Administrator's Guide*. To add a redo log group to a specific instance, specify the `INSTANCE` clause in the `ALTER DATABASE ADD LOGFILE` statement, as described in the *Oracle Database SQL Language Reference*. If you do not specify the instance when adding the redo log group, the redo log group is added to the instance to which you are currently connected.

> **See Also:** "About Designing and Deploying Oracle RAC Environments" on page 1-12 for more information about administrator and policy management for databases

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. Only the instance assigned to the undo tablespace can modify the contents of that tablespace. However, all 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 OMF 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:** *Oracle Database Administrator's Guide* for detailed information about creating and managing undo tablespaces

Oracle Automatic Storage Management in 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 within your **cluster database** environment. Oracle ASM partitions your total disk space requirements into uniformly sized units across all disks in a disk group. Oracle ASM can also automatically mirror data to prevent data loss. Because of these features, Oracle ASM also significantly reduces your administrative overhead.

To use Oracle ASM in Oracle RAC, select Oracle ASM as your storage option when you create your database with the Database Configuration Assistant (DBCA). As in single-instance Oracle databases, using Oracle ASM in 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 in Oracle RAC
- Oracle ASM Disk Group Management
- Configuring Preferred Mirror Read Disks in Extended Distance Clusters
- Converting Single-Instance Oracle ASM to Clustered Oracle ASM
- Administering Oracle ASM Instances with SRVCTL in Oracle RAC

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

Storage Management in Oracle RAC

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. Back up the parameter files and the TNS entries for nondefault Oracle Net listeners.

You can create Oracle ASM disk groups and configure mirroring for Oracle ASM disk groups using the Oracle ASM configuration assistant (ASMCA). After your Oracle RAC database is operational, you can administer Oracle ASM disk groups 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, 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.

Modifying Disk Group Configurations for Oracle ASM in Oracle RAC

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 in an Oracle RAC database compared to using it in a single-instance Oracle database. Oracle ASM automatically re-balances the storage load 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 in single-instance Oracle databases, you can use Oracle Enterprise Manager, ASMCA, SQL*Plus, and the Server Control Utility (SRVCTL) to administer disk groups for Oracle ASM in Oracle RAC. The *Oracle Database Storage 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 ASM, you must first create ASM disk groups with ASMCA before manually creating a database with DBCA. You can also use the Oracle ASM disk group management feature 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 diskgroup, 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 Database Storage 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.

You can also use the ASM Disk Groups page in ASMCA for standalone 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.

Configuring Preferred Mirror Read Disks in Extended Distance Clusters

When you configure Oracle ASM failure groups, 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 Database Storage 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 Single-Instance Oracle ASM to Clustered Oracle ASM

When installing Grid Infrastructure, any single-instance ASM instances are automatically converted to clustered ASM.

See Also:

- *Oracle Database 2 Day + Real Application Clusters Guide* for information about using Oracle Enterprise Manager Grid Control to convert single-instance Oracle ASM to clustered Oracle ASM
- *Oracle Database Storage Administrator's Guide* for complete information about configuring preferred mirror read disks in extended distance clusters
- Your platform-specific Oracle RAC installation guide 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, remove, enable, and disable an Oracle ASM instance. To issue SRVCTL commands to manage Oracle ASM, log in as the operating system user that owns the Oracle ASM home and issue the SRVCTL commands from the bin directory of the Oracle ASM home.

Use the following syntax to remove an Oracle ASM instance:

```
srvctl remove asm [-f]
```

Use the following syntax to enable an Oracle ASM instance:

```
srvctl enable asm [-n node_name]
```

Use the following syntax to disable an Oracle ASM instance:

```
srvctl disable asm [-n node_name]
```

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 [-n node_name] [-o start_options]
```

Use the following syntax to stop an Oracle ASM instance:

```
srvctl stop asm [-n node_name] [-o stop_options]
```

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

```
srvctl config asm -n node_name
```

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

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

> **See Also:** *Oracle Database Storage 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) database instances and Oracle RAC databases.

The topics in this chapter include:

- Tools for Administering Oracle RAC
- Starting and Stopping Instances and Oracle RAC Databases
- 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
- 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 online help for more information about Oracle Enterprise Manager

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: Oracle Enterprise Manager, SQL*Plus, and the SRVCTL utility. In many cases, you use these tools the same way to manage Oracle RAC environments as you would use them manage single-instance Oracle databases:

- Administering Oracle RAC with Oracle Enterprise Manager
- Administering Oracle RAC with SQL*Plus
- Administering Oracle RAC with SRVCTL

Oracle RAC Database Administration

Prior to Oracle RAC 11*g* release 2 (11.2), DBAs had to hard code parameters, such as database instance number and redo threads, to allocate specific Oracle RAC database instances to nodes within the cluster. If a node in the cluster does not start, then the

database instance does not start. In Oracle RAC 11*g* release 2 (11.2), this method of managing your Oracle RAC database continues to be available. An Oracle RAC database managed in this manner is referred to as an *administrator-managed* database. Administrator-managed databases include pre-11*g* release 2 (11.2) Oracle databases and upgraded Oracle databases. You can manage these databases using the same commands or methods (such as DBCA or Oracle Enterprise Manager) you used with previous releases of Oracle Database. All commands and utilities maintain backward compatibility.

In Oracle RAC 11*g* release 2 (11.2), Oracle introduces *policy-managed* databases to move away from any hard coding of parameters, to make it easier to replace nodes in a cluster, or expand the cluster as requirements change. Policy-managed databases must be 11*g* release 2 (11.2) 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 are created in the cluster by the cluster administrator, and it can run on different servers at different times. A database instance starts on all servers that are in the server pools defined for the database. If you are using Oracle Automatic Storage Management (Oracle ASM) with Oracle Managed Files (OMF) 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.

You can convert an administrator-managed database to a policy-managed database using the `srvctl modify database -d` *`db_unique_name`* `-g` *`server_pool`* command. Conversely, 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 create a new administrator-managed database with the `srvctl add database` command.

> **See Also:** "SRVCTL Command Reference" on page A-11 for more information about these commands

With Oracle RAC 11*g* release 2 (11.2), you define a database as a resource in Oracle Clusterware. The resource is automatically created when you create your database with DBCA or the resource can be manually created by adding your database with srvctl. This 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 VIP, which means that the resource tries to start the VIP for the node when the database instance starts. If the VIP does not start successfully, then the instance still starts. When you review the database resource for an administrator-managed database, you will 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

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).

> **See Also:** "Service Deployment Options" on page 4-3 for more information about managing services

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. It has both the Database Control and Grid Control graphical user interfaces (GUIs) for managing single-instance and Oracle RAC environments. Because there is one Oracle Enterprise Manager Agent on each node of an Oracle RAC database, for Database Control you can use any URL for that database to administer it with Oracle Enterprise Manager.

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 the Console 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" on page 3-22 for advanced administration tasks not covered in *Oracle Database 2 Day + Real Application Clusters Guide*

Administering Oracle RAC with 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.

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.

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 the *Oracle Database Storage 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

Changing the SQL*Plus Prompt

To change the SQL*Plus prompt so that it includes the name of the current instance:

```
SET SQLPROMPT '_CONNECT_IDENTIFIER> '
```

This command replaces the "SQL" string in front of the greater than symbol (>) with the user variable _CONNECT_IDENTIFIER that displays the current instance name for the duration of your current session.

To change the prompt for all sessions automatically, add an entry similar to the following entry in your glogin.sql file, found in the SQL*Plus administrative directory:

```
SET SQLPROMPT '_CONNECT_IDENTIFIER> '
```

You may include any other required text or SQL*Plus user variable between the single quotes in the command.

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 single-instance 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.
`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.

Administering Oracle RAC with SRVCTL

Note: if you currently set environment variables other than Oracle home and Oracle SID in your session or profile when accessing the Oracle database, then use the `srvctl setenv` command to set them for SRVCTL.

The Server Control Utility (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 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). SRVCTL performs other operations, such as starting and stopping instances, by configuring and managing Oracle Clusterware resources, that define agents that perform database startup and shutdown operations using Oracle Call Interface APIs.

See Also: Appendix A, "Server Control Utility Reference" for more information about SRVCTL

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 up and shut down all of the instances in an Oracle RAC database with a single step.

You can only perform certain operations when the database is in a `NOMOUNT` or `MOUNT` state. Performing other operations requires that the database be `OPEN`. In addition, some operations require that only one instance be in the required state, while other operations require that all of the instances be in an identical state.

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

The procedures in the following sections assume that you are using a server parameter file (SPFILE):

- Starting and Stopping with Oracle Enterprise Manager
- Starting Up and Shutting Down with SQL*Plus
- Starting Up and Shutting Down with SRVCTL

Before you can start an Oracle RAC instance, your clusterware and any required operating system-specific processes must be running. For more information about these processes, see your operating system documentation.

The procedure for shutting down Oracle RAC instances is identical to shutting down instances in single-instance Oracle, with the exceptions described here. See the *Oracle Database Administrator's Guide* for more information about shutting down Oracle databases.

- 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. The 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.
- `The SHUTDOWN TRANSACTIONAL` command with the `LOCAL` option is useful to shutdown an instance after all active transactions on the instance have either committed or rolled back. This is in addition to what this command does for `SHUTDOWN IMMEDIATE`. 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 the shutdown was issued either commit or rollback.

Starting and Stopping with Oracle Enterprise Manager

See the *Oracle Database 2 Day + Real Application Clusters Guide* for step-by-step instructions to start or stop a cluster database instance or a cluster database.

Starting Up and Shutting Down with 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.

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, use the `SYSASM` privilege instead of the `SYSDBA` privilege to connect to and administer the Oracle ASM instances. See the *Oracle Database Storage 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.

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 an alias to a service or with multiple addresses, you may not be connected to your intended instance.

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
```

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

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

It is not possible to start or stop multiple instances 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 shut down keywords, such as `NOMOUNT`, `MOUNT`, `IMMEDIATE`, and so on

Starting Up and Shutting Down with SRVCTL

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:

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

```
$ srvctl start instance -d db_unique_name -i instance_name_list [-o start_options]
```

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

For policy-managed databases, enter a single node name:

```
$ srvctl start instance -d db_unique_name -n node_name [-o 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 -d db_unique_name [ -i "instance_name_list" |
  -n node_name ] [ -o 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 -d orcl -i "orcl3,orcl4" -o immediate
```

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 -d db_unique_name [-o start_options]

$ srvctl stop database -d 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 -d orcl -o mount
```

See Also: Appendix A, "Server Control Utility Reference" for information about SRVCTL options and information about other administrative tasks that you can perform with SRVCTL

Verifying That Instances are Running

To verify that instances are running, on any node from a SQL*Plus prompt enter the following, where *password* is the password:

```
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 active transactions of the session are rolled back, and resources held by the session (such as locks and memory areas) are immediately released and available to other sessions.

Using this statement enables you to maintain strict application service-level agreements in Oracle RAC environments. Often, the goal of a service-level agreement is to execute 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.

To terminate sessions, follow these steps:

- Query the value of the INST_ID column in the GV$SESSION dynamic performance view to identify which session to terminate
- Issue the ALTER SYSTEM KILL SESSION and specify the session index number (SID) and serial number of a session that you identified with the GV$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.

The following examples provide a three scenarios in which a user identifies and terminates a specific session.

Example 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 SYSDBA first queries the GV$SESSION view for the SCOTT user's session to identify the session to terminate, and then issues the ALTER SYSTEM KILL SESSION statement to terminate the session on the instance INST_ID=2. 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 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 Using the IMMEDIATE parameter

The following example include 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>
```

See Also: *Oracle Database Administrator's Guide* for more information about terminating sessions

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 an Oracle ASM disk group, a **cluster file system**, or a shared raw device. 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 SPFILE file 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.

If you include the `FROM MEMORY` clause (for example, `CREATE PFILE FROM MEMORY` or `CREATE SPFILE FROM MEMORY`), 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 alter 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
```

> **Note:** The value before the dot in an SPFILE entry identifies the instance to which the particular parameter value belongs. When an asterisk precedes the dot, 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=MEMORY;
```

Then if you run the following statement on another instance, the instance with sid `prod1` also assumes the new setting of `2000`:

```
ALTER SYSTEM SET OPEN_CURSORS=2000 sid='*' SCOPE=MEMORY;
```

In the following example, the server parameter file contains these entries:

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

Issue the following statement to make Oracle Database disregard the first entry from the server parameter file:

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

Issue the following statement to reset a 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.

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

1. `$ORACLE_HOME/dbs/spfile`*`sid`*`.ora`
2. `$ORACLE_HOME/dbs/spfile.ora`
3. `$ORACLE_HOME/dbs/init`*`sid`*`.ora`

On Windows platforms, the search order is as follows:

1. `%ORACLE_HOME%\database\spfile`*`sid`*`.ora`
2. `%ORACLE_HOME%\database\spfile.ora`
3. `%ORACLE_HOME%\database\init`*`sid`*`.ora`

Note: Oracle recommends that you do not use the default SPFILE names because all instances need to 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 default 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='?/dbs/initdbname.ora'
FROM SPFILE='/dev/vx/rdsk/oracle_dg/dbspfile'
```

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, and the *Oracle Database SQL Language Reference* for more information about the `CREATE SPFILE` 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 *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 is 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	Sets the number of instances in your Oracle RAC environment. A proper setting for this parameter can improve memory use. Set the CLUSTER_DATABASE_INSTANCES parameter to the *same* value on all instances. Otherwise, instance startup can fail. Normally, set this parameter to be equal to the number of instances in your Oracle RAC database. Alternatively, you can set this parameter to a value that is *greater than* the current number of instances if you are planning to add instances.
CLUSTER_INTERCONNECTS	Specifies an alternative cluster interconnect for the private network when there are multiple interconnects. The CLUSTER_INTERCONNECTS initialization parameter is used only in Linux and UNIX environments where UDP IPC is enabled. **Notes:** ■ Oracle recommends that all Oracle databases and Oracle Clusterware use the same interconnect network. If an Oracle database runs on a separate network from Oracle Clusterware, then the timeout on error is 900s. ■ Oracle does not recommend setting the CLUSTER_INTERCONNECTS parameter except in certain situations. See "Administering Multiple Cluster Interconnects on Linux and UNIX Platforms" on page 3-19 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.
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 Database Control or Grid Control. See *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.

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

Parameter	Description
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.
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 Database 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 Chapter 4, "Introduction to Automatic Workload Management" 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:** 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.
SESSIONS_PER_USER	Each instance maintains its own SESSIONS_PER_USER count. If SESSIONS_PER_USER is set to 1 for a user, the user can log on to the database more than once if each connection is from a different instance.
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. **See Also:** *Oracle Database SQL Language Reference* for more information about the INSTANCE clause

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:

ACTIVE_INSTANCE_COUNT
ARCHIVE_LAG_TARGET
COMPATIBLE
CLUSTER_DATABASE
CLUSTER_DATABASE_INSTANCE
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 will be used whenever the database runs on *node_name*.

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.

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" on page 3-19 for more information about the CLUSTER_INTERCONNECTS initialization parameter

Oracle Database uses the INSTANCE_NUMBER parameter to distinguish among instances at startup. Oracle Database uses the INSTANCE_NAME parameter to assign redo log groups to specific instances. To simplify administration, use the same number for both the INSTANCE_NAME and INSTANCE_NUMBER parameters. With Oracle Database 11.2 using Grid Plug and Play, the instance name parameter is no longer required. If you do not specify the instance name, then instance name defaults to db_unique_name_*number*.

Specify the ORACLE_SID environment variable, which comprises the database name and the number of the INSTANCE_NAME assigned to the instance. When you specify UNDO_TABLESPACE with automatic undo management enabled, then set this parameter to a unique undo tablespace name for each instance.

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 Streams with your Oracle RAC database, the value should be greater than zero.
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 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.

Quiescing Oracle RAC Databases

The procedure for quiescing Oracle RAC databases is identical to quiescing a single-instance 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: See the *Oracle Database Administrator's Guide* for details on the quiesce database feature and the *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 where UDP IPC is enabled, you can use the CLUSTER_INTERCONNECTS initialization parameter to specify an alternative interconnect to the one Oracle Clusterware is using for the private network.

Oracle does not recommend setting the CLUSTER_INTERCONNECTS parameter, which overrides the default interconnect settings at the operating system level. Instead, the best practice is to use operating system bonding techniques (also referred to as NIC (network interface card) bonding). See your platform-specific Oracle RAC installation guide for information about setting up NIC bonding at the operating system level.

This section includes the following topics:

- Recommendations for Setting the CLUSTER_INTERCONNECTS Parameter
- Usage Examples for the CLUSTER_INTERCONNECTS Parameter

Recommendations for Setting the CLUSTER_INTERCONNECTS Parameter

Notes:

- Oracle does not recommend setting the CLUSTER_INTERCONNECTS parameter when using a policy-managed database.

  If you must set the CLUSTER_INTERCONNECTS parameter for a policy-managed database, then Oracle recommends that you use the `srvctl modify instance -n node_name -i instance_name` command for all servers in the server pool.
- 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:

- Due to operating system limitations, you cannot use NIC bonding to provide increased bandwidth using multiple network interfaces.
- The cluster is running multiple databases and you need the interconnect traffic to be separated.
- 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 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.
- 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.
- 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 instance 1 on node 1 lists the IP addresses of the `alt0:`, `fta0:`, and `ics0:` devices in that order, then the parameter for instance 2 on node 2 must list the IP addresses of the equivalent network adapters in the same order. See the examples in "Usage Examples for the CLUSTER_INTERCONNECTS Parameter" on page 3-20 for more information about setting multiple interconnects with this parameter.
- 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

Usage Examples for the CLUSTER_INTERCONNECTS Parameter

This section provides two examples for setting the CLUSTER_INTERCONNECTS parameter.

Example 1

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 as described here.

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
```

If you set multiple values for CLUSTER_INTERCONNECTS as in the preceding example, then Oracle Database uses all of the interconnects that you specify, 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.

Example 2

To use the network interface whose IP address is 129.34.137.212 for all GCS, GES, and IPQ IPC traffic, set the CLUSTER_INTERCONNECTS parameter as follows:

```
CLUSTER_INTERCONNECTS=129.34.137.212
```

Use the ifconfig or netstat command to display the IP address of a device. This command provides a map between device names and IP addresses. For example, to determine the IP address of a device, run the following command as the root user:

```
# /usr/sbin/ifconfig -a
fta0: flags=c63<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST,SIMPLEX>
      inet 129.34.137.212 netmask fffffc00 broadcast 129.34.139.255 ipmtu 1500

lo0:  flags=100c89<UP,LOOPBACK,NOARP,MULTICAST,SIMPLEX,NOCHECKSUM>
      inet 127.0.0.1 netmask ff000000 ipmtu 4096

ics0:  flags=1100063<UP,BROADCAST,NOTRAILERS,RUNNING,NOCHECKSUM,CLUIF>
      inet 10.0.0.1 netmask ffffff00 broadcast 10.0.0.255 ipmtu 7000

sl0:  flags=10<POINTOPOINT>

tun0: flags=80<NOARP>
```

In the preceding example, the interface fta0: has an IP address of 129.34.137.212 and the interface ics0: has an IP address of 10.0.0.1.

Customizing How Oracle Clusterware Manages Oracle RAC Databases

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 allow Oracle Database to manage the Oracle RAC instances. If you set the instance to manual and start it with third-party clusterware, do not use the third-party clusterware to monitor and restart database instances, 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 policy to define the degree of control. There are two policies: automatic, which is the default, and manual. The manual policy minimizes the database instance protection level and overrides the automatic policy.

These policies enable you to configure your system so that either Oracle Clusterware automatically restarts your Oracle RAC database when you restart your system, or you manually restart your Oracle RAC database. You can also use this procedure to configure your system to prevent Oracle Clusterware from auto-restarting failed database instances more than once.

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

Example 1 Display the Current Policy

For example, use the following command syntax to display the current policy where *db_unique_name* is the name of the database for which you want to change policies:

```
srvctl config database -d db_unique_name -a
```

Example 2 Change the Current Policy to Another Policy

Use the following SRVCTL command syntax to change the current policy to another policy where *policy_name* is the name of the new policy for the database (that was identified by *db_unique_name* in Example1):

```
srvctl modify database -d db_unique_name -y management_policy
```

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

Example 3 Specify a Policy for a New Database

When you add a new database using the SRVCTL command, you can use the `-y` option to specify the management policy, as shown in the following example where *db_unique_name* is the name of the database and *management_policy* is the name of the policy:

```
srvctl add database -d db_unique_name -y management_policy -o $ORACLE_HOME -d 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 policy, the Oracle Clusterware resource records the new value for the affected database.

> **See Also:** Appendix A, "Server Control Utility Reference" for more information about SRVCTL commands

Advanced Oracle Enterprise Manager Administration

You can install, configure, and monitor an Oracle RAC database from a single location using either Oracle Enterprise Manager Database Control or Oracle Enterprise Manager Grid 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" on page 11-1.

> **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 Grid Control to Discover Nodes and Instances
- Administering Jobs and Alerts in Oracle RAC

Using Oracle Enterprise Manager Grid Control to Discover Nodes and Instances

Discovering Oracle RAC database and instance targets in Oracle Enterprise Manager enables monitoring and administration from the console:

- **Database Control** does not require discovery because DBCA performs any necessary configuration while creating the database.
- **Grid Control** enables you to use the Oracle Enterprise Manager console interface to discover Oracle RAC database and instance targets.

  If the Grid Control agents are installed on a cluster that has an Oracle RAC database, 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 Grid 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 single-instance databases and listeners, or to discover all cluster databases, single-instance databases, and listeners on the cluster, then click **Continue**.

   Oracle Enterprise Manager performs discovery to locate and display the cluster database and its associated instances. The Targets Discovered on Cluster page appears. If this procedure did not discover your reconfigured cluster database and all of its instances, you can use this page to manually configure your cluster databases and single-instance 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.

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. In the event of a failure, 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 panels 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

Performing Scheduled Maintenance Using Defined Blackouts in Oracle Enterprise Manager

You can define blackouts 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

Introduction to Automatic Workload Management

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:

Overview of Automatic Workload Management

Automatic workload management enables you to manage workload distributions to provide optimal performance for users and applications. Automatic workload management comprises the following:

- **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 and route work to the optimal instances that are assigned to offer the service.
- **Connection Load Balancing**: A feature of Oracle Net Services that balances incoming connections across all of the instances that provide the requested database service.

- **High Availability Framework**: An Oracle RAC component that enables Oracle Database to always maintain components in a running state.
- **Fast Application Notification (FAN)**: The notification mechanism that Oracle RAC uses to quickly alert applications about cluster state changes and workload service level changes, such as `UP` and `DOWN` events for instances, services, or nodes.
- **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 policy that you have defined for that service.
- **Automatic Workload Repository (AWR)**: Tracks service level statistics as metrics. Server generated alerts can be placed on these metrics when they exceed or fail to meet certain thresholds. You can then respond, for example, by changing the priority of a job, stopping overloaded processes, or by modifying a service level requirement, enabling you to maintain continued service availability despite service level changes. You can configure the service level for one service to have priorities relative to other services, and you can also configure:
  - The measurement of service quality
  - Event notification and alert mechanisms to monitor service quality changes
  - Recovery scenarios for responses to service quality changes

  The Automatic Workload Repository ensures that the Oracle Clusterware workload management framework and 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.

  See Also: *Oracle Database Performance Tuning Guide* for details about the Automatic Workload Repository and *Oracle Database PL/SQL Packages and Types Reference* for details about Oracle Database packages

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

When a user or application connects to a database, Oracle recommends that you specify a service in the connect data portion of the connect string. Oracle Database automatically creates one database service when the database is created. For many installations, this may be all you need. To enable more flexibility in the management of the workload using the database, Oracle Database enables you to create multiple

services and specify which instances offer the services. Continue reading this chapter to understand the added features that you can use with services if you are interested in greater workload management flexibility.

> **Note:** The features discussed in this chapter do not work with the default database service. You must create cluster managed services to take advantage of these features. You can only manage services that you create. Any service created by the database server is managed by the database server.

You can deploy Oracle RAC and single-instance Oracle database environments to use automatic workload management 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 this chapter describes. The following section describes the various service deployment options.

Automatic Workload Repository

The **Automatic Workload Repository (AWR)** tracks service level statistics as metrics. Server generated alerts can be placed on these metrics when they exceed or fail to meet certain thresholds. You can then respond, for example, by changing the priority of a job, stopping overloaded processes, or by modifying a service level requirement, enabling you to maintain continued service availability despite service level changes. You can configure the service level for one service to have priorities relative to other services, and you can also configure:

Service Deployment Options

This section describes the following service deployment topics:

- Using Oracle Services
- Default Service Connections
- Connection Load Balancing

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 be 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.

You can define services for both policy-managed and administrator-managed databases.

> **See Also:** "Oracle RAC Database Administration" on page 3-1 for more information about policy-managed and administrator-managed databases

- **Policy-managed database**: When you define services for a policy-managed database, you define the service to a server pool where the database is running. 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). 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.
- **Administrator-managed database**: When you define a service for an administrator-managed database, you define which instances normally support that service. These are known as the `PREFERRED` instances. You can also define other instances to support a service if the service's preferred instance fails. These are known as `AVAILABLE` instances.

  When you specify `PREFERRED` instances, you are specifying the number of instances on which a service normally runs. 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. You cannot control which `AVAILABLE` instance to which Oracle Clusterware relocates the services if there are multiple instances in the list. 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 by using FAN callouts.

When you define a service, you can also define the management policy for that service. You can choose either an *automatic* or a *manual* management policy.

- **automatic**: The service always starts when the database starts.

  > **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.

- **manual**:Requires that you start the service manually after the database starts. Prior to Oracle RAC 11*g* release 2 (11.2), all services worked as though they were defined with a manual management policy.

If you configured Oracle Data Guard in your environment, you can define a role for each service. When you specify a role for a service, Oracle Clusterware automatically starts it only when the database role matches the role you specified for the service. Valid roles are `PRIMARY`, `PHYSICAL_STANDBY`, `LOGICAL_STANDBY`, and `SNAPSHOT_STANDBY`.

See Also:

- *Oracle Data Guard Concepts and Administration* for more information about database roles
- "Creating Services with SRVCTL" on page 4-31 for more information

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.

Services are integrated with Resource Manager, which enables you to restrict the resources that are used by the users who connect with a service in an instance. The Resource Manager enables you to map a consumer group to a service so that users who connect with the service are members of the specified consumer group. Also, the Automatic Workload Repository (AWR) enables you to monitor performance by service.

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 define the method you want the listener to use for load balancing by setting the connection load balancing goal, `CLB_GOAL`. You can also specify a single transparent application failover (TAF) policy for all users of a service by defining the `FAILOVER_METHOD`, `FAILOVER_TYPE`, and so on. (See the *Oracle Database Net Services Administrator's Guide* for more information about configuring TAF.)

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. FAN has two methods for publishing events to clients, the Oracle Notification Service (ONS) that is used by Java Database Connectivity (JDBC) clients including the Oracle Application Server, and Oracle Streams, and Advanced Queueing that is used by Oracle Call Interface and Oracle Data Provider for .NET (ODP.NET) clients. When using Advanced Queueing, you must enable the service to use the queue by setting `AQ_HA_NOTIFICATIONS` to true.

With run time connection load balancing, applications can use load balancing advisory events to provide better service to users. The Oracle JDBC, Oracle Universal Connection Pool (UCP) for Java, Oracle Call Interface, Connection Manager (CMAN), and ODP.NET 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. The load balancing advisory also recommends how much of the workload should be sent to that instance. In addition, Oracle Net Services provides connection load balancing to enable you to spread user connections across all of the instances that support a service. To enable the load balancing advisory, set the `GOAL` parameter on the service.

Distributed transaction processing applications have unique requirements. To make it easier to use Oracle RAC with global transactions, set the distributed transaction processing parameter on the service so that all tightly coupled branches of a distributed transaction processing transaction are run on the same instance.

See Also: "Services and Distributed Transaction Processing in Oracle RAC" on page 4-24 for more information about distributed transaction processing in Oracle RAC

Default Service Connections

A special Oracle database service is created by default for your Oracle RAC database. 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. The database also 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

Both of these internal services support all of the automatic workload management features. You cannot stop or disable either of these internal services.

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.

Connection Load Balancing

Oracle Net Services provides the ability to balance 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 balances the connection requests across the listeners. With server-side load balancing, the SCAN listener directs a connection request to the best instance currently providing the service by using the load balancing advisory. In an Oracle RAC database, client connections should use both types of connection load balancing.

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:

- **LONG**: Use the LONG connection load balancing method for applications that have long-lived connections. This is typical for connection pools and SQL*Forms sessions. LONG is the default connection load balancing goal. The following is an example of modifying a service, POSTMAN, with the srvctl utility to define the connection load balancing goal for long-lived sessions:

  ```
  srvctl modify service -d db_unique_name -s POSTMAN -j LONG
  ```

- **SHORT**: Use the SHORT connection load balancing method for applications that have short-lived connections. When using connection pools that are integrated with FAN, set the CLB_GOAL to SHORT. The following example modifies the service known as ORDER, using SRVTCL to set the goal to SHORT:

  ```
  srvctl modify service -d db_unique_name -s ORDER -j SHORT
  ```

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

- Configures and enables server-side load balancing
- Sets the local and remote listener parameters (**Note:** If you do not use DBCA, you should set the LOCAL_LISTENER and REMOTE_LISTENER database parameters manually, especially if you do not use port 1521.)
- Creates a sample client-side load balancing connection definition in the tnsnames.ora file on the server

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 TAF for the connection, then Oracle Database moves the session to 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 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. To define a TAF policy for a service, use the `srvctl` utility as in the following example:

```
$ srvctl modify service -d crm -s  gl.us.oracle.com -q TRUE -P BASIC
   -e SELECT -z 180 -w 5 -j LONG
```

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

Client-side load balancing is defined in your client connection definition by setting the parameter `LOAD_BALANCE=ON` (the default is `ON` for description lists). 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.

The SCAN listener redirects the connection request to the local listener of the instance that is least loaded and providing 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. When using SCAN, Oracle Net automatically load balances client connection requests across the three IP addresses you defined for the SCAN, except when using EZConnect. To see what services a listener supports, run the `lsnrctl services` command.

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 within which Oracle Net waits for a response from the listener.

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.

For Oracle Call Interface 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 Oracle Call Interface client is in seconds. The `sqlnet.ora` file affects all connections using this client. With Oracle Database 11*g* Release 2 (11.2), Oracle Net Services introduces the ability to add the `connect_timeout` and `retry_count` parameters to individual `tnsnames.ora` connection strings.

```
(CONNECT_TIMEOUT=10)(RETRY_COUNT=3)
```

The granularity is seconds. Oracle Net waits for 10 seconds to receive a response, after which it assumes a failure. Oracle Net goes through the address list three times before it returns a failure to the client.

Note: Do *not* configure the connection timeout in the `sqlnet.ora` file on the server.

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

Fast Application Notification

This section provides a detailed description of FAN under the following topics:

- Overview of Fast Application Notification
- Application High Availability with Services and FAN
- Managing Unplanned Outages
- Managing Planned Outages
- Fast Application Notification High Availability Events
- Using Fast Application Notification Callouts

See Also: "Oracle Clients That Are Integrated with Fast Application Notification" on page 4-14 for more information about specific client environments that you can use with FAN

Overview of Fast Application Notification

FAN is a 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. Applications can respond to FAN events and take immediate action. FAN `UP` and `DOWN` events can apply to instances, services, and 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 poll the database and detect a problem, applications can receive FAN events and react immediately. With FAN, in-flight transactions can be 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 Database JDBC, Oracle Database ODP.NET, and Oracle Database Oracle Call Interface. This includes applications that use TAF. The integrated Oracle clients must be Oracle Database 10*g* release 2 (10.2) or later to take advantage of the load balancing advisory FAN events. (See the *Oracle Database Net Services Administrator's Guide* for more information about configuring TAF.)
2. Applications can use FAN programmatically by using the JDBC and Oracle RAC FAN application programming interface (API) or by using callbacks with Oracle Call Interface 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.

For `DOWN` events, the disruption to the application can be minimized because sessions to the failed instance or node can be terminated. Incomplete transactions can be 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 can be created so that the application can immediately take advantage of the extra resources. Through server-side callouts, you can also use FAN to:

- 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 to shut down services when the number of available instances decreases, for example, if nodes fail
- Automate the fail back of a service to `PREFERRED` instances if needed

FAN events are published using ONS and an Oracle Streams Advanced Queuing. The publication mechanisms are automatically configured as part of your Oracle RAC installation.

The CMAN and Oracle Net Services listeners are integrated with FAN events, enabling the listener and CMAN to immediately de-register services provided by the failed instance and to avoid erroneously sending connection requests to failed instances.

If you use the service goal `CLB_GOAL_SHORT`, 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.

Application High Availability with Services and FAN

Oracle Database focuses on maintaining service availability. In Oracle RAC, Oracle 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.

Managing Unplanned Outages

You can assign services in Oracle RAC to one or more instances. 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 relocates the service to an available instance in the cluster. FAN events can occur at various levels within the Oracle Database architecture and are published through ONS and Advanced Queuing. You can also program notification using FAN callouts.

> **Note:** Oracle Database does not run Oracle RAC callouts with guaranteed ordering. Callouts are run asynchronously and they are subject to scheduling variabilities.

Notification occurs 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, Oracle RAC provides interfaces that relocate, disable, and enable services to minimize service disruption to application users. Once you complete the operation, you can return the service to normal operation.

Due to dependencies, if you manually shutdown your database, then all of your services 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.

Fast Application Notification High Availability Events

Table 4–1 describes the FAN event record parameters and the event types, followed by name-value pairs for the event properties. The event type is always the first entry and the timestamp is always the last entry as in the following example:

```
FAN event type: SERVICEMEMBER VERSION=1.0
service=mix1 database=ractest instance=rac1host=staih01 status=up
reason=FAILURE card=1 timestamp=2009-07-02 22:06:02
```

Table 4–1 Event Record Parameters and Descriptions

Parameter	Description
VERSION	Version of the event record. Used to identify release changes.
EVENT TYPE	SERVICE, SERVICE_MEMBER, DATABASE, INSTANCE, NODE, SRV_PRECONNECT. Note that Database and Instance types provide the database service, such as DB_UNIQUE_NAME.DB_DOMAIN.
DATABASE UNIQUE NAME	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.

Table 4–1 (Cont.) Event Record Parameters and Descriptions

Parameter	Description
NODE NAME	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 service in DBA_SERVICES.
STATUS	Values are UP, NODEDOWN, NOT_RESTARTING, PRECONN_UP, PRECONN_DOWN, and UNKNOWN.
REASON	Failure, Dependency, User, Autostart, Restart.
CARDINALITY	The number of service members that are currently active; included in all UP events.
INCARNATION	For NODEDOWN events; the new cluster incarnation.
TIMESTAMP	The local time zone to use when ordering notification events.

A FAN record matches the database signature of each session as shown in Table 4–2.

Table 4–2 FAN Parameters and Matching Database Signatures

FAN Parameter	Matching Oracle Database Signature
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')

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 the following 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 directory *Grid_home*/racg/usrco on every node that runs Oracle Clusterware. If you are using scripts, then set the shell as the first line of the executable. The following is an example file for the *Grid_home*/racg/usrco/callout.sh callout:

```
#! /bin/ksh
FAN_LOGFILE= [your path name]/admin/log/`hostname`_uptime.log
echo $* "reported="`date` >> $FAN_LOGFILE &
```

The following output is from the previous example:

```
NODE VERSION=1.0 host=sun880-2 incarn=23 status=nodedown reason=
timestamp=08-Oct-2004 04:02:14 reported=Fri Oct 8 04:02:14 PDT 2004
```

Example callout scripts are available in the Oracle RAC Sample Code section on Oracle Technology Network at `http://www.oracle.com/technology/sample_code/products/rac/`.

> **See Also:** Table 4–1, " Event Record Parameters and Descriptions" for information about the callout and event details

A FAN record matches the database signature of each session, as shown in Table 4–2. The signature information is also available using `OCI_ATTRIBUTES`. These attributes are available in the Oracle Call Interface Connection Handle. Use this information to take actions on sessions that match the FAN event data.

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

Overview of the Load Balancing Advisory

Load balancing distributes work across all of the available Oracle RAC database instances. Oracle recommends that applications use persistent connections that span the instances that offer a particular service. 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 `THROUGHPUT` or `SERVICE_TIME` 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, and the ODP.NET Connection Pool. The load balancing advisory is also open for third-party subscription by way of the JDBC and Oracle RAC FAN API or through callbacks with the Oracle Call Interface.

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. This function enables the load balancing advisory for that service and FAN load balancing events are published. There are two types of service-level goals for run time:

- `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:

```
srvctl modify service -d db_unique_name -s OE -B SERVICE_TIME -j 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:

```
srvctl modify service -d db_unique_name -s sjob -B THROUGHPUT -j LONG
```

Setting the goal to NONE disables load balancing for the service. You can see the goal settings for a service in the data dictionary and in the DBA_SERVICES, V$SERVICES, and V$ACTIVE_SERVICES views.

> **See Also:** "Administering Services" on page 4-26 for more information about administering services and adding goals to services

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, ODP.NET, or Oracle Call Interface. Client applications can subscribe to these events directly by way of the ONS API. Table 4–3 describes the load balancing advisory FAN event parameters.

Table 4–3 Load Balancing Advisory FAN Events

Parameter	Description
VERSION	Version of the event record. Used to identify release changes.
EVENT TYPE	SERVICE, SERVICE_MEMBER, DATABASE, INSTANCE, NODE, ASM, SRV_PRECONNECT. Note that Database and Instance types provide the database service, such as DB_UNIQUE_NAME.DB_DOMAIN.
SERVICE	The service name; matches the service in DBA_SERVICES.
DATABASE UNIQUE NAME	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.

Use the following example to monitor load balancing advisory events:

```
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 ;
```

Oracle Clients That Are Integrated with Fast Application Notification

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.

You can use the Oracle Call Interface Session Pools, CMAN session pools, and JDBC and ODP.NET connection pools. Oracle Call Interface applications with TAF enabled should use FAN high availability events for fast failover. The overall goal is to enable applications to consistently obtain connections to available instances that provide the best service. 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 an instance that is 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. The use of connection pools and FAN requires that you have properly configured database connection load balancing across all of the instances that provide the service(s) that the connection pool is using. Oracle recommends that you configure both client-side and server-side load balancing with Oracle Net Services, which is the default when you use DBCA to create your database. 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
- 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
- Balance work requests at run time using load balancing advisory events

The next sections describe how to enable FAN events for the several specific client development environments:

- Enabling JDBC Clients for Fast Connection Failover
- Enabling Oracle Call Interface Clients for Fast Connection Failover
- Enabling Oracle Call Interface Clients for Run Time Connection Load Balancing
- Enabling ODP.NET Clients to Receive FAN High Availability Events
- Enabling ODP.NET Clients to Receive FAN Load Balancing Advisory Events

Enabling JDBC Clients for Fast Connection Failover

Enabling Fast Connection Failover (FCF) for the Oracle JDBC Universal Connection Pool (UCP) enables FAN high availability events and the load balancing advisory. Your application can use the JDBC development environment for either thick or thin JDBC clients to use FAN. For Java applications, Oracle recommends the UCP. The UCP is integrated to take advantage of Load Balancing Advisory information. You can use UCP, introduced in Oracle Database 11*g* Patchset 1 (11.1.0.7), with Oracle Database 10*g* or Oracle Database 11*g*.

To configure the JDBC client, set the `FastConnectionFailoverEnabled` property before making the first `getConnection()` request to a data source. When you enable

FCF, the failover applies to every connection in the connection cache. If your application explicitly creates a connection cache using the Connection Cache Manager, then you must first set `FastConnectionFailoverEnabled`.

JDBC application developers can now programatically 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 by enabling the code to respond to FAN events 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 ONS
- *Oracle Database 2 Day + Real Application Clusters Guide* for more information about configuring JDBC clients

Using FAN with Thick JDBC and Thin JDBC Clients

You can enable FCF Oracle's Implicit Connection Cache or UCP. Oracle recommends using the UCP for Java. The Implicit Connection Cache will be deprecated in the next release of Oracle Database. Enabling FCF with thin or thick JDBC clients enables the connection pool to receive and react to all FAN events.

This procedure explains how to enable FAN events for JDBC. For thick JDBC clients, if you enable FCF, do not enable TAF, either on the client or for the service. To enable FCF, you must first enable the UCP or the Implicit Connection Cache, as described in the following procedure:

1. On a cache enabled DataSource, set the DataSource property `FastConnectionFailoverEnabled` to `true` as in the following example to enable FAN for the Oracle JDBC Implicit Connection Cache:

```
OracleDataSource ods = new OracleDataSource()
...
ods.setConnectionCachingEnabled(True);
ods.setFastConnectionFailoverEnabled(True);
ods.setConnectionCacheName("MyCache");
ods.setConnectionCacheProperties(cp);
ods.setURL("jdbc:oracle:thin:@(DESCRIPTION=
   (LOAD_BALANCE=on)
   (ADDRESS=(PROTOCOL=TCP)(HOST=MYRACSCAN)(PORT=1521))
   (CONNECT_DATA=(service_name=service_name)))");
```

   Alternatively, you can use UCP for Java and set the `PoolDataSource` property for `FastConnectionFailoverEnabled` and the `ONSConfiguration`, as shown in the following syntax example. Applications must have both `ucp.jar` and `ons.jar` in their classpath.

```
PoolDataSource pds = PoolDataSourceFactory.getPoolDataSource();
pds.setONSConfiguration("nodes=racnode1:4200,racnode2:4200");
pds.setFastConnectionFailoverEnabled(true);
......
```

> **Note:** Use the following system property to enable FAN without making data source changes: `-D oracle.jdbc.FastConnectionFailover=true`.

2. Review the ONS and eONS configurations on each node that is running Oracle Clusterware as in the following example:

```
srvctl config nodeapps -s
```

> **Note:** ONS and eONS configuration should have been automatically completed during the Oracle Clusterware installation.

3. The information in the Oracle Cluster Registry (OCR) for ONS daemons is automatically configured for Oracle Database 10*g* and higher. If you are upgrading from an Oracle9*i* version of the database, then add ONS daemons to remote nodes (nodes outside the cluster), with the following command:

```
srvctl modify nodeapps -t host_port_list
```

4. Configure remote ONS subscription. Remote ONS subscription offers the following advantages:

   - Support for an All Java mid-tier software
   - An ONS 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

   When using remote ONS subscription for Fast Connection Failover, an application uses `setONSConfiguration`, using the string `remoteONSConfig`, on an Oracle DataSource instance with Implicit Connection Cache configured, as in the following example:

```
ods.setONSConfiguration("nodes=racnode1:6251,racnode2:6251");
```

   If you are using the Universal Connection Pool, then use the following example instead, where the port number used is the same as the remote port configured in Step 2:

```
pds.setONSConfiguration("nodes=racnode1:6251,racnode2:6251");
```

5. When you start the application, ensure that the `ons.jar` file is located on the application `CLASSPATH`. The `ons.jar` file is part of the Oracle client installation.

> **See Also:** *Oracle Database JDBC Developer's Guide* for more information about JDBC

Enabling Oracle Call Interface Clients for Fast Connection Failover

Oracle Call Interface clients can enable Fast Connection Failover by registering to receive notifications about Oracle RAC high availability FAN events and respond when events occur. This improves the session failover response time in Oracle Call Interface and also removes terminated connections from connection and session pools. This feature works on Oracle Call Interface applications, including those that use TAF, connection pools, or session pools.

First, you must enable a service for high availability events. This automatically populates the advanced queuing ALERT_QUEUE. If your application is using TAF, then enable the TAF settings for the service. Configure client applications to connect to an Oracle RAC database. Clients can 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, Oracle Call Interface:

- Terminates affected connections at the client and returns an error
- Removes connections from the Oracle Call Interface connection pool and the Oracle Call Interface 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.

Note: Oracle Call Interface does not manage UP events.

Perform the following steps to configure Fast Connection Failover with an Oracle Call Interface client:

1. Ensure that the service that you are using has Advanced Queuing notifications enabled by setting the services' values of AQ_HA_NOTIFICATIONS to TRUE. For example:

```
$ srvctl modify service -d crm -s gl.us.oracle.com -q TRUE -P BASIC
   -e SELECT -z 180 -w 5 -j LONG
```

2. Enable OCI_EVENTS at environment creation time on the client as follows:

```
( OCIEnvCreate(...) )
```

3. Client applications must link with the client thread or operating system library.
4. Optionally register a client EVENT callback

To see the alert information, you can query the views DBA_OUTSTANDING_ALERTS and DBA_ALERT_HISTORY.

See Also: *Oracle Call Interface Programmer's Guide* for more information about Oracle Call Interface and the *Oracle Database Net Services Administrator's Guide* for more information about configuring TAF

Enabling Oracle Call Interface Clients for Run Time Connection Load Balancing

As of Oracle Database 11*g* release 2 (11.2), Oracle Call Interface session pooling enables multiple threads of an application to use a dynamically managed set of pre-created database sessions. Oracle Database continually reuses the sessions in the pool to form nearly permanent channels to the instances, thus saving the overhead of creating and closing sessions every time applications need them.

To fully optimize session pools with Oracle RAC, sessions given to the requesting threads are appropriately mapped to instances based on the instance load. The Oracle Call Interface session pools use service metrics information received from the Load Balancing Advisory in Oracle RAC to allocate the sessions, resulting in better performance

Oracle Call Interface dynamically maps sessions to the threads based on the instance load. It uses service metrics provided by the Load Balancing Advisory to perform run time connection load balancing.[1] The number of connections can exceed the number of threads at certain times, but the Oracle Call Interface eventually adjusts this automatically, without any need for user intervention.

If the session pool supports services that span multiple instances, then the work requests are distributed across instances so that the instances providing better service are given more requests. For session pools that support services at only one instance, the first available session in the pool is adequate.

Run time connection load balancing is enabled by default in a Oracle Database 11*g* release 2 (11.2) (or higher) client talking to a server running Oracle Database 10*g* release 2 (10.2) (or higher). To disable run time connection load balancing, set the mode parameter to `OCI_SPC_NO_RLB` when calling `OCISessionPoolCreate()`.

To receive the service metrics based on the service time, ensure you have met the following conditions:

1. Link the application with the threads library.
2. Create the Oracle Call Interface environment in `OCI_EVENTS` and `OCI_THREADED` mode.
3. Enable event notification for the service for which the session pool is created.

   To enable event notification, use srvctl with the `-q` option (for AQ HA notifications) set to `TRUE`.
4. Modify the service to set up its Runtime Load Balancing Goal (with the `-B` option) and Connection Load Balancing Goal (with the `-j` option) as shown in the following example:

```
$ srvctl modify service -d crm -s  myService -B SERVICE_TIME -j SHORT
```

Enabling ODP.NET Clients to Receive FAN High Availability Events

ODP.NET connection pools can subscribe to 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 that stops and ODP.NET proactively disposes connections that are no longer valid. ODP.NET establishes connections to existing Oracle RAC instances if the removal of severed connections brings the total number of connections below the value that is set for the `MIN_POOL_SIZE` parameter.

The procedures for enabling ODP.NET are similar to the procedures for enabling JDBC in that you must set parameters in the connection string to enable Fast Connection Failover. Perform the following steps to enable FAN:

1. Enable Advanced Queuing notifications by using SRVCTL as shown in the following example:

```
$ srvctl modify service -d crm -s  gl.us.oracle.com -q TRUE -j LONG
```

2. Execute the following for the users that will be connecting by way of the .Net Application, where *`user_name`* is the user name:

```
EXECUTE DBMS_AQADM.GRANT_QUEUE_PRIVILEGE('DEQUEUE','SYS.SYS$SERVICE_METRICS',
user_name);
```

[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.

3. Enable Fast Connection Failover for ODP.NET connection pools by subscribing to FAN high availability events. Do this by setting the `ha events` connection string attribute to `true` at connection time. Note that this only works if you are using connection pools. In other words, do this if you have set the `pooling` attribute to `true`, which is the default. The following example shows this in more detail where *`user_name`* is the name of the user and *`password`* is the password:

```
// C#
using System;
using Oracle.DataAccess.Client;

class HAEventEnablingSample
{
  static void Main()
  {
    OracleConnection con = new OracleConnection();

    // Open a connection using ConnectionString attributes
    // Also, enable "load balancing"
    con.ConnectionString =
      "User Id=user_name;Password=password;Data Source=oracle;" +
      "Min Pool Size=10;Connection Lifetime=120;Connection Timeout=60;" +
      "HA Events=true;Incr Pool Size=5;Decr Pool Size=2";

    con.Open();

    // Create more connections and carry out work against the DB here.

    // Dispose OracleConnection object
    con.Dispose();
  }
}
```

Enabling ODP.NET Clients to Receive FAN Load Balancing Advisory Events

Use the following procedures to enable ODP.NET to receive FAN load balancing advisory events:

1. Enable Advanced Queuing notifications by using SRVCTL, and set the Connection Load Balancing Goal as shown in the following example:

```
$ srvctl modify service -d crm -s gl.us.oracle.com -q TRUE -j LONG -m BASIC -e
SELECT
```

2. Ensure Oracle Net Services is configured for connection load balancing.

3. Execute the following for the user that will be connecting by way of the .Net Application where *`user_name`* is the name of the user:

```
EXECUTE DBMS_AQADM.GRANT_QUEUE_PRIVILEGE('DEQUEUE','SYS.SYS$SERVICE_METRICS',
user_name);
```

4. 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:

```
// C#
using System;
using Oracle.DataAccess.Client;

class LoadBalancingEnablingSample
{
  static void Main()
  {
    OracleConnection con = new OracleConnection();

    // Open a connection using ConnectionString attributes
    // Also, enable "load balancing"
    con.ConnectionString =
      "User Id=user_name;Password=password;Data Source=oracle;" +
      "Min Pool Size=10;Connection Lifetime=120;Connection Timeout=60;" +
      "Load Balancing=true;Incr Pool Size=5;Decr Pool Size=2";

    con.Open();

    // Create more connections and carry out work against the DB here.

    // Dispose OracleConnection object
    con.Dispose();
  }
}
```

Note: ODP.NET does not support connection re-distribution when a node starts. However, if you have enabled failover on the server-side, then ODP.NET can migrate connections to available instances.

See Also:

- *Oracle Data Provider for .NET Developer's Guide* for more information about ODP. NET
- "srvctl modify service" in Appendix A, "Server Control Utility Reference".

Enabling Event Notification for Connection Failures in Oracle RAC

Event notification is enabled if the `SQL_ORCLATTR_FAILOVER_CALLBACK` and `SQL_ORCLATTR_FAILOVER_HANDLE` attributes of the `SQLSetConnectAttr` function are set when a connection failure occurs in an Oracle RAC database environment. The symbols for the new attributes are defined in the `sqora.h` file. The `SQL_ORCLATTR_FAILOVER_CALLBACK` attribute is used to specify the address of a routine to call when a failure event takes place.

The `SQL_ORCLATTR_FAILOVER_HANDLE` attribute is used to specify a context handle that is passed as a parameter in the callback routine. This attribute is necessary for the ODBC application to determine which connection the failure event is taking place on.

The function prototype for the callback routine is as follows:

```
void failover_callback(void *handle, SQLINTEGER fo_code)
```

The `handle` parameter is the value that was set by the `SQL_ORCLATTR_FAILOVER_HANDLE` attribute. Null is returned if the attribute has not been set.

The `fo_code` parameter identifies the failure event that is taking place. The failure events map directly to the events defined in the Oracle Call Interface programming interface. The list of possible events is as follows:

- `ODBC_FO_BEGIN`
- `ODBC_FO_ERROR`
- `ODBC_FO_ABORT`
- `ODBC_FO_REAUTH`
- `ODBC_FO_END`

The following is a sample program that demonstrates how to use this feature where *`user_name`* is the name of the user and *`password`* is the password:

```
/*
NAME
ODBCCallbackTest
DESCRIPTION
Program to demonstrate the connection failover callback feature.
PUBLIC FUNCTION(S)
main
PRIVATE FUNCTION(S)
NOTES
Command Line: ODBCCallbackTest filename [odbc-driver]
*/
#include <malloc.h>
#include <stdio.h>
#include <string.h>
#include <sql.h>
#include <sqlext.h>
#include <sqora.h>
#define TRUE  1
#define FALSE 0
/*
 * Funtion Prototypes
 */
void display_errors(SQLSMALLINT HandleType, SQLHANDLE Handle);
void failover_callback(void *Handle, SQLINTEGER fo_code);
/*
 * Macros
 */
#define ODBC_STS_CHECK(sts) \
if (sts != SQL_SUCCESS) \
{ \
display_errors(SQL_HANDLE_ENV, hEnv); \
display_errors(SQL_HANDLE_DBC, hDbc); \
display_errors(SQL_HANDLE_STMT, hStmt); \
return FALSE; \
}
/*
 * ODBC Handles
 */
SQLHENV *hEnv = NULL; // ODBC Environment Handle
SQLHANDLE *hDbc = NULL; // ODBC Connection Handle
SQLHANDLE *hStmt = NULL; // ODBC Statement Handle
/*
 * MAIN Routine
```

```
 */
main(int argc, char **argv)
{
SQLRETURN rc;
/*
 * Connection Information
 */
SQLTCHAR *dsn = "odbctest";
SQLTCHAR *uid = "user_name";
SQLTCHAR *pwd = "password";
SQLTCHAR *szSelect = "select * from emp";
/*
 * Allocate handles
 */
rc = SQLAllocHandle(SQL_HANDLE_ENV, SQL_NULL_HANDLE, (SQLHANDLE *)&hEnv);
ODBC_STS_CHECK(rc)
rc = SQLSetEnvAttr(hEnv, SQL_ATTR_ODBC_VERSION, (SQLPOINTER)SQL_OV_ODBC3, 0);
ODBC_STS_CHECK(rc);
rc = SQLAllocHandle(SQL_HANDLE_DBC, hEnv, (SQLHANDLE *)&hDbc);
ODBC_STS_CHECK(rc);
/*
 * Connect to the database
 */
rc = SQLConnect(hDbc, dsn, (SQLSMALLINT)strlen(dsn),
uid, (SQLSMALLINT)strlen(uid),
pwd, (SQLSMALLINT)strlen(pwd));
ODBC_STS_CHECK(rc);
/*
 * Set the connection failover attributes
 */
rc = SQLSetConnectAttr(hDbc, SQL_ORCLATTR_FAILOVER_CALLBACK, &failover_
callback,0);
ODBC_STS_CHECK(rc);
rc = SQLSetConnectAttr(hDbc, SQL_ORCLATTR_FAILOVER_HANDLE, hDbc, 0);
ODBC_STS_CHECK(rc);
/*
 * Allocate the statement handle
 */
rc = SQLAllocHandle(SQL_HANDLE_STMT, hDbc, (SQLHANDLE *)&hStmt);
ODBC_STS_CHECK(rc);
/*
 * Wait for connection failovers
 */
while (TRUE)
{
  sleep(5000);
  rc = SQLExecDirect(hStmt,szSelect, strlen(szSelect));
  ODBC_STS_CHECK(rc);
  rc = SQLFreeStmt(hStmt, SQL_CLOSE);
  ODBC_STS_CHECK(rc);
}
/*
 * Free up the handles and close the connection
 */
rc = SQLFreeHandle(SQL_HANDLE_STMT, hStmt);
ODBC_STS_CHECK(rc);
rc = SQLDisconnect(hDbc);
ODBC_STS_CHECK(rc);
rc = SQLFreeHandle(SQL_HANDLE_DBC, hDbc);
ODBC_STS_CHECK(rc);
```

```
rc = SQLFreeHandle(SQL_HANDLE_ENV, hEnv);
ODBC_STS_CHECK(rc);
return TRUE;
}
/*
 * Failover Callback Routine
 */
void failover_callback(void *Handle, SQLINTEGER fo_code)
{
switch (fo_code)
{
case ODBC_FO_BEGIN:
    printf("ODBC_FO_BEGIN recevied");
        break;
case ODBC_FO_ERROR:
    printf("ODBC_FO_ERROR recevied");
        break;
case ODBC_FO_ABORT:
        printf("ODBC_FO_ABORT recevied");
        break;
case ODBC_FO_REAUTH:
        printf("ODBC_FO_REAUTH recevied");
        break;
case ODBC_FO_END:
        printf("ODBC_FO_END recevied");
        break;
default:
        printf("Invalid or unknown ODBC failover code recevied");
        break;
};
return;
}
/*
 * Retrieve the errors associated with the handle passed
 * and display them.
 */
void display_errors(SQLSMALLINT HandleType, SQLHANDLE Handle)
{
SQLTCHAR MessageText[256];
SQLTCHAR SqlState[5+1];
SQLSMALLINT i=1;
SQLINTEGER NativeError;
SQLSMALLINT TextLength;
SQLRETURN sts = SQL_SUCCESS;
if (Handle == NULL) return;
/*
 * Make sure all SQLState text is null terminated
 */
SqlState[5] = '\0';
/*
 * Fetch and display all diagnostic records that exist for this handle
 */
while (sts == SQL_SUCCESS)
{
NativeError = 0;
TextLength = 0;
sts = SQLGetDiagRec(HandleType, Handle, i, SqlState, &NativeError,
(SQLTCHAR *)&MessageText, sizeof(MessageText),&TextLength);
if (sts == SQL_SUCCESS)
{
```

```
  printf("[%s]%s\n",NativeError, &MessageText);
  if (NativeError != 0)
  {
    printf("Native Error Code: %d", NativeError);
  }
  i++;
}
}
return;
}
```

Services and Distributed Transaction Processing in Oracle RAC

An XA transaction can span Oracle RAC instances by default, allowing any application that uses XA to take full advantage of the Oracle RAC environment to enhance the availability and scalability of the application.

This feature is controlled through the `GLOBAL_TXN_PROCESSES` initialization parameter, which is set to `1` by default. This parameter specifies the initial number of GTX*n* background processes for each Oracle RAC instance. Keep this parameter at its default value clusterwide to allow distributed transactions to span multiple Oracle RAC instances.

This 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. Tightly coupled XA transactions no longer require the special type of singleton services (that is, Oracle Distributed Transaction Processing (DTP) services) to be deployed on Oracle RAC database. XA transactions are transparently supported on Oracle RAC databases with any type of services configuration.

> **See Also:** *Oracle Database Advanced Application Developer's Guide* for complete information about using Oracle XA with Oracle RAC, and *Oracle Database Reference* for information about the `GLOBAL_TXN_PROCESSES` initialization parameter

To provide improved application performance with distributed transaction processing in Oracle RAC, you may want to take advantage of the specialized service referred to as a DTP Service. 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.

In addition, connection pools at the application server tier that load balance across multiple connections to an Oracle RAC database can use this method 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

To leverage all of the instances in a cluster, create one or more 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.

An external transaction manager, such as 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.

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 Advanced Application Developer's Guide* for more information about transaction branch management in Oracle RAC

Enabling Distributed Transaction Processing for Services

For services that you are going to use for distributed transaction processing, create the service using Oracle Enterprise Manager, or SRVCTL and define only one instance as the preferred instance. You can have as many `AVAILABLE` instances as you want. For example, the following SRVCTL command creates a singleton service for database `crm, xa_01.service.us.oracle.com`, whose preferred instance is `RAC01`:

```
$ srvctl add service -d crm -s xa_01.service.us.oracle.com -r RAC01 -a RAC02,
RAC03
```

Then mark the service for distributed transaction processing by setting the `DTP` parameter to `TRUE`; the default is `FALSE`. Oracle Enterprise Manager enables you to set this parameter on the Cluster Managed Database Services: Create Service or Modify Service page. You can also use SRVCTL to modify the DTP property of the singleton service, as follows:

```
$ srvctl modify service -d crm -s xa_01.service.us.oracle.com -x TRUE
```

If, for example, `RAC01` (that provides service `XA_01`) fails, then the singleton service that it provided fails over to another instance, such as `RAC02` or `RAC03`.

If services migrate to other instances after the cold-start of the Oracle RAC database, then you might have to force the relocation of the service to evenly re-balance the load on all of the available hardware. Use data from the `GV$ACTIVE_SERVICES` view to determine whether to do this.

Administering Services

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 and Oracle Database can measure the performance for each service. 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 each service's automatic workload management characteristics. The characteristics of a service include:

- A unique global name to identify the service.
- An Oracle Net Services name that a client uses to connect to the service.
- One or more roles for the service, which enable the service to be automatically started when the database starts if the role of the service matches that of the database, and the service has an automatic management policy.
- A management policy for the service, either AUTOMATIC to automatically start the service when the database starts, or MANUAL.
- For policy-managed databases, the name of the server pool in which the service runs, and whether the service is UNIFORM (the service can run on all active servers in the server pool) or SINGLETON (the service can run on exactly one server in the server pool).
- For administrator-managed databases, the preferred instances where you want the service to be enabled in a normal running environment and the available instances where the service can be enabled if a preferred instance fails.
- A service goal that 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), as determined by the load balancing advisory.
- An indicator that determines whether the service is used for distributed transactions.
- An indicator that determines whether Oracle RAC high availability events are sent to Oracle Call Interface and ODP.NET clients that have registered to receive them through Advanced Queuing.

  See Also: "Enabling Oracle Call Interface Clients for Fast Connection Failover" on page 4-16 for more details

- The characteristics of session failovers such as whether failovers occur, whether sessions can use pre-existing connections on the failover instance, and whether failed over sessions can continue to process interrupted queries. The service definition can also define the number of times that a failed session attempts to reconnect to the service and how long it should wait between reconnection attempts. The service definition can also include a connection load balancing goal that informs the listener how to balance connection requests across the instances that provide the service.
- The method for load balancing connections for each service. This method is used by the listener when Oracle Database creates connections. Connections are classified as `LONG` (such as connection pools and SQL*FORMS) which tells the

listener to use session based, or SHORT which tells the listener to use CPU based. If load balancing advisory is enabled, its information is used to balance connections otherwise CPU utilization is used.

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 to that database. You must manually restart the services when you start the database.
- Map a Service to a Consumer Group. Oracle Database can automatically 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, for example, if you are performing an upgrade and you want to prevent connection requests.
- 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.
- If you execute a SQL statement in parallel, the parallel processes only run on the instances that offer the service with which you originally connected to the database. This is the default behavior. This does not affect other parallel operations such as parallel recovery or the processing of GV$ queries. To override this behavior, set a value for the PARALLEL_INSTANCE_GROUP initialization parameter.

Notes:

- 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.
- Using service names to access a queue provides location transparency for the queue in an Oracle RAC database.
- 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 and SRVCTL

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:

- Administering Services with Oracle Enterprise Manager
- Administering Services with SRVCTL

Administering Services with Oracle Enterprise Manager

The Cluster Managed Database Services page is the master page for beginning all tasks related to services. To access this page, go to the Cluster Database Availability page, then click **Cluster Managed Database Services** in the Services section. You can use this page and drill down from this page to:

- View a list of services for the cluster
- View the instances on which each service is currently running
- 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

See Also: *Oracle Enterprise Manager* online help for more information about administering services with Oracle Enterprise Manager

Service-Related Tasks That You Can Perform with Oracle Enterprise Manager

You can perform service-related tasks as described for the following Oracle Enterprise Manager pages:

- Cluster Managed Database Services Page

- Cluster Managed Database Services Detail Page
- Create Services Page

Cluster Managed Database Services Page

The Cluster Managed Database Services page enables you to:

- View a list of services for the cluster, the instances on which each service is currently running, and the status for each service
- Start or stop a service, or enable or disable a service
- Access the Create Service and Edit Service pages
- View all the TNS strings for a service
- Access the Services Detail page to perform instance-level tasks for a service
- Test the connection for a service
- Delete a service

Cluster Managed Database Services Detail Page

The Cluster Managed Database Services Detail page enables you to:

- View the status of a service on all of its preferred and available instances; the status can be `Running`, `Stopped`, or `Disabled`
- View the attributes of the service
- Stop or start a service for an instance of a cluster database
- Disable or enable a service for an instance of a cluster database
- Relocate a service to manually re-balance the services load
- View performance charts for the service

Create Services Page

The Create Services page enables you to:

- Create the service with and specify its high availability configuration and service properties
- Specify whether or not the service should be started after it has been created, and whether or not the tnsnames.ora file should be updated with the new service information
- Select the desired service properties, such as:
  - TAF policy
  - Distributed transaction processing
  - Whether or not the Load Balancing Advisory should be enabled for the service
  - Connection Load Balancing Goal
  - Runtime Load Balancing Goal
  - Whether or not Fast Application Notification (FAN) for Oracle Call Interface and ODP.NET Applications should be enabled
  - Service threshold levels (for generating alerts with Enterprise Manager)
  - Resource management consumer group or job scheduler mappings

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

Accessing the Oracle Enterprise Manager Services Pages

To access the Cluster Managed Database Services page and detail pages for service instances:

1. On the Cluster Database Home page, click **Availability** to view the Availability subpage.
2. From the Cluster Database Availability page, under the Services heading, click **Cluster Managed Database Services**.

   The Cluster and Database Login page appears.
3. Enter credentials for the database and for the cluster that hosts the cluster database, then click **Continue**.

   The Cluster Managed Database Services page appears and displays services that are available on the cluster database instances. For information about performing tasks on this page, see the online help for this page.

   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`.
4. To manage a service at the instance level, either click a service name or select a service name, then select **Manage** from the Actions drop-down list and click **Go**.

   The Cluster Managed Database Services Detail page for the service appears. For information about performing tasks on this page, see the online help for this page.

To access the Relocate page:

1. Perform steps 1 - 3 from the previous procedure set.
2. From the Cluster Managed Database Services page, either click a service name or select a service name, then select **Manage** in the Actions drop-down list, then click **Go**.

   The Cluster Managed Database Services Detail page for the service appears.
3. Select the instance from which you want to move the service, then click **Relocate**.

   The Relocate Service from Instance page appears, where you can perform manual load balancing of services.

   For information about performing tasks on this page, see the online help for this page.

Administering Services with SRVCTL

When you create a service with SRVCTL, you must start it 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
- Starting and Stopping Services with SRVCTL

- Enabling and Disabling Services with SRVCTL
- Relocating Services with SRVCTL
- Obtaining the Statuses of Services with SRVCTL
- Obtaining the Configuration of Services with SRVCTL

> **See Also:** Appendix A, "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, enter a command from the command line using the following syntax:

```
srvctl add service -d db_unique_name -s service_name {-r preferred_list
[-a available_list]} | {-g server_pool [-c {UNIFORM | SINGLETON}] [-k net_number]}
[-P {BASIC | NONE}] [-l {[PRIMARY] | [PHYSICAL_STANDBY] | [LOGICAL_STANDBY] |
[SNAPSHOT_STANDBY]}] [-y {AUTOMATIC | MANUAL}] [-q {TRUE | FALSE}]
[-x {TRUE | FALSE}] [-j {SHORT | LONG}] [-B {NONE | SERVICE_TIME | THROUGHPUT}]
 [-e {NONE |SESSION | SELECT}] [-m {NONE | BASIC}] [-z failover_retries]
 [-w failover_delay]
```

> **Note:** The *service_name* parameter entry has a 4KB limit. Therefore, the total length of the names of all services assigned to an instance cannot exceed 4KB.

Starting and Stopping Services with SRVCTL

Enter the following SRVCTL syntax from the command line:

```
srvctl start service -d database_unique_name [-s service_name_list] [-i inst_name]
[-o start_options]

srvctl stop service -d database_unique_name -s service_name_list [-i inst_name]
[-o start_options]
```

Enabling and Disabling Services with SRVCTL

Use the following SRVCTL syntax from the command line to enable and disable services:

```
srvctl enable service -d database_unique_name -s service_name_list [-i inst_name]
srvctl disable service -d database_unique_name -s service_name_list [-i inst_name]
```

Relocating Services with SRVCTL

Run the `srvctl relocate service` command from the command line to relocate a service. For example, the following command relocates the `crm` service from instance apps1 to instance apps3:

```
srvctl relocate service -d apps -s crm -i apps1 -t apps3
```

Obtaining the Statuses of Services with SRVCTL

Run the `srvctl relocate service` command from the command line to obtain the status of a service. For example, the following command returns the status of the `crm` service that is running on the crm database:

```
srvctl status service -d apps -s crm
```

Obtaining the Configuration of Services with SRVCTL

Run the `srvctl relocate 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 `crm` service that is running on the `crm` database:

```
srvctl config service -d apps -s crm -a
```

> **See Also:** Appendix A, "Server Control Utility Reference" for information about other administrative tasks that you can perform with SRVCTL

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 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 Automatic Workload Repository (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_MONTIOR` PL/SQL package as follows:

```
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', ACTION_NAME => NULL);
```

Use the `DBA_ENABLED_AGGREGATIONS` view to verify that you have enabled monitoring.

Statistics aggregation and tracing by service are global in scope for Oracle RAC databases. In addition, they are persistent across instance restarts and service relocations for both Oracle RAC and single-instance 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 every five seconds. 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 ;
```

Service Thresholds and Alerts

Service level thresholds enable you to compare achieved service levels against accepted minimum required levels. 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.

You can explicitly specify two performance thresholds for each service: the response time for calls, or `SERVICE_ELAPSED_TIME`, and the CPU time for calls, or `SERVICE_CPU_TIME`. The response time goal indicates that the elapsed time should not exceed a certain value, and the response time represents wall clock time. 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.

The service time and CPU time are calculated as the moving average of the elapsed, server-side call time. The AWR monitors the service time and CPU time and publishes AWR alerts when the performance exceeds the thresholds. You can then respond to these alerts by changing the priority of a job, stopping overloaded processes, or by relocating, expanding, shrinking, starting or stopping a service. This permits you to maintain service availability despite changes in demand.

Example of Services and Thresholds Alerts

To check the thresholds for the payroll service, use the AWR report. You should record output from the report over several successive intervals during which time the system is running optimally. For example, assume that for an e-mail server, the AWR report runs each Monday during the peak usage times of 10:00 a.m. to 2:00 p.m. The AWR report would contain the response time, or DB time, and the CPU consumption time, or CPU time, for calls for each service. The AWR report would also provide a breakdown of the work done and the wait times that are contributing to the response times.

Using `DBMS_MONITOR`, set a warning threshold for the payroll service at 0.5 seconds and a critical threshold for the payroll service at 0.75 seconds. You must set these

thresholds at all instances within an Oracle RAC database. You can schedule actions using Oracle Enterprise Manager jobs for alerts, or you can schedule actions to occur programmatically when the alert is received. In this example, thresholds are added for the `servall` service and set 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 => 'servall');
```

Verify the threshold configuration 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. As an example, set the following:

- Under the `ERP` service, enable monitoring for the exceptions pay action in the `payroll` module.
- Under the `ERP` service, enable monitoring for the all actions in the `payroll` module.
- Under the `HOT_BATCH` service, enable monitoring for all actions in the `posting` module.

```
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', action_name => NULL);
EXECUTE DBMS_MONITOR.SERV_MOD_ACT_STAT_ENABLE(service_name => 'hot_batch',
module_name =>'posting', action_name => NULL);
```

Verify the enabled service, module, action configuration with 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 might appear as follows:

```
AGGREGATION           SERVICE               MODULE        ACTION
------------           --------------------   ----------    -------------
SERVICE_MODULE_ACTION  ERP                    PAYROLL       EXCEPTIONS PAY
SERVICE_MODULE_ACTION  ERP                    PAYROLL
SERVICE_MODULE_ACTION  HOT_BATCH              POSTING
```

5

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 RMAN 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.

> **Tip:** When should you perform a backup? Oracle recommends performing a backup immediately after running the `root.sh` script. Also, after you perform an Oracle RAC addition or deletion, you should back up the voting disk (or disks).

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 Recovery Manager. 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. In Oracle RAC, the snapshot control file is only needed on the nodes on which RMAN performs backups; the snapshot

control file does not need to be globally available to all instances in an Oracle RAC environment.

You can specify a **cluster file system** file or a raw device destination for the location of your snapshot control file. 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 globally sets the configuration for the location of the snapshot control file throughout your **cluster database**. Therefore, ensure that the directory `$ORACLE_HOME/dbs/scf` exists on 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. You can also specify a cluster file system file or a raw device destination for the location of your snapshot control file. This file is shared across all nodes in the **cluster** just like other data files in Oracle RAC.

> **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. This means that the default location for the SPFILE must be available to all nodes in your Oracle RAC database.

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 at 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 Channel" on page 5-3.

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 Channel

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 Channel

To configure one RMAN channel for each Oracle RAC instance, use the following syntax:

```
CONFIGURE CHANNEL DEVICE TYPE sbt CONNECT
'SYS/change_on_install@node1'
CONFIGURE CHANNEL DEVICE TYPE sbt CONNECT
'SYS/change_on_install@node2'
...
```

After this one-time configuration step, you can issue the `BACKUP` or `RESTORE` commands. In addition, you can use the `PARMS` command in this example to set vendor-specific parameters.

Managing Archived Redo Logs Using RMAN in Oracle RAC

When a node generates an archived redo log, Oracle Database always records the filename 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 filename on its file system, the file must be readable by any node that needs to access this archived redo log. For example, if node 1 archives a log to `/oracle/arc_dest/log_1_100_23452345.arc`, then node 2 can only 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 need to 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, one 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 this number.

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. This section illustrates the archived redo log naming issues for configuring archiving in your cluster database. The scenario described here is for a noncluster file system archiving scheme. Assume that the following conditions are met:

- Configure each node to write to a local archiving directory that is named the same on each node.
- Do *not* set up a cluster file system (in other words, each node can only read from and write to its own local file system). See the information about cluster file systems later in this chapter.
- Do not use NFS (network file system) or mapped drives to enable the nodes in the cluster to gain read/write access to one another.

Example 5–1 Example Configuration for the initialization parameters file

```
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)
```

Assume that the filenames of the archived redo logs are recorded in the control file as follows:

```
/arc_dest_1/log_1_62_23452345.arc
/arc_dest_2/log_2_100_23452345.arc
/arc_dest_2/log_2_101_23452345.arc
/arc_dest_3/log_3_70_23452345.arc
/arc_dest_1/log_1_63_23452345.arc
```

During recovery, as long as the archived log destinations are visible from the node that performs the recovery, 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 filename extension.

Table 5–1 Archived Redo Log Filename Format Parameters

Parameter	Description	Example
%r	Resetlogs identifier	log_1_62_23452345
%R	Padded resetlogs identifier	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 thread parameters, in either upper or lower case, are mandatory for Oracle RAC. This enables 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 filename 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 files.

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 with a different disk group for your recovery set than for your data files. Alternatively, you can use a cluster file system archiving scheme.

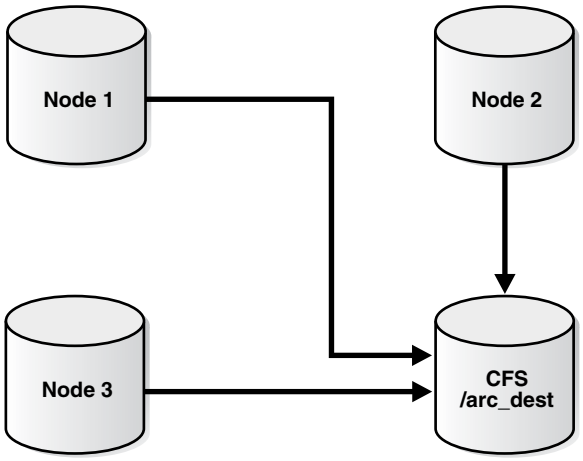
In this case, each node writes to a single cluster file system archived redo log destination and can read the archived redo log files of the other nodes. Read access is

achieved for all nodes with a cluster file system. For example, if node 1 archives a log 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.

> **Note:** The archive log naming format in this example is only for a cluster file system example. Oracle ASM uses an Oracle Managed Files (OMF)-based naming format.

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.

Figure 5–1 Cluster File System Archiving Scheme



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 filename 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. To configure this, set values for the `LOG_ARCH_DEST_`*n* parameter for each instance using the `sid` designator as in the following example:

```
sid1.LOG_ARCHIVE_DEST_1="LOCATION=/arc_dest"
sid2.LOG_ARCHIVE_DEST_1="LOCATION=/arc_dest"
sid3.LOG_ARCHIVE_DEST_1="LOCATION=/arc_dest"
```

The following list shows archived redo log entry examples that would appear in the RMAN catalog or the 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 node 3
/arc_dest/log_3_1563_23452345.arc <- thread 3 archived in node 2
```

```
/arc_dest/log_2_753_23452345.arc <- thread 2 archived in node 1
/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 directory `/arc_dest` in the cluster file system, each node can read the logs written by itself as well as any other node.

Noncluster File System Local Archiving Scheme

In the noncluster file system local archiving scheme, 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 archiving directories on the other nodes. In addition, if you are in recovery and if 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, if you want to distribute the backup processing onto multiple nodes, then the easiest method is to configure channels as described in the backup scenarios in Chapter 7, "Managing Backup and Recovery".

Initialization Parameter Settings for Noncluster File System Local Archiving

You can set the archiving destination values as follows in the initialization parameter file:

```
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"
```

The following list shows the possible archived redo log entries in the database control file. Note that any node is able to archive logs from any of the threads:

```
/arc_dest_1/log_1_1000_23435343.arc
/arc_dest_2/log_1_1001_23452345.arc <- thread 1 archived in node 2
/arc_dest_2/log_3_1563_23452345.arc <- thread 3 archived in node 2
/arc_dest_1/log_2_753_23452345.arc <- thread 2 archived in node 1
/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 5–2, each node 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.

Table 5–2 UNIX/NFS Location Log Examples, Noncluster File System Local Archiving

Node...	Can read 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 5–3.

Table 5–3 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.

Changing the Archiving Mode in Oracle RAC

You can run the ALTER DATABASE SQL statement to change the archiving mode in Oracle RAC as long as 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: 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.

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.

See Also: *Oracle Database Reference* for more information about the database views

6

Cloning Oracle RAC to Nodes in a Cluster

This chapter describes how to clone the grid infrastructure and 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 Chapter 8, "Using Cloning to Extend Oracle RAC to Nodes in the Same Cluster".

This chapter describes a noninteractive cloning technique that you implement through the use of 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 Grid 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

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 a new 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

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, and you install Oracle Clusterware.

Step 1 Install Oracle RAC

Use the detailed instructions in the *Oracle Real Application Clusters Installation Guide* 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, 11.2.0.*n*).
3. Apply one-off patches, if necessary.

Step 2 Create a backup of the source home

Create a copy of the Oracle RAC home. You will use this file to copy the Oracle RAC home to each node in the cluster (as described in "Cloning Oracle RAC to Nodes in a Cluster" on page 6-3).

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/11g/db_1
# tar -zcvf /pathname/db1120.tgz .
```

Step 3 Install and start Oracle Clusterware

Before you can use cloning to create a new Oracle RAC home, you must first install and start Oracle Clusterware on the node on 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

Cloning Oracle RAC to Nodes in a Cluster

After you complete the prerequisite tasks described in "Preparing to Clone Oracle RAC" on page 6-2, you can deploy cloned Oracle homes.

Deploying Oracle RAC Database Homes

Deploying the Oracle RAC database home to a cluster is a multiple-step process.

This section provides step-by-step instructions that describe how to:

1. Prepare the new cluster nodes
2. Deploy the Oracle RAC database software
3. Run the clone.pl script on each node
4. Run the $ORACLE_HOME/root.sh script on each node
5. Run DBCA to create the Oracle RAC instances on each node

Step 1 Prepare the new cluster nodes

Perform the Oracle RAC preinstallation steps, including such things as:

- Specify the kernel parameters.
- Use short, nondomain-qualified names for all names in the Hosts file.
- Verify that you can ping the public and interconnect names.
- Ensure Oracle Clusterware is active.
- Ensure that Oracle ASM is active and there at least one Oracle ASM disk group exists and is mounted.

See your platform-specific Oracle RAC installation guide for a complete preinstallation checklist.

Step 2 Deploy the Oracle RAC database software

To deploy the Oracle RAC software, you need to:

1. Restore the Oracle home to all nodes. For example:

```
[root@node1 root]# mkdir -p /opt/oracle/product/11g/db
[root@node1 root]# cd /opt/oracle/product/11g/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.

2. Change the ownership of all files to the `oracle` and `oinstall` group. For example:

```
[root@node1 db]# chown -R oracle:oinstall /opt/oracle/product/11g/db
```

> **Note:** You can perform this step at the same time you perform steps 3 and 4 to run the `clone.pl` and `ORACLE_HOME/root.sh` scripts on each cluster node.

Step 3 Run the clone.pl script on each node

To run the `clone.pl` script, which performs the main Oracle RAC cloning tasks, you must:

- Supply the environment variables and cloning parameters in the `start.sh` script, as described in Table 6–2 and Table 6–3. 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.
- Run the script as the `oracle` operating system user.

Table 6–1 lists and describes the `clone.pl` script parameters.

Table 6–1 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 not required.
`OSDBA_GROUP=`*`group_name`*	Specify the operating system group you want to use as the OSDBA privileged group.
`OSOPER_GROUP=`*`group_name`*	Specify the operating system group you want to use as the OSOPER privileged group.
`OSASM_GROUP=`*`group_name`*	Specify the operating system group you want to use as the OSASM privileged group.
`-O`	The `clone.pl` script passes anything following this flag to the Oracle Universal Installer command line.

Table 6–1 (Cont.) clone.pl Script Parameters

Parameter	Description
-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.

Example 6–1 shows an excerpt from the start.sh script that calls the clone.pl script.

Example 6–1 Excerpt From the start.sh Script to Clone Oracle RAC for Linux and UNIX

```
ORACLE_BASE=/opt/oracle
ORACLE_HOME=/opt/oracle/product/11g/db
cd $ORACLE_HOME/clone
THISNODE=`hostname -s`

E01=ORACLE_HOME=/opt/oracle/product/11g/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
```

Example 6–2 shows an excerpt from the start.bat script that the user must create that calls the clone.pl script.

Example 6–2 Excerpt From the start.bat Script to Clone Oracle RAC for Windows

```
set ORACLE_home=Oracle_home
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="-O'CLUSTER_NODES={node1,node2}'"
set C02="-O'LOCAL_NODE=%THISNODE%'"
perl clone.pl %E01% %E02% %E03% %C01% %C02%
```

Table 6–2 describes the environment variables E01, E02, and E03 that are shown in bold typeface in Example 6–1.

Table 6–2 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_NAME	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.

Table 6–3 describes the cloning parameters C01 and C02, that are shown in bold typeface in Example 6–1.

Table 6–3 Cloning Parameters Passed to the clone.pl Script.

Variable	Name	Parameter	Description
C01	Cluster Nodes	CLUSTER_NODES	Lists the nodes in the cluster.
C02	Local Node	LOCAL_NODE	The name of the local node.

Step 4 Run the $ORACLE_HOME/root.sh script on each node

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/11g/db/root.sh -silent
```

Note that you can run the script on each node simultaneously:

```
[root@node2 root]# /opt/oracle/product/11g/db/root.sh -silent
```

Ensure the script has completed on each node before proceeding to the next step.

Step 5 Run DBCA to create the Oracle RAC instances on each node

This step shows how to run the DBCA in silent mode and provide response file input to create the Oracle RAC instances.

The following example creates Oracle RAC instances on each node, registers the instances in the 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:

```
[oracle@node1 oracle]$ export ORACLE_HOME=/opt/oracle/product/11g/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 \
-nodeinfo node1,node2 -characterset WE8ISO8859P1 \
-obfuscatedPasswords false -sampleSchema true \
-oratabLocation /etc/oratab
```

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/cloneActions 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 prior to cloning as well as during the cloning operations.

- $ORACLE_HOME/clone/logs/error timestamp.log

  Contains information about errors that occur prior to cloning as well as during cloning operations.

Table 6–4 describes how to find the location of the Oracle inventory directory.

Table 6–4 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	Obtain the location from the Windows Registry key: HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\INST_LOC

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 Flash 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 Flash 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: the *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 its own local archived redo log files. For example, node 1 cannot access the archived redo log files on node 2 or node 3 unless you configure the network file system for remote access. If you configure a network file system file for backups, then each node will back up its archived redo logs to a local directory.

RMAN Restore Scenarios for Oracle RAC

This section describes the following common RMAN restore scenarios:

- Cluster File System Restore Scheme
- Noncluster File System Restore Scheme

- 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 single-instance scenarios.

Cluster File System Restore Scheme

The scheme that this section describes assumes that you are using the "Oracle Automatic Storage Management and Cluster File System Archiving Scheme" on page 5-5. In this scheme, assume that node 3 performed the backups to a CFS. If node 3 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 node 3 performed the backups but is unavailable, then configure a media management device for one of the remaining nodes and make the backup media from node 3 available to this node.

Noncluster File System Restore Scheme

The scheme that this section describes assumes that you are using the "Noncluster File System Local Archiving Scheme" on page 5-7. In this scheme, each node archives locally to a different directory. For example, node 1 archives to `/arc_dest_1`, node 2 archives to `/arc_dest_2`, and node 3 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 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 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.

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. You can also use callout programs as described in the example on Oracle Technology Network (OTN) to trigger application recovery.

Applications that were running continue by using failure recognition and recovery. This provides consistent and uninterrupted service in the event of hardware or software failures. 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, as long as one instance survives, 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 single-instance environments. See the Oracle Backup and Recovery documentation set for more information about single-instance RMAN backup procedures.

Channel Connections to Cluster Instances

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 `user1/pwd1@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 'user1/pwd1@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 'user1/pwd1@node1';
CONFIGURE CHANNEL 2.. CONNECT 'user2/pwd2@node2';
CONFIGURE CHANNEL 3.. CONNECT 'user3/pwd3@node3';
```

In the previous example, it is assumed that `node1`, `node2` and `node3` 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 'user1/pwd1@node1';
    ALLOCATE CHANNEL CH2 CONNECT 'user2/pwd2@node2';
    ALLOCATE CHANNEL CH3 CONNECT 'user3/pwd3@node3';
    BACKUP DATABASE PLUS ARCHIVED LOG;
}
```

During a backup operation, as long as at least one of the channels allocated has access to the archived log, 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 CFS archiving scheme, assuming that every node writes to the archived logs in the same CFS, 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 node1 instance has the database mounted while the node2 and node3 instances have the database open, then the backup fails.

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, 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 node 1 has faster read/write access to data files 7, 8, and 9 than the other nodes, then node 1 has greater node affinity to those files than nodes 2 and 3.

> **See Also:** *Oracle Database Backup and Recovery Reference* for more information about the `CONNECT` clause of the `CONFIGURE CHANNEL` statement

Deleting Archived Redo Logs after a Successful Backup

Assuming that you have configured the automatic channels as defined in section "Channel Connections to Cluster Instances" on page 7-3, 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, as long as at least one of the channels allocated has access to the archived log, RMAN will automatically schedule 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 CFS archiving scheme, assuming that every node writes to the archived logs on the same CFS, 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 needs to 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 node 1, while log 1002 is backed up to the drive attached to node 2. If you then allocate channels that connect to each node, then the channel connected to node 1 can restore log 1001 (but not 1002), and the channel connected to node 2 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 single-instance 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 this 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 to 0.

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 Flash Recovery Area in Oracle RAC

To use a flash 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 flash 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 flash 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 Flash Recovery Area section of the page.
4. Click **Help** on this page for more information.

8

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 Automatic Storage Management (Oracle ASM) and Oracle RAC to nodes in a new cluster, see Chapter 6, "Cloning Oracle RAC to Nodes in a Cluster".

> **See Also:** "Introduction to Cloning Oracle RAC" on page 6-1 for an overview of cloning and a discussion about the benefits of cloning

Adding Nodes Using Cloning in Oracle RAC Environments

This section explains how to add nodes to existing Oracle RAC environments by using Oracle cloning for Linux and UNIX system environments and Windows system 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" on page 6-6 for more information.

Complete the following steps to clone Oracle Database with Oracle RAC software on Linux and UNIX systems:

1. Follow the steps in the "Preparing to Clone Oracle RAC" on page 6-2 to create a copy of an Oracle home that you then use to perform the cloning procedure on one or more nodes.
2. If you do not have a shared Oracle Database home, then tar the Oracle home from the existing node and copy it to the new node. Assume that the location of the destination Oracle home on the new node is `$ORACLE_HOME`. Otherwise, skip this step.
3. If you do not have a shared Oracle Database home, then on the new node go to the `$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=/opt/oracle ORACLE_HOME=$ORACLE_HOME
 ORACLE_HOME_NAME=Oracle_home_name '-O-noConfig'
```

If you have a shared Oracle Database home, then append the `-cfs` option to the command example in this step for the cluster file system:

```
perl clone.pl '-O"CLUSTER_NODES={existing_node,new_node2,new_node3}"'
'-O"LOCAL_NODE=new_node2"' ORACLE_BASE=/opt/oracle ORACLE_HOME=$ORACLE_HOME
 ORACLE_HOME_NAME=Oracle_home_name '-O-noConfig'  '-O-cfs'
```

4. Run the following command on the existing node from the `$ORACLE_HOME/oui/bin` directory 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 node:

```
./runInstaller -updateNodeList ORACLE_HOME=$ORACLE_HOME "CLUSTER_
NODES={existing_node,new_node2,new_node3}"
```

5. On the new node, go to the `$ORACLE_HOME` directory and run the following command:

```
./root.sh
```

6. On the new node, run Net Configuration Assistant (NETCA) to add a listener.
7. From the node that you cloned, run Database Configuration Assistant (DBCA) to add the new instance.

Using Cloning to Add Nodes to Oracle RAC Environments on Windows Systems

Complete the following steps to clone Oracle Database with Oracle RAC software on Windows systems:

1. If you do *not* have a shared Oracle Database home, then zip the Oracle Database home with Oracle RAC on the existing node and copy it to the new node. Unzip the Oracle Database with Oracle RAC home 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%`. Skip this step if you have a shared Oracle Database 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, *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 '-O"CLUSTER_NODES={existing_node,new_node}"' '-O"LOCAL_
NODE=new_node"' -O-noConfig
```

If you have a shared Oracle Database home with Oracle RAC, then append the `-O-cfs` option to the command example in this step and provide a complete path location for the cluster file system.

3. On the existing node, from the `ORACLE_HOME\oui\bin` directory run the following command where *Oracle_home* is the Oracle Database home with Oracle RAC, *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. On the new node, run NETCA to add a listener.
5. From the node that you cloned, run DBCA to add the database instance to the new node.

9

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 Chapter 6, "Cloning Oracle RAC to Nodes in a 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: The phrase "target node" as used in this chapter refers to the node to which you plan to extend 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 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 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*`/oui/bin` directory on `node1` and run the `addNode.sh` script using the following syntax:

   ```
   $ ./addNode.sh -silent "CLUSTER_NEW_NODES={node2}"
   ```

2. Run the `root.sh` script on `node2` as `root`.

3. If you store your policy-managed database on Oracle Automatic Storage Management (Oracle ASM), **Oracle Managed Files (OMF)** is enabled, and if there is space in a server pool for `node2`, then `crsd` adds the Oracle RAC instance to `node2` and no further action is necessary. If OMF 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 `srvctl modify srvpool` 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 instance to `node2`.

4. If you have an administrator-managed database, then add a new instance on `node2` as described in "Adding Oracle RAC Database Instances to Target Nodes" on page 9-2.

Note: Oracle recommends that you back up your voting disk and Oracle Cluster Registry (OCR) files after you complete the node addition process.

Adding 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 under the following topics:

- Using DBCA in Interactive Mode to Add Database Instances to Target Nodes
- Using DBCA in Silent Mode to Add Database Instances to Target Nodes

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 Chapter 4, "Introduction to Automatic Workload Management".

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 correctly.
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 verify that the Oracle ASM instance is ready for addition to a node, as follows:

   ```
   cluvfy stage -pre storadd -s path [-t asm] [-verbose]
   ```

   Replace *`path`* with the directory path by which Oracle ASM is identified within the system.

   > **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 **Oracle Real Application Clusters database**, click **Next**, and DBCA displays the Operations page.
4. If your database is administrator-managed, select **Instance Management**, click **Next**, and DBCA displays the Instance Management page.

   If your database is policy-managed, then the **Instance Management** option is not available. To increase the number of database instances, add more nodes to the server pool.
5. 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`.
6. From the List of Cluster Databases page, select the active Oracle RAC database to which you want to add an instance. Enter user name and password for the database user that has `SYSDBA` privileges. 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.
7. Click **Next** to add a new instance and DBCA displays the Adding an Instance page.
8. 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 target node name from the list, click **Next**, and DBCA displays the Services Page.
9. Enter the services information for the target node's instance, click **Next**, and DBCA displays the Instance Storage page.
10. If you are using raw devices or raw partitions, then on the Instance Storage page select the Tablespaces folder and expand it. Select the undo tablespace storage object and a dialog appears on the right-hand side. Change the default data file name to the raw device name for the tablespace.

11. If you are using raw devices or raw partitions or if you want to change the default redo log group file name, then on the Instance Storage page select and expand the Redo Log Groups folder. For each redo log group number that you select, DBCA displays another dialog box.
12. If you are using a cluster file system, then click **Finish** on the Instance Storage page. If you are using raw devices, then repeat step 11 for all of the other redo log groups, click **Finish**, and DBCA displays a Summary dialog.
13. 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. When DBCA completes the instance addition operation, DBCA displays a dialog asking whether you want to perform another operation.
14. 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 target nodes:

    ```
    cluvfy comp admprv -o db_config -d oracle_home -n nodelist [-verbose]
    ```

15. Perform any needed service configuration procedures, as described in Chapter 4, "Introduction to Automatic Workload Management".

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 where *password* is the password:

```
dbca -silent -addInstance -nodeList node -gdbName gdbname [-instanceName instname]
 -sysDBAUserName sysdba -sysDBAPassword password
```

Table 9–1 Variables in the DBCA Silent Mode Syntax

Variable	Description
node	The node on which you want to add (or delete) the instance.
gdbname	Global database name.
instname	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.
password	Password for the SYSDBA user.

Before you run the dbca command, ensure that you have set the $ORACLE_HOME environment variable correctly on the existing nodes.

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 prior to removing the node from the cluster.

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. Alternatively, you can shrink a policy-managed database.

Deleting Policy-Managed Databases

Deleting a policy-managed database involves decreasing 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 -d db_unique_name -n node_name
$ srvctl relocate server -n node_name -g Free
```

The first command stops on the 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" on page 9-7 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, use either `srvctl` or Oracle Enterprise Manager to do the following:

- If you have services configured, 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 Oracle Enterprise Manager and SRVCTL" on page 4-28

The procedures in this section explain how to use DBCA in interactive or silent 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

This section includes the following topics:

- Using DBCA in Interactive Mode to Delete Instances from Nodes
- Using DBCA in Silent Mode to Delete Instances from Nodes

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 the 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 Welcome page select **Oracle Real Application Clusters Database**, click **Next**. DBCA displays the Operations page.

4. On the DBCA Operations page, select **Instance Management** and click **Next**. DBCA displays the Instance Management page.

5. On the DBCA Instance Management page, select the instance to be deleted, select **Delete Instance,** and click **Next**.

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.

   b. Enter a user name and password for the database user that has `SYSDBA` privileges. Click **Next**.

   c. 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. When DBCA completes this operation, DBCA displays a dialog asking whether you want to perform another operation.

      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 by querying the `V$LOG` view. 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 the OCR by running the following command:

   ```
   srvctl config database -d 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 9–1 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] -gdbName gdbname -instanceName
 instname -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 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
   ```

3. Depending on whether you have a shared or nonshared Oracle home, complete one of the following two procedures to remove the Oracle RAC software:

   - For a shared home, detach the node instead of deinstalling it by running the following command from the `$ORACLE_HOME/oui/bin` directory on each of the nodes that you want to delete:

     ```
     $ ./runInstaller -detachHome ORACLE_HOME=Oracle_home_location
     ```

   - For a nonshared home, deinstall the Oracle home from the node that you are deleting by running the following command from the `$ORACLE_HOME/deinstall` directory:

     See Also: *Oracle Clusterware Administration and Deployment Guide* for more information about deleting nodes

     ```
     $ ./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:

   ```
   $ ./runInstaller -updateNodeList ORACLE_HOME=Oracle_home_location
   "CLUSTER_NODES={remaining_node_list}"
   ```

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

10

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

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. For example, 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 is down in the eyes of your users.

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 losing one component cannot result in downtime for other hardware components 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 if fits in the service level requirements. It 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 that fit 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`

This section includes the following topics:

- 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

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:

- Remove TCP/IP timeouts 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 Chapter 4, "Introduction to Automatic Workload Management" 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.
- Place the OCR with I/O service times in the order of 2 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 the *Oracle Database 2 Day + Real Application Clusters Guide* for more information 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 use RMAN in Oracle RAC from Oracle Enterprise Manager Backup Manager or from a command line. See Chapter 5, "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 sequence number generation performance. 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`. This combination has the most negative effect on performance compared to other caching and ordering combinations.
- 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 database or Oracle RAC databases.

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 database—must be able to obtain CPU resources in a timely fashion and must be given higher priority in the system.

Oracle recommends that the number of real time LMS*n* processes 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 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 Global Cache Service Processes (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 fewer busy LMS*n* processes. Oracle Database calculates the number of processes when the instance is started, and you must restart the instance if you want to change the value.

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 be exactly 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, to 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:

- Chapter 6, "Cloning Oracle RAC to Nodes in a Cluster"
- Chapter 8, "Using Cloning to Extend Oracle RAC to Nodes in the Same Cluster"
- Chapter 9, "Adding and Deleting Oracle RAC from Nodes on Linux and UNIX 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 hash partitioned 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.

If you are using sequence numbers, then always use the `CACHE` option. If you use sequence numbers with the `CACHE` option, then:

- Your system may lose sequence numbers
- There is no guarantee of the ordering of the sequence numbers

> **Note:** If your environment cannot tolerate sequence number gaps, then consider pre-generating the sequence numbers or use the `ORDER` and `CACHE` options.

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 single-instance 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 single-instance 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.

To ensure this, create 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:

- "Services and Distributed Transaction Processing in Oracle RAC" on page 4-24 for more details about enabling services and distributed transactions
- *Oracle Database Advanced Application Developer's Guide* for more information about distributed transactions in Oracle RAC

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 in the event of 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 single instance 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 (DOP) 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

Oracle Database's parallel execution feature uses multiple processes to run SQL statements on one or more CPUs. Parallel execution is available on both single-instance 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 DOP assigned to each table or index.

> **See Also:** *Oracle Database Performance Tuning Guide* for more information about the query optimizer

Data Security Considerations in Oracle RAC

This section describes the following two Oracle RAC security considerations:

- Transparent Data Encryption and Wallets
- Windows Firewall Considerations

Transparent Data Encryption and Wallets

Wallets used by Oracle RAC instances for Transparent Data Encryption may be one of: a local copy of a common wallet, a common wallet stored on shared media that is directly accessed, or a hardware-security module (HSM) based wallet. In every case, you must open the wallet on each Oracle RAC instance before transparent data encryption can be used.

Consider the following when setting up the wallet on Oracle RAC:

1. Ensure that only one Oracle RAC instance sets the Master Key. Once the Master Key is set, manually copy the wallet to all instances of the Oracle RAC database to the locations indicated by the `sqlnet.ora` file of each instance. Reopen the wallet on all of the instances after completing the copying of the wallet. Only after all of the Oracle RAC instances have reopened the wallet can you run a `Master Key Rekey` command.
2. Ensure that only one Oracle RAC instance issues the `Master Key Rekey` command at any given time. After the instance has issued the `Master Key Rekey` command, copy the wallet to all instances of the Oracle RAC database. Again, reopen the wallet after it has been copied. Only after all the instances have reopened the wallet can you run another `Master Key Rekey` command.
3. Do not issue any wallet `open` or `close` commands while setting up or changing the Master Key.
4. A wallet must be opened on all instances of the Oracle RAC database in order to have the wallet open in Oracle RAC.
5. A wallet must be closed on all instances of the Oracle RAC database in order to have the wallet closed in Oracle RAC.

Deployments where no shared storage exists require that each Oracle RAC node maintain a local wallet.

As an alternative to copying the wallet to each node, you can use a hardware-security module (HSM) device to store the master encryption key. HSM devices are recommended over shared storage on a network drive, because an HSM device automatically clones the master key.

> **Note:** For shared access to a key, Oracle recommends using HSM devices. Oracle does not recommend using a shared copy residing on a network file system because when one instance re-keys the master key, other instances are not notified about the change and use the old master key, which can no longer decrypt the column keys. If you need to re-key an Oracle RAC environment, you must copy the wallet to all of the remaining instances and then close and reopen the wallet on each instance.

After you create and provision a wallet on a single node, you must copy the wallet and make it available to all of the other nodes, as follows:

- For systems using Transparent Data Encryption with encrypted wallets, you can use any standard file transport protocol, though Oracle recommends using a secured file transport.
- For systems using Transparent Data Encryption with obfuscated wallets, file transport through a secured channel is recommended.

To specify the directory in which the wallet must reside, set the `WALLET_LOCATION` or `ENCRYPTION_WALLET_LOCATION` parameter in the `sqlnet.ora` file. The local copies of the wallet need not be synchronized for the duration of Transparent Data Encryption usage until the server key is re-keyed though the `ALTER SYSTEM SET KEY` SQL statement. Each time you issue the `ALTER SYSTEM SET KEY` statement on a database instance, you must again copy the wallet residing on that node and make it available to all of the other nodes. Then, you must close and reopen the wallet 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 Administrator's Guide* for more information about creating and provisioning a wallet

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.

This section includes these topics:

- Oracle Executables Requiring Firewall Exceptions
- Configuring the Windows Firewall
- Troubleshooting Windows Firewall Exceptions

Oracle Executables Requiring Firewall Exceptions Table 10–1 lists the Oracle Database 10*g* release 1 (10.1) or later executables that listen on TCP ports on Windows. If they are in use and accepting connections from a remote client computer, then Oracle recommends that you add them to the Windows Firewall exceptions list to ensure correct operation. Except as noted, they can be found in *ORACLE_HOME*\bin.

The RMI registry application and daemon executable listed in Table 10–1 are used by Oracle Ultra Search to launch a remote crawler. They must be added to the Windows Firewall exception list if you are using the Ultra Search remote crawler feature, and if the remote crawler is running on a computer with the Windows Firewall enabled.

Note: If multiple Oracle homes are in use, then several firewall exceptions may be needed for the same executable: one for each home from which that executable loads.

Table 10–1 Oracle Executables Requiring Windows Firewall Exceptions

File Name	Executable Name
Grid_home\bin\crsd.exe	OracleCRService
Grid_home\bin\evmd.exe	OracleEVMService
Grid_home\bin\evmlogger.exe	Event manager logging daemon
Grid_home\bin\GuiOracleObjManager.exe	Oracle Object Link Manager (GUI version)
Grid_home\bin\ocssd.exe	OracleCSService
Grid_home\bin\OracleObjManager.exe	Oracle Object Link Manager (CLI version)
Grid_home\bin\racgvip.exe	Virtual Internet Protocol Configuration Assistant
Grid_home\cfs\Ocfsfindvol.exe	Oracle Cluster Volume Service
dgmgrl.exe	Data Guard Manager
emagent.exe	Oracle Database Control
extproc.exe	External Procedures
hdodbc.exe	Generic Connectivity
oidldapd.exe	Oracle Internet Directory LDAP Server
omtsreco.exe	Oracle Services for Microsoft Transaction Server
oracle.exe	Oracle Database
ORACLE_HOME\apache\apache\apache.exe	Apache Web Server
ORACLE_HOME\bin\emagent.exe	Oracle Enterprise Manager Agent
ORACLE_HOME\bin\omtsreco.exe	Oracle Services for Microsoft Transaction Server
ORACLE_HOME\bin\ORACLE.EXE	Oracle
ORACLE_HOME\bin\racgimon.exe	RACG
ORACLE_HOME\bin\TNSLSNR.exe	Oracle listener
ORACLE_HOME\jdk\bin\java.exe	Java Virtual Machine
ORACLE_HOME\jdk\bin\rmid.exe	RMI daemon executable
ORACLE_HOME\jdk\bin\rmiregistry.exe	RMI registry application

Table 10–1 (Cont.) Oracle Executables Requiring Windows Firewall Exceptions

File Name	Executable Name
ORACLE_HOME\jdk\jre\bin\rmiregistry.exe	RMI registry application
ORACLE_HOME\opmn\bin\ons.exe	Oracle Notification Service
ORACLE_HOME\opmn\bin\opmn.exe	Oracle Process Manager
pg4arv.exe	Oracle Procedural Gateway for APPC
pg4mqc.exe	Oracle Procedural Gateway for Websphere MQ
pg4mqs.exe	Oracle Procedural Gateway for Websphere MQ
pg4t4ic.exe	Oracle Procedural Gateway for APPC
tg4drsrv.exe	Oracle Transparent Gateway for DRDA
tg4msql.exe	Oracle Transparent Gateway for MS-SQL Server
tg4sybs.exe	Oracle Transparent Gateway for SYBASE
tg4tera.exe	Oracle Transparent Gateway for Teradata
tnslsnr.exe	Oracle TNS listener
WINDOWS_HOME\system32\drivers\Ocfs.sys	System file for Oracle Cluster File System

Configuring the Windows Firewall Post-installation configuration for the Windows Firewall must be undertaken if *all* of the following conditions are met:

- Oracle server-side components are installed.

  These components include Oracle Database, network listeners, and any Web servers or services.

- The computer services connections from other computers over a network.

  If no other computers connect to the computer with the Oracle software, then no post-installation configuration steps are required and the Oracle software will function as expected.

- The Windows Firewall is enabled.

  If the Windows Firewall is not enabled, then no post-installation configuration steps are required.

You can configure Windows Firewall by opening specific static TCP ports in the firewall or by creating exceptions for specific executables so that they are able to receive connection requests on any ports they choose. To configure the firewall, choose **Control Panel > Windows Firewall > Exceptions** or enter `netsh firewall add...` at the command line.

Alternatively, Windows will inform you if a foreground application is attempting to listen on a port, and it will ask you if you wish to create an exception for that executable. If you choose to do so, then the effect is the same as creating an exception for the executable either in the Control Panel or from the command line.

Troubleshooting Windows Firewall Exceptions If you cannot establish certain connections even after granting exceptions to the executables listed in Table 10–1, then follow these steps to troubleshoot the installation:

1. Examine Oracle configuration files (such as `*.conf` files), the Oracle key in the Windows registry, and network configuration files in *ORACLE_HOME*`\network\admin`.
2. Pay particular attention to any executable listed in *ORACLE_HOME*`\network\admin\listener.ora` in a `PROGRAM=` clause. Each of these must be granted an exception in the Windows Firewall, because a connection can be made through the TNS listener to that executable.
3. Examine Oracle trace files, log files, and other sources of diagnostic information for details on failed connection attempts. Log and trace files on the database client computer may contain useful error codes or troubleshooting information for failed connection attempts. The Windows Firewall log file on the server may contain useful information as well.
4. If the preceding troubleshooting steps do not resolve a specific configuration issue on Windows XP Service Pack 2, then provide the output from command `netsh firewall show state verbose=enable` to Oracle Support for diagnosis and problem resolution.

See Also:

- `http://www.microsoft.com/downloads/details.aspx?FamilyID=a7628646-131d-4617-bf68-f0532d8db131&displaylang=en` for information on Windows Firewall troubleshooting
- `http://support.microsoft.com/default.aspx?scid=kb;en-us;875357` for more information on Windows Firewall configuration

11

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

This section includes the following topics:

See Also:

- *Oracle Database 2 Day + Real Application Clusters Guide*
- The 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 the Oracle RAC and Oracle Clusterware environment. 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, as well as related hardware and software. For example, you can monitor your Oracle RAC database performance from your office, home, or a remote site, as long as you have access to a Web browser.

Both Oracle Enterprise Manager Database Control and Oracle Enterprise Manager Grid Control are cluster-aware and provide 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 as well as those 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 Performance Page

The Cluster Database Home Page

When you log in to Oracle Enterprise Manager using a client browser, the Cluster Database Home page appears where you can 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 the OCR, voting disk issues (if any), and node evictions

The Cluster Database Home page is similar to a single-instance 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, as well as 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, the overall throughput on the private interconnect, individual throughput on each of the network interfaces, error rates (if any) 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 also is available as collections that have a historic view. This is useful in conjunction 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 Performance Page

Also, 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.

For example, 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 single-instance tuning practices for Oracle Database apply to Oracle RAC databases. Therefore, implement the single-instance 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, query the V$CLUSTER_INTERCONNECTS and the V$CONFIGURED_INTERCONNECTS. For example:

Example 11–1 Verify Interconnect Settings with V$CLUSTER_INTERCONNECTS

```
SQL> SELECT * FROM V$CLUSTER_INTERCONNECTS;

NAME            IP_ADDRESS      IS_ SOURCE
--------------- --------------- --- ------------------------------
eth2            10.137.20.181   NO  Oracle Cluster Repository
```

Example 11–2 Verify Interconnect Settings with V$CONFIGURED_INTERCONNECTS

```
SQL> SELECT * FROM V$CONFIGURED_INTERCONNECTS;

NAME            IP_ADDRESS      IS_   SOURCE
--------------- --------------- ---   ------------------------------
eth2            10.137.20.181   NO    Oracle Cluster Repository
eth0            10.137.8.225    YES   Oracle Cluster Repository

SQL> DESC V$CONFIGURED_INTERCONNECTS
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
NAME                                                VARCHAR2(15)
IP_ADDRESS                                          VARCHAR2(16)
IS_PUBLIC                                           VARCHAR2(3)
SOURCE                                             VARCHAR2(31)
```

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 IPC buffer sizes.

The Oracle Cluster Registry (OCR) stores your system's interconnect information. Use the `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

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 the `oifcfg` command as described in the *Oracle Clusterware Administration and Deployment Guide* for more information about enabling and using the Oracle Interface Configuration (OIFCFG) command-line utility 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* installation guides for more information about creating your Oracle RAC database

Oracle RAC Performance Statistics

This section provides an overview of the V$ and GV$ views that provide statistics that you can use evaluate block transfers in your **cluster**. Use these statistics to analyze interconnect block transfer rates as well as the overall performance of your Oracle RAC database.

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. You 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.

See Also:

- *Oracle Database Performance Tuning Guide* for information about AWR
- *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
- *Oracle Database Reference* for more information about the `DBMS_ADDM_FINDINGS` view

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 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.

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

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, 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 and the `spdoc.txt` file for details about 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 sessions are waiting for 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 now 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

In order 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 statistics for current and cr blocks, for example, gc current blocks received, gc cr blocks received, and so on
- The Global Cache Service 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 in order to guarantee **cache coherency** and Oracle RAC inherently does not cause any more I/O to disk than a non-clustered instance.

Analyzing Performance Using GCS and GES Statistics

This section describes how to monitor Global Cache Service performance by identifying data blocks and objects which are frequently used (*hot*) by all instances. High concurrency on certain blocks may be identified by Global Cache Service 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 means that the blocks were pinned or held up by sessions or delayed by a log write on a remote instance or that a session on the same instance is already accessing a block which is in transition between instances and the current session needs to wait behind it (for example, `gc current block busy`).

The `V$SESSION_WAIT` view to identify objects and data blocks with contention. The `gc` 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 recourse to the query on `V$SESSION_WAIT` 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.

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 Global Cache Service 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 in most cases are 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, in most cases, 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 earlier 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.

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.

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 the most important and 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 (gc), 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.

The most important wait events for Oracle RAC include various categories, including:

- Block-oriented
  - gc current block 2-way
  - gc current block 3-way
  - gc cr block 2-way
  - gc cr block 3-way
- Message-oriented
  - gc current grant 2-way

    - gc cr grant 2-way
- Contention-oriented
    - gc current block busy
    - gc cr block busy
    - gc buffer busy acquire/release
- Load-oriented
    - gc current block congested
    - gc cr block congested

The block-oriented 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.

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.

The message-oriented 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).

The contention-oriented 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 load-oriented 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.

A

Server Control Utility Reference

This appendix includes a complete reference for the Server Control Utility (SRVCTL).

> **See Also:** Chapter 3, "Administering Database Instances and Cluster Databases" for more information about using SRVCTL to manage an Oracle RAC database

This appendix includes the following topics:

- Using SRVCTL
  - Overview of SRVCTL
  - Operational Notes for SRVCTL
  - Additional Topics on SRVCTL
  - Deprecated Subprograms or Commands
- SRVCTL Command Reference

Using SRVCTL

This section includes topics that relate to using the SRVCTL utility.

- Overview of SRVCTL
- Operational Notes for SRVCTL
- Additional Topics on SRVCTL

Overview of SRVCTL

Use SRVCTL to manage 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 11*g* release 2 (11.2) 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.

Operational Notes for SRVCTL

SRVCTL is installed on each node by default.

This section discusses the following topics:

- Usage Information
- Character Set and Case Sensitivity of Object Values
- Summary of Tasks for Which SRVCTL Is Used
- Using SRVCTL Help
- Privileges and Security

Usage Information

To use SRVCTL, log in to the operating system of a node and enter the SRVCTL command and its options in case-sensitive syntax as described in "SRVCTL Command Reference" on page A-11.

Guidelines for using SRVCTL are:

- Only use the version of SRVCTL that is provided with Oracle Database 11*g* on Oracle RAC databases that are created or upgraded for Oracle Database 11*g*. 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, issue 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.

Using Comma-Delimited Lists

When specifying a comma-delimited list as part of a SRVCTL command, there should not be any spaces between the items in the list. 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.

Specifying a Continuation of Command Line Entries

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 slash "\" symbol.

Character Set and Case Sensitivity of Object Values

SRVCTL interacts with many different types of objects. The character set and name length limitations, and whether or not 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
db_unique_name	Alpha-numeric characters, underscore (_), number sign (#), and dollar sign ($); the first 8 characters must be unique because those characters are used to form instance names for policy-managed databases	No	30 characters but the first 8 characters must be unique relative to any other database in the same cluster
diskgroup_name	Naming disk groups have the same limitations as naming other database objects. **See Also:** *Oracle Database SQL Language Reference* for more information about database object naming rules	No (all names are converted to uppercase)	
instance_name	Alpha-numeric characters	Depends on the platform	8 characters
listener_name			
node_name		No	
scan_name	The first character must be an alphabetic character	No	
server_pool	Alpha-numeric characters, underscore (_), number sign (#), period (.), and dollar sign ($); 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

SRVCTL is used to manage databases, instances, cluster databases, cluster database instances, Oracle ASM instance and disk groups, services, listeners, or other clusterware resources.

- Cluster Database Configuration Tasks
  - Add, modify, and delete cluster database configuration information.
  - Add an instance or a service to, and delete an instance or service from the configuration of a cluster database.
  - Move instances and services in a cluster database configuration and modify service configurations.

  - Set and unset the environment for an instance or service in a cluster database configuration.
  - Set and unset the environment for an entire cluster database in a cluster database configuration.
- General Cluster Database Administration Tasks
  - Start and stop cluster databases
  - Start and stop cluster database instances
  - Start, stop, and relocate cluster database services
  - Obtain statuses of cluster databases, cluster database instances, or cluster database services
- Node-Level Tasks
  - Adding and deleting node level applications, server pools, and VIPs
  - Setting and unsetting the environment for node-level applications
  - Administering disk groups
  - Administering server pools
  - Administering node applications
  - Administering Oracle ASM instances
  - Starting and stopping a group of programs that includes virtual IP addresses (VIPs), listeners, and Oracle Notification Services

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

Using SRVCTL Help

To see help for all SRVCTL commands, from the command line enter:

```
srvctl -h
```

To see the command syntax and a list of options for each SRVCTL command, from the command line enter:

```
srvctl command (or verb) object (or noun) -h
```

To see the SRVCTL version number enter:

```
$ srvctl -V
```

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 grid infrastructure, then log in as the database software owner (for example, `ora_db`) to manage databases and log in as the 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 (ONS), 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.

Additional Topics on SRVCTL

Difference between Using SRVCTL and CRSCTL

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.

Stopping Active SRVCTL Commands

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 Subprograms or Commands

The following command options have been deprecated for Oracle Database 11*g* release 2 (11.2):

Table A–2 Deprecated Commands and Options for SRVCTL

Command	Deprecated Options

`srvctl add asm`	`-n node_name`
	`-i instance_name`
	`-o Oracle_home`
	`-p spfile`

`srvctl add database`	`-A {name | IP_address}/netmask`
`srvctl add listener`	`-n node_name`
`srvctl config database`	`-t`
`srvctl config listener`	`-n node_name`
`srvctl config nodeapps`	`-n node_name` `-l`
`srvctl config asm`	`-i instance_name`
`srvctl disable nodeapps`	`-n node_name`
`srvctl enable asm`	`-i instance_name`
`srvctl enable nodeapps`	`-n node_name`

`srvctl getenv instance`	`-d db_unique_name`
	`-i instance_name`
	`-t "name=val_list"`

`srvctl getenv nodeapps`	`-n node_name`

`srvctl getenv service`	`-d db_unique_name`
	`-s service_name`
	`-t "name=val_list"`

`srvctl modify asm`	`-o Oracle_home`
`srvctl modify instance`	`-s asm_inst_name` `-r`
`srvctl remove asm`	`-n node_name` `-i instance_name`
`srvctl remove listener`	`-n node_name`
`srvctl remove nodeapps`	`-n "node_name_list"`

`srvctl setenv instance`	`-d db_unique_name`
	`-i instance_name`
	`-t "name=val_list"`
	`-T "name=val"`

`srvctl setenv nodeapps`	`-n node_name`

`srvctl setenv service`	`-d db_unique_name`
	`-s service_name`
	`-t "name=val_list"`
	`-T "name=val"`

`srvctl start asm`	`-i instance_name`

Table A–2 (Cont.) Deprecated Commands and Options for SRVCTL

Command	Deprecated Options
`srvctl status instance`	`-S` *`level`*
`srvctl status nodeapps`	`-n` *`node_name`*
`srvctl stop asm`	`-i` *`instance_name`*
`srvctl unsetenv instance`	`-d` *`db_unique_name`* `-i` *`instance_name`* `-t "`*`name=val_list`*`"`
`srvctl unsetenv nodeapps`	`-n` *`node_name`*
`srvctl unsetenv service`	`-d` *`db_unique_name`* `-s` *`service_name`* `-t "`*`name=val_list`*`"`

SRVCTL Command Reference

SRVCTL Command Syntax and Options

SRVCTL commands, object names, and options 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 [options]
```

In SRVCTL syntax:

- *command* is a verb such as `start`, `stop`, or `remove`
- *object* is the target or object on which SRVCTL performs the command, such as database or instance. You can also use object abbreviations.
- *options* extend the use of a preceding command combination to include additional parameters for the command. For example, the `-i` option indicates that a comma-delimited list of preferred instance names follows; sometimes the `-i` option only permits one value and not a list of names. The `-n` option indicates that a node name or a comma-delimited list of node names follows. 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–3 Summary of SRVCTL Commands

Command	Description
`add` on page A-14	Adds node applications, databases, database instances, Grid Naming Service (GNS), listeners, single client access names (SCANs), Oracle ASM instances, server pools, services, or virtual IPs (VIPs).
`config` on page A-26	Lists the configuration for GNS, the node applications, database, Oracle ASM instance, or service.
`disable` on page A-32	Disables the database, database instance, GNS, Oracle ASM instance, or service.
`enable` on page A-39	Enables the database, database instance, GNS, Oracle ASM instance, or service.
`getenv` on page A-46	Displays the environment variable in the configuration for the node applications, database, database instance, or service.
`modify` on page A-49	Modifies the node applications, database, database instance, GNS, or service configuration.
`relocate` on page A-61	Relocates GNS, OC4J, SCANs, and servers from one node to another.
`remove` on page A-65	Removes the node applications, database, database instance, GNS, Oracle ASM instance, or service.
`setenv` on page A-73	Sets the environment variable in the configuration for the node applications, database, database instance, or service.

Table A–3 (Cont.) Summary of SRVCTL Commands

Command	Description
`start` on page A-76	Starts the node applications, database, database instance, GNS, Oracle ASM instance, or service.
`status` on page A-85	Displays the status of the node applications, database, database instance, GNS, Oracle ASM instance, or service.
`stop` on page A-93	Stops the node applications, database, database instance, GNS, Oracle ASM instance, or service.
`unsetenv` on page A-103	Unsets the environment variable in the configuration for the node applications, database, database instance, or service.

SRVCTL Objects Summary

Table A–4 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–4 Object Keywords and Abbreviations

Object	Keyword	Purpose
Oracle Automatic Storage Management	`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.
Database	`database`	To add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, and remove databases.
Instance	`instance`	To add, modify, enable, disable, start, stop, obtain the status of, and remove database instances.
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).
Grid Naming Service (GNS)	gns	To add, modify, list the configuration of, enable, disable, stop, start, obtain the status of, and remove a GNS daemon.
Home directory (for patching)	`home`	To start, stop, and obtain the status of all clusterware resources related to a Home directory.f
Listener	`listener`	To add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, and remove listeners.

Table A–4 (Cont.) Object Keywords and Abbreviations

Object	Keyword	Purpose
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` or `eons`	To add, configure, enable, start, obtain the status of, stop, disable, and remove ONS instances for Oracle Restart **Note:** Oracle Notification Service is enhanced for 11*g* release 2 (11.2) and is abbreviated as eONS.
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 pool	`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
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

add

The `srvctl add` command adds the configuration and the Oracle Clusterware applications to the 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 `-i` must match the `ORACLE_SID` parameter. The database name given with `-d 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 `-m db_domain` must match the `DB_DOMAIN` setting.

Table A–5 srvctl add Summary

Command	Description
`srvctl add asm` on page A-14	Adds Oracle ASM instances
`srvctl add database` on page A-15	Adds a database and configuration
`srvctl add eons` on page A-16	Adds an eONS daemon
`srvctl add filesystem` on page A-16	Adds a volume to Oracle ACFS
`srvctl add gns` on page A-17	Adds the Grid Naming Service (GNS) to a cluster
`srvctl add instance` on page A-18	Adds one or more instance and configuration
`srvctl add listener` on page A-18	Adds a listener to the node
`srvctl add nodeapps` on page A-19	Adds node applications
`srvctl add oc4j` on page A-20	Adds OC4J instances
`srvctl add ons` on page A-20	Adds ONS daemons
`srvctl add scan` on page A-21	Adds SCAN VIPs
`srvctl add scan_listener` on page A-22	Adds SCAN listeners
`srvctl add service` on page A-22	Adds services
`srvctl add srvpool` on page A-24	Adds a server pool to a cluster
`srvctl add vip` on page A-25	Adds a VIP to a node

srvctl add asm

Adds a record for an Oracle ASM instance to the entire cluster. This command needs to be run only one time from the Oracle grid infrastructure home.

Note: To manage Oracle ASM on Oracle Database 11*g* release 2 (11.2) 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 Options

Use the `srvctl add asm` command with the following syntax:

```
srvctl add asm [-l listener_name]
```

This command has only one option, `-l`, which calls for the name of a listener. If you do not specify this option, then the listener name defaults to `LISTENER`.

Example

To add a clusterware resource for Oracle ASM on every node in the cluster, use the following command:

```
srvctl add asm
```

srvctl add database

Adds a database configuration to your cluster database configuration.

Syntax and Options

Use the `srvctl add database` command with the following syntax:

```
srvctl add database -d db_unique_name -o oracle_home
     [-x node_name] [-m domain_name] [-p spfile]
     [-r {PRIMARY | PHYSICAL_STANDBY | LOGICAL_STANDBY | SNAPSHOT_STANDBY}]
     [-s start_options] [-t stop_options] [-n db_name]
     [-y {AUTOMATIC | MANUAL}] [-g server_pool_list] [-a disk_group_list]
```

Table A–6 srvctl add database Options

Syntax	Description
`-d db_unique_name`	Unique name for the database.
`-o oracle_home`	The path for the Oracle database home directory.
`-x node_name`	Node name on which you want to register a single-instance (non-RAC) database. **Note:** This option is available only with Oracle Clusterware and cannot be used with the `-g` option.
`-m db_domain`	The domain for the database **Note:** You must use this option if you set the `DB_DOMAIN` initialization parameter set for the database.
`-p spfile`	The path name of the database server parameter file.
`-r {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
`-s start_options`	Startup options for the database, such as `OPEN`, `MOUNT`, and `NOMOUNT`.

Table A–6 (Cont.) srvctl add database Options

Syntax	Description
`-t stop_options`	Stop options for the database, such as `NORMAL`, `TRANSACTIONAL`, `IMMEDIATE`, and `ABORT`
`-n db_name`	The name of the database, if it is different from the unique name given by the `-d` option
`-y {AUTOMATIC | MANUAL}`	Management policy for the database, either automatic or manual
`-g server_pool_list`	Comma-delimited list of server pool names used to control database placement. If you do not specify this option, then it defaults to the Generic server pool. **Note:** This option is available only with Oracle Clusterware and cannot be used with the `-x` option.
`-a "disk_group_list"`	Comma-delimited list of Oracle ASM disk groups if database uses Oracle ASM storage

Example

An example of this command is:

```
srvctl add database -d crm -o /u01/oracle/product/112/mydb -m foo.com
     -p +diskgroup1/crm/spfilecrm.ora -r PRIMARY -s open -t normal
     -n database2 -y AUTOMATIC -g svrpool1,svrpool2 -a "diskgroup1,diskgroup2"
```

srvctl add eons

Adds an eONS daemon to be managed by Oracle Restart.

> **Note:** This command is only available with Oracle Restart.

Syntax and options

```
srvctl add eons [-m multicast_ip_address] [-p port_number]
                [-e eons_listen_port][-v]
```

Table A–7 srvctl add eons Options

Option	Description
`-m multicast_ip_address`	The multicast IP address for eONS
`-p port_number`	The port number for eONS
`-e eons_listen_port`	Local listen port for eONS daemon **Note:** If `port` is not specified, then the default value of 2016 is used.
`-v`	Verbose output

Example

An example of this command is:

```
# srvctl add eons -p 2018
```

srvctl add filesystem

Adds a disk device to Oracle ACFS. This command needs to be run only one time from the Oracle grid infrastructure home.

An Oracle ACFS file system resource is typically created for use with application resource dependency lists. For example, if an Oracle ACFS file system is configured for use as an Oracle Database home, then a resource created for the file system can be included in the resource dependency list of the Oracle Database application. This will cause the file system and stack to be automatically mounted as a result of the start action of the database application.

> **Note:** To manage Oracle ACFS on Oracle Database 11*g* release 2 (11.2) 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 ACFS.

Syntax and Options

Use the `srvctl add filesystem` command with the following syntax:

```
srvctl add filesystem -d volume_device -v volume_name -g diskgroup_name
                      [-m mountpoint_path] [-u user_name]
```

> **Note:** This command is only available with Oracle Clusterware.

Table A–8 srvctl add filesystem Options

Option	Description
`-d volume_device`	The Volume device path
`-v volume_name`	The name of the volume
`-g diskgroup_name`	The name of the Oracle ACFS disk group to which the device should be added
`-m mountpoint_path`	The mount point path name for the disk device
`-u user_name`	The name of the user authorized to mount and unmount the filesystem

Example

An example of this command is the following:

```
srvctl add filesystem -d /dev/asm/d1volume1-295 -v VOLUME1 -d RAC_DATA \
  -m /oracle/cluster1/acfs1
```

srvctl add gns

Use this command to add the Grid Naming Service (GNS) to a cluster when you are using a DHCP public network.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl add gns` command with the following syntax:

```
srvctl add gns -i ip_address -d domain
```

Table A–9 srvctl add gns Options

Option	Description
`-i` *ip_address*	The Grid virtual IP (VIP) address
`-d` *domain*	The network subdomain that is used for Forward DNS Lookup of cluster IP addresses

Example

An example of this command is:

```
srvctl add gns -i 192.168.16.17 -d cluster.mycompany.com
```

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 is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl add instance` command with the following syntax:

```
srvctl add instance -d db_unique_name -i instance_name -n node_name
```

Table A–10 srvctl add instance Options

Option	Description
`-d` *db_unique_name*	The unique name of the database you are adding the instance to
`-i` *instance_name*	The name of the instance you are adding
`-n` *node_name*	The name of the node on which you are creating the instance

Examples

Examples of this command are:

```
$ srvctl add instance -d crm -i  crm01 -n gm01
$ srvctl add instance -d crm -i  crm02 -n gm02
$ srvctl add instance -d crm -i  crm03 -n gm03
```

srvctl add listener

Adds a listener to every node in a cluster.

Syntax and Options

Use the `srvctl add listener` command with the following syntax:

```
srvctl add listener [-l listener_name] [-o Oracle_home]
    [-p "[TCP:]port_list[/IPC:key][/NMP:pipe_name][/TCPS:s_port][/SDP:port]"]
    [-k network_number] [-s]
```

Table A–11 srvctl add listener Options

Option	Description
`-l` *listener_name*	Specify a listener name. If this option is not specified, the default name of LISTENER is used.
`-o` *oracle_home*	Specify an Oracle home for the cluster database. If you do not include this option, then it uses the Grid home by default.

`-p "[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.

`-k` *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.
`-s`	Indicates you want to skip the checking of ports

Example

The following command adds a listener named `listener112` that is listening on port 1341 and runs from the `/ora/ora112` home directory on every node in the cluster:

```
$ srvctl add listener -l listener112 -p 1341 -o /ora/ora112
```

srvctl add nodeapps

Adds a node application configuration to the specified node.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl add nodeapps` command with one the following syntax models:

```
srvctl add nodeapps -n node_name -A {name | ip}/netmask[/if1[|if2|...]]
  [-m multicast_ip_address] [-p multicast_port_number] [-l ons_local_port]
  [-r ons_remote-port] [-t host[:port][,host[:port],...]] [-v]

srvctl add nodeapps -S subnet/netmask[/if1[|if2|...]] [-d dhcp_server_type]
  [-m multicast_ip_address] [-p multicast_port_number] [-l ons_local_port]
  [-r ons_remote-port] [-t host[:port][,host[:port],...]] [-v]
```

Table A–12 srvctl add nodeapps Options

Option	Description
`-n` *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.
`-A`	This specification creates a traditional VIP node application on the specified node: {*name* | *ip*}`/`*netmask*`[/if1[|if2|...]]`. **Note:** This option must be used for upgrade configurations and new, non-DHCP configurations.

Table A–12 (Cont.) srvctl add nodeapps Options

Option	Description
`-S subnet/netmask [/if1[|if2|...]]`	Creates a DHCP subnet.
`-m multicast_ip_address`	The multicast address for the eONS daemon
`-p multicast_port_number`	The multicast port number for the eONS daemon
`-l ons_local_port`	The ONS daemon listener port on its node. If you do not specify this value, the ONS daemon listener port defaults to 6100.
`-r ons_remote_port`	The port number for remote ONS daemon connections. If you do not specify a port number, the default value of 6200 is used for the ONS remote port.
`-t host[:port], [host[:port],[...]`	A list of `host:port` pairs of remote hosts that are part of the ONS 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

Note: 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 -n crmnode1 -A 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 Options

```
srvctl add oc4j [-v]
```

This command has only one option, `-v`, which displays verbose output.

Example

An example of this command is:

```
srvctl add oc4j
```

srvctl add ons

Adds an ONS daemon to an Oracle Restart configuration.

Note: This command is only available with Oracle Restart.

Syntax and options

```
srvctl add ons [-l ons_local_port] [-r ons_remote_port]
    [-t host[:port][,host[:port]][...]] [-v]
```

Table A–13 srvctl add ons Options

Option	Description
`-l ons_local_port`	The ONS daemon listening port for local client connections
`-r ons_remote_port`	The ONS daemon listening port for connections from remote hosts
`-t host[:port][,host[:port]][...]`	A list of comma-delimited `host:port` pairs of remote hosts that are part of the ONS 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 DHCP network and Oracle GNS configuration.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl add scan` command with the following syntax:

```
srvctl add scan -n scan_name [-k network_number]
  [-S subnet/netmask[/if1[|if2|...]]]
```

Table A–14 srvctl add scan Options

Option	Description
`-n scan_name`	A fully qualified host name, which includes the domain name.
`-k network_number`	The optional network number from which SCAN VIPs are obtained. If not specified, the SCAN VIPs are obtained from the same default network from which the `nodeapps` VIP is obtained.
`-S subnet/netmask [/if1 [|if2|...]]`	Creates the `network_number`. This option must be specified when `network_number` does not exist.

Example

An example of this command is:

```
# srvctl add scan -n scan.mycluster.example.com
```

srvctl add scan_listener

Adds Oracle Clusterware resources to 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 Options

Use the `srvctl add scan_listener` command with the following syntax:

```
srvctl add scan_listener [-l lsnr_name_prefix] [-s]
  [-p "[TCP:]port_list[/IPC:key][/NMP:pipe_name][/TCPS:s_port] [/SDP:port]"]
```

Table A–15 srvctl add scan_listener Options

Option	Description
`-l lsnr_name_prefix`	The SCAN listener name prefix.
`-s`	Use the `-s` option to skip checking of the ports.
`-p "[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 option is not specified, then the default TCP port of 1521 is used.

Example

An example of this command is:

```
# srvctl add scan_listener -l 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.

Syntax and Options

Use the `srvctl add service` command with one of the following syntax models:

```
srvctl add service -d db_unique_name -s service_name
   -r preferred_list [-a available_list] [-P {BASIC | NONE | PRECONNECT}]
   [-l [PRIMARY | PHYSICAL_STANDBY | LOGICAL_STANDBY | SNAPSHOT_STANDBY]
   [-y {AUTOMATIC | MANUAL}] [-q {TRUE | FALSE}] [-j {SHORT | LONG}]
   [-B {NONE | SERVICE_TIME | THROUGHPUT}] [-e {NONE | SESSION | SELECT}]
   [-m {NONE | BASIC}] [-x {TRUE | FALSE}]
   [-z failover_retries] [-w failover_delay]

srvctl add service -d db_unique_name -s service_name
   -u {-r preferred_list | -a available_list}

srvctl add service -d db_unique_name -s service_name
   -g server_pool [-c {UNIFORM | SINGLETON}] [-k network_number]
   [-l [PRIMARY | PHYSICAL_STANDBY | LOGICAL_STANDBY | SNAPSHOT_STANDBY]
   [-y {AUTOMATIC | MANUAL}] [-q {TRUE | FALSE}] [-j {SHORT | LONG}]
   [-B {NONE | SERVICE_TIME | THROUGHPUT}] [-e {NONE | SESSION | SELECT}]
   [-m {NONE | BASIC}] [-P {BASIC | NONE | PRECONNECT}] [-x {TRUE | FALSE}]
   [-z failover_retries] [-w failover_delay]
```

Table A–16 lists and describes all the `srvctl add service` options and whether they can be used when adding a service to a single-instance or Oracle RAC database.

Table A–16 srvctl add service Options

Option	Description
`-d` *db_unique_name*	Unique name for the database.
`-s` *service_name*	The service name.
`-r` *preferred_list*	A list of preferred instances used when the database is administrator managed. The list of preferred instances must be mutually exclusive with the list of available instances. **Note:** This option is available only with Oracle RAC and only for administrator-managed databases.
`-a` *available_list*	A list of available instances used when the database is administrator managed. The list of available instances must be mutually exclusive with the list of preferred instances. **Note:** This option is available only with Oracle RAC and only for administrator-managed databases.
`-g` *server_pool*	The name of a server pool used when the database is policy managed. **Note:** This option is available only with Oracle RAC and only for policy-managed databases.
`-c {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 option is available only with Oracle RAC and only for policy-managed databases.
`-k` *network_number*	The optional network number from which SCAN VIPs are obtained. If not specified, the SCAN VIPs are obtained from the same default network from which the `nodeapps` VIP is obtained. **Note:** This option is available only with Oracle RAC and only for policy-managed databases.
`-P {BASIC | NONE | PRECONNECT}`	TAF policy specification (for administrator-managed databases only). **Note:** You can only use `PRECONNECT` when you specify the `-r` and `-a` options.
`-l {[PRIMARY] | [PHYSICAL_STANDBY] | [LOGICAL_STANDBY] | [SNAPSHOT_STANDBY]}`	The service role. You use this option to indicate that the service should should only be automatically started when the Oracle Data Guard database role matches one of the specified service roles. **See Also:** *Oracle Data Guard Concepts and Administration* for more information about database roles
`-y {AUTOMATIC | MANUAL}`	Service management policy. `AUTOMATIC`: The service starts when the database starts. `MANUAL`: The service does not start when the database starts.
`-q {TRUE | FALSE}`	Indicates whether AQ HA notifications should be enabled (`TRUE`) for this service.

Table A–16 (Cont.) srvctl add service Options

Option	Description
-x {TRUE | FALSE}	Indicates whether or not 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 option is available only with Oracle RAC.
-j {SHORT | LONG}	Assign a connection load balancing goal to the service: SHORT if using an integrated connection pool, LONG for long running connections that you want balanced by the number of sessions per node for the service
-B {NONE | SERVICE_TIME | THROUGHPUT}	Goal for the Load Balancing Advisory.
-e {NONE | SESSION | SELECT}	Failover type.
-m {NONE | BASIC}	Failover method. If the failover type (-e) is set to a value other than NONE, then you should choose BASIC for this option.
-u	**Note:** This option is available only with Oracle RAC.
-z *failover_retries*	The number of failover retry attempts.
-w *failover_delay*	The time delay between failover attempts.

Examples

Use this example syntax to add the gl.example.com service to the my_rac database with AQ HA notifications enabled, a failover method of BASIC, a Connection Load Balancing Goal of LONG, a failover type of SELECT, and 180 failover retries with a delay of 5:

```
srvctl add service -d my_rac -s gl.example.com -q TRUE -m BASIC \
-e SELECT -z 180 -w 5 -j LONG
```

Use this example syntax to add a named service to a database with preferred instances in list one and available instances in list two, using preconnect failover for the available instances:

```
srvctl add service -d crm -s sales -r crm01,crm02 -a crm03 -P PRECONNECT
```

srvctl add srvpool

Adds a server pool to a cluster.

Note: This command is only available with Oracle Clusterware.

Syntax and Options

Use the srvctl add srvpool command with the following syntax:

```
srvctl add srvpool -g server_pool [-i importance] [-l min_size] [-u max_size]
     [-n node_list] [-f]
```

Table A–17 srvctl add srvpool Options

Option	Description
-g *server_pool*	The name of the server pool.

Table A–17 (Cont.) srvctl add srvpool Options

Option	Description
-i *importance*	The importance of the server pool (default is 0).
-l *min_size*	The minimum size of the server pool (default is 0).
-u *max_size*	The maximum size of the server pool. A value of -1 indicated the size is unlimited.
-n *node_names*	A comma-separated list of candidate node names. 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.
-f	Add the server pool, even if requires stopping resources in other server pools.

Example

An example of this command is:

```
$ srvctl add srvpool -g srvpool1 -i 1 -l 3 -u 7 -n mynode1,mynode2
```

srvctl add vip

Adds a VIP to a node.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl add vip` command with the following syntax:

```
srvctl add vip -n node_name -A {name|ip}/netmask[/if1[if2|...]]
     [-k network_number] [-v]
```

Table A–18 srvctl add vip Options

Option	Description
-n *node_name*	The name of the node on which you are adding the VIP
-A {*name*|*ip*}/*netmask* [/*if1*[|*if2*|...]]	This specification creates a traditional VIP node application on the specified node
-k *network_number*	The optional network number from which SCAN VIPs are obtained. If not specified, the SCAN VIPs are obtained from the same default network from which the `nodeapps` VIP is obtained.
-v	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 vip -n node7 -A 192.168.16.17/255.255.255.0 -k 2
```

The preceding example creates a network number, 2, for the VIP just added. You can specify the network number after the -k option in other SRVCTL commands.

config

The `srvctl config` command displays the configuration stored in the Oracle Clusterware resource attributes.

Table A–19 srvctl config Summary

Command	Description
`srvctl config asm` on page A-26	Displays the configuration for the Oracle ASM instances on the node
`srvctl config database` on page A-27	Displays the configuration information of the cluster database
`srvctl config eons` on page A-27	Displays configuration information for eONS
`srvctl config filesystem` on page A-27	Displays the configuration information for an ACFS volume
`srvctl config gns` on page A-28	Displays the GNS configuration
`srvctl config listener` on page A-28	Displays a list of configured listeners that are registered with Oracle Clusterware on a given node
`srvctl config nodeapps` on page A-28	Displays the configuration information for the node applications
`srvctl config oc4j` on page A-29	Displays the configuration of the OC4J instance
`srvctl config ons` on page A-29	Displays configuration information for ONS
`srvctl config scan` on page A-29	Displays the configuration information for SCAN VIPs
`srvctl config scan_listener` on page A-30	Displays the configuration information for SCAN listeners
`srvctl config service` on page A-30	Displays the configuration information for the services
`srvctl config srvpool` on page A-30	Displays configuration information for a specific server pool
`srvctl config vip` on page A-31	Displays the configuration information for the VIP

srvctl config asm

Displays the configuration for all Oracle ASM instances.

> **Note:** To manage Oracle ASM on Oracle Database 11*g* release 2 (11.2) 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 Options

Use the `srvctl config asm` command with the following syntax:

```
srvctl config asm [-a]
```

Table A–20 srvctl config asm Options

Option	Description
-a	Print detailed configuration information

Example

An example of this command is:

```
srvctl config asm -a
```

srvctl config database

Displays the configuration for an Oracle RAC database or lists all configured databases that are registered with Oracle Clusterware.

Syntax and Options

Use the `srvctl config database` command with the following syntax:

```
srvctl config database [-d db_unique_name] [-a]
```

Table A–21 srvctl config database Options

Option	Description
-d *db_unique_name*	Unique name for the database. If you do not specify this option, then the utility displays the configuration of all database resources.
-a	Print detailed configuration information

Examples

To list the configuration of all databases, use the following example:

```
srvctl config database
```

To show the configuration for a specific database, use the following example:

```
srvctl config database -d db_erp -a
```

srvctl config eons

Displays configuration information for the eONS daemon.

> **Note:** This command is only available with Oracle Restart.

```
srvctl config eons
```

srvctl config filesystem

Displays the configuration for an Oracle Automatic Storage Management Cluster File System (Oracle ACFS) device.

Syntax and Options

Use the `srvctl config filesystem` command with the following syntax:

```
srvctl config filesystem -d volume_device_path
```

Table A–22 srvctl config filesystem Options

Option	Description
`-d volume_device_path`	The path name of a device that an Oracle ACFS volume uses.

Examples

To list the configuration of all databases, use the following example:

```
srvctl config database
```

To show the configuration for a specific database, use the following example:

```
srvctl config database -d db_erp -a
```

srvctl config gns

Displays the configuration for GNS.

> **Note:** This option is available only for Oracle Clusterware.

Syntax and Options

Use the `srvctl config gns` command with the following syntax:

```
srvctl config gns
```

srvctl config listener

Displays a list of configured listeners that are registered with Oracle Clusterware or displays detailed configuration information for a specific listener.

Syntax and Options

Use the `srvctl config listener` command with the following syntax:

```
srvctl config listener [-l listener_name] [-a]
```

Table A–23 srvctl config listener Options

Option	Description
`-l listener_name`	Listener name. If you do not specify this option, then the name of the listener defaults to LISTENER.
`-a`	Print detailed configuration information

Example

An example of this command is:

```
srvctl config listener
```

srvctl config nodeapps

Displays the VIP configuration for each node in the cluster.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl config nodeapps` command with the following syntax:

```
srvctl config nodeapps [-a] [-g] [-s] [-e]
```

Table A–24 srvctl config nodeapps Option

Option	Description
-a	Displays the VIP address configuration
-g	Displays the GSD configuration
-s	Displays the ONS configuration
-e	Displays the eONS configuration

Example

An example of this command is:

```
$ srvctl config nodeapps -a -g -s -e
```

srvctl config oc4j

Displays configuration information for the OC4J instance.

> **Note:** This command is only available with Oracle Clusterware.

```
srvctl config oc4j
```

srvctl config ons

Displays configuration information for the ONS daemon.

> **Note:** This command is only available with Oracle Restart.

```
srvctl config ons
```

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 Options

Use the `srvctl config scan` command with the following syntax:

```
srvctl config scan [-i ordinal_number]
```

The only option available for this command is `-i` *ordinal_number*, 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 config scan -i 1
```

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 Options

Use the `srvctl config scan_listener` command with the following syntax:

```
srvctl config scan_listener [-i ordinal_number]
```

The only option available for this command is `-i` *ordinal_number*, 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 config scan_listener -i 1
```

srvctl config service

Displays the configuration for a service.

Syntax and Options

Use the `srvctl config service` command with the following syntax:

```
srvctl config service -d db_unique_name [-s service_name] [-a]
```

Table A–25 srvctl config service Options

Option	Description
`-d` *db_unique_name*	Unique name for the database
`-s` *service_name*	Service name. If this option is not specified, then the configuration information for all services configured for the database are displayed.
`-a`	Print detailed configuration information

Example

An example of this command is:

```
$ srvctl config service -d crm -s crm_dev
```

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 Options

Use the `srvctl config srvpool` command with the following syntax:

```
srvctl config srvpool [-g server_pool]
```

Example

An example of this command is:

```
$ srvctl config srvpool -g 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 Options

Use the `srvctl config vip` command with one of the following syntax models:

```
srvctl config vip -n node_name

srvctl config vip -i vip_name
```

Table A–26 srvctl config vip Options

Option	Description
`-n node_name`	Node name
`-i vip_name`	The VIP name

Example

An example of this command is:

```
$ srvctl config vip -n crmnode1
```

disable

Disables a specified object (cluster database, database instance, Oracle ASM instance, or service). The `srvctl disable` command is intended to be used when a object needs to be repaired or shut down for maintenance, and should not be restarted automatically.

When you issue 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 `-i` *`instance_name`* or `-n` *`node_name`*, then SRVCTL only disables the service from running on the specified instance or node.

Table A–27 srvctl disable Summary

Command	Description
`srvctl disable asm` on page A-32	Disables an Oracle ASM instance
`srvctl disable database` on page A-33	Disables the cluster database
`srvctl disable diskgroup` on page A-33	Disables a disk group on a number of specified nodes
`srvctl disable eons` on page A-34	Disables the eONS daemon
`srvctl disable filesystem` on page A-34	Disables an Oracle ACFS volume
`srvctl disable gns` on page A-34	Disables GNS
`srvctl disable instance` on page A-35	Disables an instance
`srvctl disable listener` on page A-35	Disables a listener
`srvctl disable nodeapps` on page A-36	Disables a node application and GSD
`srvctl disable oc4j` on page A-36	Disables OC4J instances
`srvctl disable ons` on page A-36	Disables the ONS daemon
`srvctl disable scan` on page A-36	Disables SCAN VIPs
`srvctl disable scan_listener` on page A-37	Disables SCAN listeners
`srvctl disable service` on page A-37	Disables a service
`srvctl disable vip` on page A-38	Disables a VIP

srvctl disable asm

Disables an Oracle ASM instance.

Note: To manage Oracle ASM on Oracle Database 11*g* release 2 (11.2) 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 Options

Use the `srvctl disable asm` command with the following syntax:

```
srvctl disable asm [-n node_name]
```

Table A–28 srvctl disable asm Options

Option	Description
`-n node_name`	Node name **Note:** This option is available only with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl disable asm -n crmnode1
```

srvctl disable database

Disables a database. If the database is a cluster database, then its instances are also disabled.

Syntax and Options

Use the `srvctl disable database` command with the following syntax:

```
srvctl disable database -d db_unique_name [-n node_name]
```

Table A–29 srvctl disable database Options

Option	Description
`-d database_name`	Database name
`-n node_name`	Disables the database from running on the named node **Note:** This option is available only with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl disable database -d mydb1
```

srvctl disable diskgroup

Disables a specific disk group on a number of specified nodes.

Syntax and Options

Use the `srvctl disable diskgroup` command with the following syntax:

```
srvctl disable diskgroup -g diskgroup_name [-n node_list]
```

Table A–30 srvctl disable diskgroup Options

Option	Description
`-g diskgroup_name`	The Oracle ASM disk group name

Table A–30 (Cont.) srvctl disable diskgroup Options

Option	Description
`-n node_list`	Comma-delimited list of node names on which to disable the disk group This option is only available with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl disable diskgroup -g diskgroup1 -n mynode1, mynode2
```

srvctl disable eons

Disables the eONS daemon for Oracle Restart installations.

> **Note:** This command is only available with Oracle Restart.

```
srvctl disable eons [-v]
```

The only option for this command is `-v`, which indicates that verbose output should be displayed.

srvctl disable filesystem

Disables an Oracle ACFS volume.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl disable filesystem` command with the following syntax:

```
srvctl disable filesystem -d volume_device_name
```

Table A–31 srvctl disable filesystem Options

Option	Description
`-d volume_device_name`	Name of the Oracle ACFS volume

Example

An example of this command is:

```
$ srvctl disable filesystem -d /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 Options

Use the `srvctl disable gns` command with the following syntax:

```
srvctl disable gns [-n node_name]
```

Table A–32 srvctl disable gns Options

Option	Description
`-n node_name`	Name of a node in the cluster

Example

An example of this command is:

```
$ srvctl disable gns -n crm7
```

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.

Syntax and Options

Use the `srvctl disable instance` command with the following syntax:

```
srvctl disable instance -d db_unique_name -i instance_name_list
```

Table A–33 srvctl disable instance Options

Option	Description
`-d db_unique_name`	Unique name for the database
`-i instance_name_list`	Comma-delimited list of instance names

Example

An example of this command is:

```
$ srvctl disable instance -d crm -i "crm1,crm3"
```

srvctl disable listener

Disables a listener resource.

Syntax and Options

Use the `srvctl disable listener` command with the following syntax:

```
srvctl disable listener [-l listener_name] [-n node_name]
```

Table A–34 srvctl disable listener Options

Option	Description
`-l listener_name`	Name of a listener resource. If you do not specify this option, the name of the listener defaults to `LISTENER`.
`-n node_name`	Name of a cluster node on which the listener you want to disable is running. This option is only available with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl disable listener -l listener_crm -n node5
```

srvctl disable nodeapps

Disables node applications on all nodes in the cluster.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl disable nodeapps` command with the following syntax:

```
srvctl disable nodeapps [-g] [-v]
```

Table A–35 srvctl disable nodeapps Options

Option	Description
`-g`	Disables GSD
`-v`	Verbose output

Example

An example of this command is:

```
$ srvctl disable nodeapps -g -v
```

srvctl disable oc4j

Disables the OC4J instance on all nodes or on a specific node.

Syntax and Options

Use the `srvctl disable oc4j` command with the following syntax:

```
srvctl disable oc4j [-n node_name] [-v]
```

Table A–36 srvctl disable oc4j Options

Option	Description
`-n` *`node_name`*	The name of a node in the cluster
`-v`	Verbose output

Example

An example of this command is:

```
$ srvctl disable oc4j -n crm3
```

srvctl disable ons

Disables the ONS daemon for Oracle Restart installations.

```
srvctl disable ons [-v]
```

The only option for this command is `-v`, 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 Options

Use the `srvctl disable scan` command with the following syntax:

```
srvctl disable scan [-i ordinal_number]
```

The only option available for this command is `-i` *`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 -i 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 Options

Use the `srvctl disable scan_listener` command with the following syntax:

```
srvctl disable scan_listener [-i ordinal_number]
```

The only option available for this command is `-i` *`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 -i 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 Options

Use the `srvctl disable service` command with the following syntax:

```
srvctl disable service -d db_unique_name
      -s "service_name_list" [-i instance_name | -n node_name]
```

If you do not specify either the `-i` *`instance_name`* or `-n` *`node_name`* options, then the command disables the service on all nodes.

Table A–37 srvctl disable service Options

Option	Description
`-d db_unique_name`	Unique name for the database
`-s "service_name_list"`	Comma-delimited list of service names, or a single service name
`-i instance_name`	The name of the instance that you want to disable the service for. **Note:** Use this option with administrator-managed databases **Note:** This option is available only with Oracle Clusterware and Oracle RAC.
`-n node_name`	The name of the node on which to disable the service **Note:** Use this option with policy-managed databases **Note:** This option is available only with Oracle Clusterware and Oracle RAC.

Examples

The following example globally disables two services for the `CRM` database:

```
$ srvctl disable service -d crm -s "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 -d crm -s crm -i crm1
```

srvctl disable vip

Disables a specific VIP.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl disable vip` command with the following syntax:

```
srvctl disable vip -i vip_name [-v]
```

Table A–38 srvctl disable vip Options

Option	Description
`-i vip_name`	The VIP name
`-v`	Verbose output

Example

An example of this command is:

```
$ srvctl disable vip -i vip1 -v
```

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 the command is ignored. Enabled objects can be started, and disabled objects cannot be started.

Table A–39 srvctl enable Summary

Command	Description
`srvctl enable asm` on page A-39	Enables an Oracle ASM instance
`srvctl enable database` on page A-40	Enables the database resource
`srvctl enable diskgroup` on page A-40	Enables a specified disk group on a number of specified nodes
`srvctl enable eons` on page A-41	Enables the eONS daemon
`srvctl enable filesystem` on page A-41	Enables an Oracle ACFS volume
`srvctl enable gns` on page A-41	Enables GNS
`srvctl enable instance` on page A-41	Enables the instance
`srvctl enable listener` on page A-42	Enables a listener
`srvctl enable nodeapps` on page A-42	Enables node applications and GSD
`srvctl enable oc4j` on page A-43	Enables OC4J instances
`srvctl enable ons` on page A-43	Enables the ONS daemon
`srvctl enable scan` on page A-43	Enables SCAN VIPs
`srvctl enable scan_listener` on page A-44	Enables SCAN listeners
`srvctl enable service` on page A-44	Enables a service
`srvctl enable vip` on page A-45	Enables a VIP

srvctl enable asm

Enables an Oracle ASM instance.

> **Note:** To manage Oracle ASM on Oracle Database 11*g* release 2 (11.2) 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 Options

Use the `srvctl enable asm` command with the following syntax:

```
srvctl enable asm [-n node_name]
```

Table A–40 srvctl enable asm Option

Option	Description
-n *node_name*	Node name **Note:** This option is available only with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl enable asm -n crmnode1
```

srvctl enable database

Enables a cluster database and its instances.

Syntax and Options

Use the `srvctl enable database` command with the following syntax:

```
srvctl enable database -d db_unique_name [-n node_name]
```

Table A–41 srvctl enable database Options

Option	Description
-d *database_name*	Database name
-n *node_name*	The name of the node for which the database resource should be enabled **Note:** This option is available only with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl enable database -d mydb1
```

srvctl enable diskgroup

Enables a specific disk group on a number of specified nodes.

Syntax and Options

Use the `srvctl enable diskgroup` command with the following syntax:

```
srvctl enable diskgroup -g diskgroup_name [-n node_list]
```

Table A–42 srvctl enable diskgroup Options

Option	Description
-g *diskgroup_name*	The Oracle ASM disk group name
-n *node_list*	Comma-delimited list of node names on which to enable the disk group This option is only available with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl enable diskgroup -g diskgroup1 -n mynode1,mynode2
```

srvctl enable eons

Enables the eONS daemon for Oracle Restart installations.

```
srvctl enable eons [-v]
```

The only option for this command is `-v`, which indicates that verbose output should be displayed.

srvctl enable filesystem

Enables an Oracle ACFS volume.

Note: This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl enable filesystem` command with the following syntax:

```
srvctl enable filesystem -d volume_device_name
```

Table A–43 srvctl enable filesystem Options

Option	Description
`-d volume_device_name`	Device name of the Oracle ACFS volume

Example

An example of this command is:

```
$ srvctl enable filesystem  -d /dev/asm/racvol1
```

srvctl enable gns

Enables GNS on all nodes or a specific node.

Note: This command is available only with Oracle Clusterware.

Syntax and Options

Use the `srvctl enable gns` command with the following syntax:

```
srvctl enable gns [-n node_name]
```

Table A–44 srvctl enable gns Options

Option	Description
`-n node_name`	Name of the node on which to enable GNS. If this option is not specified, then GNS is enabled on all nodes in the cluster.

Example

An example of this command is:

```
$ srvctl enable gns
```

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.

Syntax and Options

Use the `srvctl enable instance` command with the following syntax:

```
srvctl enable instance -d db_unique_name -i instance_name_list
```

Table A–45 srvctl enable instance Option

Option	Description
`-d db_unique_name`	Unique name for the database
`-i instance_name_list`	Comma-delimited list of instance names.

Example

An example of this command is:

```
$ srvctl enable instance  -d crm -i "crm1,crm2"
```

srvctl enable listener

Enables a listener resource.

Syntax and Options

Use the `srvctl enable listener` command with the following syntax:

```
srvctl enable listener [-l listener_name] [-n node_name]
```

Table A–46 srvctl enable listener Options

Option	Description
`-l listener_name`	Name of a listener resource. If you do not specify this option, the name of the listener defaults to LISTENER
`-n node_name`	Name of a cluster node **Note:** This option is available only with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl enable listener -l listener_crm -n 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 Options

Use the `srvctl enable nodeapps` command with the following syntax:

```
srvctl enable nodeapps [-g] [-v]
```

Table A–47 srvctl enable nodeapps Options

Option	Description
-g	Enables GSD
-v	Verbose output

Example
An example of this command is:

```
$ srvctl enable nodeapps -g -v
```

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 Options
Use the `srvctl enable oc4j` command with the following syntax:

```
srvctl enable oc4j [-n node_name] [-v]
```

Table A–48 srvctl enable oc4j Options

Option	Description
-n *node_name*	The name of a node in the cluster
-v	Verbose output

Example
An example of this command is:

```
$ srvctl enable oc4j -n crm3
```

srvctl enable ons

Enables the ONS daemon.

> **Note:** This command is only available with Oracle Restart.

Syntax and Options
Use the `srvctl enable ons` command with the following syntax:

```
srvctl enable ons [-v]
```

The only option for this command is -v, which indicates that verbose output should be displayed.

Example
An example of this command is:

```
$ srvctl enable oc4j -n crm3
```

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 Options

Use the `srvctl enable scan` command with the following syntax:

```
srvctl enable scan [-i ordinal_number]
```

The only option available for this command is `-i ordinal_number`, which identifies any 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 -i 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 Options

Use the `srvctl enable scan_listener` command with the following syntax:

```
srvctl enable scan_listener [-i ordinal_number]
```

The only option available for this command is `-i ordinal_number`, which identifies any one of the three SCAN listeners, and takes a range of values from 1 to 3.

Example

An example of this command is:

```
$ srvctl enable scan_listener -i 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 Options

Use the `srvctl enable service` command with the following syntax:

```
srvctl enable service -d db_unique_name -s "service_name_list"
     [-i instance_name | -n node_name]
```

Table A–49 srvctl enable service Options

Option	Description
`-d db_unique_name`	Unique name for the database
`-s service_name_list`	Comma-delimited list of service names

Table A–49 (Cont.) srvctl enable service Options

Option	Description
`-i instance_name`	Name of the database instance where you want the service to run Use this option for administrator-managed databases **Note:** This option is available only with Oracle Clusterware and Oracle RAC.
`-n node_name`	Name of the node where you want the service to run Use this option for policy-managed databases **Note:** This option is available only with Oracle Clusterware and Oracle RAC.

Examples

The following example globally enables a service:

```
$ srvctl enable service -d crm -s crm
```

The following example enables a service to use a preferred instance:

```
$srvctl enable service -d crm -s crm -i crm1
```

srvctl enable vip

Enables a specific VIP.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl enable vip` command with the following syntax:

```
srvctl enable vip -i vip_name [-v]
```

Table A–50 srvctl enable vip Options

Option	Description
`-i vip_name`	The VIP name
`-v`	Verbose output

Example

An example of this command is:

```
$ srvctl enable vip -i crm1-vip -v
```

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–51 srvctl getenv Summary

Command	Description
`srvctl getenv asm` on page A-46	Gets the Oracle ASM environment values
`srvctl getenv database` on page A-46	Gets the database environment values
`srvctl getenv listener` on page A-47	Gets the listener environment values
`srvctl getenv nodeapps` on page A-47	Gets the node application environment values
`srvctl getenv vip` on page A-48	Gets the service environment values

srvctl getenv asm

Displays the values for environment variables associated with Oracle ASM.

Syntax and Options

Use the `srvctl getenv asm` command with the following syntax:

```
srvctl getenv asm [-t "name_list"]
```

Table A–52 srvctl getenv asm Options

Options	Description
`-t "name_list"`	Comma-delimited list of the names of environment variables. If this option 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 Options

Use the `srvctl getenv database` command with the following syntax:

```
srvctl getenv database -d db_unique_name [-t "name_list"]
```

Table A–53 srvctl getenv database Options

Options	Description
`-d db_unique_name`	Unique name for the database

Table A–53 (Cont.) srvctl getenv database Options

Options	Description
`-t "name_list"`	Comma-delimited list of the names of environment variables If this option 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 -d crm
```

srvctl getenv listener

Gets the environment variables for the specified listener.

Syntax and Options

Use the `srvctl getenv listener` command with the following syntax:

```
srvctl getenv listener [-l listener_name] [-t "name_list"]
```

Table A–54 srvctl getenv listener Options

Options	Description
`-l listener_name`	Listener name If this option is not specified, the name of the listener defaults to LISTENER
`-t "name_list"`	Comma-delimited list of the names of environment variables If this option is not specified, then the values of all environment variables associated with the listener are displayed.

Example

The following example lists all environment variables for the default listener:

```
$ srvctl getenv listener
```

srvctl getenv nodeapps

Gets the environment variables for the node application configurations.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl getenv nodeapps` command with the following syntax:

```
srvctl getenv nodeapps [-a] [-g] [-s] [-e] [-t "name_list"] [-v]
```

Table A–55 srvctl getenv nodeapps Options

Options	Description
`-a`	Displays the VIP address configuration
`-g`	Displays the GSD configuration
`-s`	Displays the ONS configuration
`-e`	Displays eONS daemon configuration

Table A–55 (Cont.) srvctl getenv nodeapps Options

Options	Description
`-t "name_list"`	Comma-delimited list of the names of environment variables If this option is not specified, then the values of all environment variables associated with the nodeapps are displayed.
`-v`	Verbose output

Example

The following example lists all environment variables for the node applications:

```
$ srvctl getenv nodeapps -a
```

srvctl getenv vip

Gets the environment variables for the specified VIP.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl getenv vip` command with the following syntax:

```
srvctl getenv vip -i vip_name [-t "name_list"] [-v]
```

Table A–56 srvctl getenv vip Options

Options	Description
`-i vip_name`	The name of the VIP
`-t "name_list"`	Comma-delimited list of the names of environment variables If this option is not specified, then the values of all environment variables associated with the VIP are displayed.
`-v`	Verbose output

Example

The following example lists all environment variables for the specified VIP:

```
$ srvctl getenv vip -i node1-vip
```

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 re-entered. 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–57 srvctl modify Summary

Command	Description
`srvctl modify asm` on page A-49	Modifies the configuration for Oracle ASM
`srvctl modify database` on page A-50	Modifies the configuration for a database
`srvctl modify eons` on page A-51	Modifies the network configuration for the eONS daemon
`srvctl modify filesystem` on page A-52	Modifies the user authorized to mount and unmount the Oracle ACFS volume
`srvctl modify gns` on page A-52	Modifies the GNS configuration
`srvctl modify instance` on page A-53	Modifies the configuration for an instance
`srvctl modify listener` on page A-53	Modifies the listener configuration on a node
`srvctl modify nodeapps` on page A-54	Modifies the configuration for a node application
`srvctl modify oc4j` on page A-55	Modifies the RMI port for an OC4J instance
`srvctl modify ons` on page A-55	Modifies the network configuration for the ONS daemon
`srvctl modify scan` on page A-56	Modifies the SCAN VIP configuration to match that of a specific SCAN VIP
`srvctl modify scan_listener` on page A-56	Updates the SCAN listener configuration to match that of the current SCAN VIP configuration
`srvctl modify service` on page A-57	Modifies the configuration for a service
`srvctl modify srvpool` on page A-60	Modifies a specific server pool

srvctl modify asm

Modify the listener used by Oracle ASM, the owner of the Oracle ASM software directory, the disk group discovery string used by Oracle ASM, or the SPFILE used by Oracle ASM for a single-instance database or a cluster database.

Note: To manage Oracle ASM on Oracle Database 11*g* release 2 (11.2) 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 Options

Use the `srvctl modify asm` command with the following syntax:

```
srvctl modify asm [-n node_name] [-l listener_name]
     [-d asm_diskstring] [-p spfile_path_name]
```

Table A–58 srvctl modify asm Options

Option	Description
`-n node_name`	Node name **Note:** This option is available only with Oracle Clusterware.
`-l listener_name`	The listener name that Oracle ASM registers with. **Note:** This option is available only with Oracle Clusterware and Oracle Restart.
`-d asm_diskstring`	The new Oracle ASM disk group discovery string. **Note:** This option is available only with Oracle Restart.
`-p spfile_path_name`	The path name of the new spfile to be used by Oracle ASM. **Note:** This option is only available for Oracle Restart.

Example

An example of this command to modify the configuration of Oracle ASM is:

```
$ srvctl modify asm -l lsnr1
```

srvctl modify database

Modifies the configuration for a database.

Syntax and Options

Use the `srvctl modify database` command with the following syntax:

```
srvctl modify database -d db_unique_name [-n db_name]
     [-o oracle_home] [-u user_name] [-m db_domain] [-p spfile]
     [-r {PRIMARY|PHYSICAL_STANDBY|LOGICAL_STANDBY|SNAPSHOT_STANDBY}]
     [-s start_options] [-t stop_options] [-y {AUTOMATIC | MANUAL}]
     [-g "server_pool_list"] [{-a "diskgroup_list" | -z}]
```

Table A–59 srvctl modify database Options

Option	Description
`-d db_unique_name`	Unique name for the database
`-n 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 `-d` option
`-o oracle_home`	Path for the Oracle home for the database
`-u user_name`	The name of the user that owns the Oracle home directory **Note:** If you specify the `-u` option, you must run this command in privileged mode.
`-m db_domain`	Domain for the database **Note:** If the database has the initialization parameter `DB_DOMAIN` set, then you must specify this option.
`-p spfile`	Path name of the server parameter file for the database

Table A–59 (Cont.) srvctl modify database Options

Option	Description
`-r role [PRIMARY | PHYSICAL_STANDBY | LOGICAL_STANDBY | SNAPSHOT_STANDBY]`	Role of the database in an Oracle Data Guard configuration (`PRIMARY`, `PHYSICAL_STANDBY`, `LOGICAL_STANDBY`, or `SNAPSHOT_STANDBY`)
`-s start_options`	Startup options for the database, for example, `OPEN`, `MOUNT`, or `NOMOUNT`
`-t stop_options`	Stop options for the database, for example `NORMAL`, `TRANSACTIONAL`, `IMMEDIATE`, or `ABORT`
`-y [AUTOMATIC | MANUAL]`	Management policy for the database resource
`-g 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 option 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 option is available only with Oracle Clusterware and Oracle RAC.
`-a "diskgroup_list"`	Comma-delimited list of Oracle ASM disk groups
`-z`	To remove the database's dependency on Oracle ASM disk groups

Examples

The following example changes the role of a database to a logical standby:

```
$ srvctl modify database -d crm -r logical_standby
```

The following example directs the `racTest` database to use the `SYSFILES`, `LOGS`, and `OLTP` Oracle ASM disk groups.

```
$ srvctl modify database -d racTest -a "SYSFILES,LOGS,OLTP"
```

srvctl modify eons

Modifies the ports used by the eONS daemon that is registered with Oracle Restart or with Oracle Clusterware.

Syntax and Options

Use the `srvctl modify eons` command with the following syntax:

```
srvctl modify eons [-m multicast_ip_address] [-p multicast_port_number]
     [-e eons_listen_port][-v]
```

Table A–60 srvctl modify eons Options

Option	Description
`-m multicast_ip_address`	The multicast IP address for eONS
`-p multicast_port_number`	The port number for eONS

Table A–60 (Cont.) srvctl modify eons Options

Option	Description
`-e eons_listen_port`	Local listen port for eONS daemon **Note:** If you do not specify *port*, then the utility uses the default value of 2016.
`-v`	Verbose output

Example

An example of this command is:

```
# srvctl modify eons -p 2018
```

srvctl modify filesystem

Modifies the name of the user that is authorized to mount and unmount the Oracle ACFS volume.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl modify filesystem` command with the following syntax:

```
srvctl modify filesystem -d volume_device_name -u user_name
```

Table A–61 srvctl modify filesystem Options

Option	Description
`-d volume_device_name`	Device name of the Oracle ACFS volume
`-u user_name`	Name of the user that is authorized to mount and unmount the Oracle ACFS volume

Examples

The following example changes the authorized user to `sysad` for the `RACVOL1` volume:

```
$ srvctl modify filesystem -d /dev/asm/racvol1 -u sysad
```

srvctl modify gns

Modifies the IP address or domain used by GNS

Syntax and Options

Use the `srvctl modify gns` command with the following syntax:

```
srvctl modify gns [-i ip_address] [-d domain]
```

Table A–62 srvctl modify gns Options

Option	Description
`-i ip_address`	The IP address for GNS
`-d domain`	The network domain for GNS

Example

An example of this command is:

```
$ srvctl modify gns -i 192.000.000.003
```

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:** This command is only available with Oracle Clusterware and Oracle RAC.

Syntax and Options

Use the `srvctl modify instance` command with the following syntax:

```
srvctl modify instance -d db_unique_name -i instance_name
     {-n node_name | -z}
```

Table A–63 srvctl modify instance Options

Option	Description
`-d db_unique_name`	Unique name for the database
`-i instance_name`	Database instance name
`-n node_name`	Name of the node to run the instance on
`-z`	To remove existing node instance mapping for a policy-managed database

Examples

An example of this command to relocate a database instance is:

```
$ srvctl modify instance -d crm -i crm1 -n mynode
```

The following example of this command causes the database `crm` when and if it runs on `mynode`, to use the instance name `crm1`:

```
$ srvctl modify instance -d crm -i crm1 -n mynode
```

The following example removes the directive established by the previous example:

```
$ srvctl modify instance -d crm -i crm1 -z
```

srvctl modify listener

Modifies the listener name, `ORACLE_HOME` path or the listener endpoints, either for the default listener, or a specific listener, or for all the listeners represented in a given list of listener names, that are registered with Oracle Restart or with Oracle Clusterware on the specified node.

If you want to change the name of a listener, then use the `srvctl remove listener` and `srvctl add listener` commands.

Syntax and Options

Use the `srvctl modify listener` command with the following syntax:

```
srvctl modify listener [-l listener_name] [-o oracle_home] [-u user_name]
     [-p "[TCP:]port_list[/IPC:key][/NMP:pipe_name][/TCPS:s_port][/SDP:port]"]
     [-k network_number]
```

Table A–64 srvctl modify listener Options

Option	Description
`-l listener_name`	The name of the listener. If you do not specify this option, then the utility uses the name `LISTENER`.
`-o oracle_home`	When this option is specified, SRVCTL moves the listener to run from the specified Oracle home. **Note:** When using this option, the command should be run as privileged user to enable SRVCTL to update resource ownership corresponding to the new `ORACLE_HOME` owner
`-u user_name`	The name of the operating system user who will own the specified Oracle home **Note:** This option is available only with Oracle Clusterware.
`-p "[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.
`-k network_number`	This option changes the public subnet on which the listener listens. **Note:** You should have at least one listener on the default network at all times. Do not use this option to change the network of the only listener that listens on the default network.

Example

The following example changes the TCP ports for the default listener on the node `mynode1`:

```
$ srvctl modify listener -n mynode1 -p "TCP:1521,1522"
```

srvctl modify nodeapps

Modifies the configuration for a node application.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl modify nodeapps` command with one of the following syntax models:

```
srvctl modify nodeapps [-n node_name -A new_vip_address]
[-S subnet/netmask[/if1[|if2|...]]
     [-m multicast_ip_address] [-p multicast_port_number]
     [-e eons_listen_port] [-l ons_local_port] [-r ons_remote_port]
     [-t host[:port][,host:port,...]] [-v]
```

Table A–65 srvctl modify nodeapps Options

Option	Description
`-n node_name`	Node name.
`-A new_vip_address`	Node level Virtual IP address. The address specified by name or IP must match the subnet number of the default network. **Note:** This option must be used for upgrade configurations and new non-DHCP configurations

Table A–65 (Cont.) srvctl modify nodeapps Options

Option	Description
-S *subnet*/*netmask* [/*if1*[|*if2*|...]]	Specifies a subnet number for the public network. The netmask and interfaces specified, if any, must match 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.
-m *multicast_ip_address*	Multicast IP address for the eONS daemon.
-p *multicast_port_number*	Multicast port number for the eONS daemon.
-e *eons_listen_port*	Port on which the eONS daemon listens for local client connections. If you do not specify this option, then the utility uses the default port number.
-l *ons_local_port*	Port on which the ONS daemon listens for local client connections.
-r *ons_remote_port*	Port on which the ONS daemon listens for connections from remote hosts.
-t *host*:*port*, [*host*:*port*,...]	List of *host*:*port* pairs of remote hosts that are part of the ONS 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*.
-v	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 -n mynode1 -A 100.200.300.40/255.255.255.0/eth0
```

srvctl modify oc4j

Modifies the RMI port for the OC4J instances.

Syntax and Options

Use the srvctl modify oc4j command with the following syntax:

```
srvctl modify oc4j -p oc4j_rmi_port [-v]
```

Table A–66 srvctl modify oc4j Options

Option	Description
-p *oc4j_rmi_port*	The RMI port number used by the OC4J instance
-v	Verbose output

Example

An example of this command is:

```
$ srvctl modify oc4j -p 5385
```

srvctl modify ons

Modifies the ports used by the ONS daemon that is registered with Oracle Restart or with Oracle Clusterware.

Syntax and Options

Use the `srvctl modify ons` command with the following syntax:

```
srvctl modify ons [-l ons_local_port] [-r ons_remote_port]
     [-t host[:port][,host[:port]][...]] [-v]
```

Table A–67 srvctl modify ons Options

Option	Description
`-l ons_local_port`	The ONS daemon listening port for local client connections
`-r ons_remote_port`	The ONS daemon listening port for connections from remote hosts
`-t host[:port] [,host[:port]][...]]`	A list of `host:port` pairs of remote hosts that are part of the ONS 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`.
`-v`	Display verbose output

srvctl modify scan

Modifies the SCAN VIP configuration to the match that of another SCAN VIP you specify with `scan_name`. If `scan_name` currently resolves to more IP addresses than when it was initially configured, then the utility creates new Oracle Clusterware resources for those additional IP addresses. If `scan_name` currently resolves to fewer IP addresses, then the utility removes Oracle Clusterware resources for SCAN VIP addresses with numerically higher ordinal numbers until the remaining SCAN VIP resources match the number of IP addresses to which `scan_name` resolves.

Note: This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl modify scan` command with the following syntax:

```
srvctl modify scan -n scan_name
```

The only option available for this command is `-n scan_name`, which identifies the SCAN VIP that has the configuration you want to use.

Example

An example of this command is:

```
$ srvctl modify scan -n scan1
```

srvctl modify scan_listener

Modifies the SCAN listener to match SCAN VIP's or modifies the SCAN listener endpoints.

Note: This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl modify scan_listener` command with the following syntax:

```
srvctl modify scan_listener {-p [TCP:]port[/IPC:key][/NMP:pipe_name]
```

```
[/TCPS:s_port][/SDP:port] | -u }
```

Table A–68 srvctl modify scan_listener Options

Option	Description
`-p [TCP:]port[/IPC:key] [/NMP:pipe_name] [/TCPS:s_port][/SDP:port]`	The new SCAN listener end points
`-u`	Updates SCAN listener configuration to match the current SCAN VIP configuration. This option adds new resources or removes existing SCAN listener resources to match the SCAN VIP resources.

Example

An example of this command is:

```
$ srvctl modify scan_listener -u
```

srvctl modify service

Moves a service member from one instance to another. Additionally, this command changes which instances are to be the preferred and the available instances for a service. This command supports some online modifications to the service, such as:

- Service attributes from `DBMS_SERVICE` (for example, failover delay, Runtime Load Balancing Goal, and so on) can be changed online but 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 as described on page A-82.
- 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.

As a result 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.

> **Important: 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.**

Syntax and Options

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:

> **Note:** This form of the command is only available with Oracle Clusterware.

```
srvctl modify service -d db_unique_name -s service_name -i old_instance_name
     -t new_instance_name [-f]
```

Table A–69 srvctl modify service Options for Moving a Service

Option	Description
`-d db_unique_name`	Unique name for the database
`-s service_name`	Service name
`-i old_instance_name`	Old instance name
`-t new_instance_name`	New instance name
`-f`	Disconnect all sessions during stop or relocate service operations

To change an available instance to a preferred instance for a service:

> **Note:** This form of the command is only available with Oracle Clusterware.

```
srvctl modify service -d db_unique_name -s service_name -i avail_inst_name -r [-f]
```

Table A–70 srvctl modify service Options for Changing an Available Instance to a Preferred Instance

Option	Description
`-d db_unique_name`	Unique name for the database
`-s service_name`	Service name
`-i available_inst_name`	Name of the available instance to change
`-r`	Change instance status to preferred
`-f`	Disconnect all sessions during stop or relocate service operations

To change the available and preferred status for multiple instances:

> **Note:** This form of the command is only available with Oracle Clusterware.

```
srvctl modify service -d db_unique_name -s service_name -n -i preferred_list
     [-a available_list] [-f]
```

Table A–71 srvctl modify service Options for Changing Available and Preferred Status of Multiple Instances

Option	Description
`-d` *db_unique_name*	Unique name for the database
`-s` *service_name*	Service name
`-n`	Uses only the instances named for this service (unnamed instances already assigned to the service are removed)
`-i` *preferred_instance_list*	List of preferred instances
`-a` *available_instance_list*	List of available instances
`-f`	Disconnect all sessions during stop or relocate service operations

To modify other service attributes or to modify a service for Oracle Restart:

```
srvctl modify service -d db_unique_name -s service_name
     [-c {UNIFORM|SINGLETON}] [-P {BASIC|PRECONNECT|NONE}]
     [-l {[PRIMARY] | [PHYSICAL_STANDBY] | [LOGICAL_STANDBY] | [SNAPSHOT_STANDBY]}
     [-q {TRUE|FALSE}] [-x {TRUE|FALSE}] [-j {SHORT|LONG}]
     [-B {NONE|SERVICE_TIME|THROUGHPUT}] [-e {NONE|SESSION|SELECT}]
     [-m {NONE|BASIC}] [-z failover_retries] [-w failover_delay]
     [-y {AUTOMATIC | MANUAL}]
```

Table A–72 srvctl modify service Options

Option	Description
`-d` *db_unique_name*	Unique name for the database
`-s` *service_name*	Service name
`-c {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 option is available only with Oracle Clusterware.
`-P {BASIC|PRECONNECT|NONE}`	TAF failover policy
`-l {[PRIMARY] | [PHYSICAL_STANDBY] | [LOGICAL_STANDBY] | [SNAPSHOT_STANDBY]}`	The database modes for which the service should be started automatically.
`-q {TRUE | FALSE}`	Indicates whether AQ HA notifications should be enabled (`TRUE`) for this service
`-x {TRUE | FALSE}`	Indicates whether or not Distributed Transaction Processing should be enabled for this service **Note:** This option is available only with Oracle Clusterware.
`-j {SHORT | LONG}`	Connection Load Balancing Goal
`-B {NONE | SERVICE_TIME | THROUGHPUT}`	Runtime Load Balancing Goal
`-e {NONE | SESSION | SELECT}`	Failover type
`-m {NONE | BASIC}`	Failover method

Table A–72 (Cont.) srvctl modify service Options

Option	Description
`-z failover_retries`	The number of failover retry attempts
`-w failover_delay`	The time delay between failover attempts
`-y {AUTOMATIC | MANUAL}`	Service management policy

Examples

An example of moving a service member from one instance to another is:

```
$ srvctl modify service -d crm -s crm -i crm1 -t crm2
```

An example of changing an available instance to a preferred instance is:

```
srvctl modify service -d crm -s crm -i crm1 -r
```

The following command exchanges a preferred and available instance:

```
$ srvctl modify service -d crm -s crm -n -i crm1 -a 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 Options

Use the `srvctl modify srvpool` command with the following syntax:

```
srvctl modify srvpool -g server_pool [-i importance] [-l min_size]
     [-u max_size] [-n node_name_list] [-f]
```

Table A–73 srvctl modify srvpool Options

Option	Description
`-g server_pool`	The name of the server pool to modify,
`-i importance`	The new importance of the server pool.
`-l min_size`	The new minimum size of the server pool. The default value is 0.
`-u max_size`	The new maximum size of the server pool. A value of `-1` sets the server pool maximum size to UNLIMITED.
`-n node_name_list`	A comma-delimited list of candidate server names.
`-f`	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` on the nodes `mynode3` and `mynode4`:

```
$ srvctl modify srvpool -g srvpool1 -i 0 -l 2 -u 4 -n mynode3, mynode4
```

relocate

Relocates the named service names from one named instance to another named instance. The `srvctl relocate` 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. The target instance must be on the preferred or available list for the service.

The relocation of the service is temporary until you modify the configuration. The `modify` command described on page A-61 permanently changes the service configuration.

Table A–74 srvctl relocate Summary

Command	Description
`srvctl relocate gns` on page A-61	Relocates GNS to a new node
`srvctl relocate oc4j` on page A-61	Relocates an OC4J instance to a new node
`srvctl relocate scan` on page A-62	Relocates a SCAN VIP from its current hosting server to another server within the cluster
`srvctl relocate scan_listener` on page A-62	Relocates a SCAN listener from its current hosting server to another server within the cluster
`srvctl relocate server` on page A-63	Relocates named servers to another server pool
`srvctl relocate service` on page A-63	Relocates the named service names from one named instance to another named instance

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 Options

Use the `srvctl relocate gns` command with the following syntax:

```
srvctl relocate gns [-n node_name]
```

Table A–75 srvctl relocate gns Options

Option	Description
`-n node_name`	The name of the node to which you want to move GNS

Example

An example of this command is:

```
$ srvctl relocate gns -n node1
```

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 Options

Use the `srvctl relocate oc4j` command with the following syntax:

```
srvctl relocate oc4j [-n node_name] [-v]
```

Table A–76 srvctl relocate oc4j Options

Option	Description
`-n node_name`	The name of the node to relocate the OC4J instance to.
`-v`	Display verbose output

Example

An example of this command is:

```
$ srvctl relocate oc4j -n staih01 -v
```

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 Options

Use the `srvctl relocate scan` command with the following syntax:

```
srvctl relocate scan -i ordinal_number [-n node_name]
```

Table A–77 srvctl relocate scan Options

Option	Description
`-i ordinal_number`	An ordinal number that identifies which SCAN VIP you want to relocate. The range of values you can specify for this option is 1 to 3.
`-n node_name`	The name of a single node. If you do not specify this option, then the utility chooses the node to which the SCAN VIP is relocated.

Example

An example of this command is:

```
$ srvctl relocate scan -i 1 -n 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 Options

Use the `srvctl relocate scan_listener` command with the following syntax:

```
srvctl relocate scan_listener -i ordinal_number [-n node_name]
```

Table A–78 srvctl relocate scan_listener Options

Option	Description
`-i ordinal_number`	An ordinal number that identifies which SCAN VIP you want to relocate. The range of values you can specify for this option is 1 to 3.
`-n node_name`	The name of a single node. If you do not specify this option, then the utility chooses the node to which the SCAN VIP is relocated.

Example

An example of this command is:

```
$ srvctl relocate scan_listener -i 1
```

srvctl relocate server

Relocates servers to a server pool in the cluster.

Syntax and Options

Use the `srvctl relocate server` command with the following syntax:

```
srvctl relocate server -n "server_name_list" -g server_pool_name [-f]
```

Table A–79 srvctl relocate server Options

Option	Description
`-n "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.
`-g server_pool_name`	The name of the server pool to which you want to move servers.
`-f`	Use the `-f` option to force the relocation of servers even if it means stopping some resources.

Example

An example of this command is:

```
$ srvctl relocate server -n "server1, server2" -g sp3
```

srvctl relocate service

Temporarily relocates a service member to run on another instance.

Note: This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl relocate service` command with the following syntax:

```
srvctl relocate service -d db_unique_name -s service_name
    {-c source_node -n target_node | -i old_instance_name -t new_instance_name}
    [-f]
```

Table A–80 srvctl relocate service Options

Option	Description
`-d db_unique_name`	Unique name for the database
`-s service_name`	Service name
`-c source_node`	Name of the node where the service is currently running
`-n target_node`	Name of node where the service should be relocated
`-i old_instance_name`	Old instance name
`-t new_instance_name`	New instance name
`-f`	Disconnect all sessions during stop or relocate service operations

Example

To temporarily relocate a named service member from `crm1` to `crm3`:

```
$ srvctl relocate service -d crm -s crm -i crm1 -t crm3
```

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, ONS, eONS, and listeners.

If you do not use the force flag (`-f`), then Oracle Clusterware or Oracle Restart prompts you to confirm whether to proceed. If you use the force (`-f`) option, 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 flag (`-f`) option 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 on page A-93.

Table A–81 srvctl remove Summary

Command	Description
`srvctl remove asm` on page A-66	Removes Oracle ASM instances
`srvctl remove database` on page A-66	Removes a database and configuration
`srvctl remove diskgroup` on page A-66	Removes a disk group from the Oracle Clusterware or Oracle Restart configuration
`srvctl remove eons` on page A-67	Removes the eONS daemon from the Oracle Clusterware or Oracle Restart configuration
`srvctl remove filesystem` on page A-67	Removes the configuration for an Oracle ACFS volume
`srvctl remove gns` on page A-67	Removes GNS
`srvctl remove instance` on page A-68	Removes instances and configurations of administrator-managed databases
`srvctl remove listener` on page A-68	Removes the listener from the specified node
`srvctl remove nodeapps` on page A-69	Removes node applications
`srvctl remove oc4j` on page A-69	Removes the OC4J instance configuration
`srvctl remove ons` on page A-70	Removes ONS instances
`srvctl remove scan` on page A-70	Removes all Oracle Clusterware resources for all SCAN VIPs
`srvctl remove scan_listener` on page A-70	Removes all Oracle Clusterware resources for all SCAN listeners
`srvctl remove service` on page A-71	Removes services from the Oracle Clusterware or Oracle Restart configuration

Table A–81 (Cont.) srvctl remove Summary

Command	Description
`srvctl remove srvpool` on page A-71	Removes a specific server pool
`srvctl remove vip` on page A-72	Removes specific VIPs

srvctl remove asm

Removes the Oracle ASM resource.

> **Note:** To manage Oracle ASM on Oracle Database 11*g* release 2 (11.2) 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 Options

Use the `srvctl remove asm` command with the following syntax:

```
srvctl remove asm [-f]
```

The `-f` option is the only option you can use with this command and it forcefully removes an Oracle ASM resource.

Example

An example of this command is:

```
$ srvctl remove asm -f
```

srvctl remove database

Removes a database configuration.

Syntax and Options

Use the `srvctl remove database` command with the following syntax:

```
srvctl remove database -d db_unique_name [-f] [-y]
```

Table A–82 srvctl remove database Options

Options	Description
`-d` *db_unique_name*	Unique name for the database
`-f`	Force remove
`-y`	Suppress prompts

Example

An example of this command is:

```
$ srvctl remove database -d crm
```

srvctl remove diskgroup

Removes a specific Oracle ASM disk group resource from Oracle Clusterware or Oracle Restart.

Syntax and Options

Use the `srvctl remove diskgroup` command with the following syntax:

```
srvctl remove diskgroup -g diskgroup_name [-n node_list] [-f]
```

Table A–83 srvctl remove diskgroup Options

Option	Description
`-g diskgroup_name`	The Oracle ASM disk group name.
`-n node_list`	Comma-delimited list of node nodes. **Note:** This option is available only with Oracle Clusterware.
`-f`	Force remove.

Example

An example of this command is:

```
$ srvctl remove diskgroup -g DG1 -f
```

srvctl remove eons

Removes eONS the Oracle Restart or Oracle grid infrastructure home.

```
srvctl remove eons [-f] [-v]
```

Table A–84 srvctl remove eons Options

Options	Description
`-f`	Force remove
`-v`	Verbose output

srvctl remove filesystem

Removes a specific Oracle ACFS volume resource.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl remove filesystem` command with the following syntax:

```
srvctl remove filesystem -d volume_device_name [-f]
```

Table A–85 srvctl remove filesystem Options

Option	Description
`-d volume_device_name`	The Oracle ACFS volume device name
`-f`	Force remove

Example

An example of this command is:

```
$ srvctl remove filesystem -d /dev/asm/racvol1
```

srvctl remove gns

Removes GNS from the cluster.

Note: This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl remove gns` command with the following syntax:

```
srvctl remove gns [-f]
```

The only option for this command is `-f`, which indicates GNS should be removed regardless of any errors that might occur.

Example

An example of this command is:

```
$ srvctl remove gns
```

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 -f option, 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.

Syntax and Options

Use the `srvctl remove instance` command with the following syntax:

```
srvctl remove instance -d db_unique_name -i instance_name [-f]
```

Table A–86 srvctl remove instance Options

Options	Description
`-d db_unique_name`	Unique name for the database.
`-i instance_name`	Instance name.
`-f`	Specify this option to skip checking that the instance is not running, and remove it even though it is running. This option 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 -d crm -i crm01
```

srvctl remove listener

Removes the listener from the Oracle Clusterware or Oracle Restart configuration.

Syntax and Options

Use the `srvctl remove listener` command with the following syntax:

```
srvctl remove listener [-l listener_name] [-f]
```

Table A–87 srvctl remove listener Options

Options	Description
`-l` *`listener_name`*	Name of the listener that you want to remove. If you do not specify this option, then the listener name defaults to `LISTENER`.
`-f`	Specify this option 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 listener `lsnr01` from the `node1` node:

```
$ srvctl remove listener -l lsnr01
```

srvctl remove nodeapps

Removes the node application 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

Use the `srvctl remove nodeapps` command as follows:

```
srvctl remove nodeapps [-f] [-y] [-v]
```

Example

An example of this command is:

```
$ srvctl remove nodeapps
```

Table A–88 srvctl remove nodeapps Options

Options	Description
`-f`	Force remove
`-y`	Suppress prompts
`-v`	Verbose output

srvctl remove oc4j

Removes the OC4J instance from the Oracle Clusterware configuration.

Note: This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl remove oc4j` command with the following syntax:

```
srvctl remove oc4j [-f] [-v]
```

Table A–89 srvctl remove oc4j Options

Options	Description
-f	Force remove
-v	Verbose output

srvctl remove ons

Removes ONS from the grid infrastructure home.

Note: This command is only available with Oracle Restart.

Syntax and Options

Use the `srvctl remove ons` command with the following syntax:

```
srvctl remove ons [-f] [-v]
```

Table A–90 srvctl remove ons Options

Options	Description
-f	Force remove
-v	Verbose output

srvctl remove scan

Removes Oracle Clusterware resources from all SCAN VIPs.

Note: This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl remove scan` command with the following syntax:

```
srvctl remove scan [-f]
```

The only option available for this command is `-f`, which indicates the removal should proceed regardless of any errors received.

Example

An example of this command is:

```
$ srvctl remove scan -f
```

srvctl remove scan_listener

Removes Oracle Clusterware resources from all SCAN listeners.

Note: This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl remove scan_listener` command with the following syntax:

```
srvctl remove scan_listener [-f]
```

The only option available for this command is `-f`, which indicates the removal should proceed regardless of any errors received.

Example

An example of this command is:

```
$ srvctl remove scan_listener -f
```

srvctl remove service

Removes the configuration for a service.

Syntax and Options

Use the `srvctl remove service` command as follows:

```
srvctl remove service -d db_unique_name -s service_name [-i instance_name] [-f]
```

Table A–91 srvctl remove service Options

Options	Description
`-d db_unique_name`	Unique name for the database
`-s service_name`	Service name
`-i instance_name`	Instance name **Note:** This option is available only for Oracle Clusterware.
`-f`	Force remove

Examples

An example of this command is:

```
$ srvctl remove service -d crm -s sales
```

The following example removes the services from specific instances:

```
$ srvctl remove service -d crm -s sales -i crm01,crm02
```

srvctl remove srvpool

Removes a specific server pool. If there are databases or services that depend upon this server pool, then remove them first so this operation succeeds.

If you successfully remove *`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.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl remove srvpool` command with the following syntax:

```
srvctl remove srvpool -g server_pool
```

Example

An example of this command is:

```
$ srvctl remove srvpool -g srvpool1
```

srvctl remove vip

Removes specific VIPs.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl remove vip` command with the following syntax:

```
srvctl remove vip -i "vip_name_list" [-f] [-y] [-v]
```

Table A–92 srvctl remove vip Options

Option	Description
`-i "vip_name_list"`	A comma-delimited list of VIP names surrounded by double quotation marks (`""`)
`-f`	Force remove
`-y`	Suppress prompts
`-v`	Verbose output

Example

An example of this command is:

```
$ srvctl remove vip -i "vip1,vip2,vip3" -f -y -v
```

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–93 srvctl setenv Summary

Command	Description
`srvctl setenv asm` on page A-73	Administers environment configuration for Oracle ASM
`srvctl setenv database` on page A-73	Administers cluster database environment configurations
`srvctl setenv listener` on page A-74	Administers listener environment configurations **Note:** You cannot use this command to administer SCAN listeners.
`srvctl setenv nodeapps` on page A-74	Administers node application environment configurations
`srvctl setenv vip` on page A-75	Administers VIP environment configurations

srvctl setenv asm

Administers Oracle ASM environment configurations.

Syntax and Options

Use the `srvctl setenv asm` command with the following syntax:

```
srvctl setenv asm {-t "name=val[,name=val][...]" | -T "name=val"}
```

Table A–94 srvctl setenv asm Options

Options	Description
`-t "name=val[,name=val][...]"`	Comma-delimited list of name-value pairs of environment variables
`-T "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 -t LANG=en
```

srvctl setenv database

Administers cluster database environment configurations.

Syntax and Options

Use the `srvctl setenv database` command with the following syntax:

```
srvctl setenv database -d db_unique_name
     {-t "name=val[,name=val][...]" | -T "name=val"}
```

Table A–95 srvctl setenv database Options

Options	Description
`-d db_unique_name`	Unique name for the database
`-t "name=val,..."`	Comma-delimited list of name-value pairs of environment variables
`-T "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 -d crm -t LANG=en
```

srvctl setenv listener

Administers listener environment configurations.

Syntax and Options

Use the `srvctl setenv listener` with the following syntax:

```
srvctl setenv listener [-l listener_name]
     {-t "name=val[,name=val][...]" | -T "name=val"}
```

Table A–96 srvctl setenv listener Options

Options	Description
`-l listener_name`	Name of the listener. If you do not specify this option, then the listener name defaults to `LISTENER`.
`-t "name=val"`	Comma-delimited list of name-value pairs of environment variables.
`-T "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 -t 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 Options

Use the `srvctl setenv nodeapps` command as follows:

```
srvctl setenv nodeapps {-t "name=val[,name=val][...]" | -T "name=val"} [-v]
```

Table A–97 srvctl setenv nodeapps Options

Options	Description
`-t "name=val[,name=val][...]"`	Comma-delimited list of name-value pairs of environment variables
`-T "name=val"`	Enables single environment variable to be set to a value that contains commas or other special characters
`-v`	Verbose output

Example

To set an environment variable for a node application:

```
$ srvctl setenv nodeapps -T "CLASSPATH=/usr/local/jdk/jre/rt.jar" -v
```

srvctl setenv vip

Administers cluster VIP environment configurations.

Note: This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl setenv vip` command with the following syntax:

```
srvctl setenv vip -i vip_name {-t "name=val[,name=val,...]" | -T "name=val"}
```

Table A–98 srvctl setenv vip Options

Options	Description
`-i vip_name`	Name of the VIP
`-t "name=val,..."`	Comma-delimited list of name-value pairs of environment variables
`-T "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 VIP:

```
$ srvctl setenv vip -i crm1-vip -t 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–99 srvctl start Summary

Command	Description
`srvctl start asm` on page A-76	Starts Oracle ASM instances
`srvctl start database` on page A-79	Starts the cluster database and its instances
`srvctl start diskgroup` on page A-77	Starts a specified disk group on a number of nodes
`srvctl start eons` on page A-78	Starts the eONS daemon for Oracle Restart
`srvctl start filesystem` on page A-78	Starts the Oracle ACFS volume resource
`srvctl start gns` on page A-78	Starts GNS
`srvctl start home` on page A-79	Starts Oracle Clusterware-managed or Oracle Restart-managed resources in a specific Oracle home
`srvctl start instance` on page A-79	Starts the instance
`srvctl start listener` on page A-80	Starts the specified listener or listeners
`srvctl start nodeapps` on page A-80	Starts the node applications
`srvctl start oc4j` on page A-81	Starts the OC4J instance
`srvctl start ons` on page A-81	Starts the ONS daemon for Oracle Restart
`srvctl start scan` on page A-81	Starts all SCAN VIPs
`srvctl start scan_listener` on page A-82	Starts all SCAN listeners
`srvctl start service` on page A-82	Starts the service
`srvctl start vip` on page A-83	Starts a VIP

srvctl start asm

Starts an Oracle ASM instance.

Notes: To manage Oracle ASM on Oracle Database 11*g* release 2 (11.2) 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 Options

Use the `srvctl start asm` command with the following syntax:

```
srvctl start asm [-n node_name] [-o start_options]
```

Table A–100 srvctl start asm Option

Option	Description
`-n node_name`	Node name **Note:** This option is available only with Oracle Clusterware.
`-o start_options`	Options to startup command, for example `OPEN`, `MOUNT`, or `NOMOUNT`

Examples

An example of this command to start an Oracle ASM instance on a single node of a cluster is:

```
$ srvctl start asm -n crmnode1
```

An example to start an Oracle ASM instance on all nodes in the cluster, or for a single-instance database, is:

```
$ srvctl start asm
```

srvctl start database

Starts a cluster database and its enabled instances.

Syntax and Options

Use the `srvctl start database` command with the following syntax:

```
srvctl start database -d db_unique_name [-o start_options]
```

Table A–101 srvctl start database Options

Option	Description
`-d db_unique_name`	Unique name for the database
`-o start_options`	Options for startup command (for example: `OPEN`, `MOUNT`, or `NOMOUNT`)

Example

An example of this command is:

```
$ srvctl start database -d crm -o open
```

srvctl start diskgroup

Starts a specific disk group resource on a number of specified nodes.

Syntax and Options

Use the `srvctl start diskgroup` command with the following syntax:

```
srvctl start diskgroup -g diskgroup_name [-n node_list]
```

Table A–102 srvctl start diskgroup Options

Option	Description
`-g diskgroup_name`	The Oracle ASM disk group name

Table A–102 (Cont.) srvctl start diskgroup Options

Option	Description
`-n node_list`	Comma-delimited list of node names on which to start the disk group resource **Note:** This option is available only with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl start diskgroup -g diskgroup1 -n mynode1,mynode2
```

srvctl start eons

Starts the eONS daemon.

> **Note:** This command is only available with Oracle Restart.

Syntax and Options

Use the `srvctl start eons` command with the following syntax:

```
srvctl start eons [-v]
```

There is only one option for this command, `-v`, which is used to indicate that verbose output should be displayed.

srvctl start filesystem

Starts the Oracle ACFS volume resource.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl start filesystem` command with the following syntax:

```
srvctl start filesystem -d volume_device_name [-n node_name]
```

Table A–103 srvctl start filesystem Options

Option	Description
`-d volume_device_name`	The Oracle ACFS volume device name
`-n node_name`	The name of the node on which the Oracle ACFS volume resource should be started. If you do not specify this option, then the utility starts the Oracle ACFS volume resource on all the available nodes in the cluster.

srvctl start gns

Starts GNS on a specific node, or all nodes in the cluster.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl start gns` command with the following syntax:

```
srvctl start gns [-l log_level] [-n node_name]
```

Table A–104 srvctl start gns Options

Option	Description
`-l 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).
`-n node_name`	The name of a node in the cluster where you want to start GNS.

srvctl start home

Starts all the Oracle Restart-managed or Oracle Clusterware-managed resources on the specified Oracle home.

Syntax and Options

Use the `srvctl start home` command with the following syntax:

```
srvctl start home -o Oracle_home -s state_file [-n node_name]
```

Table A–105 srvctl start home Options

Option	Description
`-o Oracle_home`	The path to the Oracle home for which you want to start the Oracle Restart or Oracle Clusterware-managed resources
`-s state_file`	The path name of the state file you specified when you ran either the `srvctl stop home` or the `srvctl status home` command.
`-n node_name`	The name of the node where the Oracle home resides. **Note:** This option is available only with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl start -o /u01/app/oracle/product/11.2.0/db_1 -s ~/state.txt
```

srvctl start instance

Starts instances in the cluster database.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl start instance` command with the following syntax:

```
srvctl start instance -d db_unique_name {-n node_name -i "instance_name_list"}
     [-o start_options]
```

In Windows, you must enclose the list of comma-delimited instance names in double quotation marks (" ").

Table A–106 srvctl start instance Options

Option	Description
`-d db_unique_name`	Unique name for the database
`-n node_name`	The name of a single node **Note:** Use this option for policy-managed databases.
`-i "instance_name_list"`	Specify either exactly one instance name or a comma-delimited list of instance names **Note:** Use this option for administrator-managed databases.
`-o start_options`	Options for startup command (for example: `OPEN`, `MOUNT`, or `NOMOUNT`)

Example

An example of starting an instance for a policy-managed database is:

```
$ srvctl start instance -d crm -n node2
```

An example of starting an instance for an administrator-managed database is:

```
$ srvctl start instance -d crm -i "crm2,crm3"
```

srvctl start listener

Starts the default listener on the specified *node_name*, or starts all of the listeners represented in a given list of listener names, that are registered with Oracle Clusterware on the given node.

Syntax and Options

Use the `srvctl start listener` command with the following syntax:

```
srvctl start listener [-n node_name] [-l listener_name_list]
```

Table A–107 srvctl start listener Options

Option	Description
`-n node_name`	Node name **Note:** This option is available only with Oracle Clusterware.
`-l listener_name_list`	Listener name If you do not specify this option, then the listener name defaults to `LISTENER`.

Example

An example of this command is:

```
$ srvctl start listener -n mynode1
```

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 Options

Use the `srvctl start nodeapps` command with the following syntax:

```
srvctl start nodeapps [-n node_name] [-v]
```

Table A–108 srvctl start nodeapps Options

Option	Description
`-n node_name`	Node name If you do not specify this option, then the utility starts the nodeapps on all active nodes in the cluster.
`-v`	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 Options

Use the `srvctl start oc4j` command with the following syntax:

```
srvctl start ocj4 [-v]
```

There is only one option for this command, `-v`, which is used to indicate that verbose output should be displayed.

srvctl start ons

Starts the ONS daemon.

> **Note:** This command is only available with Oracle Restart.

Syntax and Options

Use the `srvctl start ons` command with the following syntax:

```
srvctl start ons [-v]
```

There is only one option for this command, `-v`, 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 Options

Use the `srvctl start scan` command with the following syntax:

```
srvctl start scan [-i ordinal_number] [-n node_name]
```

Table A–109 srvctl start scan Options

Option	Description
`-i ordinal_number`	An ordinal number that identifies which SCAN VIP you want to start. The range of values you can specify for this option is 1 to 3. If you do not specify this option, then the utility starts all the SCAN VIPs.
`-n node_name`	The name of a single node. If you do not specify this option, 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 `node1` node, use the following command:

```
$ srvctl start scan -i 1 -n node1
```

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 Options

Use the `srvctl start scan_listener` command with the following syntax:

```
srvctl start scan_listener [-n node_name] [-i ordinal_number]
```

Table A–110 srvctl start scan_listener Options

Option	Description
`-i ordinal_number`	An ordinal number that identifies which SCAN Listener you want to start. The range of values you can specify for this option is 1 to 3. If you do not specify this option, then the utility starts all the SCAN listeners.
`-n node_name`	The name of a single node. If you do not specify this option, then the utility starts the SCAN listeners on all nodes in the cluster.

Example

An example of this command is:

```
$ srvctl start scan_listener -i 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.

Syntax and Options

Use the `srvctl start service` command with the following syntax:

```
srvctl start service -d db_unique_name
            [-s "service_name_list" [-n node_name | -i instance_name]]
            [-o start_options]
```

Table A–111 srvctl start service Options

Option	Description
`-d db_unique_name`	Unique name for the database
`-s "service_name_list"`	Comma-delimited list of service names. If you do not include this option, then SRVCTL starts all of the services for the specified database.
`-n node_name`	The name of the node where the service should be started. Use this option for policy-managed databases. **Note:** This option is available only with Oracle Clusterware.
`-i instance_name`	The name of the instance for which the service should be started. Use this option for administrator-managed databases. **Note:** This option is available only with Oracle Clusterware.
`-o start_options`	Options to startup command (for example: `OPEN`, `MOUNT`, or `NOMOUNT`)

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 -d crm -s crm
```

The following example starts a named service on a specified instance:

```
$ srvctl start service -d crm -s crm -i 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 Options

Use the `srvctl start vip` command with the following syntax:

```
srvctl start vip {-n node_name | -i vip_name } [-v]
```

Table A–112 srvctl start vip Options

Option	Description
`-n node_name`	Node name
`-i vip_name`	The VIP name
`-v`	Verbose start

Example

An example of this command is:

```
$ srvctl start vip -i crm1-vip -v
```

status

Displays the current state of a named database, instances, services, disk group, listener, node application, or other resource managed by Oracle Clusterware.

Table A–113 srvctl status Summary

Command	Description
`srvctl status asm` on page A-85	Displays the status of Oracle ASM instances
`srvctl status database` on page A-86	Displays the status of a database
`srvctl status diskgroup` on page A-86	Displays status of a specific disk group on a number of nodes
`srvctl status eons` on page A-87	Displays the status of the eONS daemon
`srvctl status filesystem` on page A-87	Displays the status of an Oracle ACFS volume
`srvctl status gns` on page A-87	Displays the status of GNS
`srvctl status home` on page A-87	Displays the status of the resources associated with the specified Oracle home
`srvctl status instance` on page A-88	Displays the status of a instance
`srvctl status listener` on page A-88	Displays the status of a listener resource
`srvctl status nodeapps` on page A-89	Displays the status of node applications
`srvctl status oc4j` on page A-89	Determines which node is running the Oracle QoS Management server
`srvctl status ons` on page A-89	Displays the status of ONS
`srvctl status scan` on page A-90	Displays the status of SCAN VIPs
`srvctl status scan_listener` on page A-90	Displays the status of SCAN listeners
`srvctl status server` on page A-90	Displays the status of servers
`srvctl status service` on page A-91	Displays the status of services
`srvctl status srvpool` on page A-91	Displays the status of server pools
`srvctl status vip` on page A-92	Displays the status of VIPs

srvctl status asm

Displays the status of an Oracle ASM instance.

> **Note:** To manage Oracle ASM on Oracle Database 11*g* release 2 (11.2) 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 Options

Use the `srvctl status asm` command with the following syntax:

```
srvctl status asm [-n node_name] [-a]
```

Table A–114 srvctl status asm Options

Option	Description
`-n node_name`	Node name. If you do not specify this option, the SRVCTL displays the status of all Oracle ASM instances. **Note:** This option is available only with Oracle Clusterware.
`-a`	Print detailed status information

Example

An example of this command is:

```
$ srvctl status asm -n crmnode1 -a
```

srvctl status database

Displays the status of instances and their services.

Syntax and Options

Use the `srvctl status database` command with the following syntax:

```
srvctl status database -d db_unique_name [-f] [-v]
```

Table A–115 srvctl status database Options

Option	Description
`-d db_unique_name`	Unique name for the database
`-f`	Include disabled applications
`-v`	Verbose output

Example

An example of this command is:

```
$ srvctl status database -d crm -v
```

srvctl status diskgroup

Displays the status of a specific disk group on a number of specified nodes.

Syntax and Options

Use the `srvctl status diskgroup` command with the following syntax:

```
srvctl status diskgroup -g diskgroup_name [-n node_list] [-a]
```

Table A–116 srvctl status diskgroup Options

Option	Description
`-g diskgroup_name`	The Oracle ASM disk group name
`-n node_list`	Comma-delimited list of node names on which to check status of the disk group **Note:** This option is available only with Oracle Clusterware.
`-a`	Display enabled status information of disk group

Example

An example of this command is:

```
$ srvctl status diskgroup -g diskgroup1 -n mynode1,mynode2 -a
```

srvctl status eons

Displays the current state of the eONS daemon.

Note: This command is only available with Oracle Restart.

Syntax and Options

Use the `srvctl status eons` command with the following syntax:

```
srvctl status eons
```

srvctl status filesystem

Displays the status of the specified Oracle ACFS volume.

Syntax and Options

Use the `srvctl status filesystem` command with the following syntax:

```
srvctl status filesystem -d volume_device_name
```

Table A–117 srvctl status filesystem Options

Option	Description
`-d volume_device_name`	The device name of the Oracle ACFS volume

Example

An example of this command is:

```
$ srvctl status filesystem -d /dev/asm/racvol_1
```

srvctl status gns

Displays the current state of GNS.

Note: This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl status gns` command with the following syntax:

```
srvctl status gns [-n node_name]
```

Table A–118 srvctl status gns Options

Option	Description
`-n node_name`	Specify a node on which GNS is running for which you want to display the state

srvctl status home

Displays the status of all the Oracle Restart-managed or Oracle Clusterware-managed resources for the specified Oracle home.

Syntax and Options

Use the `srvctl status home` command with the following syntax:

```
srvctl status home -o Oracle_home -s state_file [-n node_name]
```

Table A–119 srvctl status home Options

Option	Description
`-o Oracle_home`	The path to the Oracle home for which you want to start the Oracle Restart or Oracle Clusterware-managed resources
`-s state_file`	The path name the text file that holds the state information generated by this command.
`-n node_name`	The name of the node where the Oracle home resides. **Note:** This option is available only with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl status home -o /u01/app/oracle/product/11.2.0/db_1 -s ~/state.txt
```

srvctl status instance

Displays the status of instances.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl status instance` command with the following syntax:

```
srvctl status instance -d db_unique_name {-n node_name | -i "instance_name_list"}
     [-f] [-v]
```

Table A–120 srvctl status instance Options

Option	Description
`-d db_unique_name`	Unique name for the database
`-n node_name`	Node name **Note:** Use this option for policy-managed databases
`-i "instance_name_list"`	Comma-delimited list of instance names **Note:** Use this option for administrator-managed databases
`-f`	Include disabled applications
`-v`	Verbose output

Example

An example of this command is:

```
$ srvctl status instance -d crm -i "crm1,crm2" -v
```

srvctl status listener

Displays the status of listener resources.

Syntax and Options

Use the `srvctl status listener` command with the following syntax:

```
srvctl status listener [-l listener_name] [-n node_name]
```

Table A–121 srvctl status listener Options

Option	Description
`-l listener_name`	Name of a listener. If you do not specify this option, then the listener name defaults to `LISTENER`
`-n node_name`	Name of a cluster node. **Note:** This option is available only for Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl status listener -n node2
```

srvctl status nodeapps

Displays the status of node applications.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl status nodeapps` command with the following syntax:

```
srvctl status nodeapps
```

srvctl status oc4j

Determines which node is running the Oracle QoS Management server.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl status oc4j` command with the following syntax:

```
srvctl status oc4j [-n node_name]
```

The only option available for this command is the `-n node_name` option, where *node_name* is the name of a node in the cluster.

srvctl status ons

Displays the current state of the ONS daemon.

> **Note:** This command is only available with Oracle Restart.

Syntax and Options

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 Options

Use the `srvctl status scan` command with the following syntax:

```
srvctl status scan [-i ordinal_number]
```

The only option available for this command is `-i`, which is an ordinal number that identifies a specific SCAN VIP. The range of values you can specify for this option is 1 to 3. If you do not specify this option, then the utility displays the status of all SCAN VIPs in the cluster.

Example

An example of this command is:

```
$ srvctl status scan -i 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 Options

Use the `srvctl status scan_listener` command with the following syntax:

```
srvctl status scan_listener [-i ordinal_number]
```

The only option available for this command is `-i`, which is an ordinal number that identifies a specific SCAN Listener. The range of values you can specify for this option is 1 to 3. If you do not specify this option, then the utility displays the status of all SCAN listeners in the cluster.

Example

An example of this command is:

```
$ srvctl status scan_listener -i 1
```

srvctl status server

Displays the current state of named servers.

Syntax and Options

Use the `srvctl status server` command with the following syntax:

```
srvctl status server -n "server_name_list" [-a]
```

Table A–122 srvctl status server Options

Option	Description
`-n "server_name_list"`	Comma-delimited list of server names.
`-a`	Print detailed status information.

Example

The following example displays the status of a named server:

```
$ srvctl status server -n server11 -a
```

srvctl status service

Displays the status of a service.

Syntax and Options

Use the `srvctl status service` command with the following syntax:

```
srvctl status service -d db_unique_name [-s "service_name_list"] [-f] [-v]
```

Table A–123 srvctl status service Options

Option	Description
`-d db_unique_name`	Unique name for the database
`-s "service_name_list"`	Comma-delimited list of service names. If you do not specify this option, then the utility lists the status of all the services for the specified database.
`-f`	Include disabled applications
`-v`	Verbose output

Example

The following example displays the status of a named service globally across the clustered database:

```
$ srvctl status service -d crm -s crm -v
```

srvctl status srvpool

Displays all server pool names and number of servers (and names of servers if you specify the `-a` option) that are currently assigned to each server pool, if you do not specify the `-g` option. When you specify the `-g` option, the command displays the preceding information for the specified server pool.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl status srvpool` command with the following syntax:

```
srvctl status srvpool [-g server_pool] [-a]
```

Table A–124 srvctl status srvpool Options

Option	Description
`-g server_pool`	Name of the server pool
`-a`	Print detailed status information

Example

An example of this command is:

```
$ srvctl status srvpool -g srvpool1 -a
```

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 Options

Use the `srvctl status vip` command with the following syntax:

```
srvctl status vip {-n node_name | -i vip_name}
```

Table A–125 srvctl status vip Options

Option	Description
`-n node_name`	Node name
`-i vip_name`	The VIP name

Example

An example of this command is:

```
$ srvctl status vip -i node1-vip
```

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 starting or running are stopped. Objects running outside of Oracle Clusterware are not stopped. Stops node-level applications and all dependent Oracle Clusterware applications on the node.

You should disable an object that you intend to remain stopped after you issue a `srvctl stop` command. See the `srvctl disable` command starting with `srvctl disable database` on page A-33.

> **Note:** If the object is stopped and is not disabled, then it can restart as a result of another planned operation. The object does *not* restart as a result of a failure. Oracle recommends that you disable any object that should remain stopped after you issue a `stop` command.

Table A–126 srvctl stop Summary

Command	Description
`srvctl stop asm` on page A-93	Stops Oracle ASM instances
`srvctl stop database` on page A-94	Stops the cluster database
`srvctl stop diskgroup` on page A-95	Stops a specific disk group on a specified number of nodes
`srvctl stop eons` on page A-95	Stops the eONS daemon
`srvctl stop filesystem` on page A-95	Stops the Oracle ACFS volume resource
`srvctl stop gns` on page A-96	Stops GNS
`srvctl stop home` on page A-96	Stops the resources for the specified Oracle home
`srvctl stop instance` on page A-97	Stops the instance
`srvctl stop listener` on page A-98	Stops the specified listener or listeners
`srvctl stop nodeapps` on page A-98	Stops the node-level applications
`srvctl stop oc4j` on page A-99	Stops the OC4J instance
`srvctl stop ons` on page A-99	Stops ONS
`srvctl stop scan` on page A-100	Stops all SCAN VIPs
`srvctl stop scan_listener` on page A-100	Stops all SCAN listeners
`srvctl stop service` on page A-101	Stops the service
`srvctl stop vip` on page A-101	Stops VIP resources

srvctl stop asm

Stops an Oracle ASM instance.

Notes:

- To manage Oracle ASM on Oracle Database 11*g* release 2 (11.2) 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.

Syntax and Options

Use the `srvctl stop asm` command with the following syntax:

```
srvctl stop asm [-n node_name] [-o stop_options] [-f]
```

Table A–127 srvctl stop asm Option

Option	Description
`-n node_name`	The name of the node on which to stop the Oracle ASM instance. If you do not specify this option, then the utility stops the Oracle ASM instance on every active node in the cluster. **Note:** This option is available only with Oracle Clusterware.
`-o stop_options`	Options for shutdown command, for example, `NORMAL`, `TRANSACTIONAL`, `IMMEDIATE`, or `ABORT`
`-f`	Use this option to stop disk groups, file systems and databases that depend on Oracle ASM

Example

An example of this command is:

```
$ srvctl stop asm -n crmnode1 -i asm1
```

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 Options

Use the `srvctl stop database` command with the following syntax:

```
srvctl stop database -d db_unique_name [-o stop_options] [-f]
```

Table A–128 srvctl stop database Options

Option	Description
`-d db_unique_name`	Unique name for the database.
`-o stop_options`	Use this option to specify `shutdown` command options (for example: `NORMAL`, `TRANSACTIONAL`, `IMMEDIATE`, or `ABORT`).
`-f`	This option stops the database, its instances, its services, and any resources that depend on those services

Example

An example of this command is:

```
$ srvctl stop database -d crm
```

srvctl stop diskgroup

Stops a specific disk group resource on a number of specified nodes.

Syntax and Options

Use the `srvctl stop diskgroup` command with the following syntax:

```
srvctl stop diskgroup -g diskgroup_name [-n node_list] [-f]
```

Table A–129 srvctl stop diskgroup Options

Option	Description
`-g` *diskgroup_name*	The Oracle ASM disk group name
`-n` *node_list*	Comma-delimited list of node names on which to stop the disk group **Note:** This option is available only with Oracle Clusterware.
`-f`	This option 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 -g diskgroup1 -n mynode1,mynode2 -f
```

srvctl stop eons

Stops the eONS daemon.

> **Note:** This command is only available with Oracle Restart.

Syntax and Options

Use the `srvctl stop eons` command with the following syntax:

```
srvctl stop eons [-f] [-v]
```

The only options for this command are the `-v` option, which specifies that verbose output should be displayed and the `-f` option, which indicates the daemon should be shut down forcefully, if necessary.

Example

An example of this command is:

```
srvctl stop eons -f
```

srvctl stop filesystem

Stops the Oracle ACFS volume resource.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl stop filesystem` command with the following syntax:

```
srvctl stop filesystem -d volume_device_name [-n node_name] [-f]
```

Table A–130 srvctl stop filesystem Options

Option	Description
`-d volume_device_name`	The Oracle ACFS volume device name
`-n node_name`	The name of a node If you do not specify this option, then the utility stops the volume resource on all active nodes in the cluster.
`-f`	This option stops the file system and also stops any databases or other resources that depend on this file system.

Example

An example of this command is:

```
$ srvctl stop filesystem -d /dev/asm/racvol_1 -f
```

srvctl stop gns

Stops GNS for the cluster.

Note: This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl stop gns` command with the following syntax:

```
srvctl stop gns [-n node_name [-v] [-f]
```

Table A–131 srvctl stop gns Options

Option	Description
`-n node_name`	The name of a node on which GNS is running
`-v`	Verbose output
`-f`	Force stop

Example

An example of this command is:

```
$ srvctl stop gns
```

srvctl stop home

Stops all the Oracle Restart-managed or Oracle Clusterware-managed resources on the specified Oracle home.

Syntax and Options

Use the `srvctl stop home` command with the following syntax:

```
srvctl stop home -o Oracle_home -s state_file [-t stop_options]
                 [-n node_name] [-f]
```

Table A–132 srvctl stop home Options

Option	Description
-o *Oracle_home*	The path to the Oracle home for which you want to start the Oracle Restart or Oracle Clusterware-managed resources
-s *state_file*	The path name where you want the state file to be written.
-t *stop_options*	Shutdown options for the database (for example: NORMAL, TRANSACTIONAL, IMMEDIATE, or ABORT)
-n *node_name*	The name of the node where the Oracle home resides. **Note:** This option is available only with Oracle Clusterware.
-f	Stop the resources even if errors are reported.

Example

An example of this command is:

```
$ srvctl stop -o /u01/app/oracle/product/11.2.0/db_1 -s ~/state.txt
```

srvctl stop instance

Stops instances and stops any services running on those instances.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl stop instance` command with the following syntax:

```
srvctl stop instance -d db_unique_name {[-n node_name]|[-i "instance_name_list"]}
     [-o stop_options] [-f]
```

In Windows, you must enclose the list of comma-delimited instance names in double quotation marks (`" "`).

Table A–133 srvctl stop instance Options

Option	Description
-d *db_unique_name*	Unique name for the database
-n *node_name*	The name of a single node **Note:** Use this option for policy-managed databases.
-i "*instance_name*"	Comma-delimited instance names enclosed in double quotation marks **Note:** Use this option for administrator-managed databases.
-o *stop_options*	Options for shutdown command (for example: NORMAL, TRANSACTIONAL, IMMEDIATE, or ABORT)
-f	This option stops the instance and its running services and any resources that depend on those services.

Example

An example of stopping an instance in a policy-managed database is:

```
$ srvctl stop instance -d crm -n node1
```

An example of stopping an instance in an administrator-managed database is:

```
$ srvctl stop instance -d crm -i crm1
```

srvctl stop listener

Stops the default listener on the specified *node_name*, or the listeners represented in a given list of listener names, that are registered with Oracle Clusterware on the given node.

This command can also be used to stop a listener on a single-instance database from the single-instance database home. SRVCTL does not accept the `-n` option, however, when run from a single-instance database home.

Syntax and Options

Use the `srvctl stop listener` command with the following syntax:

```
srvctl stop listener [-n node_name] [-l listener_name_list] [-f]
```

Table A–134 srvctl stop listener Options

Option	Description
`-n` *node_name*	The name of a single node on which a particular listener runs. **Note:** This option is available only with Oracle Clusterware.
`-l` *listener_name*	The name of the listener you want to stop. If you do not specify this option, then the listener name defaults to `LISTENER`.
`-f`	Force stop

Example

An example of this command is:

```
$ srvctl stop listener -n 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 Options

Use the `srvctl stop nodeapps` command with the following syntax:

```
srvctl stop nodeapps [-n node_name] [-r] [-v]
```

Table A–135 srvctl stop nodeapps Options

Option	Description
`-n` *node_name*	Node name If you do not specify this option, then the utility stops the nodeapps on all active nodes in the cluster.

Table A–135 (Cont.) srvctl stop nodeapps Options

Option	Description
`-r`	Relocate VIP **Note:** If you specify this option, then you must also specify the `-n` *`node_name`* option.
`-v`	Display verbose output

Example

An example of this command is:

```
$ srvctl stop nodeapps
```

srvctl stop oc4j

Stops the OC4J instance.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl stop oc4j` command with the following syntax:

```
srvctl stop oc4j [-f] [-v]
```

Table A–136 srvctl stop oc4j Options

Option	Description
`-f`	Force stop of the SCAN Listener.
`-v`	Display verbose output

Example

An example of this command is:

```
$ srvctl stop oc4j -f -v
```

srvctl stop ons

Stops the ONS daemon.

> **Note:** This command is only available with Oracle Restart.

Syntax and Options

Use the `srvctl stop ons` command with the following syntax:

```
srvctl stop ons [-v]
```

The only option for this command is the `-v` option, which specifies that verbose output should be displayed.

Example

An example of this command is:

```
$ srvctl stop ons -v
```

srvctl stop scan

Stops 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 Options

Use the `srvctl stop scan` command with the following syntax:

```
srvctl stop scan [-i ordinal_number] [-f]
```

Table A–137 srvctl stop scan Options

Option	Description
`-i` *ordinal_number*	An ordinal number that identifies which SCAN VIP you want to stop. The range of values you can specify for this option is 1 to 3. If you do not specify this option, then the utility stops all the SCAN VIPs.
`-f`	Force stop of the SCAN VIP.

Example

An example of this command is:

```
$ srvctl stop scan -i 1
```

srvctl stop scan_listener

Stops all SCAN listeners, by default, or a specific listener identified by *ordinal_number*.

> **Note:** This command is only available with Oracle Clusterware.

Syntax and Options

Use the `srvctl start scan_listener` command with the following syntax:

```
srvctl stop scan_listener [-i ordinal_number] [-f]
```

Table A–138 srvctl stop scan_listener Options

Option	Description
`-i` *ordinal_number*	An ordinal number that identifies which SCAN listener you want to stop. The range of values you can specify for this option is 1 to 3. If you do not specify this option, then the utility stops all the SCAN listeners.
`-f`	Force stop of the SCAN listener.

Example

An example of this command is:

```
$ srvctl stop scan_listener -i 1
```

srvctl stop service

Stops one or more services globally across the cluster database, or on the specified instance.

Syntax and Options

Use the `srvctl stop service` command with the following syntax:

```
srvctl stop service -d db_unique_name [-s "service_name_list"
     [-n node_name | -i instance_name] [-f]
```

In Windows, you must enclose the list of comma-delimited service names in double quotation marks (`""`).

Table A–139 srvctl stop service Options

Option	Description
`-d db_unique_name`	Unique name for the database
`-s 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.
`-n node_name`	The name of the node on which the services should be stopped. Use this option for policy-managed databases. **Note:** This option is available only with Oracle Clusterware.
`-i instance_name`	The name of the instance for which the services should be stopped. Use this option for administrator-managed databases. **Note:** This option is available only with Oracle Clusterware.
`-f`	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 `-f` option, 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 globally across a cluster database:

```
$ srvctl stop service -d crm -s crm
```

The following example stops a service on a specified instance:

```
$ srvctl stop  service -d crm -s crm -i 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 Options

Use the `srvctl stop vip` command with the following syntax:

```
srvctl stop vip {-n node_name | -i vip_name} [-r] [-v]
```

Table A–140 srvctl stop vip Options

Option	Description
`-n node_name`	Node name
`-i vip_name`	The VIP name
`-r`	Relocate VIP **Note:** If you specify this option, you must also specify the `-n node_name` option.
`-v`	Verbose output

Example

To stop all the VIPs on `mynode1`, including any failed-over VIPS, use the following command:

```
$ srvctl stop vip -n mynode1 -v
```

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–141 srvctl unsetenv Command Summary

Command	Description
`srvctl unsetenv asm` on page A-103	Unsets the value for one or more Oracle ASM environment variables
`srvctl unsetenv database` on page A-103	Unsets the value for one or more cluster database environment variables
`srvctl unsetenv listener` on page A-104	Unsets the value for one or more listener environment variables.
`srvctl unsetenv nodeapps` on page A-104	Unsets the value for one or more node application environment variables
`srvctl unsetenv vip` on page A-104	Unsets the value for one or more VIP environment variables

srvctl unsetenv asm

Unsets the Oracle ASM environment configurations.

Syntax and Options

Use the `srvctl unsetenv asm` command as follows:

```
srvctl unsetenv asm -t "name_list"
```

Table A–142 srvctl unsetenv asm Options

Options	Description
`-t "name_list"`	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 -t CLASSPATH
```

srvctl unsetenv database

Unsets the cluster database environment configurations.

Syntax and Options

Use the `srvctl unsetenv database` command as follows:

```
srvctl unsetenv database -d db_unique_name -t "name_list"
```

Table A–143 srvctl unsetenv database Options

Options	Description
`-d db_unique_name`	Unique name for the database

Table A–143 (Cont.) srvctl unsetenv database Options

Options	Description
`-t "name_list"`	Names of environment variables

Example
The following example unsets the environment configuration for a cluster database environment variable:

```
$ srvctl unsetenv database -d crm -t CLASSPATH
```

srvctl unsetenv listener

Unsets the environment configuration for a listener.

Syntax and Options
Use the `srvctl unsetenv listener` command as follows:

```
srvctl unsetenv listener [-l listener_name] -t "name_list"
```

Table A–144 srvctl unsetenv listener Options

Options	Description
`-l listener_name`	Name of the listener. If you do not specify this option, then the name of the listener defaults to `LISTENER`.
`-t "name_list"`	Names of environment variables

Example
The following example unsets the environment configuration for the node applications:

```
$ srvctl unsetenv listener -t "TNS_ADMIN"
```

srvctl unsetenv nodeapps

Unsets the environment configuration for the node applications.

Syntax and Options
Use the `srvctl unsetenv nodeapps` command as follows:

```
srvctl unsetenv nodeapps -t "name_list" [-v]
```

Table A–145 srvctl unsetenv nodeapps Options

Options	Description
`-t "name_list"`	Names and values of environment variables
`-v`	Display verbose output

Example
The following example unsets the environment configuration for the node applications:

```
$ srvctl unsetenv nodeapps -t "test_var1,test_var2"
```

srvctl unsetenv vip

Unsets the environment configuration for the specified cluster VIP.

Syntax and Options

Use the `srvctl unsetenv vip` command as follows:

```
srvctl unsetenv vip -i vip_name -t "name_list" [-v]
```

Table A–146 srvctl unsetenv vip Options

Options	Description
`-i vip_name`	Name of the cluster VIP
`-t "name_list"`	Comma-delimited list of the names of environment variables
`-v`	Verbose output

Example

The following example unsets the `CLASSPATH` environment variable for a cluster VIP:

```
$ srvctl unsetenv vip -i crm2-vip -t CLASSPATH
```

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

> **Note:** The trace and log files 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 single-instance 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. 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 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. 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* for more information about monitoring errors and alerts in trace files

Trace files are 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.

  > **See Also:** *Oracle Database Administrator's Guide* for information about managing diagnostic data

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` parameter in the `initdb_name.ora` initialization parameter file.

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 the 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 Oracle Enterprise Manager Database Control graphical user interfaces (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
```

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.

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.

cluster

Multiple interconnected computers or servers that appear as if they are one server to end users and applications.

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 database

The generic term for a Oracle RAC database.

Cluster Ready Services Daemon (CRSD)

The primary Oracle Clusterware process that performs high availability recovery and management operations, such as maintaining the 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.

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.

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 needs to be located at both sites, and therefore one needs to look at alternatives for mirroring the storage.

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.

forced disk write

In Real Application Clusters, 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 implement 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 a Real Application Clusters 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 by default.

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 11*g* release 2 (11.2) combines these infrastructure products into one software installation called the *grid infrastructure home* (`Grid_home`).

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 in the event of 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 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 up 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.

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.

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 an instance resides.

Object Link Manager (OLM)

The Oracle interface that maps symbolic links to logical drives and displays them in the OLM graphical user interface.

OCLSMON

A process that runs on nodes in Oracle RAC environments.`OCLSOMON` monitors CSS hangs that result from load or scheduling issues. If `OCLSOMON` detects either of these, the process shuts down the affected node to prevent database corruption.

OCLSVMON

OCLSVMON is a process that runs when vendor clusterware is installed. Basically due to the existence of this process, the Oracle Clusterware can delay log flushes. This delay is useful in diagnosing problems because the vendor clusterware allows a delay of eviction processing until after log files have flushed.

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.

OPROCD

A Linux or UNIX process monitor for a cluster. Note that this process will only appear on platforms that do not use vendor clusterware with Oracle Clusterware.

Oracle Cluster File Systems

Oracle offers two cluster file systems, OCFS for Windows and OCFS2 for Linux. While OCFS for Windows 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. The OCR also manages information about Oracle Clusterware resource profiles for customized applications.

Oracle Enterprise Manager Configuration Assistant (EMCA)

A graphical user interface-based configuration assistant that you can use to configure Oracle Enterprise Manager features.

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 Hardware Assisted Resilient Data (HARD)

The Oracle Hardware Assisted Resilient Data (HARD) Initiative prevents data corruptions. The HARD initiative uses Oracle data validation algorithms inside storage devices to prevent writing corrupted data to permanent storage.

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 single-instance 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 (OMF)

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 Services (ONS)

A publish and subscribe service for communicating information about all FAN events.

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 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).

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.

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.

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 Real Application Clusters 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 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.

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.

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.

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.

voting disk

A file that manages information about node membership.

Index

Symbols

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X