Contents

Send Us Your Comments ...................................................................................................................... xiii

Preface .................................................................................................................................................... xv
  Audience ................................................................................................................................................. xv
  Documentation Accessibility .................................................................................................................... xv
  Structure .............................................................................................................................................. xvi
  Related Documents .............................................................................................................................. xix
  Conventions ........................................................................................................................................... xix

What's New in Real Application Clusters Administration and Deployment? ..... xxiii
  Oracle Database 10g Release 2 (10.2) New Features in RAC Administration and Deployment xxiii
  Oracle Database 10g Release 1 (10.1) New Features for RAC Administration .................. xxx

1 Introduction to Oracle Clusterware and Oracle Real Application Clusters
  Oracle Clusterware and Oracle Real Application Clusters ............................................................... 1-1
  The Oracle Clusterware Architecture and Oracle Clusterware Processing ................................. 1-2
    Oracle Clusterware Software Component Processing Details ...................................................... 1-2
    The Oracle Clusterware Software Components ............................................................................ 1-4
  The Real Application Clusters Architecture and Real Application Clusters Processing ............. 1-5
    The Real Application Clusters Software Components ................................................................. 1-6
  Oracle Clusterware Components and High Availability ............................................................... 1-7
    The Oracle Clusterware Voting Disk and Oracle Cluster Registry ............................................ 1-7
    The Oracle Clusterware High Availability Framework and the Application Programming Interface 1-8
  Workload Management with Real Application Clusters ............................................................... 1-8
  Introduction to Installing the Oracle Clusterware and Real Application Clusters .................... 1-9
    Oracle Clusterware Installation Process Description ................................................................. 1-9
    Real Application Clusters Installation and Database Creation Process Description ............ 1-10
  Additional Considerations and Features for Real Application Clusters ..................................... 1-11
  Managing Real Application Clusters Environments ................................................................... 1-11
    Designing Real Application Clusters Environments ................................................................. 1-12
    Administrative Tools for Real Application Clusters Environments ........................................ 1-12
    Monitoring Real Application Clusters Environments ............................................................... 1-13
    Evaluating Performance in Real Application Clusters Environments ................................... 1-14
# 2 Introduction to Oracle Clusterware and RAC Administration and Deployment

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Application Clusters Documentation Overview</td>
<td>2-1</td>
</tr>
<tr>
<td>Platform-Specific Oracle Real Application Clusters Installation and Configuration Guides.</td>
<td>2-2</td>
</tr>
<tr>
<td>Introduction to Administering Real Application Clusters</td>
<td>2-2</td>
</tr>
<tr>
<td>Administering Real Application Clusters</td>
<td>2-2</td>
</tr>
<tr>
<td>Voting Disk and Oracle Cluster Registry Device Administration</td>
<td>2-3</td>
</tr>
<tr>
<td>Database Instance Management and Database Administration in RAC</td>
<td>2-3</td>
</tr>
<tr>
<td>Storage Management in Real Application Clusters</td>
<td>2-3</td>
</tr>
<tr>
<td>Oracle Clusterware for Real Application Clusters</td>
<td>2-4</td>
</tr>
<tr>
<td>Additional Real Application Clusters Administrative Topics</td>
<td>2-4</td>
</tr>
<tr>
<td>Overview of Using Enterprise Manager with Real Application Clusters</td>
<td>2-4</td>
</tr>
<tr>
<td>Overview of Deploying Applications on Real Application Clusters</td>
<td>2-5</td>
</tr>
<tr>
<td>Code Changes are Not Required for Applications</td>
<td>2-6</td>
</tr>
<tr>
<td>Implementing Oracle Features with Real Application Clusters</td>
<td>2-6</td>
</tr>
<tr>
<td>Automatic Storage Management</td>
<td>2-6</td>
</tr>
<tr>
<td>Cluster File Systems in Real Application Clusters</td>
<td>2-6</td>
</tr>
<tr>
<td>Storage Management Features and Real Application Clusters</td>
<td>2-7</td>
</tr>
<tr>
<td>Services in Oracle Database 10g</td>
<td>2-7</td>
</tr>
<tr>
<td>The Oracle Clusterware and High Availability in Real Application Clusters</td>
<td>2-8</td>
</tr>
<tr>
<td>Additional Oracle High Availability Features and Solutions</td>
<td>2-9</td>
</tr>
</tbody>
</table>

# 3 Administering Oracle Clusterware Components

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administering Voting Disks in Real Application Clusters</td>
<td>3-1</td>
</tr>
<tr>
<td>Backing up Voting Disks</td>
<td>3-1</td>
</tr>
<tr>
<td>Recovering Voting Disks</td>
<td>3-2</td>
</tr>
<tr>
<td>Changing the Voting Disk Configuration after Installing Real Application Clusters</td>
<td>3-2</td>
</tr>
<tr>
<td>Administering the Oracle Cluster Registry in Real Application Clusters</td>
<td>3-2</td>
</tr>
<tr>
<td>Adding, Replacing, Repairing, and Removing the OCR</td>
<td>3-3</td>
</tr>
<tr>
<td>Managing Backups and Recovering the OCR Using OCR Backup Files</td>
<td>3-5</td>
</tr>
<tr>
<td>Diagnosing OCR Problems with the OCRDUMP and OCRCHECK Utilities</td>
<td>3-7</td>
</tr>
<tr>
<td>Overriding the Oracle Cluster Registry Data Loss Protection Mechanism</td>
<td>3-7</td>
</tr>
<tr>
<td>Administering the Oracle Cluster Registry with OCR Exports</td>
<td>3-8</td>
</tr>
<tr>
<td>Implementing the Oracle Hardware Assisted Resilient Data Initiative for the OCR</td>
<td>3-9</td>
</tr>
<tr>
<td>Upgrading and Downgrading the OCR Configuration in Real Application Clusters</td>
<td>3-9</td>
</tr>
</tbody>
</table>

# 4 Administering Storage

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview of Storage in Oracle Real Application Clusters</td>
<td>4-1</td>
</tr>
<tr>
<td>Datafile Access in Real Application Clusters</td>
<td>4-2</td>
</tr>
<tr>
<td>Redo Log File Storage in Real Application Clusters</td>
<td>4-2</td>
</tr>
<tr>
<td>Automatic Undo Management in Real Application Clusters</td>
<td>4-2</td>
</tr>
<tr>
<td>Automatic Storage Management in Real Application Clusters</td>
<td>4-2</td>
</tr>
<tr>
<td>Automatic Storage Management Components in RAC</td>
<td>4-3</td>
</tr>
<tr>
<td>Modifying Disk Group Configurations for ASM in RAC</td>
<td>4-3</td>
</tr>
<tr>
<td>Standalone ASM Disk Group Management</td>
<td>4-3</td>
</tr>
<tr>
<td>Administering ASM Instances and Disk Groups with Enterprise Manager in RAC</td>
<td>4-4</td>
</tr>
<tr>
<td>Administering ASM Instances with SRVCTL in RAC</td>
<td>4-5</td>
</tr>
</tbody>
</table>
5 Administering Database Instances and Cluster Databases

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview of Real Application Clusters Management Tools</td>
<td>5-1</td>
</tr>
<tr>
<td>Overview of Administering Real Application Clusters with Enterprise Manager</td>
<td>5-1</td>
</tr>
<tr>
<td>Overview of Administering Real Application Clusters with SQL*Plus</td>
<td>5-2</td>
</tr>
<tr>
<td>Overview of Administering Real Application Clusters with SRVCTL</td>
<td>5-3</td>
</tr>
<tr>
<td>Starting and Stopping Instances and Real Application Clusters Databases</td>
<td>5-3</td>
</tr>
<tr>
<td>Starting Up and Shutting Down with Enterprise Manager</td>
<td>5-4</td>
</tr>
<tr>
<td>Starting Up and Shutting Down with SQL*Plus</td>
<td>5-4</td>
</tr>
<tr>
<td>Starting Up and Shutting Down with SRVCTL</td>
<td>5-5</td>
</tr>
<tr>
<td>Customizing How Oracle Clusterware Manages RAC Databases</td>
<td>5-6</td>
</tr>
<tr>
<td>Switching Between the Automatic and Manual Policies</td>
<td>5-6</td>
</tr>
<tr>
<td>Overview of Initialization Parameter Files in Real Application Clusters</td>
<td>5-7</td>
</tr>
<tr>
<td>Setting Server Parameter File Parameter Values for Real Application Clusters</td>
<td>5-7</td>
</tr>
<tr>
<td>Parameter File Search Order in Real Application Clusters</td>
<td>5-8</td>
</tr>
<tr>
<td>Initialization Parameter Use in Real Application Clusters</td>
<td>5-8</td>
</tr>
<tr>
<td>Parameters that Must Have Identical Settings on All Instances</td>
<td>5-9</td>
</tr>
<tr>
<td>Parameters That Must Have Unique Settings on All Instances</td>
<td>5-9</td>
</tr>
<tr>
<td>Parameters that Should Have Identical Settings on All Instances</td>
<td>5-9</td>
</tr>
<tr>
<td>Summary of Parameter Use in Real Application Clusters Databases</td>
<td>5-10</td>
</tr>
<tr>
<td>Backing Up the Server Parameter File</td>
<td>5-13</td>
</tr>
</tbody>
</table>

6 Introduction to Workload Management

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction to Workload Management and Application High Availability</td>
<td>6-1</td>
</tr>
<tr>
<td>Services Deployment Options</td>
<td>6-2</td>
</tr>
<tr>
<td>Using Oracle Services</td>
<td>6-2</td>
</tr>
<tr>
<td>Default Service Connections</td>
<td>6-3</td>
</tr>
<tr>
<td>Connection Load Balancing</td>
<td>6-3</td>
</tr>
<tr>
<td>Fast Application Notification</td>
<td>6-4</td>
</tr>
<tr>
<td>Overview of Fast Application Notification</td>
<td>6-4</td>
</tr>
<tr>
<td>Application High Availability with Services and FAN</td>
<td>6-5</td>
</tr>
<tr>
<td>Managing Unplanned Outages</td>
<td>6-6</td>
</tr>
<tr>
<td>Managing Planned Outages</td>
<td>6-6</td>
</tr>
<tr>
<td>Fast Application Notification High Availability Events</td>
<td>6-6</td>
</tr>
<tr>
<td>Using Fast Application Notification Callouts</td>
<td>6-7</td>
</tr>
<tr>
<td>Load Balancing Advisory</td>
<td>6-8</td>
</tr>
<tr>
<td>Overview of the Load Balancing Advisory</td>
<td>6-8</td>
</tr>
<tr>
<td>Administering the Load Balancing Advisory</td>
<td>6-8</td>
</tr>
<tr>
<td>Load Balancing Advisory FAN Events</td>
<td>6-9</td>
</tr>
<tr>
<td>Oracle Clients that are Integrated with Fast Application Notification</td>
<td>6-10</td>
</tr>
<tr>
<td>Enabling Oracle Call Interface to Receive FAN High Availability Events</td>
<td>6-10</td>
</tr>
<tr>
<td>Enabling Java Database Connectivity to Receive FAN Events</td>
<td>6-11</td>
</tr>
<tr>
<td>Enabling ODP.NET to Receive FAN High Availability Events</td>
<td>6-13</td>
</tr>
<tr>
<td>Enabling ODP.NET to Receive FAN Load Balancing Advisory Events</td>
<td>6-14</td>
</tr>
<tr>
<td>Services and Distributed Transaction Processing in RAC</td>
<td>6-15</td>
</tr>
<tr>
<td>Enabling Distributed Transaction Processing for Services</td>
<td>6-16</td>
</tr>
<tr>
<td>Administering Services</td>
<td>6-17</td>
</tr>
</tbody>
</table>
Administering Services with Enterprise Manager, DBCA, PL/SQL, and SRVCTL .......... 6-19
Administering Services with Enterprise Manager .................................................. 6-19
Administering Services with the Database Configuration Assistant ................... 6-21
Administering Services with the PL/SQL DBMS_SERVICE Package .................. 6-22
Administering Services with SRVCTL ............................................................... 6-23
Measuring Performance by Service Using the Automatic Workload Repository ....... 6-24
Service Thresholds and Alerts .............................................................................. 6-26

7 Configuring Recovery Manager and Archiving
Overview of Configuring RMAN for Real Application Clusters ......................... 7-1
Configuring the RMAN Snapshot Control File Location ....................................... 7-1
Configuring the RMAN Control File and SPFILE Autobackup Feature ................. 7-2
Configuring Channels for RMAN in Real Application Clusters .......................... 7-2
Configuring Channels to use Automatic Workload Balancing ........................... 7-3
Configuring Channels to Use a Specific Channel ............................................. 7-3
Managing Archived Redo Logs Using RMAN in Real Application Clusters ......... 7-3
Archived Redo Log File Conventions in RAC ................................................... 7-4
RMAN Archiving Configuration Scenarios ......................................................... 7-5
  Automatic Storage Management and Cluster File System Archiving Scheme ...... 7-5
  Non-Cluster File System Local Archiving Scheme ....................................... 7-6
Changing the Archiving Mode in Real Application Clusters ............................... 7-8
  Monitoring the Archiver Processes ............................................................. 7-8

8 Managing Backup and Recovery
RMAN Backup Scenarios for Real Application Clusters .................................... 8-1
  Non-Cluster File System Backup Scheme ..................................................... 8-1
RMAN Restore Scenarios for Real Application Clusters ..................................... 8-1
  Cluster File System Restore Scheme ......................................................... 8-2
  Non-Cluster File System Restore Scheme .................................................. 8-2
  Using RMAN or Enterprise Manager to Restore the Server Parameter File (SPFILE) ......................................................... 8-2
RMAN Recovery Through Resetlogs in Real Application Clusters .................... 8-3
RMAN and Oracle Net in Real Application Clusters .......................................... 8-3
Instance Recovery in Real Application Clusters ................................................ 8-3
  Single Node Failure in Real Application Clusters ....................................... 8-3
  Multiple-Node Failures in Real Application Clusters .................................... 8-4
  Using RMAN to Create Backups in Real Application Clusters .................... 8-4
  Channel Connections to Cluster Instances ............................................... 8-4
  Node Affinity Awareness of Fast Connections ....................................... 8-5
  Deleting Archived Redo Logs after a Successful Backup ............................. 8-5
  Autolocation for Backup and Restore Commands .................................... 8-5
Media Recovery in Real Application Clusters .................................................. 8-6
Parallel Recovery in Real Application Clusters ............................................... 8-6
  Parallel Recovery with RMAN ................................................................. 8-6
  Disabling Parallel Recovery ................................................................. 8-6
Using a Flash Recovery Area in Real Application Clusters ................................. 8-7

9 Administrative Options

Enterprise Manager Tasks for Real Application Clusters ................................................................. 9-1
Using Enterprise Manager Grid Control to Discover Nodes and Instances ........................................ 9-2
Enterprise Manager Pages for Real Application Clusters ................................................................. 9-2
  Databases Summary Page ................................................................................................................. 9-2
  Cluster Database Home Page ........................................................................................................... 9-3
  Cluster Database Instances Pages .................................................................................................... 9-3
  The Databases Overview Page for Real Application Clusters .......................................................... 9-4
The Cluster Home Page for Real Application Clusters ....................................................................... 9-4

Real Application Clusters Administration Procedures for Enterprise Manager .................................... 9-4

Administering Enterprise Manager Jobs in Real Application Clusters .............................................. 9-4
Administering Alerts in Real Application Clusters with Enterprise Manager ....................................... 9-5
Performing Scheduled Maintenance Using Defined Blackouts in Enterprise Manager ..................... 9-5

Additional Information About SQL*Plus in Real Application Clusters ............................................. 9-5

How SQL*Plus Commands Affect Instances ..................................................................................... 9-5
Verifying that Instances are Running .................................................................................................. 9-6

Quiesced Real Application Clusters Databases .................................................................................. 9-6

Quiesced State and Cold Backups ......................................................................................................... 9-7

Administering System and Network Interfaces with OIFCFG .......................................................... 9-7

Defining Network Interfaces with OIFCFG ....................................................................................... 9-7
Syntax and Commands for the OIFCFG Command-Line Tool ............................................................ 9-8

Changing VIP Addresses .................................................................................................................... 9-9

10 Adding and Deleting Nodes and Instances

Cloning Oracle Clusterware and RAC Software in Grid Environments ............................................. 10-1
Adding and Deleting Instances Using Enterprise Manager ............................................................... 10-2

Quick-Start Procedures for Adding and Deleting Nodes and Instances on UNIX-Based Systems ...... 10-3

Adding an Oracle Clusterware Home to a New Node ......................................................................... 10-3
Adding an Oracle Home with RAC on UNIX-Based Systems ............................................................... 10-4
Deleting an Oracle Home with RAC from a Node.............................................................................. 10-5
Deleting an Oracle Clusterware Home from a Node .......................................................................... 10-7
Adding and Deleting Instances Using the DBCA Silent Mode .............................................................. 10-8

Overview of Node Addition Procedures ............................................................................................. 10-9

Step 1: Connecting New Nodes to the Cluster .................................................................................... 10-10
  Making Physical Connections ............................................................................................................ 10-10
  Installing the Operating System ........................................................................................................ 10-10
  Creating Oracle Users ....................................................................................................................... 10-10
  Verify the Installation with the Cluster Verification Utility .............................................................. 10-10
  Checking the Installation ................................................................................................................... 10-11

Step 2: Extending Clusterware and Oracle Software to New Nodes .................................................. 10-12
  Adding Nodes at the Vendor Clusterware Layer ............................................................................... 10-12
  Adding Nodes at the Oracle Clusterware Layer .............................................................................. 10-12

Step 3: Preparing Storage for RAC on New Nodes ............................................................................ 10-15
Raw Device Storage Preparation for New Nodes ............................................................... 10-16
Step 4: Adding Nodes at the Oracle RAC Database Layer .................................................. 10-16
Step 5: Adding Database Instances to New Nodes ............................................................. 10-18
Adding Nodes that Already Have Clusterware and Oracle Software to a Cluster .......... 10-20
Deleting Instances from Real Application Clusters Databases ........................................ 10-21
Deleting Nodes from Real Application Clusters Databases .............................................. 10-22
ASM Instance Clean-Up Procedures for Node Deletion .................................................... 10-25

11 Design and Deployment Techniques

Service Configuration Recommendations for High Availability ....................................... 11-1
Service Topologies and Workload Management in Real Application Clusters ................. 11-1
Recommended Real Application Clusters Service Configurations ................................... 11-1
Automatic Workload Repository ......................................................................................... 11-2
Setting Service Levels and Thresholds ............................................................................... 11-2
How the Oracle Clusterware Manages Service Relocation .............................................. 11-3

General Database Deployment Topics for Real Application Clusters ............................. 11-3
Tablespace Use in Real Application Clusters ................................................................. 11-3
Object Creation and Performance in Real Application Clusters ..................................... 11-3
Node Addition and Deletion and the SYSAUX Tablespace in Real Application Clusters .. 11-3
Distributed Transactions and Real Application Clusters ..................................................... 11-4

12 Monitoring Performance

Overview of Monitoring Real Application Clusters Databases ........................................... 12-1
Verifying the Interconnect Settings for Real Application Clusters .................................... 12-1
Influencing Interconnect Processing .................................................................................. 12-1

Performance Views in Real Application Clusters ............................................................. 12-2
Real Application Clusters Performance Statistics ............................................................ 12-2
The Content of Real Application Clusters Statistics ......................................................... 12-2

Automatic Workload Repository in Real Application Clusters Environments ............... 12-3

Monitoring Real Application Clusters Statistics and Events ........................................... 12-3
RAC Statistics and Events in AWR and Statspack Reports ............................................... 12-3
Real Application Clusters Wait Events ............................................................................. 12-3
Monitoring Performance by Analyzing GCS and GES Statistics .................................... 12-4
Analyzing Cache Fusion Transfer Impact Using GCS Statistics ........................................ 12-5
Analyzing Response Times Based on Wait Events ............................................................ 12-6

Monitoring Performance with Oracle Enterprise Manager ............................................ 12-7
Overview of Enterprise Manager Monitoring .................................................................... 12-7
Collection-Based Monitoring ............................................................................................ 12-8
Real-Time Performance Monitoring .................................................................................. 12-9
Using the Cluster Database Performance Page ............................................................... 12-10
Using the Cluster Database Instance Performance Page .................................................. 12-16
Using the Cluster Performance Page ................................................................................ 12-17
Using the Cluster Interconnects Page ................................................................................. 12-17

13 Making Applications Highly Available Using the Oracle Clusterware

Overview of Using the Oracle Clusterware Commands to Enable High Availability .......... 13-1
Overview of Managing Custom Applications with Oracle Clusterware Commands ....................... 13-3
Creating Application Profiles ........................................................................................................ 13-3
   Application Resource Profiles ................................................................................................. 13-4
Example of Using Oracle Clusterware Commands to Create Application Resources .............. 13-7
   Using crs_profile to Create An Application Resource Profile ............................................. 13-7
   The Oracle Clusterware Required Resources List ................................................................. 13-8
   Application Placement Policies .............................................................................................. 13-8
   Optional Resources in Placement Decisions ............................................................................ 13-9
Oracle Clusterware Action Program Guidelines ................................................................. 13-9
   How the Oracle Clusterware Runs Action Programs ............................................................. 13-10
Using Oracle Clusterware Commands ........................................................................................ 13-11
   Registering Application Resources ...................................................................................... 13-12
   Starting Application Resources ............................................................................................ 13-12
   Relocating Applications and Application Resources ............................................................. 13-13
   Stopping Applications and Application Resources ............................................................... 13-13
   Managing Automatic Oracle Clusterware Resource Operations for Action Scripts ............. 13-14
   Unregistering Applications and Application Resources ......................................................... 13-15
   Displaying Clusterware Application and Application Resource Status Information .......... 13-15

14 Application-Specific Deployment Topics

General Deployment Strategies for Real Application Clusters-Based Applications .................. 14-1
Deploying OLTP Applications in Real Application Clusters ....................................................... 14-1
   Flexible Implementation with Cache Fusion ............................................................................ 14-1
Deploying Data Warehouse Applications with Real Application Clusters ............................. 14-2
   Speed-Up for Data Warehouse Applications on Real Application Clusters ......................... 14-2
   Parallel Execution in Data Warehouse Systems and RAC ...................................................... 14-2
   Using Parallel Instance Groups .............................................................................................. 14-2
Data Security Considerations in Real Application Clusters ..................................................... 14-2

A Troubleshooting

Overview of Troubleshooting Real Application Clusters ............................................................ A-1
Diagnosing the Oracle Clusterware High Availability Components ........................................ A-1
   Dynamic Debugging ................................................................................................................ A-2
   Component Level Debugging .................................................................................................. A-2
   Oracle Clusterware Software Shutdown and Startup .............................................................. A-3
   Enabling and Disabling Oracle Clusterware Daemons ............................................................. A-3
   Diagnostics Collection Script .................................................................................................. A-4
   The Oracle Clusterware Alerts ................................................................................................. A-4
   Resource Debugging ................................................................................................................ A-4
   Checking the Health of the Clusterware .................................................................................. A-4
   Clusterware Log Files and the Unified Log Directory Structure ............................................. A-5
   Troubleshooting the Oracle Cluster Registry ........................................................................... A-6
   Enabling Additional Tracing for Real Application Clusters High Availability ....................... A-8
Diagnosing Oracle Real Application Clusters Components .................................................... A-9
   Where to Find Files for Analyzing Errors ............................................................................... A-9
   Using Instance-Specific Alert Files in Real Application Clusters .......................................... A-10
Enabling Tracing for Java-Based Tools and Utilities in Real Application Clusters .......... A-11
Resolving Pending Shutdown Issues ........................................................................... A-11
Using the Cluster Verification Utility .......................................................................... A-11
Cluster Verification Utility Requirements ..................................................................... A-12
Understanding CVU Commands, Help, Output, and Nodelist Shortcuts ......................... A-13
Performing Various CVU Tests ................................................................................... A-15
Known Issues for the Cluster Verification Utility ......................................................... A-19

B High Availability Oracle Clusterware Command-Line Reference and C API

Using Oracle Clusterware Commands .......................................................................... B-1
Application Profile Syntax .......................................................................................... B-1
Security and Permissions ......................................................................................... B-1
The Oracle Clusterware Commands ............................................................................. B-2
   crs_getperm ............................................................................................................ B-3
   crs_profile ............................................................................................................ B-3
   crs_register ......................................................................................................... B-6
   crs_relocate ......................................................................................................... B-8
   crs_setperm ......................................................................................................... B-10
   crs_stat .............................................................................................................. B-10
   crs_start ............................................................................................................ B-13
   crs_stop ............................................................................................................. B-14
   crs_unregister .................................................................................................... B-15
C Application Programming Interface to the Oracle Clusterware ................................. B-16
   clscrs_init_crs .................................................................................................... B-16
   clscrs_term_crs ................................................................................................... B-16
   clscrs_getnodename ............................................................................................ B-17
   clscrs_env_create ............................................................................................... B-17
   clscrs_env_set ..................................................................................................... B-17
   clscrs_env_delete ............................................................................................... B-18
   clscrs_env_format ............................................................................................... B-18
   clscrs_start_resource ......................................................................................... B-18
   clscrs_stop_resource ......................................................................................... B-18
   clscrs_check_resource ....................................................................................... B-19
   clscrs_register_resource .................................................................................... B-19
   clscrs_unregister_resource ............................................................................... B-20
   clscrs_stat .......................................................................................................... B-21
Functions for Managing Resource Structures ................................................................ B-22
   Export Operations ............................................................................................... B-22

C Oracle Cluster Registry Configuration Tool Command Syntax

The OCR Configuration Tool Command Syntax and Options ......................................... C-1

D Server Control Utility Reference

Overview of SRVCTL for Administering Real Application Clusters .............................. D-1
Guidelines for Using SRVCTL in Real Application Clusters ...................................... D-1
Obtaining Command-Line Help for SRVCTL ............................................................. D-2
SRVCTL Command Syntax and Options ................................................................. D-2
SRVCTL Cluster Database Configuration Tasks ..................................................... D-2
SRVCTL General Cluster Database Administration Tasks ...................................... D-3
SRVCTL Node-Level Tasks ..................................................................................... D-3
SRVCTL Command Reference ................................................................................. D-3
SRVCTL Commands ............................................................................................... D-4
   SRVCTL Commands Summary ............................................................................... D-4
   SRVCTL Objects Summary .................................................................................... D-4
   srvctl add ................................................................................................................ D-5
   srvctl config .......................................................................................................... D-8
   srvctl enable ........................................................................................................... D-10
   srvctl disable ......................................................................................................... D-12
   srvctl start ............................................................................................................ D-14
   srvctl stop ............................................................................................................. D-17
   srvctl modify ....................................................................................................... D-20
   srvctl relocate ..................................................................................................... D-24
   srvctl status ......................................................................................................... D-25
   srvctl getenv ....................................................................................................... D-27
   srvctl setenv and unsetenv ................................................................................ D-29
   srvctl remove ....................................................................................................... D-33

E Oracle Real Application Clusters Tools Messages

   Overview of Real Application Clusters-Specific Messages .................................... E-1
   Prefixes and Message Codes for RAC-Specific Messages ...................................... E-2
   Types of Real Application Clusters Messages and Related Files ............................ E-2
   PRKA—Cluster Node Applications Messages ..................................................... E-2
   PRKC—Cluster Command Messages .................................................................... E-4
   PRKD—Global Services Daemon Messages .......................................................... E-14
   PRKE—Global Services Daemon Controller Utility Messages ............................. E-14
   PRKH—Server Manager (SRVM) Messages .......................................................... E-14
   PRKI—Cluster Pre-Install Messages ..................................................................... E-16
   PRKN—Server Manager (SRVM) System Library Messages ............................... E-17
   PRKO—Server Control (SRVCTL) Utility Messages ............................................. E-18
   PRKP—Cluster Database Management Messages ............................................... E-21
   PRKR—Cluster Registry Messages ...................................................................... E-29
   PRKS—Automatic Storage Management Messages ............................................. E-34
   PRKU—Command-Line Parser Utility Messages ............................................... E-38
   PRKV—Virtual IP Configuration Assistant Messages .......................................... E-38

Index
Send Us Your Comments

Oracle Clusterware and RAC Administration and Deployment Guide 10g Release 2 (10.2)
B14197-01

Oracle welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, please indicate the title and part number of the documentation and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: infodev_us@oracle.com
- FAX: (650) 506-7227. Attn: Server Technologies Documentation Manager
- Postal service:
  Oracle Corporation
  Server Technologies Documentation Manager
  500 Oracle Parkway, Mailstop 4op11
  Redwood Shores, CA 94065
  USA

If you would like a reply, please give your name, address, telephone number, and electronic mail address (optional).

If you have problems with the software, please contact your local Oracle Support Services.
Preface

The *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Administration and Deployment Guide* introduces Oracle Clusterware and Real Application Clusters (RAC) by describing the architectures of these products as well as providing an overview of them. This book also describes administrative and deployment topics for Oracle Clusterware and RAC.

Information in this manual applies to RAC as it runs on all platforms. 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
- Documentation Accessibility
- Structure
- Related Documents
- Conventions

### Audience

The *Oracle Database Oracle Clusterware and 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 RAC databases
- Administer and manage RAC databases
- Manage and troubleshoot clusters and networks that use RAC

To use this document, you should be familiar with the administrative procedures described in *Oracle Database 2 Day DBA* and the *Oracle Database Administrator’s Guide*. You should also read *Oracle Database Concepts* to become familiar with Oracle database concepts. You should also be familiar with installing and configuring RAC as described in the platform-specific Real Application Clusters installation guides.

### Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. 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.

**TTY Access to Oracle Support Services**

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

**Structure**

This document is organized as follows:

**Chapter 1, "Introduction to Oracle Clusterware and Oracle Real Application Clusters"**

This chapter introduces Oracle Clusterware and RAC by describing the Oracle Clusterware and RAC architectures as well as the component processing of these products. The chapter describes the Oracle Clusterware API that you can use on both single-instance Oracle databases as well as RAC databases to use the Oracle high availability features. This chapter briefly describes workload management, high availability, and services. This chapter also describes the Oracle Clusterware and RAC installation processes. In addition, the chapter introduces RAC-specific design, deployment, and administration topics.

**Chapter 2, "Introduction to Oracle Clusterware and RAC Administration and Deployment"**

This chapter introduces RAC administrative and deployment topics by describing the Oracle Database 10g RAC documentation set. This chapter provides an overview of database and instance management, storage management, and services management. This chapter also provides an overview of administering RAC environments with Oracle Enterprise Manager, as well as a description of application deployment in RAC environments. In addition, this chapter introduces the Oracle Clusterware, and additional high availability features such as load balancing, Recovery Manager (RMAN), and Data Guard.
Chapter 3, "Administering Oracle Clusterware Components"
This chapter describes how to administer the voting disks and the Oracle Cluster Registry (OCR) disks. This chapter explains how to backup and recover these components as well as how to add, replace, repair, remove, and troubleshoot them.

Chapter 4, "Administering Storage"
This chapter explains how to administer storage components in RAC by describing Automatic Storage Management (ASM) administration, datafiles, redo log files, and automatic undo tablespace administration.

Chapter 5, "Administering Database Instances and Cluster Databases"
This chapter explains how to administer RAC database instances and RAC databases. This chapter provides an overview of the tools that you can use to administer RAC environments, such as Enterprise Manager, SQL*Plus, and the Server Control Utility (SRVCTL). This chapter also describes how to start and stop instances and databases using these tools. In addition, this chapter describes parameter use in RAC and the administration of the Server Parameter File (SPFILE).

Chapter 6, "Introduction to Workload Management"
This chapter describes how to administer services and other high availability features in RAC environments. This chapter explains Connection Pool Load Balancing and connection load balancing and also describes how to change VIP addresses, how automatic restarts after failures function, and OCI event notification. In addition, this chapter explains how to administer high availability features such as services with Enterprise Manager and the Server Control Utility (SRVCTL).

Chapter 7, "Configuring Recovery Manager and Archiving"
This chapter explains how to configure Recovery Manager (RMAN) for use with RAC databases. This chapter discusses how to configure the snapshot control file location, the RMAN control file, and the automatic Server Parameter File (SPFILE) autobackup feature. This chapter also explains how to manage archived redo logs and provides RMAN configuration scenarios for RAC environments. In addition, this chapter explains how to change the archiving log mode in RAC.

Chapter 8, "Managing Backup and Recovery"
This chapter explains how to administer backup and recovery in RAC. This chapter explains instance recovery and the chapter provides Recovery Manager (RMAN) backup and restore scenarios for RAC environments. This chapter also describes RAC instance recovery, parallel backup, recovery with SQL*Plus, and using the Flash Recovery Area in RAC.

Chapter 9, "Administrative Options"
This chapter describes administrative tasks or options within Oracle tools that are specific to RAC. These include Enterprise Manager tasks, special tasks that you perform with SQL*Plus, quiescing RAC databases, and administering network interfaces with the Oracle Interface Configuration Tool (OIFCFG).

Chapter 10, "Adding and Deleting Nodes and Instances"
This chapter explains how to add and delete nodes and instances in RAC environments.
Chapter 11, "Design and Deployment Techniques"
This chapter describes design and deployment techniques for deploying applications in RAC environments. This chapter also describes high availability topics such as deploying services and how the Oracle Clusterware manages services within RAC.

Chapter 12, "Monitoring Performance"
This chapter describes how to monitor performance in RAC environments. This includes an explanation of how to verify interconnect settings, descriptions of performance views for RAC, and descriptions of RAC-specific statistics and how to monitor them. This chapter also describes the Automatic Workload Repository (AWR) and how to analyze RAC statistics and RAC wait events. This chapter also explains how to monitor performance in RAC environments with Enterprise Manager. This chapter describes the Enterprise Manager Cluster Performance and Cluster Database Performance Pages. The chapter also describes the Cluster Cache Coherency and Coherency Instances Pages.

Chapter 13, "Making Applications Highly Available Using the Oracle Clusterware"
This chapter explains how use Oracle Clusterware commands to extend the high availability of the Oracle Clusterware framework to custom applications. That is, you can use RAC high availability features to make your custom applications highly available.

Appendix A, "Troubleshooting"
This appendix explains important troubleshooting topics such as how to contact Oracle Support Services, using log and trace files, and using the Cluster Verification Utility (CVU).

Chapter B, "High Availability Oracle Clusterware Command-Line Reference and C API"
This appendix describes Oracle Clusterware application programming interface (API) command reference information. This chapter describes the Oracle Clusterware commands, information about the C-code application API, and an explanation about the functions for managing resource structures. Use information in this appendix with Chapter 13, "Making Applications Highly Available Using the Oracle Clusterware".

Appendix C, "Oracle Cluster Registry Configuration Tool Command Syntax"
This appendix describes the OCRCONFIG syntax, which is the Oracle Cluster Registry (OCR) configuration tool.

Appendix D, "Server Control Utility Reference"
This appendix is the Server Control Utility (SRVCTL) command reference. This appendix includes such topics as SRVCTL command syntax and options, RAC database configuration and administration tasks that you can perform with SRVCTL, as well as descriptions of node-level tasks.

Appendix E, "Oracle Real Application Clusters Tools Messages"
This appendix describes the messages for the Java-based RAC management tools.

Glossary
The glossary defines important RAC-specific terms.
Related Documents

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

- Platform-specific Real Application Clusters installation guides
- Oracle Database 2 Day DBA
- Oracle Database Administrator’s Guide
- Oracle Database Net Services Administrator’s Guide
- Oracle Database Platform Guide for Microsoft Windows (32-Bit)
- Oracle Database 10g Administrator’s Reference Release 1 (10.1) for UNIX Systems: AIX-Based Systems, HP-UX, Tru64 UNIX, Linux, and the Solaris Operating System (SPARC)

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

Printed documentation is available for sale in the Oracle Store at

http://oraclestore.oracle.com/

To download free release notes, installation documentation, white papers, or other collateral, please visit the Oracle Technology Network (OTN). You must register online before using OTN; registration is free and can be done at

http://www.oracle.com/membership/

If you already have a username and password for OTN, then you can go directly to the documentation section of the OTN Web site at

http://www.oracle.com/documentation/

Conventions

This section describes the conventions used in the text and code examples of this documentation set. This section describes:

- Conventions in Text
- Conventions in Code Examples
- Conventions for Windows Operating Systems

Conventions in Text

We use various conventions in text to help you more quickly identify special terms. The following table describes those conventions and provides examples of their use.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bold</td>
<td>Bold typeface indicates terms that are defined in the text or terms that appear in a glossary, or both.</td>
<td>When you specify this clause, you create an index-organized table.</td>
</tr>
</tbody>
</table>
| Italics    | Italic typeface indicates book titles or emphasis. | Oracle Database Concepts
Ensure that the recovery catalog and target database do not reside on the same disk. |
Conventions in Code Examples

Code examples illustrate SQL, PL/SQL, SQL*Plus, or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT username FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPPERCASE</td>
<td>Uppercase monospace typeface indicates elements supplied by the system.</td>
<td>You can specify this clause only for a NUMBER column.</td>
</tr>
<tr>
<td>monospace</td>
<td>Such elements include parameters, privileges, datatypes, RMAN keywords,</td>
<td>You can back up the database by using the BACKUP command.</td>
</tr>
<tr>
<td>(fixed-width)</td>
<td>SQL keywords, SQL*Plus or utility commands, packages and methods, as</td>
<td>Query the TABLE_NAME column in the USER_TABLES data dictionary view.</td>
</tr>
<tr>
<td>font</td>
<td>well as system-supplied column names, database objects and structures,</td>
<td>Use the DBMS_STATS.GENERATE_STATS procedure.</td>
</tr>
<tr>
<td></td>
<td>usernames, and roles.</td>
<td></td>
</tr>
<tr>
<td>LOWERCASE</td>
<td>Lowercase monospace typeface indicates executable programs, filenames,</td>
<td>Enter sqlplus to start SQL*Plus.</td>
</tr>
<tr>
<td>monospace</td>
<td>directory names, and sample user-supplied elements. Such elements</td>
<td>The password is specified in the orapwd file.</td>
</tr>
<tr>
<td>(fixed-width)</td>
<td>include computer and database names, net service names and connect</td>
<td>Back up the datafiles and control files in the /disk1/oracle/dbs</td>
</tr>
<tr>
<td>font</td>
<td>identifiers, user-supplied database objects and structures, column</td>
<td>directory.</td>
</tr>
<tr>
<td></td>
<td>names, packages and classes, usernames and roles, program units, and</td>
<td>The department_id, department_name, and location_id columns are in the</td>
</tr>
<tr>
<td></td>
<td>parameter values.</td>
<td>hr.departments table.</td>
</tr>
<tr>
<td>Note: Some</td>
<td>Note: Some programmatic elements use a mixture of UPPERCASE and</td>
<td>Set the QUERY_REWRITE_ENABLED initialization parameter to true.</td>
</tr>
<tr>
<td>programmatic</td>
<td>lowercase. Enter these elements as shown.</td>
<td>Connect as oe user.</td>
</tr>
<tr>
<td>elements use a</td>
<td></td>
<td>The JRepUtil class implements these methods.</td>
</tr>
<tr>
<td>mixture of</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPPERCASE and</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lowercase.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOWERCASE</td>
<td>Lowercase italic monospace font represents placeholders or variables.</td>
<td>You can specify the parallel_clause.</td>
</tr>
<tr>
<td>italic</td>
<td></td>
<td>Run old_release.SQL where old_release refers to the release you installed</td>
</tr>
<tr>
<td>monospace</td>
<td></td>
<td>prior to upgrading.</td>
</tr>
<tr>
<td>(fixed-width)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Italic</td>
<td>Italics indicates placeholders or variables for which you must supply</td>
<td>CONNECT SYSTEM/system_password</td>
</tr>
<tr>
<td></td>
<td>particular values.</td>
<td>DB_NAME = database_name</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Conventions in Code Examples**

Code examples illustrate SQL, PL/SQL, SQL*Plus, or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT username FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ ]</td>
<td>Anything enclosed in brackets is optional.</td>
<td>DECIMAL (digits [, precision ])</td>
</tr>
<tr>
<td>{ }</td>
<td>Braces are used for grouping items.</td>
<td>{ENABLE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A vertical bar represents a choice of two options.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>In addition, ellipsis points can mean an omission in code examples or</td>
</tr>
<tr>
<td>...</td>
<td>Ellipsis points mean repetition in syntax descriptions.</td>
<td>CREATE TABLE ... AS subquery;</td>
</tr>
<tr>
<td></td>
<td>In addition, ellipsis points can mean an omission in code examples or</td>
<td>SELECT col1, col2, ..., coln FROM employees;</td>
</tr>
<tr>
<td></td>
<td>text.</td>
<td></td>
</tr>
<tr>
<td>Other symbols</td>
<td>You must use symbols other than brackets ([</td>
<td>]), braces ({</td>
</tr>
<tr>
<td></td>
<td></td>
<td>acct      CONSTANT NUMBER(4) := 3;</td>
</tr>
<tr>
<td>Italic</td>
<td>Italics indicates placeholders or variables for which you must supply</td>
<td>CONNECT SYSTEM/system_password</td>
</tr>
<tr>
<td></td>
<td>particular values.</td>
<td>DB_NAME = database_name</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Conventions for Windows Operating Systems

The following table describes conventions for Windows operating systems and provides examples of their use.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPPERCASE</td>
<td>Uppercase typeface indicates elements supplied by the system. We show these terms in uppercase in order to distinguish them from terms you define. Unless terms appear in brackets, enter them in the order and with the spelling shown. Because these terms are not case sensitive, you can use them in either UPPERCASE or lowercase.</td>
<td>SELECT last_name, employee_id FROM employees; SELECT * FROM USER_TABLES; DROP TABLE hr.employees;</td>
</tr>
<tr>
<td>lowercase</td>
<td>Lowercase typeface indicates user-defined programmatic elements, such as names of tables, columns, or files. Note: Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.</td>
<td>SELECT last_name, employee_id FROM employees; sqlplus hr/hr CREATE USER mjones IDENTIFIED BY ty3MU9;</td>
</tr>
</tbody>
</table>

**Conventions for Windows Operating Systems**

The following table describes conventions for Windows operating systems and provides examples of their use.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Choose Start &gt; menu item</td>
<td>How to start a program.</td>
<td>To start the Database Configuration Assistant, choose Start &gt; Programs &gt; Oracle - HOME_NAME &gt; Configuration and Migration Tools &gt; Database Configuration Assistant.</td>
</tr>
<tr>
<td>File and directory names</td>
<td>File and directory names are not case sensitive. The following special characters are not allowed: left angle bracket (&lt;), right angle bracket (&gt;), colon (:), double quotation marks (&quot;), slash (/), pipe (</td>
<td>), and dash (-). The special character backslash () is treated as an element separator, even when it appears in quotes. If the filename begins with , then Windows assumes it uses the Universal Naming Convention.</td>
</tr>
<tr>
<td>C:&gt;</td>
<td>Represents the Windows command prompt of the current hard disk drive. The escape character in a command prompt is the caret (^). Your prompt reflects the subdirectory in which you are working. Referred to as the command prompt in this manual.</td>
<td>C:\oracle\oradata&gt;</td>
</tr>
<tr>
<td>Special characters</td>
<td>The backslash () special character is sometimes required as an escape character for the double quotation mark (&quot;) special character at the Windows command prompt. Parentheses and the single quotation mark (') do not require an escape character. Refer to your Windows operating system documentation for more information on escape and special characters.</td>
<td>C:&gt;exp HR/HR TABLES=employees QUERY=&quot;WHERE job_id='SA_REP' and salary&lt;8000&quot;</td>
</tr>
<tr>
<td>HOME_NAME</td>
<td>Represents the Oracle home name. The home name can be up to 16 alphanumeric characters. The only special character allowed in the home name is the underscore.</td>
<td>C:&gt; net start OracleHOME_NAME\TNSListener</td>
</tr>
<tr>
<td>Convention</td>
<td>Meaning</td>
<td>Example</td>
</tr>
<tr>
<td>------------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>ORACLE_HOME and ORACLE_BASE</td>
<td>In releases prior to Oracle8i release 8.1.3, when you installed Oracle components, all subdirectories were located under a top level ORACLE_HOME directory. The default for Windows NT was C:\orant. This release complies with Optimal Flexible Architecture (OFA) guidelines. All subdirectories are not under a top level ORACLE_HOME directory. There is a top level directory called ORACLE_BASE that by default is C:\oracle\product\10.1.0. If you install the latest Oracle release on a computer with no other Oracle software installed, then the default setting for the first Oracle home directory is C:\oracle\product\10.1.0\db_n, where n is the latest Oracle home number. The Oracle home directory is located directly under ORACLE_BASE.</td>
<td>Go to the ORACLE_BASE\ORACLE_HOME\rdbms\admin directory.</td>
</tr>
</tbody>
</table>

All directory path examples in this guide follow OFA conventions.

Refer to Oracle Database Installation Guide for Microsoft Windows (32-Bit) for additional information about OFA compliances and for information about installing Oracle products in non-OFA compliant directories.
What's New in Real Application Clusters Administration and Deployment?

The following section describes the new administration and deployment features for Oracle Real Application Clusters (RAC) for the following releases:

- Oracle Database 10g Release 2 (10.2) New Features in RAC Administration and Deployment
- Oracle Database 10g Release 1 (10.1) New Features for RAC Administration

See Also: Oracle Database New Features for a complete description of the new features in Oracle Database 10g release 2 (10.2) and Oracle Database 2 Day DBA for a beginner’s guide to RAC administration

Oracle Database 10g Release 2 (10.2) New Features in RAC Administration and Deployment

This section describes the Oracle Database 10g release 2 (10.2) features for RAC administration.

- Oracle Clusterware

Oracle Clusterware, formerly known as Cluster Ready Services (CRS) is an integrated cluster management solution that enables you to link multiple servers so that they function as a single system or cluster. This simplifies the infrastructure required for a RAC database by providing cluster software that is integrated with the Oracle Database. In addition, while continuing to be required for RAC databases, the Oracle Clusterware is also available for use with single-instance databases and applications that you deploy on clusters.

See Also:

- Chapter 1, “Introduction to Oracle Clusterware and Oracle Real Application Clusters”, Chapter 13, “Making Applications Highly Available Using the Oracle Clusterware”, and Appendix B, ”High Availability Oracle Clusterware Command-Line Reference and C API” for more information about the Oracle Clusterware, the Oracle Clusterware API, and the Oracle Clusterware API commands
- Your platform-specific Oracle Clusterware and RAC installation guide for more information about installing Oracle Clusterware
Real Application Clusters Deployment Book Merged into the Administration Book and Oracle Clusterware designated as a separate component

Information that was previously in the Oracle Real Application Clusters Deployment and Performance Guide and the Real Application Clusters Administrator’s Guide has been incorporated into this book. In addition, because you can now use Oracle Clusterware for single-instance Oracle databases within clustered environments, the Oracle Clusterware is presented as a separate, portable Oracle component. The title of this book is Oracle Database Oracle Clusterware and Oracle Real Application Clusters Administration and Deployment Guide.

See Also: Your platform-specific Oracle Clusterware and RAC installation guide for more information about installing Oracle Clusterware

Oracle Clusterware High Availability Application Programming Interface Support

You can now use the Oracle Clusterware high availability components to make your customized applications highly available. You can configure high availability features for applications that are inside or outside of the Oracle database environment. In addition, the functionality provided by some of the command-line interfaces is available through the application programming interfaces (API).

See Also: Chapter 13, "Making Applications Highly Available Using the Oracle Clusterware", and Appendix B, "High Availability Oracle Clusterware Command-Line Reference and C API" for more information about the Oracle Clusterware, the Oracle Clusterware API, and the Oracle Clusterware API commands

Cluster Verification Utility

The Cluster Verification Utility (CVU) verifies a wide range of cluster and RAC-specific components such as shared storage devices, networking configurations, system requirements, the Oracle Clusterware, groups, and users. You can use CVU for pre-installation checks as well as for post-installation checks of your cluster environment. You can also use CVU to verify your environment when performing administrative operations such as installation, storage management, node addition, and troubleshooting. The OUI runs CVU immediately after you successfully install the Oracle Clusterware.

See Also: "Using the Cluster Verification Utility" on page A-11 for more information about CVU

Cloning Oracle Clusterware and Oracle Real Application Clusters nodes and instances

The preferred method to extend RAC databases is to use cloning as described in Oracle Universal Installer and OPatch User’s Guide. Cloning enables you to copy base images of the Oracle Clusterware and RAC software onto other nodes that have

Note: You can install the Oracle Clusterware high availability Application Programming Interface (API) from the Oracle Database 10g release 10.2 client installation media. You can also install a demo application that uses the API from the Oracle Database 10g release 10.2 companion installation media.
identical hardware and base software. You can also use cloning to create new clusters.

**See Also:** Chapter 10, "Adding and Deleting Nodes and Instances" for more information about adding or deleting nodes

- Oracle Load Balancing Advisory

Applications using a RAC database need to balance the workload across the cluster. To assist in the balancing of application workloads across designated resources, Oracle Database 10g release 2 provides the load balancing advisory. This advisory monitors the current workload activity across the cluster and for each instance where a service is active; it provides a percentage value to indicate how much of the total workload should be sent to this instance as well as a service quality flag. The feedback is provided as an entry in the automatic workload repository and a Fast Application Notification (FAN) event is published. To take advantage of the load balancing advisory, an application can use integrated clients, or clients that use the Runtime Connection Load Balancing feature, or by subscribing directly to the FAN event.

- Oracle RAC **Runtime Connection Load Balancing** using JDBC and Oracle Data Provider for .NET (ODP.NET)

Oracle supports Runtime Connection Load Balancing to balance work requests across all instances of a RAC database using service level information to select connections from a connection pool. The Oracle Database 10g client enables you to use Runtime Connection Load Balancing when using Java Database Connectivity (JDBC) or ODP.NET connection pools.

Runtime Connection Load Balancing balances work requests across instances based on a service’s real-time information. The connection cache manager uses RAC workload metrics and the load balancing policies to select the optimal instance to process a connection request. This results in efficient database resource usage with a balanced and dynamic distribution of the workload among RAC instances based on workload metrics and distribution policy.

---

**Note:** ODP.NET and OCI do not process UP events.

---

**See Also:** Chapter 6, "Introduction to Workload Management" for more information about services and the Oracle application development documentation set for more information about this feature

- Oracle Fast Connection Failover (FCF)

You can use FCF with JDBC, OCI, and ODP.NET to recover sessions when UP or DOWN events are published from clients. In the case of a DOWN event, Oracle cleans up any sessions in the connection pool that go to the instance that stops. For UP events, Oracle creates new connections to the recently-started instance. Clients can use any of the three connection protocols to accept event information that Runtime Connection Load Balancing publishes to re-create sessions and initiate failover. In addition, your chosen connection protocol, JDBC, OCI, or ODP.NET, reacts to throughput information that Runtime Connection Load Balancing publishes to chose the most appropriate connection.

- Transparent Data Encryption and Real Application Clusters
Transparent Data Encryption protects data that is stored in Oracle datafiles by preventing access to the data using means other than the normal database access mechanisms. This feature also provides secure storage and management of the encryption keys using a module that is external to the database. Thus, you can encrypt database column access and simultaneously more effectively manage encryption key access.

Using Transparent Data Encryption in a RAC environment requires that all of the database instances have access to the same encryption keys. For this release, the only key storage mechanism that is supported is the Oracle Wallet. All of the RAC nodes must be able to access the wallet either through a shared disk or by way of a local copy. All other Transparent Data Encryption administration and usage requirements are the same as those for single-instance Oracle database deployments.

See Also: "Data Security Considerations in Real Application Clusters" on page 14-2 and Oracle Database Advanced Security Administrator’s Guide for additional information about transparent data encryption.

RAC Configuration Assistant Enhancements

The Database Configuration Assistant (DBCA), the Database Upgrade Assistant (DBUA) have been enhanced for this release as follows:

- DBCA Enhancements for Standalone ASM Configuration
  
  When you create a RAC database that uses ASM, the DBCA creates the database in the same Oracle home that the ASM instance uses. If you create the database using a different home than the Oracle home that has ASM and if the ASM version is 10.2, then the DBCA automatically extends ASM from whichever Oracle home ASM is running in. However, if the ASM version is 10.1 and if ASM instances do not yet exist on all of the selected nodes, then the DBCA displays an error, prompting the user to either run the add node script or to upgrade ASM using the Database Upgrade Assistant (DBUA).

  See Also: "Automatic Storage Management in Real Application Clusters" on page 4-2 for more information about ASM in RAC

- DBCA Enhancements for Standalone ASM Disk Group Management
  
  You can use the DBCA to create and manage an ASM instance and its diskgroups independent of the database creation process. You can use the DBCA to create, add, and mount an ASM disk group on any node in the cluster database. If an ASM instance does not yet exist on a node, then the DBCA automatically extends ASM.

  See Also: Oracle Database Administrator’s Guide for more information about ASM

- Database Configuration Assistant, General Enhancements
  
  When you use DBCA to create a database in version 10.1, you configure the database control agent and OMS on all of the hosts. For version 10.2, DBCA by default configures an agent on all of the hosts where the RAC database has an instance, but only configures OMS on the node where you invoked DBCA. However, you can always run EMCA to change this configuration and set up
multiple database console OMS configurations on other hosts. If the central agent (also called the Grid Control agent) is configured on the cluster, you can optionally register the database to the central agent.

**See Also:** Your platform-specific Oracle Clusterware and RAC installation guide for more information about the DBCA

- **Database Upgrade Assistant Enhancements**

  You can also upgrade ASM independently or upgrade both the database and ASM at the same time. During an upgrade, the Oracle Database Upgrade Assistant (DBUA) automatically locates the Oracle Database 10g release 1 (10.1) Listener, and migrates it to Oracle Database 10g release 2. The Listener migration process stops the Listener in the existing Oracle home and restarts the Listener from the new Oracle home. During migration, client applications may not be able to connect to any databases that are registered to the Listener that is being migrated.

  If you are upgrading a database from Oracle Database 10g release 1.0 to Oracle Database 10g release 2.0, then the Database Control configuration in the 10gR1 ORACLE_HOME is upgraded to Oracle Database 10g release 2.0 as well. If you are upgrading a pre-Oracle Database 10g release 1.0 database to Oracle Database 10g release 2.0, then a new release 2.0 Database Control configuration will be performed. If the Listener was migrated as part of an upgrade operation, then the Enterprise Manager configurations will be updated with new information.

  **See Also:** *Oracle Database Upgrade Guide* for more information about database upgrades

- **ASM Storage Consolidation**

  One ASM instance on a node can support both single-instance Oracle database instances and RAC instances running on that node.

  **See Also:** "Automatic Storage Management in Real Application Clusters" on page 4-2 for more information about ASM in RAC

- **Dynamic RMAN Channel Allocation for RAC Environments**

  In previous releases, to use RMAN’s parallelism in RAC, you had to manually allocate an RMAN channel for each instance. You can now use the syntax `CONFIGURE DEVICE TYPE device PARALLELISM n` in RAC in the same way as in single-instance Oracle database environments. Dynamic channel allocation is only applicable where each node can access all of the datafiles, archived logs, and so on, in a RAC environment.

  **See Also:** "Configuring Channels for RMAN in Real Application Clusters" on page 7-2 for more information about dynamic RMAN channel allocation in RAC

- ** Archived Log Redo Thread Parameter Requirements**

  You must use all three archived log redo thread identifier parameters, %r or %R, %s or %S, and %t or %T, when identifying archived redo log threads. This enables Oracle to create unique names for archive logs across the incarnation.
Changing the Archiving Mode

You no longer need to modify the CLUSTER_DATABASE parameter setting to change the archiving mode in RAC. You can change archive log mode as long as the database is mounted in the local instance and not open in any instances.

See Also: "Changing the Archiving Mode in Real Application Clusters" on page 7-8 for more information about changing the archiving mode in RAC

Failover Improvements for Distributed Transaction Processing (DTP) in RAC

Oracle DTP transaction environments should now use services to simplify management in a RAC environment. This feature automates the implementation of workarounds for using distributed transactions in RAC. This feature leverages the Oracle services framework so that failure detection, failover, and fail back are transparent to DBAs.

In this release, DTP services automate the steps that are required to configure a RAC database to support distributed transactions in DTP environments. A DTP service will only be active on one instance in the cluster at a time. By creating multiple DTP services, with one or more DTP services enabled on each RAC instance, all tightly coupled branches of a global distributed transaction go to the same instance. In this way, you can leverage all of the instances of a RAC database to balance the distributed transaction load and thereby maximize application throughput.

For current and future client implementations, such as those for JDBC, you do not need the invocation to the SYS.DBMS_SYSTEM.DIST_TXN_SYNC procedure because the OPS_FAILOVER flag is deprecated. Instead, the server manages the synchronization of in-doubt transaction information across the RAC instances for transaction recovery.

See Also: "Services and Distributed Transaction Processing in RAC" on page 6-15 for information about how to configure DTP services to support distributed transactions

Multiple Oracle Clusterware Files

When you install the Oracle Clusterware, you can select the option of using multiple voting disks that reside on independent shared physical disks. This removes the requirement that the voting disk use redundant storage; now Oracle provides the redundancy and you do not need to use third party storage solutions to duplicate the voting disk. You can also select the option of mirroring your Oracle Cluster Registry (OCR). In addition, you can replace, repair, or remove an OCR if it fails, and you can perform these operation while the OCR is online. If you do not select the OCR mirroring option during the Oracle Clusterware installation, then you can mirror the OCR later.

See Also: Chapter 3, "Administering Oracle Clusterware Components" and your platform-specific RAC installation guide for more information about multiplexing the voting disk and the OCR
Fast-Start Failover and Data Guard Environments

Fast-start failover, which is provided with the Oracle Data Guard broker, enables failovers to occur automatically when a RAC primary database becomes unavailable. This occurs without DBA intervention and with no loss of data. When fast-start failover is enabled, the broker determines if a failover is necessary and automatically initiates the failover to a pre-specified target RAC standby database instance.

Fast-start failover will not occur in a RAC environment until all instances comprising a RAC primary database have failed. Moreover, after a failover completes, the broker can automatically reinstate the former primary database as a standby database in the new configuration.

See Also: Oracle Data Guard Broker for more information about Data Guard

Expanded Enterprise Manager Monitoring Features

You can use expanded Enterprise Manager monitoring features to:

- Monitor cluster interconnects to see all of the configured interfaces and interfaces that individual cluster database instances use. A single page provides information such as the transfer rates achieved on these interfaces, and whether the interface is private or public. You can see historical data for interconnects from drilldowns across the cluster or for a specific database.

- Improve scalability when monitoring clusters with a higher number of nodes. For example, the Performance page displays the maximum, minimum, and average loads across the cluster hosts, not just average load of each node. This enables you to quickly assess whether the load is evenly distributed. You only need to drill down to detailed information if a problem appears at the higher level Performance page.

- View backup reports for a group of databases.

- Global Cache Block Access Latency drilldowns. You can View the Global Cache Block Access Latency summary chart to see the end-to-end elapsed time or latency for a block request. You can drill down from there to the Cluster Cache Coherency page to see the cluster cache coherency metrics for the entire cluster database. This enables you to identify processing trends and optimize performance for your RAC environment.

Expanded Enterprise Manager Instance Addition

You can use Enterprise Manager to add instances to a RAC database.

Expanded Enterprise Manager Service Administration

You can use expanded Enterprise Manager service features for either Database Control or Grid Control to perform the following services tasks:

- Create
- Delete
- Update
- Test connections

Server Control Utility (SRVCTL) Enhancements

If you create additional Listeners with non-default names, in other words, with names other than the name listener_nodename where nodename is the name
of the node on which the Listener resides, then you must start and stop these Listeners with SRVCTL commands.

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

- `MAX_COMMIT_PROPAGATION_DELAY`—The `MAX_COMMIT_PROPAGATION_DELAY` parameter is deprecated. By default, commits on one instance are immediately visible on all of the other instances.

  **See Also:** *Oracle Database Reference* for more information about parameters and deprecated parameters

---

**Oracle Database 10g Release 1 (10.1) New Features for RAC Administration**

This section describes the Oracle Database 10g release 1 (10.1) features for RAC administration.

- High Availability, Workload Management, and Services

  Oracle Real Application Clusters introduces integrated clusterware known as Cluster Ready Services (CRS). You install CRS on all platforms on which you can run Oracle Real Application Clusters software. CRS manages cluster database functions including node membership, group services, global resource management, and high availability.

  **See Also:** *Oracle Real Application Clusters Quick Installation Guide for Oracle Database Standard Edition for Microsoft Windows* to install the Oracle Database 10g Standard Edition with RAC on Windows systems

In Oracle Real Application Clusters, you can use services to define application workloads by creating a service for each application, group of applications, or for major components within complex applications. You can then define where and when the service runs and thus use services to control your workload.

In both cluster and non-cluster environments, the **Automatic Workload Repository (AWR)** tracks performance metrics using services. You can also set thresholds on performance metrics to automatically generate alerts if these thresholds are exceeded.

  **See Also:** *Oracle Database PL/SQL Packages and Types Reference* for more information about the `DBMS_SERVICE` PL/SQL and `DBMS_MONITOR` packages and for more information about setting thresholds.

- Enhanced Cluster Management Implementation

  In earlier releases of the Oracle Database, cluster manager implementations on some platforms were referred to as "Cluster Manager". In Oracle Database 10g release (10.1), Cluster Ready Services (CRS) serves as the clusterware software, and Cluster Synchronization Services (CSS) is the cluster manager software for all platforms. The Oracle Cluster Synchronization Service Daemon (OCSSD) performs some of the clusterware functions on UNIX-based systems. On Windows-based systems, OracleCSService, OracleCRService, and OracleEVMService replace the Oracle Database OracleCMService9i.
Oracle Enterprise Manager, the Database Configuration Assistant (DBCA), and the Server Control (SRVCTL) Utility provide tools to administer clusters, RAC databases, and services.

- Enterprise Manager Enhancements for RAC
  This release includes the new Web-based Enterprise Manager Database Control with which you can manage a RAC database, and Enterprise Manager Grid Control for administering multiple RAC databases. Administration of RAC databases is greatly simplified because of more simplified drill-down tasks and because Enterprise Manager displays cluster-wide performance information. This is available for both single-instance Oracle and RAC databases.
  Enterprise Manager has several summary pages that show cluster database performance information at a glance; you no longer have to log in to each cluster database or display instance-specific pages to obtain a global view of cluster database performance.

- Expanded Enterprise Manager Service Administration
  You can use expanded Enterprise Manager service features for either Database Control or Grid Control to perform the following services tasks:
  - Edit
  - Enable
  - Disable
  - Start
  - Stop

- Enhancements for Flash Recovery Area and Automatic Disk-Based Backup and Recovery
  A flash recovery area is an Automatic Storage Management (ASM) disk group, a file system, or a directory that serves as a default storage area for recovery files. RAC supports the Automatic Disk-Based Backup and Recovery feature that simplifies managing disk space and backup and recovery files.

- Database Configuration Assistant (DBCA) Enhancements
  Use the DBCA to perform instance addition and deletion as well as database deletion.

- Database Upgrade Assistant (DBUA) Enhancements
  Use the DBUA to upgrade from an earlier RAC version to Oracle Database 10g with RAC. When you upgrade from a Primary/Secondary environment, the DBUA creates one service and assigns it to one instance as a preferred instance, and to the other instance as its available instance.

- Server Control (SRVCTL) Enhancements
  Enhancements to SRVCTL support the management of services and Automatic Storage Management (ASM) instances within RAC.

- Enhanced Recovery Parallelism on Multiple CPU Systems
  The default for instance, crash, and media recovery is to operate in parallel mode on multiple-CPU systems.

- Revised Error Messages for High Availability and Management Tools in Real Application Clusters
- The high availability error messages have been enhanced for this release.

- **Oracle Cluster Registry (OCR) Enhancements**

  The OCR contains configuration details for the cluster database and for high availability resources such as services, Virtual Interconnect Protocol (VIP) addresses, and so on.

- **GCS_SERVER_PROCESSES Parameter**

  There is a new, static parameter to specify the number of server processes for an instance’s Global Cache Service (GCS) for routing inter-instance traffic among RAC instances. The default number of GCS server processes is calculated based on system resources with a minimum of 2. You can set this parameter to different values on different instances.
Introduction to Oracle Clusterware and Oracle Real Application Clusters

This chapter introduces the Oracle Clusterware and Oracle Real Application Clusters (RAC) by describing these products as well as how to install, administer, and deploy them. This chapter describes the Oracle Clusterware and RAC architectures as well as the software and hardware components for both of these products. This chapter also briefly describes workload management, services, and high availability for both single-instance Oracle databases and RAC environments. This chapter includes the following topics:

- Oracle Clusterware and Oracle Real Application Clusters
- The Oracle Clusterware Architecture and Oracle Clusterware Processing
- The Real Application Clusters Architecture and Real Application Clusters Processing
- Oracle Clusterware Components and High Availability
- Workload Management with Real Application Clusters
- Introduction to Installing the Oracle Clusterware and Real Application Clusters
- Additional Considerations and Features for Real Application Clusters
- Managing Real Application Clusters Environments

Oracle Clusterware and Oracle Real Application Clusters

A cluster comprises multiple interconnected computers or servers that appear as if they are one server to end users and applications. Oracle Database 10g Real Application Clusters (RAC) enables the clustering of the Oracle Database. RAC uses the Oracle Clusterware for the infrastructure to bind multiple servers so that they operate as a single system.

Oracle Clusterware is a portable cluster management solution that is integrated with the Oracle database. The Oracle Clusterware is also a required component for using RAC. In addition, the Oracle Clusterware enables both single-instance Oracle databases and RAC databases to use the Oracle high availability infrastructure. The Oracle Clusterware enables you to create a clustered pool of storage to be used by any combination of single-instance and RAC databases.

Oracle Clusterware is the only clusterware that you need for most platforms on which RAC operates. You can also use clusterware from other vendors if the clusterware is certified for RAC.
Single-instance Oracle databases have a one-to-one relationship between the Oracle database and the instance. RAC environments, however, have a one-to-many relationship between the database and instances. In RAC environments, the cluster database instances access one database. The combined processing power of the multiple servers can provide greater throughput and scalability than is available from a single server. RAC is the Oracle Database option that provides a single system image for multiple servers to access one Oracle database. In RAC, each Oracle instance usually runs on a separate server.

RAC is a unique technology that provides high availability and scalability for all application types. The 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. 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 RAC databases can operate without code changes.

The Oracle Clusterware Architecture and Oracle Clusterware Processing

The Oracle Clusterware is software that when installed on servers running the same operating system, enables the servers to be bound together to operate as if they were one server. The Oracle Clusterware requires two clusterware components: a voting disk to record node membership information and the Oracle Cluster Registry (OCR) to record cluster configuration information. The voting disk and the OCR must reside on shared storage. The Oracle Clusterware requires that each node be connected to a private network by way of a private interconnect.

The private interconnect that the Oracle Clusterware requires is a separate network that you configure between the cluster nodes. This interconnect, which is required by RAC, can be the same network that the clusterware uses, but the interconnect should not be accessible by nodes that are not part of the cluster.

Oracle recommends that you use User Datagram Protocol (UDP) on a Gigabit Ethernet for your cluster interconnect. Oracle also recommends that you provide redundancy for the interconnect to prevent the interconnect from being a single point of failure. Crossover cables are not supported for use with the Oracle Clusterware or RAC databases.

The Oracle Clusterware manages node membership and prevents split brain syndrome in which two or more instances attempt to control the database. This can occur in cases where there is a break in communication between nodes through the interconnect.

The Oracle Clusterware architecture supports high availability by automatically restarting stopped components. In a RAC environment, all Oracle processes are under the control of the Oracle clusterware. The Oracle Clusterware also provides an application programming interface (API) that enables you to control other Oracle processes with the Oracle Clusterware.

Oracle Clusterware Software Component Processing Details

The Oracle Clusterware comprises several background processes that facilitate cluster operations. The Cluster Synchronization Service (CSS), Event Management (EVM), and Oracle Cluster components communicate with other cluster component layers in the other instances within the same cluster database environment. These components are also the main communication links between the Oracle Clusterware high
availability components and the Oracle Database. In addition, these components monitor and manage database operations.

**See Also:** Chapter 13, "Making Applications Highly Available Using the Oracle Clusterware" for more detailed information about the Oracle Clusterware API

The following list describes the functions of some of the major Oracle Clusterware components. This list includes these components which are processes on Unix and Linux operating systems or services on Windows.

---

**Note:** On Windows-based operating systems, many of the components are threads of the Oracle process instead of separate processes.

- **Cluster Synchronization Services (CSS)**—Manages the cluster configuration by controlling which nodes are members of the cluster and by notifying members when a node joins or leaves the cluster. If you are using third-party clusterware, then the css process interfaces with your clusterware to manage node membership information.

- **Cluster Ready Services (CRS)**—The primary program for managing high availability operations within a cluster. Anything that the crs process manages is known as a cluster resource which could be a database, an instance, a service, a Listener, a virtual IP (VIP) address, an application process, and so on. The crs process manages cluster resources based on the resource's configuration information that is stored in the OCR. This includes start, stop, monitor and failover operations. The crs process generates events when a resource status changes. When you have installed RAC, crs monitors the Oracle instance, Listener, and so on, and automatically restarts these components when a failure occurs. By default, the crs process makes five attempts to restart a resource and then does not make further restart attempts if the resource does not restart.

- **Event Management (EVM)**—A background process that publishes events that crs creates. The evm process scans a designated callout directory and invokes all of the scripts in that directory when an event occurs.

- **Oracle Notification Service (ONS)**—A publish and subscribe service for communicating Fast Application Notification (FAN) events.

- **RACG**—Extends clusterware to support Oracle-specific requirements and complex resources. Runs server callout scripts when FAN events occur.

- **Process Monitor Daemon (OPROCD)**—This process is locked in memory to monitor the cluster and provide I/O fencing. OPROCD performs its check, stops running, and if the wake up is beyond the expected time, then OPROCD resets the processor and reboots the node. An OPROCD failure results in the Oracle Clusterware restarting the node. OPROCD uses the hangcheck timer on Linux platforms.

In the following table, if a process has a (r) beside it, then the process runs as the root user. Otherwise the process runs as the oracle user.
The Oracle Clusterware Architecture and Oracle Clusterware Processing

When the Oracle Clusterware operates, several platform-specific processes or services will also be running on each node in the cluster to support the Oracle Clusterware. The Oracle Clusterware platform-specific UNIX-based processes and Windows-based services are described under the following headings:

- **Oracle Clusterware Processes on UNIX-Based Systems**
- **Oracle Clusterware Services on Windows-Based Systems**

### Oracle Clusterware Processes on UNIX-Based Systems

The Oracle Clusterware processes on UNIX-based systems are:

- **crsd**—Performs high availability recovery and management operations such as maintaining the OCR and managing application resources. This process runs as the root user, or by a user in the admin group on Mac OS X-based systems. This process restarts automatically upon failure.

- **evmd**—Event manager daemon. This process also starts the racgevt process to manage FAN server callouts.

- **ocssd**—Manages cluster node membership and runs as the oracle user; failure of this process results in cluster restart.

- **oprocd**—Process monitor for the cluster. Note that this process only appears on platforms that do not use vendor clusterware with the Oracle Clusterware.

### Oracle Clusterware Services on Windows-Based Systems

The Oracle Clusterware services on Windows-based systems are:

- **OracleCRService**—Performs high availability recovery and management operations such as maintaining the OCR and managing application resources. This process runs as the root user, or by a user in the admin group on Mac OS X-based systems. This process restarts automatically upon failure.

---

**Table 1–1  List of Processes and Windows Services associated with Oracle Clusterware**

<table>
<thead>
<tr>
<th>Oracle Clusterware Component</th>
<th>Linux/Unix Process</th>
<th>Windows Services</th>
<th>Windows Processes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process Monitor Daemon</td>
<td>oprocd (r)</td>
<td>OraFenceService</td>
<td></td>
</tr>
<tr>
<td>RACG</td>
<td>racgmain, racgimon</td>
<td>racgmain.exe</td>
<td>racgimon.exe</td>
</tr>
<tr>
<td>Oracle Notification Service</td>
<td>ons</td>
<td>ons.exe</td>
<td></td>
</tr>
<tr>
<td>Service (ONS)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Event Manager</td>
<td>evmd (r), evmlog</td>
<td>OracleEVMService</td>
<td>evmlogger.exe,</td>
</tr>
<tr>
<td></td>
<td>rb</td>
<td></td>
<td>evmd.exe</td>
</tr>
<tr>
<td>Cluster Ready</td>
<td>crsd.bin (r)</td>
<td>OracleCRService</td>
<td>crsd.exe</td>
</tr>
<tr>
<td>Cluster Synchronization</td>
<td>init.cssd (r), ocss</td>
<td>OracleCSService</td>
<td>ocssd.exe</td>
</tr>
<tr>
<td>Services</td>
<td>dd (r), ocssd.bin</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:** RAC on Linux platforms can have multiple threads that appear as separate processes with separate process identifiers.
The Real Application Clusters Architecture and Real Application Clusters Processing

A RAC database is a logically or physically shared everything database. All datafiles, control files, PFILEs, and redo log files in RAC environments must reside on cluster-aware shared disks so that all of the cluster database instances can access them. All of the instances must also share the same interconnect. In addition, RAC databases can share the same interconnect that the Oracle Clusterware uses.

Because a RAC database uses a shared everything architecture, RAC requires cluster-aware storage for all database files. It is your choice as to how to configure your disk, but you must use a supported cluster-aware storage solution. Oracle Database 10g provides Automatic Storage Management (ASM), which is the recommended solution to manage your disk. However you may also use a cluster-aware volume manager or a cluster file system (not required). In RAC, the Oracle Database software manages disk access and the Oracle software is certified for use on a variety of storage architectures. A RAC database can have up to 100 instances. Depending on your platform, you can use the following file storage options for RAC:

- ASM, which Oracle recommends
- Oracle Cluster File System (OCFS), which is available for Linux and Windows platforms, or a third-party cluster file system that is certified for RAC
- A network file system
- Raw devices

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

- At least one additional thread of redo for each instance
- An instance-specific undo tablespace

All nodes in a 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 enables the load balancing of application connections across all of the instances in a RAC database.

Users can access a RAC database using a client-server configuration or through one or more middle tiers, with or without connection pooling. Users can be DBAs, 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. RAC database instances can be accessed through a database’s defined, default IP address and through VIP addresses.

**Note:** Do not to use the interconnect or the private network for user communication because Cache Fusion uses the private interconnect for inter-instance communications.

In addition to the node’s host name and IP address, you must also assign a virtual host name and an IP address to each node. The virtual host name or VIP should be used to connect to the database instance. For example, you might enter the virtual host name CRM in the address list of the tnsnames.ora file.

A virtual IP address is an alternate public address that client connections use instead of the standard public IP address. To configure VIP addresses, you need to reserve a spare IP address for each node that uses the same subnet as the public network.

If a node fails, then the node’s VIP fails over to another node on which the VIP cannot accept connections. Clients that attempt to connect to the VIP receive a rapid connection refused error instead of waiting for TCP connect timeout messages. You configure VIP addresses in the address list for your database connection definition to enable connectivity. The following section describes the RAC software components in more detail.

### The Real Application Clusters Software Components

RAC databases have two or more database instances that each contain memory structures and background processes. A RAC database has the same processes and memory structures as a single-instance Oracle database as well as additional process and memory structures that are specific to RAC. Any one instance’s database view is nearly identical to any other instance’s view within the same 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, 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 RAC are greater than the SGA requirements for single-instance Oracle databases due to Cache Fusion.

To ensure that each RAC database instance obtains the block that it needs to satisfy a query or transaction, 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 System Global Area for a 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. RAC uses the private interconnect for inter-instance communication and block transfers. The Global Enqueue Service
Monitor and the Instance Enqueue Process manages access to Cache Fusion resources as well as enqueue recovery processing.

These RAC-specific processes and the GRD collaborate to enable Cache Fusion. The RAC-specific processes and their identifiers are as follows:

- **LMS**—Global Cache Service Process
- **LMD**—Global Enqueue Service Daemon
- **LMON**—Global Enqueue Service Monitor
- **LCK0**—Instance Enqueue Process

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.

---

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

---

**Oracle Clusterware Components and High Availability**

When you combine the Oracle Clusterware and RAC, you can achieve excellent scalability and high availability. The Oracle Clusterware achieves this using the components that this section describes under the following topics:

- **The Oracle Clusterware Voting Disk and Oracle Cluster Registry**
- **The Oracle Clusterware High Availability Framework and the Application Programming Interface**
- **The Oracle Clusterware Software Components**

**The Oracle Clusterware Voting Disk and Oracle Cluster Registry**

The Oracle Clusterware requires the following two critical files:

- **Voting Disk**—Manages cluster membership by way of a health check and arbitrates cluster ownership among the instances in case of network failures. RAC uses the voting disk to determine which instances are members of a cluster. The voting disk must reside on shared disk. For high availability, Oracle recommends that you have multiple voting disks. The Oracle Clusterware enables multiple voting disks but you must have an odd number of voting disks, such as three, five, and so on. If you define a single voting disk, then you should use external mirroring to provide redundancy.

- **Oracle Cluster Registry (OCR)**—Maintains cluster configuration information as well as configuration information about any cluster database within the cluster. The OCR also manages information about processes that the Oracle Clusterware controls. The OCR stores configuration information in a series of key-value pairs within a directory tree structure. The OCR must reside on shared disk that is accessible by all of the nodes in your cluster. The Oracle Clusterware can multiplex the OCR and Oracle recommends that you use this feature to ensure cluster high availability. You can replace a failed OCR online, and you can update the OCR through supported APIs such as Enterprise Manager, the Server Control Utility (SRVCTL), or the Database Configuration Assistant (DBCA).
The Oracle Clusterware High Availability Framework and the Application Programming Interface

Oracle Clusterware provides a high availability application programming interface (API) that you can use to enable the Oracle Clusterware to manage applications or processes that run a cluster. This enables you to provide high availability for all of your applications. The Oracle Clusterware with ASM enables you to create a consolidated pool of storage to support both the single-instance Oracle databases and the RAC databases that are running on your cluster.

To maintain high availability, the Oracle Clusterware components can respond to status changes to restart applications and processes according to defined high availability rules. In addition, you can use the Oracle Clusterware high availability framework by registering your applications with the Oracle Clusterware and configuring the clusterware to start, stop, or relocate your application processes. That is, you can make custom applications highly available by using the Oracle Clusterware to create profiles that monitor, relocate, and restart your applications. The Oracle Clusterware responds to FAN events that are created by a RAC database. Oracle broadcasts FAN events when cluster servers may become unreachable and network interfaces are slow or non-functional.

See Also: Chapter 13, "Making Applications Highly Available Using the Oracle Clusterware" for more detailed information about the Oracle Clusterware API

Workload Management with Real Application Clusters

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 10g with RAC includes many features that can enhance workload management such as connection load balancing, fast connection failover (FCF), the load balancing advisory, and runtime connection load balancing. Workload management provides the greatest benefits to RAC environments. You can, however, take advantage of workload management by using Oracle services in single-instance Oracle Databases, especially those that use Data Guard or Streams. Workload management comprises the following components:

- **High Availability Framework**—The RAC high availability framework enables the Oracle Database to maintain components in a running state at all times. Oracle high availability implies that the Oracle Clusterware monitors and restarts critical components if they stop, unless you override the restart processing. The Oracle Clusterware and RAC also provide alerts to clients when configurations change. This enables 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 processes in an Oracle database instance to failing over the processing of an entire instance to other available instances.

---

**Note:** Both the voting disks and the OCRs must reside on either cluster file system files or on shared raw devices that you configure before you install the Oracle Clusterware and RAC.
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 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 10g introduces a powerful automatic workload management facility, called services, to enable the enterprise grid vision. Services are entities that you can define in 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 assigns to process workloads and to monitor workload resources. Applications that you assign to services transparently acquire the defined 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 the Scheduler, to map services to specific job classes.

In RAC databases, the service performance rules that you configure control the amount of work that Oracle 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.

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 6, "Introduction to Workload Management" for more information about workload management and services

Introduction to Installing the Oracle Clusterware and Real Application Clusters

This section introduces the storage options for RAC and the installation processes for both the Oracle Clusterware and RAC under the following topics:

- Oracle Clusterware Installation Process Description
- Real Application Clusters Installation and Database Creation Process Description

Oracle Clusterware Installation Process Description

The Oracle Clusterware is distributed on the Oracle Database 10g installation media. The Oracle Universal Installer (OUI) installs the Oracle Clusterware into a directory structure, which can be referred to as CRS_HOME, that is separate from other Oracle software running on the machine. Because the Oracle Clusterware works closely with the operating system, system administrator access is required for some of the
installation tasks. In addition, some of the Oracle Clusterware processes must run as the system administrator, which is generally the root user on Unix and Linux systems and the System Administrator user on Windows systems.

Before you install the Oracle Clusterware, Oracle recommends that you run the Cluster Verification Utility (CVU) to ensure that your environment meets the Oracle Clusterware installation requirements. The OUI also automatically runs CVU at the end of the clusterware installation to verify various clusterware components. The CVU simplifies the installation, configuration, and overall management of the Oracle Clusterware installation process by identifying problems in cluster environments.

During the Oracle Clusterware installation, you must identify three IP addresses for each node that is going to be part of your installation. One IP address is for the private interconnect and the other is for the public interconnect. The third IP address is the virtual IP address that clients will use to connect to each instance.

The Oracle Clusterware installation process creates the voting disk and OCR on cluster-aware storage. If you select the option for normal redundant copies during the installation process, then the Oracle Clusterware automatically maintains redundant copies of these files to prevent the files from becoming single points of failure. The normal redundancy feature also eliminates the need for third party storage redundancy solutions. When you use normal redundancy, the Oracle Clusterware automatically maintains two copies of the Oracle Cluster Registry (OCR) file and three copies of the Voting Disk file.

---

**Note:** If you choose external redundancy for the OCR and voting disk, then to enable redundancy, your disk subsystem must be configurable for RAID mirroring. Otherwise, your system may be vulnerable because the OCR and voting disk are single points of failure.

---

**Real Application Clusters Installation and Database Creation Process Description**

The RAC software is distributed as part of the Oracle Database 10g installation media. By default, the standard Oracle Database 10g software installation process installs the RAC option when it recognizes that you are performing the installation on a cluster. The OUI installs RAC into a directory structure, which can be referred to as ORACLE_HOME, that is separate from other Oracle software running on the machine. Because the OUI is cluster-aware, it installs the RAC software on all of the nodes that you defined to be part of the cluster. If you are using a certified cluster file system for the Oracle home, then only select the node that you are connected to for the installation.

You must first install the Oracle Clusterware before installing RAC. After the Oracle Clusterware is operational, you can use the OUI to install the Oracle database software with the RAC components. During the installation, the OUI runs the DBCA to create your RAC database according to the options that you select. The DBCA also runs the Net Configuration Assistant (NetCA) to configure the network for your RAC environment.

Oracle recommends that you select ASM during the installation to simplify storage management; ASM automatically manages the storage of all database files within disk groups. You can also configure services during installation, depending on your processing requirements. If you are using the Oracle Database 10g Standard Edition, then you must use ASM for storing all of the database files.
By default, the DBCA creates one service for your environment and this service is for the database. The default service is available on all instances in a RAC environment, unless the database is in restricted mode.

**Additional Considerations and Features for Real Application Clusters**

In addition to configuring services to manage your workloads, also consider using the following features when you deploy RAC:

- **Scaling Your RAC Database**—You can add nodes and instances to your RAC environment using Oracle cloning described in Chapter 10. In addition, you can extend your database by using the manual procedures that are also described in Chapter 10, "Adding and Deleting Nodes and Instances".

- **Enterprise Manager**—Use Enterprise Manager to administer your entire RAC environment, not just the RAC database. Use Enterprise Manager to create and modify services, and to start and stop the cluster database instances and the cluster database. Enterprise Manager has additional features as detailed in the section "Overview of Using Enterprise Manager with Real Application Clusters" on page 2-4.

- **Recovery Manager (RMAN)**—RMAN backs up, restores, and recovers datafiles, control files, server parameter files (SPFILEs) and archived redo logs. 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 RAC databases. In RAC, RMAN channels can be dynamically allocated across all of the RAC instances. Channel failover enables failed operations on one node to continue on another node. You can use RMAN in RAC from the Oracle Enterprise Manager Backup Manager or from a command line.

- **Automatic undo management**—Automatically manages undo processing.

- **Automatic segment space management (ASSM)**—Automatically manages segment freelists and freelist groups.

- **Locally managed tablespaces**—Enhances space management performance.

- **Cluster Verification Utility (CVU)**—Use CVU to verify the status of your clusterware if you experience problems or use it whenever you reconfigure your cluster.

- **Sequences**—If you use Oracle sequences, then use the `CACHE` option for optimal performance.

- **Indexes**—If you use indexes, 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.

  **See Also:** Chapter 7, "Configuring Recovery Manager and Archiving" for more information about RMAN

**Managing Real Application Clusters Environments**

This section describes the following RAC environment management topics:

- **Designing Real Application Clusters Environments**

- **Administrative Tools for Real Application Clusters Environments**

- **Monitoring Real Application Clusters Environments**
Evaluating Performance in Real Application Clusters Environments

Designing Real Application Clusters Environments

Consider performing the following steps during the design and development of applications that you are deploying on a RAC database. Consider tuning:

1. The design and the application
2. The memory and I/O
3. Contention
4. The operating system

Note: If an application does not scale on an SMP machine, then moving the application to a 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 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.

Administrative Tools for Real Application Clusters Environments

Oracle enables you to administer a cluster database as a single system image through Enterprise Manager, SQL*Plus, or through RAC command-line interfaces such as Server Control (SRVCTL). You can also use several tools and utilities to manage your RAC environment and its components as follows:

- Enterprise Manager—Enterprise Manager has both the Database Control and Grid Control GUI interfaces for managing both single instance and RAC environments.

See Also: Chapter 9 and Chapter 12 for more information about Enterprise Manager

- Cluster Verification Utility (CVU)—CVU is a command line tool that you can use to verify a range of cluster and RAC-specific components such as shared storage
devices, networking configurations, system requirements, and the Oracle Clusterware, as well as operating system groups and users. You can use CVU for pre-installation checks as well as for post-installation checks of your cluster environment. CVU is especially useful during pre-installation and during installation of the Oracle Clusterware and RAC components. The OUI runs CVU after the Oracle Clusterware and the Oracle installation to verify your environment.

See Also: "Using the Cluster Verification Utility" on page A-11 for more information about CVU

- Server Control (SRVCTL)—SRVCTL is a command line interface that you can use to manage a 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.

See Also: Appendix D for more information about SRVCTL

- Cluster Ready Services Control (CRSCTL)—CRSCTL is a command line tool that you can use to manage the Oracle Clusterware. You can use CRSCTL to start and stop the Oracle Clusterware. CRSCTL has many options such as enabling online debugging,

See Also: "Diagnosing the Oracle Clusterware High Availability Components" on page A-1 for more information about CRSCTL

- Oracle Interface Configuration Tool (OIFCFG)—OIFCFG is a command-line tool for both single-instance Oracle databases and RAC environments that you can use to allocate and de-allocate network interfaces to components. You can also use OIFCFG to direct components to use specific network interfaces and to retrieve component configuration information.

See Also: "Administering System and Network Interfaces with OIFCFG" on page 9-7 for more information about OIFCFG

- OCR Configuration Tool (OCRCONFIG)—OCRCONFIG is a command line tool for OCR administration. You can also use the OCRCHECK and OCRDUMP utilities to troubleshoot configuration problems that affect the OCR.

See Also: Chapter 3 and Appendix C for more information about managing the OCR

Monitoring Real Application Clusters Environments

Web-based Enterprise Manager Database Control and Grid Control enable you to monitor a RAC database. The Enterprise Manager Console is a central point of control for the Oracle environment that you access by way of a graphical user interface (GUI). Use the Enterprise Manager Console to initiate cluster database management tasks. Use Enterprise Manager Grid Control to administer multiple RAC databases. Also note the following points about monitoring RAC environments:

- The global views, or GV$ views, are based on V$ views. The script $ORACLE_HOME/rdbms/admin/catclustdb.sql creates the GV$ views. Run this script if you do not create your database with the DBCA. Otherwise, the DBCA runs this script for you.

- Statspack is RAC-aware.
Evaluating Performance in Real Application Clusters Environments

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

Some of the RAC-specific performance features are:

- **Dynamic Resource Allocation**
  - Oracle 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 RAC performance monitoring
  - Enterprise Manager Database Control and Grid Control are Integrated with RAC
Introduction to Oracle Clusterware and RAC Administration and Deployment

This chapter provides an overview of administering the Oracle Clusterware and Real Application Clusters (RAC) environments. This chapter includes the following topics:

- Real Application Clusters Documentation Overview
- Introduction to Administering Real Application Clusters
- Administering Real Application Clusters
- Database Instance Management and Database Administration in RAC

Real Application Clusters Documentation Overview

This section describes the RAC documentation set. This book, the Oracle Database Oracle Clusterware and Oracle Real Application Clusters Administration and Deployment Guide, provides RAC-specific administration and application deployment information.

This book describes how to administer the Oracle Clusterware components such as the voting disks and the Oracle Cluster Registry (OCR). This book also explains how to administer storage and how to use RAC scalability features to add and delete instances and nodes. This book also discusses how to use Recovery Manager (RMAN), and how to perform backup and recovery in RAC.

The Oracle Database Oracle Clusterware and Oracle Real Application Clusters Administration and Deployment Guide describes RAC deployment topics by explaining how to deploy automatic workload management and take advantage of high availability by using services. This book describes how the Automatic Workload Repository (AWR) tracks and reports service levels and how you can use service level thresholds and alerts to improve high availability in your RAC environment. This book also explains how to make your applications highly available by using the Oracle Clusterware.

The Oracle Database Oracle Clusterware and Oracle Real Application Clusters Administration and Deployment Guide explains how to monitor and tune performance in RAC environments by using Oracle Enterprise Manager and by using information in AWR and Oracle performance views. This book also highlights application-specific deployment techniques for online transaction processing and data warehousing environments.

The appendices in this book describe:

- Troubleshooting tips such as debugging and log file use and general information about installing and using the Cluster Verification Utility (CVU)
The Oracle Clusterware command-line reference
- The Oracle Cluster Registry command syntax
- How to use the Server Control (SRVCTL) utility to start and stop the database and instances, to manage configuration information, and to delete or move instances and services
- RAC tools error messages

In addition to this book, refer to the platform-specific Oracle Real Application Clusters installation and configuration guide on your product installation media as described under the following heading.

**Platform-Specific Oracle Real Application Clusters Installation and Configuration Guides**

Each platform-specific Oracle Database 10g installation media contains a copy of an Oracle Real Application Clusters platform-specific installation and configuration guide in HTML and PDF formats. These RAC installation books contain the pre-installation, installation, and post-installation information for the various UNIX- and Windows-based platforms on which RAC operates.

---

**Note:** Additional information for this release may be available in the Oracle Database 10g README or Release Notes. If these documents are available for this release, then they are on your Oracle product installation media.

---

**Introduction to Administering Real Application Clusters**

Install the Oracle Clusterware and your Oracle Database 10g software with the Oracle Universal Installer (OUI) and create your database with the Database Configuration Assistant (DBCA). This ensures that your RAC environment has the optimal network configuration, database structure, and parameter settings for the environment that you selected. As a DBA, after installation your tasks are to administer your RAC environment at three levels:

- Instance Administration
- Database Administration
- Cluster Administration

**Administering Real Application Clusters**

Use the following tools to perform administrative tasks in RAC:

- Cluster Verification Utility (CVU)—Install and use CVU before you install RAC to ensure that your configuration meets the minimum RAC installation requirements. Also use the CVU for on-going administrative tasks, such as node addition and node deletion.

  **See Also:** Your platform-specific Oracle Clusterware and RAC installation guide for information about how to manually install CVU

- Enterprise Manager—Oracle recommends that you use Enterprise Manager to perform administrative tasks whenever feasible
- Task-specific GUIs such as the Database Configuration Assistant (DBCA) and the Virtual Internet Protocol Configuration Assistant (VIPCA)
- Command-line tools such as SQL*Plus, Server Control (SRVCTL), the Oracle Clusterware command-line interface, and the Oracle Interface Configuration tool (OIFCFG)

**Voting Disk and Oracle Cluster Registry Device Administration**

You can use multiple disks for both the voting disk and the Oracle Cluster Registry (OCR) disk to prevent them from becoming single points of failure. Administer these components as described in Chapter 3, "Administering Oracle Clusterware Components".

**Database Instance Management and Database Administration in RAC**

Use Enterprise Manager, SQL*Plus, or SRVCTL to administer database instances and RAC databases as described in Chapter 5, "Administering Database Instances and Cluster Databases".

**Storage Management in Real Application Clusters**

When you create your database, you can create Automatic Storage Management (ASM) disk groups and configure mirroring for ASM disk groups using the DBCA. After your RAC database is operational, you can administer ASM disk groups with Enterprise Manager or the SRVCTL utility as described in Chapter 4, "Administering Storage".

You can configure ASM in a separate standalone ASM-only home. This enables instances for non-RAC and RAC databases to share a single ASM instance on a node. You also have the option to upgrade ASM independently of your database upgrades.

The Oracle tools that you use to manage ASM, including DBCA, Database Upgrade Assistant (DBUA), Enterprise Manager, and the silent mode install and upgrade commands, include options to manage ASM instances and disk groups. For example, you can run the DBCA to create a new ASM instance or ASM diskgroup independently of creating a database.

When you perform ASM-related options during installs, upgrades, or other operations, the tool you are using may automatically extend ASM to other nodes in your cluster. This can include installing ASM software into the same home as on the current node and starting the ASM instance. For example, if you use DBCA to create a database using a new Oracle home, then DBCA will attempt to extend ASM to the new Oracle home on all of the nodes you select.

In some cases, your current configuration may not be compatible with an ASM activity that you are trying to perform, either explicitly or with an automatic ASM extension to other nodes. Should you be using DBCA, as described earlier, to build a database using a new Oracle home, and if the ASM version is from an earlier release of the Oracle software but does not exist on all of the nodes you selected for the database, then ASM cannot be extended. Instead, the DBCA session displays an error, prompting you either to run the add node script or to upgrade ASM using the DBUA.
Oracle Clusterware for Real Application Clusters

When you create a RAC database, you can also create services and assign them to instances using the DBCA. After your RAC database is operational, you can use Enterprise Manager, the DBCA, or SRVCTL to create and administer services and high availability components as described in Chapter 6, "Introduction to Workload Management".

Other high availability components include node resources such as the Virtual Internet Protocol (VIP) address, the Global Services Daemon, the Oracle Notification Service, and the Oracle Net Listeners. These resources are automatically started when the Oracle Clusterware starts the node and then automatically restarts them if they stop. The application level resources are the instances and the Oracle Clusterware background processes that run on each instance.

You can use the VIPCA to administer VIP addresses and you can use SRVCTL to administer other node resources. The information that describes the configuration of these components is stored in the Oracle Cluster Registry (OCR) that you can administer as described in Chapter 3, "Administering Oracle Clusterware Components".

Additional Real Application Clusters Administrative Topics

This book contains the following additional RAC administrative topics:

- Scalability—Adding instances and nodes to a RAC database as described in Chapter 10, "Adding and Deleting Nodes and Instances"
- Backup and Recovery—Configuring Recovery Manager (RMAN) and performing backup and recovery processing as described in Chapter 7, "Configuring Recovery Manager and Archiving" and Chapter 8, "Managing Backup and Recovery"
- Making Applications Highly Available—Wrapping your custom applications with the Oracle Clusterware commands to use the Oracle Clusterware infrastructure to keep your applications highly available as described in Chapter 13, "Making Applications Highly Available Using the Oracle Clusterware" and Appendix B, "High Availability Oracle Clusterware Command-Line Reference and C API"
- Log Files—Administering information that Oracle records in log files as described in Appendix A, "Troubleshooting"
- Using SRVCTL—Using SRVCTL to administer RAC instances, databases, services, and so on, as described in Appendix D, "Server Control Utility Reference"
- Error Messages—Interpreting error messages for RAC high availability and management tools as described in Appendix E, "Oracle Real Application Clusters Tools Messages"

Overview of Using Enterprise Manager with Real Application Clusters

Enterprise Manager is a Web-based tool with RAC-specific administration and performance-related features. If you create your RAC database with the Database Configuration Assistant (DBCA), then the Enterprise Manager Database Control tool is automatically configured for your RAC environment. This means that all instances that were part of your installation have an Enterprise Manager Agent running on them. Enterprise Manager Database Control enables you to manage a single RAC database with its instance targets, Oracle Net Services Listener targets, host targets, and a cluster target.
You can configure Enterprise Manager Grid Control on additional hosts, which were not part of your initial RAC installation, either inside or outside your cluster environment. Enterprise Manager Grid Control enables you to manage multiple cluster databases, cluster database instances, and the hosts on which cluster database components operate.

Grid Control enables you to monitor and administer your entire computing environment from one network location. Use Grid Control to manage all of your enterprise services, including hosts, databases, Listeners, application servers, HTTP Servers, and Web applications, as one cohesive unit. Enterprise Manager Grid Control only requires one Agent on one host in your cluster environment to perform cluster database and instance discovery. Install Enterprise Manager Grid Control from a your Oracle Database 10g installation media.

See Also: Oracle Enterprise Manager Concepts for more information about using Enterprise Manager

You can also use both Enterprise Manager Database Control and the Enterprise Manager Grid Control to:

- Administer database services—Create, configure, start, stop, relocate, obtain status, and so on
- Create and assign resource plans—Assign resource plans to cluster database instances
- Administer storage—Assign undo tablespaces and re-assign them from one instance to another, administer redo log assignments among cluster database instances, and switch archive log modes
- Administer Automatic Storage Management—Administer ASM instances and ASM disk groups if the database uses ASM
- Perform general database activities—Start up and shut down RAC databases and instances, perform backup and recovery operations, edit server parameter file (spfile) settings for instances or for entire cluster databases, and so on
- Display host configurations—Memory, CPU, device I/O, network interfaces, the operating system and installed patches

The following sections introduce Oracle Real Application Clusters (RAC) application deployment and performance by explaining the main points that you need to remember when you deploy applications on RAC:

- Overview of Deploying Applications on Real Application Clusters
- Implementing Oracle Features with Real Application Clusters

Overview of Deploying Applications on Real Application Clusters

To optimally deploy applications on RAC, remember the following points:

- Storage for RAC datafiles must be shared storage—When you install RAC, use Automatic Storage Management (ASM) or a Cluster File System for datafile storage when available.
- Create your database with the Database Configuration Assistant (DBCA).
- Define services for your environment with the DBCA or Enterprise Manager and administer them with Oracle Enterprise Manager or the Server Control (SRVCTL) Utility.
- Use the Server Parameter File (SPFILE)—The SPFILE should be located on either a cluster file system file or on a shared raw device.
- Use Automatic Undo Management.
- Use Automatic Segment-Space Management.
- Use the Automatic Database Diagnostic Monitor (ADDM) to reduce the effort required to tune Oracle systems.

See Also: Your platform-specific Real Application Clusters installation guide for more information about configuring these features for Oracle Database 10g Real Application Clusters

Code Changes are Not Required for Applications

Applications that perform well on single-instance Oracle databases do not require code changes to perform well on RAC databases. The same compatibility guidelines that apply to Oracle single-instance databases also apply to RAC. For example, if an application scales well in an SMP environment, then the application will scale well with RAC.

The primary feature of RAC that enables applications to scale across multiple instances is Cache Fusion. Cache Fusion enables each instance to locate the most useful version of a block of data for its needs, whether that block resides on disk or in the memory of another instance. The cluster interconnect provides a fast pathway for Cache Fusion to transfer required blocks of data between instances when necessary.

Implementing Oracle Features with Real Application Clusters

The Oracle features described in this section enhance the performance and simplify the administration of your RAC environment. The features discussed in this section are:
- Automatic Storage Management
- Cluster File Systems in Real Application Clusters
- Storage Management Features and Real Application Clusters
- Services in Oracle Database 10g
- The Oracle Clusterware and High Availability in Real Application Clusters
- Additional Oracle High Availability Features and Solutions

Automatic Storage Management

Automatic Storage Management (ASM) simplifies database administration by eliminating the need for you to manage Oracle database files. Instead, ASM enables you to create disk groups that comprise disks and the files that reside on them.

Cluster File Systems in Real Application Clusters

Depending on your hardware platform, you can store Oracle homes and Oracle datafiles on a cluster file system. Cluster file systems are simpler to configure and manage than raw device storage. Cluster file systems also offer scalable, low latency, highly resilient storage that significantly reduces costs.
Storage Management Features and Real Application Clusters

The advanced storage features of Oracle Automatic Storage Management greatly enhance manageability for RAC just as with single instance Oracle. Other storage features include Oracle-managed files, automatic segment-space management, and automatic undo management. Refer to the Oracle database documentation for more information about using storage management features.

See Also:
- Oracle Database 2 Day DBA
- Oracle Database Administrator’s Guide
- Oracle Database Application Developer’s Guide - Object-Relational Features

Services in Oracle Database 10g

With Oracle Database 10g, you can define application workloads as services so that you can individually manage and control them. You can create a service for each application or for major components within a complex application. Once created, you can define where and when a service runs. Your entire database workload can be separated into a few services, each of which can be managed independently, reducing the need to manage individual users or sessions for many tasks.

In a RAC database, you can use services to maximize the use of your cluster’s processing resources. You can assign each service to one or more instances for normal startup (preferred), depending on its processing requirements. Additionally, you can define one or more alternate (available) instances that a service can use if one of the service’s assigned (preferred) instances becomes unavailable.

In both cluster and non-cluster environments, you can track performance metrics by service using the Automatic Workload Repository (AWR). You can set thresholds on performance metrics so that your system automatically generates alerts if the thresholds are exceeded. You can also map services to Resource Manager consumer groups to provide more fine-grained resource allocation controls such as placing limits on CPU consumption. Other Oracle tools and facilities such as Job Scheduler, Parallel Query, and Oracle Streams Advanced Queuing can also use services to manage their workloads.

The settings for Resource Manager are instance specific, that is, you can have different Resource Manager settings on different instances. Therefore, ensure that your Resource Manager settings are appropriate for the instances on which you are using it. The Automatic Workload Management features of Oracle Database 10g are integrated with the resource manager. The only difference in the way you use Resource Manager with RAC is that Resource Manager has instance specific settings.

Note: An example that shows how to manage a complex environment with database services is available on the Oracle Technology Network at the URL http://www.oracle.com/index.html. The example includes entries from the related files, such as listener, database, and Net Services parameter files, as well as the commands to create and use the services for application management.

Runtime Connection Load Balancing is used when selecting connections from a connection pool. For connection pools that support services at one instance only, the first available connection in the pool is used. When connection pools support services
that span multiple instances using a policy such as service metrics, Runtime Connection Load Balancing distributes work requests across instances that are adequately serving a service. This avoids sending work to slow, hung, failed or restricted instances.

For connection load balancing, the method for balancing connections across a service uses four metrics: session count by instance, run queue length of the node, goodness by service, and weighted session count by service. The metrics used depend on what goals have been defined for the service and how Oracle Net Services has been configured.

Fast Connection Failover is used to prevent connections being directed to failed nodes or instances. Integrated connection pools will clean up connections when a failure occurs or add additional connections when new instances become available. This allows the application to be immediately aware of cluster configuration changes and react to them without any programming or configuration changes at the application tier.

See Also: Chapter 6, "Introduction to Workload Management" for more detailed information about these features

The Oracle Clusterware and High Availability in Real Application Clusters

This section introduces the following high availability features:

■ High Availability with the Oracle Clusterware
■ The Oracle Clusterware and High Availability

High Availability with the Oracle Clusterware

Oracle Real Application Clusters 10g introduces a complete, integrated clusterware management solution on all Oracle Database 10g 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 install the Oracle Clusterware as a separate install that you can complete independently or as a pre-requisite to the RAC installation process. Oracle database features such as Oracle 10g services use the underlying Oracle Clusterware mechanisms to provide their capabilities. Oracle also continues to support select third-party clusterware products on specified platforms.

See Also: Chapter 13, "Making Applications Highly Available Using the Oracle Clusterware" for more information about the Oracle Clusterware

The Oracle Clusterware and High Availability

High availability configurations have redundant hardware and software that maintain operations by avoiding single points-of-failure. During outages, the Oracle Clusterware relocates the processing performed by the inoperative component to a backup component. The Oracle recovery processes quickly re-master resources, recover partial or failed transactions, and rapidly restore the system.

You can combine many Oracle products and features to create highly reliable computing environments. Doing this requires capacity and redundancy planning. In addition, consider your overall system costs and your return on investment. There are also other practical considerations such as selecting the appropriate hardware and
deciding whether to use idle machines that are part of your high availability configuration.

**Additional Oracle High Availability Features and Solutions**

This section describes the following additional high availability solutions:

- Connection Load Balancing with Real Application Clusters
- Recovery Manager (RMAN) in Real Application Clusters
- Data Guard
- Primary/Secondary Instance Configurations in Earlier Releases

**Connection Load Balancing with Real Application Clusters**

The connection load balancing feature of Oracle Net Services automatically distributes connections among active instances. Connection load balancing does this based on the workload of each node and instance in a cluster.

**Recovery Manager (RMAN) in Real Application Clusters**

Recovery Manager (RMAN) is an Oracle tool that you can use to backup, copy, restore, and recover datafiles, control files, SPFILEs, and archived redo logs. You can invoke RMAN as a command-line utility or use Oracle Enterprise Manager.

A best practice is to configure RMAN so that all instances can access all of the archive log threads throughout your cluster database. In the event of media recovery, the recovering instance requires access to all of the archived redo log threads. You can simplify media recovery administration by ensuring that a recovering instance can access a local copy of the archive log threads from all of the instances in your cluster database.

*See Also:* Chapter 7, “Configuring Recovery Manager and Archiving” for details about configuring RMAN for use with RAC and *Oracle Database Backup and Recovery Advanced User’s Guide* for detailed information about RMAN

**Data Guard**

Oracle Data Guard works with standby databases to protect your data against errors, failures, and corruptions that might otherwise destroy your database. Data Guard protects critical data by automating the creation, management, and monitoring aspects of standby database environments. Oracle Data Guard automates the otherwise manual process of maintaining a transactional consistent copy of an Oracle database to recover from the loss of or damage to the production database.

**Primary/Secondary Instance Configurations in Earlier Releases**

If you are upgrading from a pre-Oracle 10g Primary/Secondary configuration, then the Database Upgrade Assistant (DBUA) creates a service on your database with one preferred instance and one available instance.
Administering Oracle Clusterware Components

The Oracle Clusterware includes two important components: the voting disk and the Oracle Cluster Registry (OCR). The voting disk is a file that manages information about node membership and the OCR is a file that manages cluster and RAC database configuration information. This chapter describes how to administer the voting disks and the Oracle Cluster Registry (OCR) under the following topics:

- Administering Voting Disks in Real Application Clusters
- Administering the Oracle Cluster Registry in Real Application Clusters

Administering Voting Disks in Real Application Clusters

Oracle recommends that you select the option to configure multiple voting disks during the Oracle Clusterware installation to improve availability. After installation, use the following procedures to regularly backup your voting disks and to recover them as needed:

- Backing up Voting Disks
- Recovering Voting Disks

**Note:** You can dynamically add voting disks after you complete the Oracle Clusterware and RAC installation processes.

**See Also:** "Administering the Oracle Cluster Registry in Real Application Clusters" on page 3-2 for more information about administering the OCR

Backing up Voting Disks

Run the following command to back up a voting disk. Perform this operation on every voting disk as needed where `voting_disk_name` is the name of the active voting disk and `backup_file_name` is the name of the file to which you want to back up the voting disk contents:

```
dd if=voting_disk_name of=backup_file_name
```

**Note:** You can use the `ocopy` command in Windows environments or use the `crsctl` commands described in the following note.
Recovering Voting Disks

Run the following command to recover a voting disk where `backup_file_name` is the name of the voting disk backup file and `voting_disk_name` is the name of the active voting disk:

```
dd if=backup_file_name of=voting_disk_name
```

**Note:** If you have multiple voting disks, then you can remove the voting disks and add them back into your environment using the `crsctl delete css votedisk path` and `crsctl add css votedisk path` commands respectively, where `path` is the complete path of the location on which the voting disk resides.

Changing the Voting Disk Configuration after Installing Real Application Clusters

You can dynamically add and remove voting disks after installing Real Application Clusters. Do this using the following commands where `path` is the fully qualified path for the additional voting disk. Run the following command as the root user to add a voting disk:

```
crsctl add css votedisk path
```

Run the following command as the root user to remove a voting disk:

```
crsctl delete css votedisk path
```

**Note:** If your cluster is down, then you can use `-force` option to modify the voting disk configuration with either of these commands without interacting with active Oracle Clusterware daemons. However, using the `-force` option while any cluster node is active may corrupt your configuration.

Administering the Oracle Cluster Registry in Real Application Clusters

This section describes how to administer the OCR. The OCR contains information about the cluster node list, instance-to-node mapping information, and information about Oracle Clusterware resource profiles for applications that you have customized as described in Chapter 13, "Making Applications Highly Available Using the Oracle Clusterware". The procedures discussed in this section are:

- Adding, Replacing, Repairing, and Removing the OCR
- Managing Backups and Recovering the OCR Using OCR Backup Files
- Diagnosing OCR Problems with the OCRDUMP and OCRCHECK Utilities
- Overriding the Oracle Cluster Registry Data Loss Protection Mechanism
- Administering the Oracle Cluster Registry with OCR Exports
- Implementing the Oracle Hardware Assisted Resilient Data Initiative for the OCR
- Upgrading and Downgrading the OCR Configuration in Real Application Clusters

**See Also:** Appendix A, "Troubleshooting" for information about the ocrdump utility
Adding, Replacing, Repairing, and Removing the OCR

The Oracle installation process for RAC gives you the option of automatically mirroring the OCR. This creates a second OCR to duplicate the original OCR. You can put the mirrored OCR on an Oracle cluster file system disk, on a shared raw device, or on a shared raw logical volume.

You can also manually mirror the OCR if you:

- Upgraded to release 10.2 but did not choose to mirror the OCR during the upgrade
- Created only one OCR during the Oracle Clusterware installation

Note: Oracle strongly recommends that you use mirrored OCRs if the underlying storage is not RAID. This prevents the OCR from becoming a single point of failure.

In addition to mirroring the OCR, you can also replace the OCR if Oracle displays an OCR failure alert in Enterprise Manager or in the Oracle Clusterware alert log file. You can also repair an OCR location if there is a mis-configuration or other type of OCR error. In addition, you can remove an OCR location if, for example, your system experiences a performance degradation due to OCR processing or if you transfer your OCR to RAID storage devices and chose to no longer use multiple OCRs. Use the following procedures to perform these tasks:

- Adding an Oracle Cluster Registry
- Replacing an Oracle Cluster Registry
- Repairing an Oracle Cluster Registry Configuration on a Local Node
- Removing an Oracle Cluster Registry

Note: The operations in this section affect the OCR cluster-wide: they change the OCR configuration information in the ocr.loc file on UNIX-based systems and the Registry keys on Windows-based systems. However, the ocrconfig command cannot modify OCR configuration information for nodes that are shut down or for nodes on which the Oracle Clusterware is not running.

Adding an Oracle Cluster Registry

You can add an OCR location after an upgrade or after completing the RAC installation. If you already mirror the OCR, then you do not need to add an OCR location; Oracle automatically manages two OCRs when it mirrors the OCR. RAC environments do not support more than two OCRs, a primary OCR and a second OCR.

Note: If your OCR resides on a cluster file system file or if the OCR is on an network file system, then create the target OCR file before performing the procedures in this section.

Run the following command to add an OCR location using either destination_file or disk to designate the target location of the additional OCR:

```
ocrconfig -replace ocr destination_file or disk
```
Run the following command to add an OCR mirror location using either `destination_file` or `disk` to designate the target location of the additional OCR:

```
ocrconfig -replace ocrmirror destination_file or disk
```

---

**Note:** You must be root user to run `ocrconfig` commands.

### Replacing an Oracle Cluster Registry

You can replace a mirrored OCR using the following procedure as long as one OCR-designated file remains online:

1. Verify that the OCR that you are *not* going to replace is online.
2. Verify that the Oracle Clusterware is running on the node on which the you are going to perform the replace operation.

**Note:** The OCR that you are replacing can be either online or offline. In addition, if your OCR resides on a cluster file system file or if the OCR is on an network file system, then create the target OCR file before continuing with this procedure.

3. Run the following command to replace the OCR using either `destination_file` or `disk` to indicate the target OCR:

```
ocrconfig -replace ocr destination_file or disk
```

Run the following command to replace an OCR mirror location using either `destination_file` or `disk` to indicate the target OCR:

```
ocrconfig -replace ocrmirror destination_file or disk
```

4. If any node that is part of your current RAC environment is shut down, then run the command `ocrconfig -repair` on any node that is stopped to enable that node to rejoin the cluster after you restart the stopped node.

### Repairing an Oracle Cluster Registry Configuration on a Local Node

You may need to repair an OCR configuration on a particular node if your OCR configuration changes while that node is stopped. For example, you may need to repair the OCR on a node that was not up while you were adding, replacing, or removing an OCR. To repair an OCR configuration, run the following command on the node on which you have stopped the Oracle Clusterware daemon:

```
ocrconfig -repair ocrmirror device_name
```

This operation only changes the OCR configuration on the node from which you run this command. For example, if the OCR mirror device name is `/dev/raw1`, then use the command syntax `ocrconfig -repair ocrmirror /dev/raw1` on this node to repair its OCR configuration.

**Note:** You cannot perform this operation on a node on which the Oracle Clusterware daemon is running.
Removing an Oracle Cluster Registry

To remove an OCR location, at least one other OCR must be online. You can remove an OCR location to reduce OCR-related overhead or to stop mirroring your OCR because you moved your the OCR to redundant storage such as RAID. Perform the following procedure to remove an OCR location from your RAC environment:

1. Ensure that at least one OCR other than the OCR that you are removing is online.

   **Caution:** Do not perform this OCR removal procedure unless there is at least one other active OCR online.

2. Run the following command on any node in the cluster to remove the OCR:

   `ocrconfig -replace ocr`

   Run the following command on any node in the cluster to remove the mirrored OCR:

   `ocrconfig -replace ocmmirror`

   These commands update the OCR configuration on all of the nodes on which the Oracle Clusterware is running.

   **Note:** When removing an OCR location, the remaining OCR must be online. If you remove a primary OCR, then the mirrored OCR becomes the primary OCR.

Managing Backups and Recovering the OCR Using OCR Backup Files

This section describes two methods for copying OCR content and using it for recovery. The first method uses automatically generated OCR file copies and the second method uses manually created OCR export files.

The Oracle Clusterware automatically creates OCR backups every four hours. At any one time, Oracle always retains the last three backup copies of the OCR. The CRSD process that creates the backups also creates and retains an OCR backup for each full day and at the end of each week.

You cannot customize the backup frequencies or the number of files that Oracle retains. You can use any backup software to copy the automatically generated backup files at least once daily to a different device from where the primary OCR resides.

The default location for generating backups on UNIX-based systems is `CRS_HOME/cdata/cluster_name` where `cluster_name` is the name of your cluster. The Windows-based default location for generating backups uses the same path structure.

   **Note:** You must be root user to run ocrconfig commands.

Restoring the Oracle Cluster Registry from Automatically Generated OCR Backups

If an application fails, then before attempting to restore the OCR, restart the application. As a definitive verification that the OCR failed, run an `ocrcheck` and if the command returns a failure message, then both the primary OCR and the OCR mirror have failed. Attempt to correct the problem using one of the following platform-specific OCR restoration procedures.
Restoring the Oracle Cluster Registry on UNIX-Based Systems

Use the following procedure to restore the OCR on UNIX-based systems:

1. Identify the OCR backups using the `ocrconfig -showbackup` command. Review the contents of the backup using `ocrdump -backupfile file_name` where `file_name` is the name of the backup file.

2. Stop the Oracle Clusterware software on all of the nodes in your RAC database by executing the `init.crs stop` command on all of the nodes.

3. Perform the restore by applying an OCR backup file identified in Step 1 using the following command where `file_name` is the name of the OCR that you want to restore:

   ```
   ocrconfig -restore file_name
   ```

4. Restart the Oracle Clusterware on all of the nodes in your cluster by restarting each node or by running the `init.crs start` command.

5. Run the following command to verify the OCR integrity where the `-n all` argument retrieves a listing of all of the cluster nodes that are configured as part of your cluster:

   ```
   cluvfy comp ocr -n all [-verbose]
   ```

See Also: Appendix A, "Troubleshooting" for more information about enabling and using CVU

Restoring the Oracle Cluster Registry on Windows-Based Systems

Use the following procedure to restore the OCR on Windows-based systems:

1. Identify the OCR backups using the `ocrconfig -showbackup` command. Review the contents of the backup using `ocrdump -backupfile file_name` where `file_name` is the name of the backup file.

2. On the remaining node, disable the following OCR clients and stop them using the Service Control Panel: OracleClusterVolumeService, OracleCSService, OracleCRService, and the OracleEVMService.

3. Identify the most recent backups using the `ocrconfig -showbackup` command.

4. Execute the restore by applying an OCR backup file identified in Step 3 with the `ocrconfig -restore file name` command.

5. Start all of the services that were stopped in step 2. Restart all of the nodes and resume operations in cluster mode.

6. Run the following command to verify the OCR integrity where the `-n all` argument retrieves a listing of all of the cluster nodes that are configured as part of your cluster:

   ```
   cluvfy comp ocr -n all [-verbose]
   ```

Note: You cannot restore your configuration from an automatically created OCR backup file using the `-import` option, which is explained in "Administering the Oracle Cluster Registry with OCR Exports" on page 3-8. You must instead use the `-restore` option as described in the following sections.
cluvfy comp ocr -n all [-verbose]

See Also: "Cluster Verification Utility Oracle Clusterware Component Verifications" on page A-18 for more information about enabling and using CVU

Diagnosing OCR Problems with the OCRDUMP and OCRCHECK Utilities

You can use the OCRDUMP and OCRCHECK utilities to diagnose OCR problems as described under the following topics:

- Using the OCRDUMP Utility
- Using the OCRCHECK Utility

Using the OCRDUMP Utility

Use the OCRDUMP utility to write the OCR contents to a file so that you can examine the OCR content.

See Also: "OCRDUMP Utility Syntax and Options" on page A-6 for more information about the OCRDUMP utility

Using the OCRCHECK Utility

Use the OCRCHECK utility to verify the OCR integrity.

See Also: Using the OCRCHECK Utility on page A-7 for more information about the OCRCHECK utility

Overriding the Oracle Cluster Registry Data Loss Protection Mechanism

The OCR has a mechanism that prevents data loss due to accidental overwrites. If you configure a mirrored OCR and if the OCR cannot access the two mirrored OCR locations and also cannot verify that the available OCR contains the most recent configuration, then the OCR prevents further modification to the available OCR. The OCR prevents overwriting by prohibiting the Oracle Clusterware from starting on the node on which the OCR resides. In such cases, Oracle displays an alert message in either Enterprise Manager, the Oracle Clusterware alert log files, or both.

Sometimes this problem is local to only one node and you can use other nodes to start your cluster database. In such cases, Oracle displays an alert message in EM, the Oracle Clusterware alert log, or both.

However, if you are unable to start any cluster nodes in your environment and if you cannot repair the OCR, then you can override the protection mechanism. This procedure enables you to start the cluster using the available OCR, thus enabling you to use the updated OCR file to start your cluster. However, this can result in the loss of data that was not available at the time that the previous known good state was created.

Note: Overriding the OCR using this procedure can result in the loss of OCR updates that were made between the time of the last known good OCR update made to the currently-accessible OCR and the time at which you performed this procedure. In other words, running the ocrconfig -overwrite command can result in data loss if the OCR that you are using to perform the overwrite does not contain the latest configuration updates for your cluster environment.
Perform the following procedure to overwrite the OCR if a node cannot start up and if the alert log contains a CLSD-1009 or CLSD-1011 message.

1. Attempt to resolve the cause of the a CLSD-1009 or CLSD-1011 message. Do this by comparing the node’s OCR configuration (ocr.loc on Unix-based systems and the Registry on Windows-based systems) with other nodes on which the Oracle Clusterware is running. If the configurations do not match, then run ocrconfig -repair. If the configurations match, then ensure that the node can access all of the configured OCRs by running an ls command on Unix-based systems or a dir command on Windows-based systems. Oracle issues a warning when one of the configured OCR locations is not available or if the configuration is incorrect.

2. Ensure that the most recent OCR contains the latest OCR updates. Do this by taking output from the ocrdump command and determine whether it has your latest updates.

3. If you cannot resolve the CLSD message, then run the command ocrconfig -overwrite to bring up the node.

Administering the Oracle Cluster Registry with OCR Exports

In addition to using the automatically created OCR backup files, you should also export the OCR contents before and after making significant configuration changes, such as adding or deleting nodes from your environment, modifying Oracle Clusterware resources, or creating a database. Do this by using the ocrconfig -export command. This exports the OCR content to a file format.

Using the ocrconfig -export command enables you to restore the OCR using the -import option if your configuration changes cause errors. For example, if you have unresolvable configuration problems, or if you are unable to restart your clusterware after such changed, then restore your configuration using one of the following platform-specific procedures:

- Importing Oracle Cluster Registry Content on UNIX-Based Systems
- Importing Oracle Cluster Registry Content on Windows-Based Systems

**Note:** Most configuration changes that you make not only change the OCR contents, configuration changes also cause file and database object creation. Some of these changes are often not restored when you restore the OCR. Do not perform an OCR restore as a correction to revert to previous configurations if some of these configuration changes should fail. This may result in an OCR that has contents that do not match the state of the rest of your system.

Importing Oracle Cluster Registry Content on UNIX-Based Systems

Use the following procedure to import the OCR on UNIX-based systems:

1. Stop the Oracle Clusterware on all of the nodes in your cluster database by executing the init.crs stop command on all of the nodes.

2. Perform the restore by applying an OCR backup file, which is the file identified in Step 1, using the following command where file_name is the name of the OCR file from which you want to import OCR information:

   ocrconfig -import file_name
3. Restart the Oracle Clusterware on all of the nodes in your cluster by restarting each node.

4. Run the following Cluster Verification Utility (CVU) command to verify the OCR integrity where the \(-n\) all argument retrieves a listing of all of the cluster nodes that are configured as part of your cluster:

   `cluvfy comp ocr -n all [-verbose]`

---

**Note:** You cannot import an exported OCR backup file, which is described in "Managing Backups and Recovering the OCR Using OCR Backup Files" on page 3-5. You must instead use the \(-import\) option as described in the following sections.

**See Also:** Appendix A, "Troubleshooting" for more information about enabling and using CVU

---

**Importing Oracle Cluster Registry Content on Windows-Based Systems**

Use the following procedure to import the OCR on Windows-based systems:

1. Stop the following OCR clients on each node in your RAC environment using the Service Control Panel: OracleClusterVolumeService, OracleCMService, OracleEVMService, OracleCSService, and the OracleCRService.

2. Import an OCR export file using the `ocrconfig -import` command from one node.

3. Restart all of the affected services on all nodes.

4. Run the following Cluster Verification Utility (CVU) command to verify the OCR integrity where `node_list` is a list of all of the nodes in your cluster database:

   `cluvfy comp ocr -n all [-verbose]`

**See Also:** Appendix A, "Troubleshooting" for more information about enabling and using CVU

---

**Implementing the Oracle Hardware Assisted Resilient Data Initiative for the OCR**

The Oracle Hardware Assisted Resilient Data (HARD) initiative prevents data corruptions from being written to permanent storage. If you enable HARD, then the OCR writes HARD-compatible blocks. To determine whether the device used by the OCR supports HARD and then enable it, review the Oracle HARD white paper at:


---

**Upgrading and Downgrading the OCR Configuration in Real Application Clusters**

When you install the Oracle Clusterware, Oracle automatically runs the `ocrconfig -upgrade` command. To downgrade, follow the downgrade instructions for each component and also downgrade the OCR using the `ocrconfig -downgrade` command. If you are upgrading the OCR to Oracle Database 10g release 10.2, then you can use the `cluvfy` command to verify the integrity of the OCR. If you are downgrading, you cannot use the Cluster Verification Utility (CVU) commands to verify the OCR for pre-10.2 release formats.
HARD-Compatible OCR Blocks in Oracle9i

In Oracle9i, the OCR did not write HARD-compatible blocks. If the device used by OCR is enabled for HARD, then use the method described in the HARD white paper to disable HARD for the OCR before downgrading your OCR. If you do not disable HARD, then the downgrade operation fails.
This chapter describes storage topics such as Automatic Storage Management (ASM). The topics in this chapter are:

- Overview of Storage in Oracle Real Application Clusters
- Automatic Storage Management in Real Application Clusters

**Overview of Storage in Oracle Real Application Clusters**

Storage for RAC databases must be shared. In other words, datafiles must reside in an Automatic Storage Management (ASM) disk group, on a cluster file system, or on shared raw devices. This must include space for an undo tablespace for each instance if you are using automatic undo management. Additionally, for each instance you must create at least two redo log files that reside on shared storage. In addition, as Oracle recommends, you can use one shared server parameter file (SPFILE) with instance-specific entries. Or you can use a local file system to store client side parameter files (PFILES).

Unless otherwise noted, Oracle storage features such as ASM, Oracle Managed Files (OMF), automatic segment-space management, and so on, function the same in RAC environments as they do in single-instance Oracle database environments. Refer to Oracle Database 2 Day DBA and the Oracle Database Administrator’s Guide for additional information about these storage features.

If you do not use ASM, if your platform does not support a cluster file system, or if you do not want to use a cluster file system to store datafiles, then create additional raw devices as described in your platform-specific Oracle Real Application Clusters installation and configuration guide. However, Oracle recommends that you use ASM for datafile storage which is described later in this chapter in the section titled "Automatic Storage Management in Real Application Clusters" on page 4-2. The remainder of this section describes the following topics:

- Datafile Access in Real Application Clusters
- Redo Log File Storage in Real Application Clusters
- Automatic Undo Management in Real Application Clusters

**Note:** To create a RAC database using the Oracle Database 10g Standard Edition, you must use ASM for your datafile storage.
**Datafile Access in Real Application Clusters**

All RAC instances must be able to access all datafiles. If a datafile needs to be recovered when the database is opened, then the first 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 datafiles. Similarly, when you add a tablespace or datafile or bring a tablespace or datafile online, all instances verify access to the file or files.

If you add a datafile to a disk that other instances cannot access, then verification fails. Verification also fails if instances access different copies of the same datafile. 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 datafile access.

**Redo Log File Storage in Real Application Clusters**

Each instance has its own online redo log groups which are referred to as an instance’s thread of online redo. Create these online redo log groups and establish group members as described in the *Oracle Database Administrator’s Guide*.

Each instance must have at least two groups of online redo log files in its thread. 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 log files into its own archive log thread and update the control file with the status of its thread.

---

**Note:** MAXLOGHISTORY is useful for sites with demanding availability requirements. This option can help you administer recovery, especially when there are many instances and many log files.

---

**Automatic Undo Management in Real Application Clusters**

Oracle 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, as long as that undo tablespace is not currently used by another instance for undo generation or transaction recovery.

You assign undo tablespaces in your RAC database by specifying a different value for the `UNDO_TABLESPACE` parameter for each instance in your SPFILE or individual PFILEs. You cannot simultaneously use automatic undo management and manual undo management in a RAC database. In other words, all instances of a 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

---

**Automatic Storage Management in Real Application Clusters**

ASM automatically optimizes storage to maximize performance by managing the storage configuration across the disks that ASM manages. ASM does this by evenly distributing the storage load across all of the available storage within your *cluster database* environment. ASM partitions your total disk space requirements into
uniformly sized units across all disks in a disk group. ASM can also automatically mirror data to prevent data loss. Because of these features, ASM also significantly reduces your administrative overhead.

To use ASM in RAC, select ASM as your storage option when you create your database with the Database Configuration Assistant (DBCA). As in single-instance Oracle databases, using ASM in RAC does not require I/O tuning. The following topics describe ASM and ASM administration as follows:

- **Automatic Storage Management Components in RAC**
- **Modifying Disk Group Configurations for ASM in RAC**
- **Standalone ASM Disk Group Management**
- **Administering ASM Instances and Disk Groups with Enterprise Manager in RAC**
- **Administering ASM Instances with SRVCTL in RAC**

### Automatic Storage Management Components in RAC

When you create your database, Oracle creates one ASM instance on each node in your RAC environment if one does not already exist. Each ASM instance has either an SPFILE or PFILE type parameter file. You may need to back up the parameter files and the TNS entries for nondefault Oracle Net Listeners.

### Modifying Disk Group Configurations for ASM in RAC

When you create a disk group for a cluster or add new disks to an existing clustered disk group, you only need to prepare the underlying physical storage on shared disks. The shared disk requirement is the only substantial difference between using ASM in a RAC database compared to using it in a single-instance Oracle database. ASM automatically rebalances the storage load after you add or delete a disk or disk group.

In a cluster, each ASM instance manages its node’s metadata updates to the disk groups. In addition, each ASM instance coordinates disk group metadata with other nodes in the cluster. As in single-instance Oracle databases, you can use Enterprise Manager, the DBCA, SQL*Plus, and the Server Control Utility (SRVCTL) to administer disk groups for ASM in RAC. The *Oracle Database Administrator’s Guide* explains how to use SQL*Plus to administer ASM instances. The following sections describe how to use the other tools.

### Standalone ASM Disk Group Management

When you create a database using the DBCA and you select the ASM storage option, the DBCA creates the ASM instances for you if they do not already exist. However, you can also use the standalone ASM disk group management feature to create and manage an ASM instance and its associated disk groups independently of creating a new database. You can use Enterprise Manager or the DBCA to add disks to a disk group, to mount a disk group or to mount all of the disk groups, or to create ASM instances. Additionally, you can use Enterprise Manager to dismount and drop disk groups or to delete ASM instances.

To create an ASM instance without creating a database with the DBCA, select the **Configure Automatic Storage Management** option on the DBCA Database Options page. You can also use this option to add or mount one or more ASM disk groups. The DBCA then displays the Node Selection page on which you can identify the nodes on which you want to create the ASM instance or on which you want to manage disk groups. If necessary, the next page you must complete is the DCBA Create Instance
Page on which you add the information about the ASM instance parameter file and SYS password and, for Windows-based systems, the owner of the ASM-related service.

The DBCA session provides the same ASM Disk Groups page for standalone ASM management as it does for database creation using ASM storage. Use this page to create new disk groups, to add disks to existing disk groups, or to mount disk groups that are not currently mounted. You can also use the standalone ASM feature, where you have the ability to manage ASM instances independent of the database, in DBCA silent mode to create an ASM instance or to manage ASM disk groups. The syntax for the DBCA silent mode command is:

```
dbca -silent -nodeList nodelist -configureASM -asmPassword asm_pwd [-diskList disk_list] [-redundancy redundancy_option] [-diskGroupName dgname] [-diskString disk_discovery_string] [-recoveryGroupName recovery_dgname] [-recoveryRedundancy redundancy_option]
```

Where:

- `nodelist` is a comma-delimited list of nodes and these are the nodes on which you want to perform the ASM activities
- `asm_pwd` is the password for the SYS account in the ASM instance
- `disk_list` is a comma-delimited list of disks to be added to disk group
- `redundancy_option` is NORMAL, HIGH, or EXTERNAL
- `dgname` is either the name of an existing disk group, in which case the disk group is mounted on all of the selected nodes, or `dgname` is the name of a new disk group, in which case the disk group is created and started on all of the selected nodes
- `disk_recovery_string` identifies the location or identifier for the ASM-ready disks
- `recovery_dgname` is the name for a recovery disk group

Administering ASM Instances and Disk Groups with Enterprise Manager in RAC

You can perform administrative operations on ASM disk groups with Enterprise Manager such as adding and deleting them. You can also monitor ASM disk group performance as well as control disk group availability at the instance level. For example, some of the RAC-specific ASM-related Enterprise Manager tasks that you can perform are:

- When you add a disk group, the disk group definition includes a checkbox to indicate whether the disk group is automatically mounted to all of the cluster database instances.
- The default Disk Group Performance page displays instance-level performance details when you click a performance characteristic such as Write Response Time or I/O Throughput.
- When you mount and dismount ASM disk groups, you can use a checkbox to indicate which instances should mount or dismount a particular ASM Disk Group.

See Also: Oracle Database Administrator’s Guide and Oracle Database 2 Day DBA for detailed information about using Enterprise Manager
Administering ASM Instances with SRVCTL in RAC

You can use the Server Control Utility (SRVCTL) to add, remove, enable, and disable an ASM instance as described in the following procedures:

Use the following syntax to add configuration information about an existing ASM instance:

```
srvctl add asm -n node_name -i asm_instance_name -o oracle_home
```

**Note:** For all of the SRVCTL commands in this section for which the `-i` option is not required, if you do not specify an instance name, then the command applies to all of the ASM instances on the node.

Use the following syntax to remove an ASM instance:

```
srvctl remove asm -n node_name [-i asm_instance_name]
```

Use the following syntax to enable an ASM instance:

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

Use the following syntax to disable an ASM instance:

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

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

Use the following syntax to start an ASM instance:

```
srvctl start asm -n node_name [-i asm_instance_name] [-o start_options] [-c <connect_str> | -q]
```

Use the following syntax to stop an ASM instance:

```
srvctl stop asm -n node_name [-i asm_instance_name] [-o stop_options] [-c <connect_str> | -q]
```

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

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

Use the following syntax to obtain the status of an ASM instance:

```
srvctl status asm -n node_name
```
This chapter describes how to administer Oracle Real Application Clusters (RAC) database instances and RAC databases. This chapter explains the startup and shutdown tasks for database components and well as how to administer parameters and parameter files in RAC. The topics in this chapter are:

- Overview of Real Application Clusters Management Tools
- Starting and Stopping Instances and Real Application Clusters Databases
- Overview of Initialization Parameter Files in Real Application Clusters
- Initialization Parameter Use in Real Application Clusters
- Summary of Parameter Use in Real Application Clusters Databases
- Backing Up the Server Parameter File

See Also: Chapter 4, "Administering Storage" for information about managing Automatic Storage Management (ASM) instances

Overview of Real Application Clusters Management Tools

The following section introduces the three tools that you will most likely use to manage RAC databases and RAC instances, namely, Oracle Enterprise Manager, SQL*Plus, and the SRVCTL utility. In many cases, you use these tools the same way to manage RAC environments as you would use them manage single-instance Oracle databases. The following sections identify the differences when managing a RAC database:

- Overview of Administering Real Application Clusters with Enterprise Manager
- Overview of Administering Real Application Clusters with SQL*Plus
- Overview of Administering Real Application Clusters with SRVCTL

Overview of Administering Real Application Clusters with Enterprise Manager

Use the Web-based Enterprise Manager Database Control to manage a single RAC database. The Enterprise Manager Console provides a central point of control for the Oracle environment through a graphical user interface (GUI). You can use the Enterprise Manager Console to initiate a variety of cluster database management tasks. Use Enterprise Manager Grid Control to administer multiple RAC databases.

Enterprise Manager enables you to start, stop, and monitor databases, cluster database instances, and their Listeners, as well as to schedule jobs or set up alert thresholds for...
metrics. You can perform these tasks simultaneously on multiple cluster databases. You can also use the Console to manage schemas, security, and cluster database storage features.

**See Also:** Oracle Enterprise Manager Concepts for more information about administering RAC with Enterprise Manager

### Overview of Administering Real Application Clusters 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 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 specifying a remote instance net service name as in the following example:

```sql
CONNECT user_name/password@net_service_name
```

**Note:** You may wish to change the SQL*Plus prompt so that it includes the name of the current instance. To do this you can issue a SQL*Plus command such as the following:

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

This command replaces the "SQL" string in front of the greater than symbol (>) with the user variable _CONNECT_IDENTIFIER that will display 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:

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

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

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.

**See Also:**

- Oracle Database Net Services Administrator’s Guide for the proper specification of `net_service_name`
- The Oracle Database Administrator’s Guide for information about connecting to the database using SYSDBA or SYSOPER privileges
Overview of Administering Real Application Clusters with SRVCTL

The SRVCTL tool manages configuration information that is used by several other Oracle tools. For example, Enterprise Manager uses the configuration information that SRVCTL generates to discover and monitor nodes in your 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 calling SQL*Plus on each node.

Starting and Stopping Instances and Real Application Clusters Databases

Oracle RAC Runtime Connection Load Balancing—

You can start up and shut down instances with Enterprise Manager, SQL*Plus or SRVCTL as described in the following sections. Both Enterprise Manager and SRVCTL provide options to startup and shutdown all of the instances in a 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.

The procedures in this section assume that you are using a server parameter file (SPFILE) and are described in the following topics:

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

Before you can start a 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 RAC instances is identical to shutting down instances in single-instance Oracle, with the exceptions described here. Refer to the Oracle Database Administrator’s Guide for more information about shutting down Oracle databases.

- In RAC, shutting down one instance does not interfere with the operation of other running instances.
- To shut down a 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 Up and Shutting Down with Enterprise Manager

You can shut down all instances or any identified instance from the Cluster Database Startup and Shutdown page. To start up or shut down a cluster database instance:

1. Go to the Cluster Database Home page.
2. Click the appropriate Startup or Shutdown button for your situation. If the system is down, a Startup button appears. If the system is up, a Shutdown button appears. A Specify Host and Target Database Credentials page appears.
3. Enter valid user names and passwords for host and database credentials to change the status of the database, then click OK. To change the status of the database, you must log in to the database as either SYSDBA or SYSOPER.

To start or shut down a cluster database, that is, all of the instances known to Enterprise Manager:

1. Go to the Cluster Database Home page.
2. Click Startup/Shutdown. The Specify Credentials page appears.
3. Enter a valid User Name and Password for the cluster that hosts the cluster database, then click Continue. You must be a member of the OSDBA group.

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, will be 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 within your SQL*Plus session:

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

You can start more than one instance from a single SQL*Plus session on one node by way of Oracle Net Services. To achieve this, you must 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 your SQL*Plus session:

CONNECT /@db2 AS SYSDBA
SHUTDOWN TRANSACTIONAL

Other startup and shut down keywords, such as NOMOUNT, MOUNT, IMMEDIATE, and so on, are described in the SQL*Plus User’s Guide and Reference.

It is not possible to start up or shut down more than one instance at a time in SQL*Plus, so you cannot start or stop all of the instances for a cluster database with a single SQL*Plus command. You may wish to create a script that will connect to each instance in turn and start it up and shut it down. However, you will need to maintain this script manually if you add or drop instances.

Intermittent Windows Shutdown Issue in RAC Environments

In an Oracle Real Application Clusters release 10.1.0.2 environment on Windows, a normal Windows shutdown may cause errors that prevent the Windows shutdown from completing. As a result, you may need to perform a power reset. The following steps are recommended to avoid this during Windows shutdowns. Before shutting down or restarting any Oracle cluster node, perform a graceful shutdown of all registered Oracle Clusterware resources on the affected cluster node. Do this by using SRVCTL commands to shutdown:

- All services on the node.
- All database instances on the node.
- All ASM instances on the node.
- All node applications on the node.

Lengthy Startup of OracleDBConsole and OracleCRService on Windows

After a cluster node restart, the node may not be fully responsive for some period of time. During this time, Oracle is attempting to restart the process OracleDBConsole\sid and the OracleCRService resource. Eventually, all of the resource startup operations will complete and the computer will operate normally.

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 more than one instance name to start more than one specific instance:

```
srvctl start instance -d db_name -i "inst_name_list" [-o start_options] [-c connect_str | -q]
```

Note that this command will also start all enabled and non-running services that have the listed instances either as preferred or available instances.

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

```
srvctl stop instance -d name -i "inst_name_list" [-o stop_options] [-c connect_str | -q]
```
This command will also stop the services related to the terminated instances on the
nodes where the instances were running. As an example, the following command
provides its own connection information to shut down the two instances, orcl3 and
orcl4, using the IMMEDIATE option:

```
srvc1 stop instance -d orcl -i "orcl3,orcl4" -o immediate -c "sysback/oracle as sysoper"
```

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

```
srvc1 start database -d name [-o start_options] [-c connect_str | -q]
```

```
srvc1 stop database -d name [-o stop_options] [-c connect_str | -q]
```

The following SRVCTL command, for example, mounts all of the non-running
instances of a RAC database using the default connection information:

```
srvc1 start database -d orcl -o mount
```

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

Customizing How Oracle Clusterware Manages RAC Databases

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

To prevent the Oracle Clusterware from restarting your 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.

To use this feature, configure one of the pre-defined policies to define the degree of
control. There are two policy types: automatic, which is the default, and manual. The
manual policy, which minimizes the database instance protection level, is an override
of the automatic policy.

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

Switching Between the Automatic and Manual Policies

Use SRVCTL commands to display and change these Oracle Clusterware policies. For
example, use the following command syntax to display the current policy where
database_name is the name of the database for which you want to change policies:

```
srvc1 config database -d database_name -a
```

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 is
identified by database_name:
srvctl modify database d database_name -y policy_name

This command syntax changes the resource profile values for each applicable resource and sets the Current Policy OCR key to the new value. When you add a new database using the `srvctl` command, you can use the `-y` option to specify the policy as in the following example where `database_name` is the name of the database and `policy_name` is the name of the policy:

```bash
text
```
srvctl add database -d database_name -y policy_name

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

**See Also:** Chapter 13, "Making Applications Highly Available Using the Oracle Clusterware" for more information about using Oracle Clusterware and Appendix D, "Server Control Utility Reference" for more information about SRVCTL commands

### Overview of Initialization Parameter Files in Real Application Clusters

When you create the database, Oracle creates an SPFILE in the file location that you specify. This location can be an ASM disk group, cluster file system file, 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).

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

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, 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.

### Setting Server Parameter File Parameter Values for Real Application Clusters

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

Running the following statement makes Oracle disregard the first entry from the server parameter file:

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

To reset a parameter to its default value throughout your cluster database, enter the statement:

```
alter system reset open_cursors scope=spfile sid='prod1';
```

---

**Note:** Modifying SPFILEs with anything except Enterprise Manager or SQL*Plus can corrupt the file and prevent database startup. To repair the file, you might need to create the PFILE and regenerate the SPFILE.

---

### Parameter File Search Order in Real Application Clusters

Oracle searches for your parameter file in a particular order depending on your platform. On UNIX-based platforms, Oracle examines directories in the following order:

1. `$ORACLE_HOME/dbs/spfilesid.ora`
2. `$ORACLE_HOME/dbs/spfile.ora`
3. `$ORACLE_HOME/dbs/initsid.ora`

The search order on Windows-based platforms is:

1. `%ORACLE_HOME%\database\spfilesid.ora`
2. `%ORACLE_HOME%\database\spfile.ora`
3. `%ORACLE_HOME%\database\initsid.ora`

---

### Initialization Parameter Use in Real Application Clusters

By default, Oracle sets most parameters 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 the *Oracle Database Reference*. Other parameters must either be unique or identical as described in the following sections.
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 a RAC database. Specify these parameter values in the SPFILE, or within the individual PFILEs for each instance. The following list contains the parameters must be identical on every instance:

- ACTIVE_INSTANCE_COUNT
- ARCHIVE_LAG_TARGET
- CLUSTER_DATABASE
- CLUSTER_DATABASE_INSTANCES
- CONTROL_FILES
- DB_BLOCK_SIZE
- DB_DOMAIN
- DB_FILES
- DB_NAME
- DB_RECOVERY_FILE_DEST
- DB_RECOVERY_FILE_DEST_SIZE
- DB_UNIQUE_NAME
- UNDO_MANAGEMENT

The setting for DML_LOCKS must be identical on every instance only if set to zero.

Parameters That Must Have Unique Settings on All Instances

If you use the THREAD or ROLLBACK_SEGMENTS parameters, then Oracle recommends setting unique values for them 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.

Oracle uses the INSTANCE_NUMBER parameter to distinguish among instances at startup. Oracle uses the THREAD number to assign redo log groups to specific instances. To simplify administration, use the same number for both the THREAD and INSTANCE_NUMBER parameters.

Specify the ORACLE_SID environment variable, which comprises the database name and the number of the THREAD 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.

Parameters that Should Have Identical Settings on All Instances

Oracle recommends that you set the values for the following parameters 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:

- ARCHIVE_LAG_TARGET

Different values for instances in your RAC database are likely to increase overhead because of additional automatic synchronization performed by the database processing.
When using Streams with your RAC database, the value should be greater than zero.

- **LICENSE_MAX_USERS**
  
  This parameter determines a database-wide limit on the number of users defined in the database and 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 additional warning messages to be generated 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 archive 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, as well as 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 will not be 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 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.

- **UNDO_RETENTION**
  
  By setting different values for UNDO RETENTION in each instance, you are likely to reduce scalability and encounter unpredictable behavior following a fail over. Therefore, you should carefully consider whether you will accrue any benefits before you assign different values for this parameter to the instances in your RAC database.

### Summary of Parameter Use in Real Application Clusters Databases

This section summarizes considerations for using parameters in RAC databases.

- **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 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 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.

Note: The value for this parameter determines the maximum number of instances that you can have in your RAC database environment. If you add instances, then you may need to reset the value for this parameter to accommodate the increased number of instances.

- **CLUSTER_INTERCONNECTS**

  Specifies the cluster interconnect when there is more than one interconnect. Refer to your Oracle platform-specific documentation for the use, syntax, and behavior of this parameter.

  You typically do not need to set the **CLUSTER_INTERCONNECTS** parameter. For example, do not set this parameter for the following common configurations:

  - If you have only one cluster interconnect.
  - If the default cluster interconnect meets the bandwidth requirements of your RAC database, which is typically the case.
  - If your database is running on a Sun Solaris cluster. If you do set the parameter, its value must be the same on all instances.

  Oracle uses information from **CLUSTER_INTERCONNECTS** to distribute interconnect traffic among the various network interfaces if you specify more than one interconnect with this parameter. Note that the specified configuration inherits any limitations of the listed interconnects and the associated operating system IPC services, such as availability. 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 represents an IP address in standard dot-decimal format, for example: 144.25.16.214:

  Database One: **CLUSTER_INTERCONNECTS = ip1**
  Database Two: **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 uses all of the interconnects that you specify. This provides load balancing as long as 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.

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

---

**Note:** Setting this parameter does not provide interconnect failover. Because interconnect failover is operating-system dependent, refer to your operating system documentation for information about configuring interconnect failover.

---

- **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. Enterprise manager does not require `tnsnames.ora` entries on the client for Database Control or Grid Control. Refer to *Oracle Database Net Services Administrator’s Guide* for complete information about configuring the `DISPATCHERS` parameter and its attributes and for configuring the shared server.

- **INSTANCE_NAME**
  
  This parameter is normally set to a different value for each instance to provide a simple method for clients to connect directly to a specific instance. However, you may use the same value for multiple instances. The value of the `INSTANCE_NAME` parameter is usually the same as the value of the SID for the instance, to which it defaults, and this association of the two names may help reduce confusion when managing complex systems.

- **SERVICE_NAMES**
  
  When you use an SPFILE, all RAC database instances must use the SPFILE and the file must be on shared storage. 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. This enables 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 as long as each connection is from a different instance.

- **SPFILE**

---

Note: Setting this parameter does not provide interconnect failover. Because interconnect failover is operating-system dependent, refer to your operating system documentation for information about configuring interconnect failover.
When you use an SPFILE, all RAC database instances must use the SPFILE and the file must be on shared storage.

- **THREAD**

  If specified, this parameter must have unique values on all instances. The THREAD parameter specifies the number of the redo thread to be used by an instance. You can specify any available redo thread number as long as that thread number is enabled and is not used.

**Backing Up the Server Parameter File**

Oracle recommends that you regularly back up the server parameter file for recovery purposes. Do this using the `CREATE PFILE` statement. For example:

```sql
CREATE PFILE='?/dbs/initdbname.ora'
FROM SPFILE='/dev/vx/rdsk/oracle_dg/dbspfile'
```

You can use RMAN (Recovery Manager) 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 will not contain instance-specific values, even those that must be unique in 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 Enterprise Manager or the RMAN `CONTROLFILE AUTOBACKUP` statement to enable the RMAN autobackup feature.

**See Also:**

- *Oracle Database Backup and Recovery Basics* for more information about RMAN
- *Oracle Database SQL Reference* for more information about the `CREATE SPFILE` statement
This chapter describes how to manage workloads in Oracle Real Application Clusters (RAC) to provide high availability and scalability for your applications. This chapter contains the following topics:

- Introduction to Workload Management and Application High Availability
- Services Deployment Options
- Fast Application Notification
- Load Balancing Advisory
- Oracle Clients that are Integrated with Fast Application Notification
- Services and Distributed Transaction Processing in RAC
- Administering Services
- Administering Services with Enterprise Manager, DBCA, PL/SQL, and SRVCTL
- Measuring Performance by Service Using the Automatic Workload Repository

### Introduction to Workload Management and Application High Availability

Workload management enables you to manage the distribution of workloads to provide optimal performance for users and applications. Workload management comprises the following:

- **High Availability Framework**—Enables the Oracle Database to maintain components in a running state at all times.
- **Load balancing advisory**—Provides information to applications about the current service levels being provided by the database and its instances and provides recommendations to applications about where to direct application requests to obtain the best service based on the defined policy.
- **Services**—Oracle Database 10g introduces a powerful automatic workload management facility, called services, to enable the enterprise grid vision. Services are entities that you can define in RAC databases that enable you to group database workloads and route work to the optimal instances that are assigned to offer the service.

You can deploy RAC and single-instance Oracle database environments to use these components in many different ways. Depending on the number of nodes and the objectives and complexity of your environment, your choices for the optimal workload management and high availability configuration depend on several considerations that this chapter describes. Refer to the following section for information about various deployment options.
Services Deployment Options

This section describes the following services deployment options:

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

Using Oracle Services

To manage workloads, you can define services that you assign to a particular application or to a subset of an application's operations. You can also group other types of work under services. For example, online users can be a service, batch processing can be another, and reporting can be another service type.

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 is a separate unit of work. When you define a service, you define which instances will 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 the PREFERRED instances, you are specifying the number of instances on which a service will initially run. Afterwards, due to either instance availability or planned service relocations, a service may be running on an AVAILABLE instance. When a service moves to an available instance, Oracle does not move the service back to the PREFERRED instance when the preferred instance comes back online. This is because the service is already running on the desired number of instances. This provides a higher level of continual service availability and avoids a second service outage just for fail back processing. You can, however, easily automate fail back by using Fast Application Notification (FAN) callouts.

Services are integrated with the Resource Manager which enables you to restrict the resources that are used by a service within an instance. The Resource Manager enables you to map a consumer group to a service so that users connecting with the service are put into 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. 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.

The easiest way to use FAN is to use the Oracle Clients that have Fast Connection Failover (FCF), which means the clients have been integrated with the high availability FAN events. These include Java Database Connectivity (JDBC), Oracle Call Interface (OCI), and ODP.NET. OCI clients can include clients with TAF enabled. For OCI and ODP.NET, you must enable the service to send FAN high availability events, in other words, set AQ_HA_NOTIFICATIONS to TRUE.

With runtime connection load balancing, applications can use load balancing advisory events to provide better service to users. The Oracle JDBC 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.
Default Service Connections

By default, when you create a database, Oracle defines one service for your RAC database and this service is a special Oracle database service. The default database service is always available on all instances in a RAC environment, unless the instance is in restricted mode. You cannot alter this service or its properties.

The database also supports the following 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 workload management features. You cannot stop or disable either service.

Connection Load Balancing

Oracle Net Services provides the ability to balance client connections across the instances in a 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 Listener directs a connection request to the best instance by using the load balancing advisory. In a RAC database, client connections should use both types of connection load balancing.

FAN, FCF, 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 this 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 PL/SQL DBMS_SERVICE package and the CLB_GOAL_LONG package constant to define the connection load balancing goal for long-lived sessions:

  ```sql
  execute dbms_service.modify_service (service_name => 'POSTMAN' -
    , clb_goal => dbms_service.clb_goal_long);
  ```

- **Short**—Use this connection load balancing method for applications that have short-lived connections. The following example modifies the service known as ORDER, using the DBMS_SERVICE.CLB_GOAL_SHORT package constant to set the goal to short:

  ```sql
  execute dbms_service.modify_service (service_name => 'ORDER' -
    , clb_goal => dbms_service.clb_goal_short);
  ```

When you create a RAC database with the Database Configuration Assistant (DBCA), the DBCA configures and enables server side load balancing by default. The DBCA also creates a sample client side load balancing connection definition in the tnsnames.ora file on the server. Any services created through the DBCA have the default settings of **GOAL=NONE** and **CLB_GOAL=CLB_GOAL_LONG**.

When Oracle Net Services establishes a connection to an instance, the connection remains open until the client closes the connection, the instance is shutdown, or a failure occurs. If you configure transparent application failover (TAF) for the connection, then Oracle moves the session to a surviving instance when an outage occurs.
Fast Application Notification

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 re-execute 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. The TAF setting on a service overrides any TAF setting at the client connection definition.

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 randomly selects an address in the address list, and connects to that node’s Listener. This provides a balancing of client connections across the available Listeners in the cluster. When the Listener receives the connection request, the Listener connects the user to an instance that the Listener knows provides the requested service. To see what services a Listener supports, run the `lsnrctl services` command.

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

Fast Application Notification

This section describes Fast Application Notification (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
- Oracle Clients that are Integrated with Fast Application Notification

Overview of Fast Application Notification

FAN is a notification mechanism that RAC uses to notify other processes about configuration and service level information. RAC also uses FAN to notify applications about service status changes, such as `UP` or `DOWN` events. FAN events enable applications to take immediate action.

For cluster configuration changes, the 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 receive FAN events and can react immediately to them.

For `DOWN` events, the disruption to the application can be minimized because sessions to the failed instance or node can be terminated. Incomplete transactions are terminated and the application user is immediately notified. Application users who request connections are directed to available instances only. You can also use server side callouts to:

- Page DBAs or open support tickets when resources fail to start.
- Automatically start dependent external applications that need to be co-located with a service.
Fast Application Notification

- Change resource plans or 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.

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.

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.

FAN events are published using the Oracle Notification Service (ONS) and an Oracle Streams Advanced Queue. The publication mechanisms are automatically configured as part of the RAC installation.

You can take advantage of FAN events in the following three ways:

1. Your application can take advantage of FAN without any programmatic changes by utilizing an integrated Oracle Client. The integrated clients for FAN events include Oracle Database 10g JDBC (Oracle Database 10g Release 2 is required for load balancing), Oracle Database 10g Release 2 Oracle Data Provider for .NET (ODP.NET), and Oracle Database 10g Release 2 Oracle Call Interface (OCI). This includes applications that use TAF.

2. Applications can take advantage of FAN programmatically by using the ONS Application Programming Interface (API) to subscribe to FAN events and execute event handling actions upon the receipt of an event.

3. You can implement FAN server side callouts on your database tier.

The Connection Manager (CMAN) and Oracle Net Services Listeners also receive FAN event information. This enables the Listener and CMAN to immediately de-register services provided by the failed instance and to avoid erroneously sending connection requests to failed instances. The Listener uses the load balancing advisory when load balancing connection requests where the service has `CLB_GOAL=CLB_GOAL_SHORT`.

The Connection Manager (CMAN) and Oracle Net Services Listeners also receive FAN event information. This enables the Listener and CMAN to immediately de-register services that are provided by the failed instance. It also avoids erroneously sending connection requests to failed instances. When 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 focuses on maintaining service availability. In RAC, Oracle services are designed to be continuously available with loads shared across one or more instances. The RAC high availability framework maintains the service availability by using the Oracle Clusterware and resource profiles.

Resource profiles are automatically created when you define a service. A resource profile describes how the Oracle Clusterware should manage the service and which instance the service should failover to if the instance that the service is running on fails. Resource profiles also define service dependencies on 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.

The RAC high availability framework monitors the database and its services and sends event notifications using FAN. The Oracle Clusterware recovers and balances
services according to business rules. For example, Oracle clients such as the Oracle JDBC driver will redistribute connections in the connection pool when an outage occurs, when a system is restarted, or when additional capacity becomes available.

**Managing Unplanned Outages**

You can assign services in RAC to one or more instances. If RAC detects an outage, then the Oracle Clusterware isolates the failed component and recovers the dependent components. For services, if the failed component is an instance, then the Oracle Clusterware relocates the service to an available instance in the cluster. FAN events can occur at various levels within the Oracle Server architecture and the response can include notifying external entities by way of ONS, Advanced Queuing, or the use of FAN callouts.

---

**Note:** Oracle does not execute RAC callouts with guaranteed ordering. Callouts are executed 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 a 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 Automatic Storage Management (ASM) processes, not just the database. You can use FAN callouts to report faults to your fault management system or to initiate repair jobs.

**Managing Planned Outages**

For repairs, upgrades, and changes that require you to isolate one or more instances, 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.

Remember that due to dependencies, if you manually shutdown your database, then all of your services will be automatically stopped. If you then manually restart the database, then you must also restart the services. Use FAN callouts to automate starting the services when the database starts.

**Fast Application Notification High Availability Events**

Table 6–1 describes the FAN event record parameters and the event types, followed by name-value pairs. The event type is always the first entry and the timestamp is always the last entry as in the following example:

```
FAN event type: service_member
Properties: version=1.0 service=ERP database=FINPROD instance=FINPROD3 host=node3 status=up
```

---

**Table 6–1 Event Record Parameters and Descriptions**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VERSION</td>
<td>Version of the event record. Used to identify release changes.</td>
</tr>
<tr>
<td>EVENT TYPE</td>
<td>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.</td>
</tr>
</tbody>
</table>
Fast Application Notification

Introduction to Workload Management

A FAN record matches the database signature of each session as shown in Table 6–2. The signature information is also available using `OCI_ATTRIBUTES`. These attributes are available in the OCI Connection Handle. Use this information to take actions on sessions that match the FAN event data.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATABASE UNIQUE NAME</td>
<td>The unique database supporting the service; matches the initialization parameter value for <code>DB_UNIQUE_NAME</code>, which defaults to the value of the initialization parameter <code>DB_NAME</code>.</td>
</tr>
<tr>
<td>INSTANCE</td>
<td>The name of the instance that supports the service; matches the <code>ORACLE_SID</code> value.</td>
</tr>
<tr>
<td>NODE NAME</td>
<td>The name of the node that supports the service or the node that has stopped; matches the node name known to CSS.</td>
</tr>
<tr>
<td>SERVICE</td>
<td>The service name; matches the service in <code>DBA_SERVICES</code>.</td>
</tr>
<tr>
<td>STATUS</td>
<td>Values are <code>UP</code>, <code>DOWN</code>, <code>NOT_RESTARTING</code>, <code>PRECONN_UP</code>, <code>PRECONN_DOWN</code>, and <code>UNKNOWN</code>.</td>
</tr>
<tr>
<td>REASON</td>
<td>Failure, Dependency, User, Autostart, Restart.</td>
</tr>
<tr>
<td>CARDINALITY</td>
<td>The number of service members that are currently active; included in all <code>UP</code> events.</td>
</tr>
<tr>
<td>INCARNATION</td>
<td>For node <code>DOWN</code> events; the new cluster incarnation.</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>The local time zone to use when ordering notification events.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATABASE UNIQUE NAME</td>
<td>The unique database supporting the service; matches the initialization parameter value for <code>DB_UNIQUE_NAME</code>, which defaults to the value of the initialization parameter <code>DB_NAME</code>.</td>
</tr>
<tr>
<td>INSTANCE</td>
<td>The name of the instance that supports the service; matches the <code>ORACLE_SID</code> value.</td>
</tr>
<tr>
<td>NODE NAME</td>
<td>The name of the node that supports the service or the node that has stopped; matches the node name known to CSS.</td>
</tr>
<tr>
<td>SERVICE</td>
<td>The service name; matches the service in <code>DBA_SERVICES</code>.</td>
</tr>
<tr>
<td>STATUS</td>
<td>Values are <code>UP</code>, <code>DOWN</code>, <code>NOT_RESTARTING</code>, <code>PRECONN_UP</code>, <code>PRECONN_DOWN</code>, and <code>UNKNOWN</code>.</td>
</tr>
<tr>
<td>REASON</td>
<td>Failure, Dependency, User, Autostart, Restart.</td>
</tr>
<tr>
<td>CARDINALITY</td>
<td>The number of service members that are currently active; included in all <code>UP</code> events.</td>
</tr>
<tr>
<td>INCARNATION</td>
<td>For node <code>DOWN</code> events; the new cluster incarnation.</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>The local time zone to use when ordering notification events.</td>
</tr>
</tbody>
</table>

Using Fast Application Notification Callouts

FAN callouts are server side executables that RAC executes immediately when high availability events occur. To use FAN callouts, place an executable in the directory `CRS_HOME/racg/usrco` on every node that runs the Oracle Clusterware. If you are using scripts, then set the shell as the first line of the executable.

You can use FAN callouts to automate activities when events occur in a cluster configuration, such as:

- Opening fault tracking tickets
- Sending messages to pagers
- Sending e-mail
- Starting and stopping server side applications
- Maintaining an uptime log by logging each event as it occurs
- Relocating low-priority services when high priority services come online

The following is a callout file example for `CRS_HOME/racg/usrco/callout.sh`:

```bash
#!/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

See Also: Table 6–1 for information about the callout and event details

A FAN record matches the database signature of each session, as shown in Table 6–2. The signature information is also available using OCI_ATTRIBUTES. These attributes are available in the OCI Connection Handle. Use this information to take actions on sessions matching the FAN event data.

Load Balancing Advisory

This section describes the load balancing advisory under the following topics:

- Overview of the Load Balancing Advisory
- Administering the Load Balancing Advisory
- Load Balancing Advisory FAN Events

Overview of the Load Balancing Advisory

Load balancing distributes work across all available RAC database instances. Oracle recommends that applications use persistent connections that span the instances that offer a service. Connections are created infrequently and exist for a long duration. Work comes into the system with high frequency, borrows these connections, and exist for relatively short duration. The load balancing advisory has the task of advising the direction of incoming work to the RAC instances that provide optimal quality of service for that work.

The load balancing advisory directs the work requests to the best RAC instances to serve the work which minimizes the need to relocate the work later. By using the THROUGHPUT or SERVICE_TIME goals, feedback is built in to the system. The work is routed to provide the best service times globally, and routing responds gracefully to changing conditions in the system. In steady state, the system approaches equilibrium with improved throughput across all of the RAC instances.

Standard architectures that can use the load balancing advisory include connection load balancing, TP 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 Listener, JDBC Implicit Connection Cache 10g, and ODP.NET Connection Pool. The Load Balancing Advisory is also open for third party subscription by way of ONS.

Administering the Load Balancing Advisory

You can configure the load balancing advisory by defining service-level goals for each service that you want to balance. This enables the RAC load balancing advisory for that service and FAN load balancing events are published. There are two types of service-level goals for runtime:
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:

```sql
execute dbms_service.modify_service (service_name => 'OE'
, goal => dbms_service.goal_service_time -
, clb_goal => dbms_service.clb_goal_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:

```sql
execute dbms_service.modify_service (service_name => 'sjob' -
, goal => dbms_service.goal_service_time -
, clb_goal => dbms_service.clb_goal_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 views DBA_SERVICES, V$SERVICES, and V$ACTIVE_SERVICES.

See Also: "Administering Services" on page 6-17 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 an Oracle integrated client such as ODP.NET or JDBC. Client applications can subscribe to these events directly by way of the ONS Application Programming interface. Table 6–3 describes the load balancing advisory event parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VERSION</td>
<td>Version of the event record. Used to identify release changes.</td>
</tr>
<tr>
<td>EVENT TYPE</td>
<td>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.</td>
</tr>
<tr>
<td>SERVICE</td>
<td>The service name; matches the service in DBA_SERVICES.</td>
</tr>
<tr>
<td>DATABASE UNIQUE NAME</td>
<td>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.</td>
</tr>
<tr>
<td>INSTANCE</td>
<td>The name of the instance that supports the service; matches the ORACLE_SID value.</td>
</tr>
<tr>
<td>PERCENT</td>
<td>The percentage of work requests to send to this database instance.</td>
</tr>
<tr>
<td>FLAG</td>
<td>Indication of the service quality relative to the service goal. Valid values are GOOD, VIOLATING, NO DATA, and BLOCKED.</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>The local time zone to use when ordering notification events.</td>
</tr>
</tbody>
</table>

Use the following example to monitor the load balancing advisory events:

```sql
set pages 60 space 2 lines 132 num 8 verify off feedback off
column user_data heading "AQ Service Metrics" format A60 wrap
```
Oracle Clients that are Integrated with Fast Application Notification

Oracle has integrated FAN with many of the common client application environments to connect to RAC databases. Therefore, the easiest way to use FAN is to use an integrated Oracle Client.

You can use the OCI Session Pools, CMAN session pools, and JDBC and ODP.NET connection pools. OCI applications with TAF enabled should use FAN high availability events for fast failover. The overall goal is to enable your application to always get a connection to an available instance that provides the best service. Due to the integration with FAN, Oracle integrated clients are more aware of the current status of a RAC cluster to prevent client connections from waiting or trying to connect to an instance that is no longer available.

When instances start, FAN notifies the connection pool so that the connection pool can create connections to the new instance and take advantage of the additional resources provided. The use of connection pools and FAN requires that you have properly configured the load balancing of database connections 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.

Oracle connection pools that are integrated with FAN can:

- Balance connections across all of the RAC instances when a service starts; this is preferable to directing the sessions that are defined for the connection pool to the first 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 detects the NOT RESTARTING state, instead of making the client wait while the service repeatedly attempts to restart.
- Balance work requests at runtime using the load balancing advisory events.

The next sections describe how to configure FAN events for the following client development environments:

- Enabling Oracle Call Interface to Receive FAN High Availability Events
- Enabling Java Database Connectivity to Receive FAN Events
- Enabling ODP.NET to Receive FAN High Availability Events
- Enabling ODP.NET to Receive FAN Load Balancing Advisory Events

Enabling Oracle Call Interface to Receive FAN High Availability Events

Oracle Call Interface (OCI) clients can register to receive notifications about RAC high availability events and respond when events occur. This improves the session failover response time in OCI and also removes terminated connections from connection and session pools. This feature works on OCI applications that use TAF, connection pools, or session pools.
First, you must enable a service for high availability events. If your application is using TAF, then enable the TAF settings for the service. Then configure client applications to enable them to connect to a RAC instance to enable event notification. 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, OCI:

- Terminates affected connections at the client.
- Removes connections from the OCI connection pool and the OCI session pool—the session pool maps each session to a physical connection in the connection pool, and there can be multiple sessions per connection
- Fails over the connection if you have configured TAF.

If TAF is not configured, then the client only receives an error.

---

**Note:** OCI does not manage UP events.

---

Perform the following steps to configure Advanced Queueing notifications to an OCI 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:

   ```sql
   execute dbms_service.modify_service (service_name => 'gl.us.oracle.com' -
   , aq_ha_notifications => true -
   , failover_method => dbms_service.failover_method_basic -
   , failover_type => dbms_service.failover_type_select -
   , failover_retries => 180 -
   , failover_delay => 5 -
   , clb_goal => dbms_service.clb_goal_long);
   
   execute dbms_service.modify_service (service_name => 'your_other_service' -
   , aq_ha_notifications => true -
   , failover_method => dbms_service.failover_method_basic -
   , failover_type => dbms_service.failover_type_select -
   , failover_retries => 180 -
   , failover_delay => 5 -
   , clb_goal => dbms_service.clb_goal_long);
   ```

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

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

3. Client applications must link with the client thread or operating system library.

Clients on which you want to use this functionality that also require notifications, including TAF clients, must:

- Register for notifications
- Optionally register a client EVENT callback

An Advanced Queueing ALERT_QUE is populated automatically. To see the alert information, you can query the views DBA_OUTSTANDING_ALERTS and DBA_ALERT_HISTORY. OCI clients obtain a subscription to this queue if they meet the three criteria specified previously in this section.

---

**See Also:** *Oracle Call Interface Programmer’s Guide* for more information about OCI

### Enabling Java Database Connectivity to Receive FAN Events

Enabling FAN for the Oracle JDBC Implicit Connection Cache enables FAN high availability events in Oracle Database 10g release 1, and the load balancing advisory in Oracle Database 10g release 2. Your application can use the JDBC development environment for either thick or thin JDBC clients to take advantage of FAN. For FAN, your application must use the JDBC implicit connection cache.
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`.

**See Also:** The *Oracle Database JDBC Developer’s Guide and Reference* for information about configuring the JDBC implicit connection cache and ONS

### Using FAN with Thin JDBC and Thick JDBC Clients

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. These events can be both high availability events and load balancing advisory events. When using thin or thick JDBC clients, you can use FAN by configuring the JDBC 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 Oracle implicit JDBC connect cache:

```java
OracleDataSource ods = new OracleDataSource()
...
ods.setUser("Scott");
ods.setPassword("tiger");
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=VIP1)(PORT=1521))
{ADDRESS=(PROTOCOL=TCP)(HOST=VIP2)(PORT=1521))
{CONNECT_DATA=(SERVICE_NAME=service_name)})");
```

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

2. Ensure that you have configured ONS on each node that is running the Oracle Clusterware as in the following example using `CRS_HOME/opmn/conf/ons.config`.

```ini
localport=6100 # This is the port ONS is writing to on this node
remoteport=6200 # This is the port ONS is listening on this node
loglevel=3
useocr=on
```

The information in the Oracle Cluster Registry (OCR) for ONS daemons is automatically configured by the DBCA. If you are upgrading from a previous version of the database, then manually configure the OCR as follows. To add the
middle tier nodes or to update the RAC nodes, use racgons from the Oracle Clusterware bin directory:

- To add the ONS daemon configuration:
  ```
  racgons add_config hostname:port [hostname:port] ...
  ```

- To remove the ONS daemon configuration:
  ```
  racgons remove_config hostname[:port] [hostname:port]..
  ```

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

- Support for an All Java mid-tier software.
- No ONS daemon needed on the client machine. No need to manage this process.
- Simple configuration via DataSource property.

When using remote ONS subscription for FCF, an application invokes setONSConfiguration, using the string remoteONSConfig, on an Oracle DataSource instance as in the following example:
```java
ods.setONSConfiguration("nodes=racnode1:4200,racnode2:4200");
```

See Also: Oracle Database JDBC Developer’s Guide and Reference for more information about JDBC

### Enabling ODP.NET to Receive FAN High Availability Events

Oracle Data Provider for .NET (ODP.NET) connection pools can subscribe to notifications that indicate when nodes, services, and service members are down. After a DOWN event, Oracle 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 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 FCF. Perform the following steps to enable FAN:

1. Enable Advanced Queuing notifications by using the DBMS.SERVICE package as in the following example:
   ```
   execute dbms_service.modify_service (service_name => 'gl.us.oracle.com' -
   , aq_ha_notifications => true -
   , failover_method => dbms_service.failover_method_basic -
   , failover_type => dbms_service.failover_type_select -
   , failover_retries => 180 -
   , failover_delay => 5 -
   , clb_goal => dbms_service.clb_goal_long);
   ```

2. Enable FCF for ODP.NET connection pools by subscribing to FAN high availability events. Do this by setting the ha_events connection string attribute to true. You can do this at connection time. Note that this only works if you are using connection pools. In other words, if you have set the pooling attribute to true, which is the default. The following example describes this in detail:
   ```
   // C#
   ```

See Also: Oracle Database JDBC Developer’s Guide and Reference for more information about JDBC
```csharp
using System;
using Oracle.DataAccess.Client;

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

        // Open a connection using ConnectionString attributes
        // Also, enable "load balancing"
        con.ConnectionString =
            "User Id=scott;Password=tiger;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 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 the `DBMS.SERVICE` package as in the following example:
   ```sql
   execute dbms_service.modify_service (service_name => 'gl.us.oracle.com' -
       , aq_ha_notifications => true -
       , failover_method => dbms_service.failover_method_basic -
       , failover_type => dbms_service.failover_type_select -
       , failover_retries => 180 -
       , failover_delay => 5 -
       , clb_goal => dbms_service.clb_goal_long);
   ```

2. Set the `GOAL` and `CLB_GOAL` for the service and ensure that Oracle Net Service is configured for connection load balancing.

3. To take advantage of load balancing events with ODP.NET connection pools, set the load balancing string 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, as in the following example:
   ```csharp
   // C#
   using System;
   using Oracle.DataAccess.Client;

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

           // Open a connection using ConnectionString attributes
   ```
Services and Distributed Transaction Processing in RAC

All tightly coupled branches of a distributed transaction running on a RAC database must run on the same instance. Between transactions and between services, transactions can be load balanced across all of the database instances. This is important to remember when designing your application tier for a distributed transaction processing application that runs against a RAC database. To provide performance and load balancing for your application, 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 a RAC database must ensure that all tightly-coupled branches of a global distributed transaction run on only one 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).

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 a RAC database. All global distributed transactions performed through the DTP service are ensured to have their tightly-coupled branches running on a single RAC instance. This preserves the integrity for distributed transactions and has the following benefits:

- The changes are available locally within one RAC instance when tightly coupled branches need to see changes made by each other
- Relocation and failover of services are fully supported for DTP
- By using more DTP services than there are RAC instances, Oracle can balance the load by services across all of the RAC database instances

See Also: Oracle Data Provider for .NET Developer’s Guide for more information about ODP.NET and the Oracle Database PL/SQL Packages and Types Reference for more information about the DBMS_SERVICES PL/SQL package

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.
To leverage all of the instances in a cluster, create one or more DTP services for each 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 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 RAC.

For current and future client implementations, such as those for JDBC, you do not need the invocation to the `SYS.DBMS_SYSTEM.DIST_TXN_SYNC` procedure because the `OPS_FAILOVER` flag is deprecated. Instead, the server manages the synchronization of in-doubt transaction information across the RAC instances for transaction recovery. However, for backward compatibility you should retain the invocation to the `SYS.DBMS_SYSTEM.DIST_TXN_SYNC` procedure. If you add or delete nodes from your cluster database, then you may need to identify and relocate services that you are using for DTP transactions to ensure that you maintain performance levels.

Your transaction manager does not have to use the `XA_RECOVER()` call when the transaction manager has enough information about in-flight transactions. The transaction manager uses the `XA_RECOVER()` call only if the transaction manager needs to retrieve a list of prepared transactions from the database, in which case heuristically completed transaction information is also returned from the call.

See Also: *Oracle Database Application Developer’s Guide - Fundamentals* for more information about transaction branch management in RAC

### Enabling Distributed Transaction Processing for Services

For services that you are going to use for distributed transaction processing, create the service using Enterprise Manager, the DBCA, 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`. Enterprise Manager enables you to set this parameter on the Cluster Managed Database Services: Create Service or Modify Service page. You can also use the `DBMS_SERVICE` package to modify the DTP property of the singleton service as follows:

```
execute dbms_service.modify_service(service_name =>'xa_01.service.us.oracle.com', DTP=>TRUE);
```

If, for example, `RAC01` which provides service `XA_01` fails, then the singleton service that it provided fails over to one of the other instances, such as `RAC02` or `RAC03`.

If services have migrated to other instances since the cold-start of the RAC database, then you might need to force the relocation of the service to evenly re-balance the load on all of the available hardware. Use 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 is to achieve optimal utilization of your database infrastructure. You can create and deploy services based on business requirements and Oracle can measure the performance for each service. You can define both the application modules within a service as well as the individual actions for a module and monitor thresholds for these actions. This enables you to manage workloads to deliver capacity on demand.

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

- A unique global name to identify the service.
- A Net Service name that a client uses to connect to the service.
- The preferred instances where you want the service to be enabled in a normal running environment.
- The available instances where the service can be enabled if one of the preferred instances 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 will be used for distributed transactions.
- An indicator that determines whether RAC high availability events are sent to OCI and ODP.NET clients that have registered to receive them through Advanced Queuing.

See Also: "Enabling Oracle Call Interface to Receive FAN High Availability Events" on page 6-10 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 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 will be use to balance connections otherwise CPU utilization is used.

In addition to creating services, you may need to:
- 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 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 need to check the status on an instance or service basis. For example, you may need 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 shutdown your database, for example, by running the Server Control (SRVCTL) command `srvctl stop database -d database_name` where `database_name` is the name of the database that you want to stop, then Oracle stops all services to that database. You must manually re-start the services when you start the database.

- Map a Service to a Consumer Group—Oracle 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, the 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 need to 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 rebalance workloads, for example, after adding or deleting cluster nodes.

Notes:

- When you use services, do not set a value for the `SERVICE_NAMES` parameter; Oracle 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 provides when you set `SERVICE_NAMES`. In addition, setting a value for this parameter may override some of the benefits of using services.

- You must still use `INSTANCE_GROUPS` and `PARALLEL_INSTANCE_GROUPS` to restrict parallel execution processing to a subset of instances in a RAC database.

- Using service names to access a queue provides location transparency for the queue within a RAC database.
Administering Services with Enterprise Manager, DBCA, PL/SQL, and SRVCTL

You can create and administer services with Enterprise Manager, DBCA, and the DBMS_SERVICE PL/SQL package, and you can perform most service administration tasks with the SRVCTL utility. The following sections describe how to perform service-related tasks using each of these tools:

- Administering Services with Enterprise Manager
- Administering Services with the Database Configuration Assistant
- Administering Services with the PL/SQL DBMS_SERVICE Package
- Administering Services with SRVCTL

Administering Services with Enterprise Manager

The Cluster Managed Database Services page is the master page for initiating all tasks related to services. To access this page, go to the Cluster Database Maintenance page, then click Cluster Managed Database Services in the Services section. You can use this page and drilldowns 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 Concepts for more information about administering services with Enterprise Manager

Service-Related Tasks that You Can Perform with Enterprise Manager

You can perform service-related tasks as described for the following 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.
- Access the Services Detail page to perform instance-level tasks for a service.
Test the connection for 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.
- 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 rebalance the services load.

**Create Services Page**
The Create Services page enables you to:

- Create the service with name, high availability and performance attributes.
- Select the desired service policy for each instance configured for the cluster database.
- Select the desired service properties, such as the TAF policy, notification properties, load balancing goals, alert levels, and resource management properties.

**Accessing the Enterprise Manager Services Pages**
To access the Cluster Managed Database Services page and detail pages for service instances:

1. From the Cluster Database Home page, click the Maintenance tab.
2. From the Cluster Database Maintenance page, under the Services heading in the High Availability options list, 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 and 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, refer to 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 on 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, refer to the online help for this page.

To access the Relocate page:

1. Perform steps 1 - 3 above.
2. From the Cluster Managed Database Services page, either click on 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.

3. 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, refer to the online help for this page.

**Administering Services with the Database Configuration Assistant**

You can administer services in RAC with the DBCA as described under the following topics:

- Using the Database Configuration Assistant to Add and Modify Services
- Using the Database Configuration Assistant to Administer Services

**Using the Database Configuration Assistant to Add and Modify Services**

You can use the DBCA to create services and to perform administrative tasks on them. If you use the Database Upgrade Assistant (DBUA) to upgrade a Primary/Secondary configuration from an earlier release of the Oracle database, then the DBUA creates one service and assigns it to one instance as a preferred instance and to the other instance as an available instance.

*See Also:* *Oracle Database PL/SQL Packages and Types Reference* for more information about the `DBMS_SERVICE` PL/SQL and `DBMS_MONITOR` packages and for more information about setting thresholds

**Using the Database Configuration Assistant to Administer Services**

The DBCA Services Management feature enables you to manage service assignments and service preferences for instances and to configure TAF policies. You can perform these procedures while your RAC database is running. Even if your instances or the RAC database is not running, you can still use the DBCA to configure services, but the services will not start automatically. To add, modify, or delete services using the DBCA Services Management feature:

1. On the DBCA Welcome page, select the Oracle Real Application Clusters option and click **Next**.
2. On the DBCA Operations page, select Services Management and click **Next**.
3. On the DBCA List of Databases page, select the cluster database for which you want to configure services and click **Next**. If the database that you selected already has services assigned to it, then the DBCA displays these on this page.
4. Click **Add**, **Modify**, or **Delete**.
5. To add a service, enter the name of the service. Note that service names with the prefix `SYS$` are reserved for use by Oracle internal processes. To modify a service, edit the service and configure the service's instance preferences and TAF policies. Assign the service to instances for preferred (normal) and available (recovery) processing. The DBCA records your changes when you select another service or proceed to another page. To delete a service, select the service and click **Delete**.

---

*See Also:* *Oracle Database PL/SQL Packages and Types Reference* for more information about the `DBMS_SERVICE` PL/SQL and `DBMS_MONITOR` packages and for more information about setting thresholds
6. To remove a service, select the service and click **Remove**.

7. Click **Finish** and the DBCA displays the Summary page. Click **OK** and the DBCA displays a progress dialog while it configures your services.

When you click **Finish**, the DBCA configures the Oracle Clusterware resources for the services that you added, modified, or removed. The DBCA also configures the net service entries for these services and starts them. When you use the DBCA to remove services, the DBCA stops the service, removes the Oracle Clusterware resource for the service, and removes the net service entries.

**Administering Services with the PL/SQL DBMS_SERVICE Package**

You can manage services with the PL/SQL `DBMS_SERVICE` package procedures. These procedures are described in the following sections:

- `CREATE_SERVICE`
- `MODIFY_SERVICE`
- `DELETE_SERVICE`
- `START_SERVICE`
- `STOP_SERVICE`
- `DISCONNECT_SESSION`

**CREATE_SERVICE**

The `CREATE_SERVICE` procedure adds a new service to the RAC database. In the `CREATE_SERVICE` syntax, `service_name` is the unique, global service name, `network_name` is the TNS name for connections to the service, `goal` sets the workload management goal directive to service quality or throughput, `dtp` is the distributed transaction flag, `aq_ha_notification` is the flag to send RAC high availability events to registered OCI clients, `failover_method` is the TAF failover method for the service, `failover_type` is the TAF failover method for the service, `failover_retries` is the TAF connection retry count, `failover_delay` is the wait time between TAF connection retries, and `clb_goal` sets the connection load balancing goal. When using services with RAC, add the high availability properties, such as the `PREFERRED` and `AVAILABLE` placement, using Enterprise Manager, the DBCA or SRVCTL.

**Note:** Oracle recommends that you use Enterprise Manager or the DBCA to create services for RAC environments.

**See Also:** `Oracle Database PL/SQL Packages and Types Reference` for more information about the `CREATE_SERVICE` procedure syntax and for information about other procedures mentioned in this section.
MODIFY_SERVICE
The MODIFY_SERVICE procedure changes one or more service characteristics. In the MODIFY_SERVICE syntax the parameters are the same as those for the CREATE_SERVICE procedure.

DELETE_SERVICE
The DELETE_SERVICE procedure drops an existing service. In the DELETE_SERVICE syntax service_name is the name of the service to be dropped.

START_SERVICE
The START_SERVICE procedure starts a service on the connected instance, the named instance, or all instances. In the START_SERVICE syntax SERVICE_NAME is the name of the service to be started and INSTANCE_NAME can be NULL (to start the service on the connected instance), the name of an instance (to start the service on that instance), or the package constant DBMS_SERVICE.ALL_INSTANCES to start the service on all instances where it is defined.

STOP_SERVICE
The STOP_SERVICE procedure stops a service on the connected instance, the named instance, or all instances. In the STOP_SERVICE syntax the parameters are the same as those for the START_SERVICE procedure.

DISCONNECT_SESSION
The DISCONNECT_SESSION procedure terminates all sessions on the connected instance, or the instance where the PL/SQL procedure is run. In the DISCONNECT_SESSION syntax service_name is the name of the service for which connections are to be dropped.

Administering Services with SRVCTL
When you create a service with SRVCTL, you must start it with a separate SRVCTL command. However, you may later need to manually stop or restart the service. You may also need 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

See Also: Appendix D, "Server Control Utility Reference" for more information about SRVCTL and the other SRVCTL commands that you can use to manage services.

Creating Services with SRVCTL
To create a service with SRVCTL, enter a command from the command line using the following syntax:

srvtcl add service -d database_unique_name -s service_name -r preferred_list[-a available_list] [-P TAF_policy]
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] [-c connect_str] [-q]
```

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

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 status 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 config service` command from the command line to obtain the high availability configuration of a service. For example, the following command returns the configuration of the `crm` service that is running on the `crm` database:

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

See Also: Appendix D, "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 waits are attributable by application. Tuning by using ‘service and SQL’ replaces tuning by ‘session and SQL’ in the majority of systems where all sessions are anonymous and shared.
The AWR maintains performance statistics that include information about response time, throughput, resource consumption, and wait events for all services and work that the database performs. Oracle also maintains metrics, statistics, wait events, wait classes, and SQL-level traces for services. You can optionally augment these statistics by defining modules within your application to monitor certain statistics. You can also define the actions within those modules that business critical transactions should execute in response to particular statistical values. Enable module and action monitoring using the `DBMS_MONITOR` PL/SQL package 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);
```

To verify that you have enabled monitoring, use the `DBA_ENABLED_AGGREGATIONS` view.

Statistics aggregation and tracing by service are global in scope for RAC databases. In addition, they are persistent across instance restarts and service relocations for both 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$SERVICE_EVENTS`, `V$SERVICE_WAIT_CLASSES`, `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 RAC instances:

```
set pagesize 60 space 2 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 with respect to 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 a 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 also the 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.

Services and Thresholds Alerts Example

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 Email server, the AWR report runs each Monday during the peak usage times of 10am to 2pm. 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. In Oracle Database 10g, you must set these thresholds at all instances within a RAC database. You can schedule actions using 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:

```sql
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:

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

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:

col AGGREGATION_TYPE format a21 trunc heading 'AGGREGATION'
col PRIMARY_ID format a20 trunc heading 'SERVICE'
col QUALIFIER_ID1 format a20 trunc heading 'MODULE'
col QUALIFIER_ID2 format a20 trunc heading 'ACTION'
select * from DBA_ENABLED_AGGREGATIONS ;

The output might appear as follows:

<table>
<thead>
<tr>
<th>AGGREGATION</th>
<th>SERVICE</th>
<th>MODULE</th>
<th>ACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>SERVICE_MODULE_ACTION</td>
<td>ERP</td>
<td>PAYROLL</td>
<td>EXCEPTIONS PAY</td>
</tr>
<tr>
<td>SERVICE_MODULE_ACTION</td>
<td>ERP</td>
<td>PAYROLL</td>
<td></td>
</tr>
<tr>
<td>SERVICE_MODULE_ACTION</td>
<td>HOT_BATCH</td>
<td>POSTING</td>
<td></td>
</tr>
</tbody>
</table>
This chapter explains how to configure Recovery Manager (RMAN) for use in Oracle Real Application Clusters (RAC) environments. This chapter also provides procedures for using RMAN for archiving in RAC environments and discusses redo logs and archived redo logs considerations. The topics in this chapter include:

- Overview of Configuring RMAN for Real Application Clusters
- Configuring the RMAN Snapshot Control File Location
- Configuring the RMAN Control File and SPFILE Autobackup Feature
- Managing Archived Redo Logs Using RMAN in Real Application Clusters
- Archived Redo Log File Conventions in RAC
- RMAN Archiving Configuration Scenarios
- Changing the Archiving Mode in Real Application Clusters

Overview of Configuring RMAN for Real Application Clusters

Recovery Manager (RMAN) enables you to back up, restore, and recover datafiles, control files, server parameter files (SPFILEs) and archived redo logs. It is included with the Oracle server 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.

Configuring the RMAN Snapshot Control File Location

The snapshot control file is a temporary snapshot control file that RMAN creates to re-synchronize from a read-consistent version of the control file. RMAN only needs a snapshot control file when re-synchronizing with the recovery catalog or when making a backup of the current control file. In 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 a 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 UNIX-based 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 datafiles in RAC. Refer to Oracle Database Backup and Recovery Reference for more information about configuring the snapshot control file.

### Configuring the RMAN Control File and SPFILE Autobackup Feature

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 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. Refer to the Oracle Database Backup and Recovery Advanced User’s Guide for more information about using the control file autobackup feature.

Besides using the RMAN control file, you can also use Enterprise Manager to use the RMAN features.

---

**Note:** The Enterprise Manager interface that you use to configure RMAN backup and recovery is documented in Oracle Database Backup and Recovery Basics. This chapter, Chapter 7, only describes topics for this subject if they are different from the single-instance description.

### Configuring Channels for RMAN in Real Application Clusters

This section describes how to configure channels for RMAN. You can configure channels to use automatic workload balancing or you can specify specific channels for specific instances as described in the following topics:

- Configuring Channels to use Automatic Workload Balancing
Configuring Channels to Use Automatic Workload Balancing

To configure channels to use automatic load balancing, use the following syntax:

```sql
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 channels to use a specific channel, use the following syntax:

```sql
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 Real Application Clusters

When a node generates an archived redo log, Oracle 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 re-synchronization 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 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 non-cluster 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). Refer to 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 7–1 Example Configuration for the initialization parameters file**

```sql
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 can successfully recover the archived log data.

### Archived Redo Log File Conventions in 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>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>%r</td>
<td>Resetlogs identifier</td>
<td>log_1_62_23452345</td>
</tr>
<tr>
<td>%R</td>
<td>Padded resetlogs identifier</td>
<td>log_1_62_0023452345</td>
</tr>
<tr>
<td>%s</td>
<td>Log sequence number, not padded</td>
<td>log_251</td>
</tr>
<tr>
<td>%S</td>
<td>Log sequence number, left-zero-padded</td>
<td>log_0000000251</td>
</tr>
<tr>
<td>%t</td>
<td>Thread number, not padded</td>
<td>log_1</td>
</tr>
<tr>
<td>%T</td>
<td>Thread number, left-zero-padded</td>
<td>log_0001</td>
</tr>
</tbody>
</table>

All of the thread parameters, in either upper or lower case, are mandatory for RAC. This enables Oracle 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:
This section describes the archiving scenarios for a RAC database. The two configuration scenarios in this chapter describe a three-node UNIX cluster for a RAC database. For both scenarios, the \texttt{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.

**Automatic Storage Management and Cluster File System Archiving Scheme**

The preferred configuration for RAC is to use Automatic Storage Management (ASM) for a recovery area with a different disk group for your recovery set than for your datafiles. 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 \texttt{/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 CFS example. 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.
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_ARCHIVE_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.

Non-Cluster File System Local Archiving Scheme

In the non-cluster 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 UNIX-based systems or mapped drives on Windows-based 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 Non-Cluster File System Local Archiving

If you use non-cluster 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 8, "Managing Backup and Recovery".

Initialization Parameter Settings for Non-Cluster 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 Non-Cluster File System Local Archiving

As illustrated in Table 7–2, 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 7–2 UNIX/NFS Location Log Examples, Non-Cluster 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) |
Changing the Archiving Mode in Real Application Clusters

You can run the `ALTER DATABASE` SQL statement to change the archiving mode in 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.

### Monitoring the Archiver Processes

After your RMAN configuration is operative in your 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. Refer to *Oracle Database Reference* for more information about the database views.
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 (RAC) databases. This chapter also describes RAC instance recovery, parallel backup, recovery with SQL*Plus, and using the Flash Recovery Area in RAC. The topics in this chapter include:

- RMAN Backup Scenarios for Real Application Clusters
- RMAN Restore Scenarios for Real Application Clusters
- RMAN Recovery Through Resetlogs in Real Application Clusters
- RMAN and Oracle Net in Real Application Clusters
- Instance Recovery in Real Application Clusters
- Media Recovery in Real Application Clusters
- Parallel Recovery in Real Application Clusters
- Using a Flash Recovery Area in Real Application Clusters

**Note:** For restore and recovery in 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 datafiles. In RAC, datafiles are accessible from every node in the cluster, so any node can restore archived log files.

**RMAN Backup Scenarios for Real Application Clusters**

This section describes the options for backup scenarios.

**Non-Cluster File System Backup Scheme**

In a non-cluster file system environment, each node can back up only its own local archived redo logs. For example, node 1 cannot access the archived redo logs 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 backup its archived logs to a local directory.

**RMAN Restore Scenarios for Real Application Clusters**

This section describes the following common RMAN restore scenarios:

- Cluster File System Restore Scheme
RMAN Restore Scenarios for Real Application Clusters

- Non-Cluster File System Restore Scheme
- Using RMAN or 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 "Automatic Storage Management and Cluster File System Archiving Scheme" on page 7-5. 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.

Non-Cluster File System Restore Scheme

The scheme that this section describes assumes that you are using the "Non-Cluster File System Local Archiving Scheme" on page 7-6. 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 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. This procedure is described in Oracle Database Backup and Recovery Basics.

You can also use 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.
RMAN Recovery Through Resetlogs in Real Application Clusters

The `resetlogs` operation automatically archives online logs. This ensures that your database has the necessary archived redo logs if recovery was done to a point in time in the online or in the standby logs. You do not need to perform a full backup after a `resetlogs` operation.

The default archivelog format includes a resetlogs identifier. You do not need to change the backup scripts after performing a reset logs operation. RMAN backs up earlier incarnation logs on running `BACKUP ARCHIVELOG ALL` or `BACKUP ARCHIVELOG FROM TIME` or `BACKUP ARCHIVELOG FROM SCN` command.

RMAN and Oracle Net in Real Application Clusters

To facilitate RMAN in Oracle Net, RMAN only needs a net service name that creates a dedicated server connection. You can configure this net service name on any active instance within your RAC environment.

For any RMAN connection made through a net service name, each net service name must specify only one instance. This applies to all RMAN connections, whether you configure these connections from the command line or through the `CONNECT` clause in `ALLOCATE CHANNEL` or `CONFIGURE CHANNEL` RMAN commands. Additionally, when you use RMAN from the command line, you can only connect to one instance in a RAC database at a time. For example, assume that node1, node2, and node3 are net service names for three instances in a RAC environment. In this case, connect to the target database with only one of these net service names, for example:

% `rman TARGET SYS/oracle@node2 CATALOG rman/cat@catdb`

Instance Recovery in Real Application Clusters

Instance failure occurs when software or hardware problems disable an instance. After instance failure, Oracle automatically uses the online redo logs to perform recovery as described in this section.

Single Node Failure in Real Application Clusters

Instance recovery in 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 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 Real Application Clusters

When multiple node failures occur, as long as one instance survives, RAC performs instance recovery for any other instances that fail. If all instances of a RAC database fail, then Oracle 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 a RAC database. This recovery procedure is the same for Oracle running in shared mode as it is for Oracle running in exclusive mode, except that one instance performs instance recovery for all of the failed instances.

Using RMAN to Create Backups in Real Application Clusters

Oracle provides RMAN for backing up and restoring the database. RMAN enables you to back up, restore, and recover datafiles, control files, SPFILEs, and archived redo logs. RMAN is included with the Oracle 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 Automatic Storage Management (ASM).

The procedures for using RMAN in RAC environments do not differ substantially from those for Oracle single-instance environments. Refer to 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 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, datafiles and archived 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 datafiles 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 datafiles than to other datafiles. RMAN automatically detects this, which is known as node affinity awareness. When deciding which channel to use to back up a particular datafile, RMAN gives preference to the nodes with faster access to the datafiles 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 datafiles 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 8-4, 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 non-cluster 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 Real Application Clusters**

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 datafiles and then recover the database. The procedures for RMAN media recovery in 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 datafiles. 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.

**Parallel Recovery in Real Application Clusters**

Oracle automatically selects the optimum degree of parallelism for instance, crash, and media recovery. Oracle 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 RAC databases as described under the following topics:

- **Parallel Recovery with RMAN**
- **Disabling Parallel Recovery**

**See Also:** *Oracle Database Backup and Recovery Advanced User’s Guide* for more information on these topics

**Parallel Recovery with RMAN**

With RMAN’s `RESTORE` and `RECOVER` commands, Oracle automatically makes parallel the following three stages of recovery:

**Restoring Datafiles** When restoring datafiles, 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 datafiles.

**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 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

To disable parallel instance and crash recovery on a multi-CPU system, 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 to use non-parallel media recovery.

Using a Flash Recovery Area in Real Application Clusters

To use a flash recovery area in RAC, you must place it on an ASM disk group, a Cluster File System, or on a shared directory that is configured through a network file system file for each RAC instance. In other words, the flash recovery area must be shared among all of the instances of a RAC database. In addition, set the parameter DB_RECOVERY_FILE_DEST to the same value on all instances.

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 on the Help for this page for more information.
This chapter describes administrative tasks or options within Oracle tools that are specific to Oracle Real Application Clusters (RAC) and not discussed elsewhere in this book. In some cases, you have a choice of tools to perform a task while other tasks must be performed through a specific tool, such as Enterprise Manager or SRVCTL. In addition, this chapter describes how to quiesce a RAC database and how to administer network interfaces with the Oracle Interface Configuration Tool (OIFCFG). The topics in this chapter are:

- Enterprise Manager Tasks for Real Application Clusters
- Real Application Clusters Administration Procedures for Enterprise Manager
- Additional Information About SQL*Plus in Real Application Clusters
- Quiescing Real Application Clusters Databases
- Administering System and Network Interfaces with OIFCFG
- Changing VIP Addresses

See Also: Oracle Enterprise Manager Concepts and the Enterprise Manager online help for more information about Enterprise Manager

Enterprise Manager Tasks for Real Application Clusters

Within Enterprise Manager, 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 Enterprise Manager to administer storage, the schema, and security at the cluster database level. Or you can perform instance-specific commands such as setting parameters or creating resource plans.

Because there is one Enterprise Manager Agent on each node of a RAC database, for Database Control you can use any URL for that database to administer it with Enterprise Manager. You can manage all of the following RAC components as targets in your Enterprise Manager framework:

- Host cluster—Accessible from the Cluster Database Home Page.
- Cluster database instances—Links to the instance pages appear on the Cluster Database Home Page.
- Hosts and Listeners—Links to hosts and Listeners appear on the Cluster Database Instance Pages.
Using Enterprise Manager Grid Control to Discover Nodes and Instances

Discovering RAC database and instance targets in 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. But for Grid Control, Enterprise Manager console interface can be used to discover RAC database and instance targets. If the Grid Control agents are installed on a cluster that already has RAC database, 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 Enterprise Manager Grid Control as follows:

1. Log in to 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 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 Automatic Storage Management (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 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.

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.

Enterprise Manager Pages for Real Application Clusters

This section describes the following Enterprise Manager pages for RAC:

- Databases Summary Page
- Cluster Database Home Page
- Cluster Database Instances Pages
- The Databases Overview Page for Real Application Clusters
- The Cluster Home Page for Real Application Clusters

Databases Summary Page

This is a top-level page in Enterprise Manager Grid Control. The page shows cluster and single-instance databases. If there are cluster databases in your environment, the Databases Summary page displays "Cluster Database" in the Type column. The page
also indicates cluster database availability as well as the ratio of active instances to inactive instances. Click a cluster database link and Enterprise Manager displays the Cluster Database Home Page for that database, which is described under the following heading.

### Cluster Database Home Page

From the Cluster Database Home Page you can manage the cluster nodes and hosts as well as cluster subcomponents such as instances, Listeners, and interconnects. The Cluster Database Home Page is also a summary page for cluster database management that provides an overview of cluster database activity. Enterprise Manager uses a unique database name to identify the cluster database it represents. You can use the Administration tab on this page to perform many activities such as:

- Create undo tablespaces and redo threads and assign them to specific instances, SPFILE, create a backup
- Start, stop, and relocate database services at the cluster database level

You can use the Maintenance tab on this page to perform operations such as:

- Create backup and recovery scenarios
- Toggle the archive mode on and off
- Administer recovery settings
- Manage resource plans for the database and its instances
  
  You can define and modify the resource plans for the cluster database and also activate and deactivate resource plans for specific instances. You can also use the Resource Plan Schedule to schedule resource plan activation.

You can use the Interconnects tab on this page to perform tasks such as:

- Monitoring the interconnect interfaces
- Determining the load added by individual instances and databases on the interconnect
- Determining configuration issues
- Identifying transfer rate-related issues including excess traffic, and so on

### Cluster Database Instances Pages

Instances’ pages show instance-specific information similar to the information that you would see on a single-instance Oracle database. The RAC-specific contents of Instance Pages are:

- Configuration—You can view instance states, view and edit initialization parameters at the instance level and at the cluster database level, and view resource plan performance statistics. You can also view and modify the undo tablespaces assigned to an instance and the undo tablespace retention period.
- Sessions—You can list the statuses of connected users, view the latest SQL for specific sessions, and terminate sessions.
- Locks—You can view details for currently held User type and System type locks.
The Databases Overview Page for Real Application Clusters

The Databases Overview Page links to Cluster Home Pages and to the node or instance Home Pages.

The Cluster Home Page for Real Application Clusters

The Cluster Home Page displays an overview of activities and detailed reports at both the cluster and instance levels. The Cluster Home Page has the following sections:

- General Section—Provides a cluster status overview.
- Configuration Section—Lists the hardware platform, operating system and version, and Oracle or vendor clusterware version.
- Cluster Databases Table—Displays the cluster databases associated with a cluster, their availability, and any cluster database alerts. You can access the individual Cluster Database Home Pages from the Cluster Databases Table.
- Alerts Table—Provides alert information such as severity rating.
- Hosts Table—Displays information about the hosts or nodes in the cluster.

Real Application Clusters Administration Procedures for Enterprise Manager

The Cluster Database Home page shows all of the instances in the RAC database and provides an aggregate collection of several 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 RAC database, navigate to the Cluster Database Home page, and click the Administration tab.

Administering Enterprise Manager Jobs in Real Application Clusters

You can administer Enterprise Manager jobs at both the database and instance levels. For example, you can create a job at the cluster database level and the job will run on any active instance of the target RAC database. Or you can create a job at the instance level and the job will only run on the specific instance for which you created it. In the event of a failure, recurring jobs can run on a surviving instance.

Creating Enterprise Manager Jobs in Real Application Clusters

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. Enterprise Manager also displays job activity in several categories, namely, 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 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 Enterprise Manager displays the Database Home page, click **Maintenance**.
4. Complete the Enterprise Manage Job Wizard panels to create the job.

**Administering Alerts in Real Application Clusters with Enterprise Manager**

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

Enterprise Manager distinguishes between database- and instance-level alerts in RAC environments. Alert thresholds for instance level alerts, such as archive log alerts, can be set at the instance target level. This 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. This enables you to avoid receiving duplicate alerts at each instance.

**See Also:** OTN for an example of configuring alerts in 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 Enterprise Manager**

You can define blackouts for all managed targets of a 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.

**Additional Information About SQL*Plus in Real Application Clusters**

The following sections describe the use of SQL*Plus in RAC environments:

- **How SQL*Plus Commands Affect Instances**
- **Verifying that Instances are Running**

**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 RAC database. You do not need to run SQL*Plus commands as root on UNIX-based systems or as Administrator on Windows-based 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 9–1* describes how SQL*Plus commands affect instances.
Verifying that Instances are Running

To verify that instances are running, on any node from a SQL*Plus prompt enter:

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

Oracle 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 9–2.

Table 9–2 Descriptions of V$ACTIVE_INSTANCES Columns

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INST_NUMBER</td>
<td>Identifies the instance number.</td>
</tr>
<tr>
<td>INST_NAME</td>
<td>Identifies the host name and instance name.</td>
</tr>
</tbody>
</table>

Quiescing Real Application Clusters Databases

The procedure for quiescing 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. Once 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 a 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 will be queued until after the system is unquiesced.
These conditions apply to RAC:

- If you issued the `ALTER SYSTEM QUIESCE RESTRICTED` statement but Oracle has not finished processing it, you cannot open the database.
- You cannot open the database if it is already in a quiesced state.
- The `ALTER SYSTEM QUIESCE RESTRICTED` and `ALTER SYSTEM UNQUIESCE` statements affect all instances in a RAC environment, not just the instance that issues the command.

Quiesced State and Cold Backups

You cannot use the quiesced state to take a cold backup. This is because Oracle background processes may still perform updates for Oracle internal purposes even while the database is in quiesced state. In addition, the file headers of online datafiles 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. Refer to the Oracle Database Administrator’s Guide for details on the quiesce database feature and the Oracle Database Reference for more information about the `ALTER SYSTEM QUIESCE RESTRICTED` syntax.

Administrating System and Network Interfaces with OIFCFG

This section describes the following Oracle Interface Configuration (OIFCFG) topics:

- Defining Network Interfaces with OIFCFG
- Syntax and Commands for the OIFCFG Command-Line Tool

Use the OIFCFG command-line tool in single-instance Oracle databases and in RAC database environments to:

- Allocate and de-allocate network interfaces to components
- Direct components to use specific network interfaces
- Retrieve component configuration information

The Oracle Universal Installer (OUI) also uses OIFCFG to identify and display the interfaces available on the system.

Defining Network Interfaces with OIFCFG

The specification for a network interface uniquely identifies it using the interface name, its associated subnet, and interface type. The interface type indicates the purpose for which the network is configured. The supported interface types are:

- Public—An interface that can be used for communication with components external to RAC instances, such as Oracle Net and Virtual Internet Protocol (VIP) addresses
- Cluster interconnect—A private interface used for the cluster interconnect to provide inter-instance or Cache Fusion communication

A network interface can be stored as a global interface or as a node-specific interface. An interface is stored as a global interface when all of the nodes of a RAC cluster have the same interface connected to the same subnet (recommended). It is stored as a node-specific interface only when there are some nodes in the cluster that have a different set of interfaces and subnets. If an interface is configured as both a global and
a node-specific interface, the node-specific definition takes precedence over the global
definition.

A network interface specification is in the form of:

interface_name/subnet:interface_type.

For example, the following identifies qfe0 as a cluster interconnect located at the
address 204.152.65.32:

qfe0/204.152.65.32:cluster_interconnect

Syntax and Commands for the OIFCFG Command-Line Tool

Use the oifcfg -help command to display online help for OIFCFG. The elements of
OIFCFG commands, some of which are optional depending on the command, are:

- nodename—Name of the Oracle Clusterware node as listed in the output from the
  olsnodes command
- if_name—Name by which the interface is configured in the system
- subnet—Subnet address of the interface
- if_type—Type of interface: public or cluster_interconnect

You can use OIFCFG to list the interface names and the subnets of all of the interfaces
available on the local node by executing the iflist keyword as shown in this
example:

```
oifcfg iflist
hme0     139.185.141.0
qfe0     204.152.65.16
```

You can also retrieve specific OIFCFG information with a getif command using the
following syntax:

```
oifcfg getif [-global | -node nodename] [-if if_name[/subnet]] [-type if_type]
```

To store a new interface use the setif keyword. For example, to store the interface
hme0, with the subnet 139.185.141.0, as a global interface (to be used as an interconnect
for all of the RAC instances in your cluster), you would use the command:

```
oifcfg setif -global hme0/139.185.141.0:cluster_interconnect
```

For a cluster interconnect that exists between only two nodes, for example rac1 and
rac2, you could create the cms0 interface with the following commands, assuming
139.185.142.1 and 139.185.142.2 are the subnet addresses for the interconnect on rac1
and rac2 respectively:

```
oifcfg setif -node rac1 cms0/139.185.142.1:cluster_interconnect
oifcfg setif -node rac2 cms0/139.185.142.2:cluster_interconnect
```

Use the OIFCFG delif command to delete the stored configuration for global or
node-specific interfaces. A specific node-specific or global interface can be deleted by
supplying the interface name, with an optional subnet, on the command line. Without
the -node or -global options, the delif keyword deletes either the given interface
or all of the global and node-specific interfaces on all of the nodes in the cluster. For
example, the following command deletes the global interface named qfe0 for the
subnet 204.152.65.16:

```
oifcfg delif -global qfe0/204.152.65.16
```
On the other hand, the next command deletes all of the global interfaces stored with OIFCFG:

```
oifcfg delif -global
```

### Changing VIP Addresses

Use the following procedure to change a VIP address:

1. Stop all database and ASM instances.
2. Stop the Listeners, and node applications using the `srvctl stop nodeapps` command.
3. Run the following command to verify node connectivity between all of the nodes for which your cluster is configured. This command discovers all of the network interfaces available on the cluster nodes and verifies the connectivity between all of the nodes by way of the discovered interfaces. This command also lists all of the interfaces available on the nodes which are suitable for use as VIPs.

   ```
   cluvfy comp nodecon -n all [-verbose]
   ```

   **See Also:** Appendix A, "Troubleshooting" for more information about enabling and using the Cluster Verification Utility (CVU)

4. Run the `srvctl modify nodeapps` command with the `-A` option as described in Appendix D. Use the `crs_stat` command to identify all active node applications.
5. Restart all of the instances and node applications that you stopped in Step 1 and 2.
This chapter describes how to add and delete nodes and instances in Oracle Real Application Clusters (RAC) databases. One method to extend RAC databases is to use the Oracle cloning procedures as described in the *Oracle Universal Installer and OPatch User’s Guide*. Cloning enables you to copy images of the Oracle Clusterware and RAC software onto the other nodes that have identical hardware and software. You can also use Enterprise Manager Grid Control node and instance addition feature that automatically performs the cloning. You can also use manual procedures to add nodes and instances. The topics in this chapter are:

- **Cloning Oracle Clusterware and RAC Software in Grid Environments**
- **Adding and Deleting Instances Using Enterprise Manager**
- **Quick-Start Procedures for Adding and Deleting Nodes and Instances on UNIX-Based Systems**
- **Detailed Procedures for Adding and Deleting Nodes and Instances on UNIX-Based Systems**

**Note:** For all of the add node and delete node procedures any temporary directories, such as `/tmp`, `$TEMP`, or `$TMP`, should not be shared directories. If your temporary directories are shared, then set your temporary environment variable, such as `$TEMP`, to a non-shared location on a local node. In addition, use a directory that exists on all of the nodes.

**Cloning Oracle Clusterware and RAC Software in Grid Environments**

This section summarizes the procedure for deploying RAC in grid environments with large numbers of nodes using cloned images for the Oracle Clusterware and Oracle Real Application Clusters. To perform the cloning procedures that are summarized in this section, refer to the *Oracle Universal Installer and OPatch User’s Guide*.

The cloning process assumes that you successfully installed the Oracle Clusterware on at least one node; all root scripts must have run successfully on the original nodes from which you are extending your cluster database. To use Oracle cloning, first clone the Oracle Clusterware home and then clone the Oracle home with the RAC software.

To clone the Oracle Clusterware home, create a tar file of the Oracle Clusterware home and copy the file to the new node’s Oracle Clusterware home. After uncompressing the tar file, run the Oracle Universal Installer (OUI) in clone mode as described in the
Adding and Deleting Instances Using Enterprise Manager

Oracle Universal Installer and OPatch User’s Guide. Then run the installation scripts and repeat these steps on each node that you are adding.

The process for cloning the Oracle home onto new nodes is similar to cloning the Oracle Clusterware home. You must create a tar file and copy the file to the new nodes. Then create the required users and groups on the new nodes using the same information as you used for groups and users on the existing nodes. Uncompress the tar file and run the OUI again in clone mode, and then run the installation scripts. Repeat these steps for each node that you add. In addition, you must run the Oracle Net Configuration Assistant (NETCA) on each node to create the Listener.

If you have not already created a database, then you can run the Database Configuration Assistant (DBCA) to create the database. Finally, follow the post-cloning procedures to complete the extension of your RAC environment onto new nodes.

See Also: Oracle Universal Installer and OPatch User’s Guide for details about the Oracle cloning procedures

Adding and Deleting Instances Using Enterprise Manager

To use Enterprise Manager Database Control to add instances to a RAC database, you must have previously performed one of the following procedures to add the Oracle Clusterware and RAC software onto the nodes on which you want to add RAC database instances:

- The cloning procedures using the OUI as described under the heading “Cloning Oracle Clusterware and RAC Software in Grid Environments” on page 10-1
- The quick start node addition procedures described under the heading "Quick-Start Procedures for Adding and Deleting Nodes and Instances on UNIX-Based Systems” on page 10-3
- The detailed node addition procedures as described under the heading "Detailed Procedures for Adding and Deleting Nodes and Instances on UNIX-Based Systems” on page 10-9

In Enterprise Manager, using Database Control, from the Cluster Database page, select the Administration tab and under Change Database, select Add Instance or Delete Instance. Enterprise Manager displays either the Add Instance: Cluster Credentials or Delete Instance: Cluster Credentials Page.

Enter your operating system cluster credentials for the cluster database on which you are performing this operation. On the host selection page, select the host on which you are performing this operation. The page displays the currently configured instances for each host in that environment. Click Next and review the summary of the operation and click Submit Job to commit the change.

To use the Enterprise Manager Delete Instance page, provide the credentials and click Next. The Enterprise Manager Delete Instance page is instance-centric, in that the page lists the instances that you want can delete. Select an instance to delete from the list of instances and click Next. Review the summary page and click Submit Job.

Note: Oracle recommends that you back up your voting disk and OCR files after you complete the node deletion process.
Quick-Start Procedures for Adding and Deleting Nodes and Instances on UNIX-Based Systems

This section explains node and instance addition and deletion for UNIX-based systems using procedures that are presented in a quick-start format. For node addition, these procedures require that you install the required operating system patches and updates on the new nodes. Then configure the new nodes to be part of your cluster at the network level. Extend the Oracle Clusterware home from an existing Oracle Clusterware home to the new nodes and then extend the Oracle database software with RAC components to the new nodes. Finally, make the new nodes members of the existing cluster database. This section includes the following topics:

- Adding an Oracle Clusterware Home to a New Node
- Adding an Oracle Home with RAC on UNIX-Based Systems
- Deleting an Oracle Home with RAC from a Node
- Deleting an Oracle Clusterware Home from a Node

Note: You can add nodes on some UNIX-based systems without stopping the existing nodes if your clusterware supports this. Refer to your vendor-specific clusterware documentation for more information.

See Also: Your Oracle Real Application Clusters platform-specific installation and configuration guide for procedures about using the Database Configuration Assistant (DBCA) to create and delete RAC databases

Note: When you extend a RAC database, you must first extend the Oracle Clusterware home to the new nodes and then extend the Oracle home. In other words, you extend the software onto the new nodes in the same order as you installed the clusterware and Oracle database software components.

The following sections are based on the addition of a node named node2.

Adding an Oracle Clusterware Home to a New Node

You can use either of the following procedures to add an Oracle Clusterware home to a new node:

- Adding an Oracle Clusterware Home to a New Node Using the Oracle Universal Installer
- Adding an Oracle Clusterware Home to a New Node Using the OUI Silent Mode

1. Ensure that you have successfully installed the Oracle Clusterware on at least one node in your cluster environment. To use these procedures as shown, your $CRS_HOME environment variable must identify your successfully installed Oracle Clusterware home.
Quick-Start Procedures for Adding and Deleting Nodes and Instances on UNIX-Based Systems

1. Go to $CRS_HOME/oui/bin and run the addNode.sh script.

2. The Oracle Universal Installer (OUI) displays the Node Selection Page on which you should select the node or nodes that you want to add and click Next.

3. Verify the entries that the OUI displays on the Summary Page and click Next.

4. Run the rootaddnode.sh script from the $CRS_HOME/install/ directory on the node from which you are performing this procedure when the OUI dialog prompts you to do so.

5. Run the orainstRoot.sh script on the new node if required.

6. Run the root.sh script on the new node from $CRS_HOME to start the Oracle Clusterware on the new node.

7. Obtain the remote port, which you need to know for the next step, by running the following command from $CRS_HOME/opmn/conf:
   ```
   cat ons.config
   ```

8. From $CRS_HOME/bin on an existing node, run the racgons utility as in the following example where remote_port is the port number from the previous step and node2 is the name of the node that you are adding:
   ```
   ./racgons add_config node2:remote_port
   ```

Adding an Oracle Clusterware Home to a New Node Using the OUI Silent Mode

1. Ensure that you have successfully installed the Oracle Clusterware on at least one node in your cluster environment. To use these procedures as shown, your $CRS_HOME environment variable must identify your successfully installed Oracle Clusterware home.

2. Go to $CRS_HOME/oui/bin and run the addNode.sh script using the following syntax where node2 is the name of the new node that is getting added, node2-priv is the private node name for the new node, and node2-vip is the VIP name for the new node:
   ```
   ./addNode.sh -silent "CLUSTER_NEW_NODES={node2}" "CLUSTER_NEW_PRIVATE_NODE_ NAMES={node2-priv}" "CLUSTER_NEW_VIRTUAL_HOSTNAMES={node2-vip}"
   ```

3. Perform the OUI-related steps 5-9 as in the previous procedure, “Adding an Oracle Clusterware Home to a New Node Using the Oracle Universal Installer”.

Adding an Oracle Home with RAC on UNIX-Based Systems

You can use either of the following procedures to add an Oracle home with the RAC software to a new node:

- Adding an Oracle Home to a New Node Using the Oracle Universal Installer
- Adding an Oracle Home to a New Node Using the OUI in Silent Mode

Adding an Oracle Home to a New Node Using the Oracle Universal Installer

1. Ensure that you have successfully installed Oracle with the RAC software on at least one node in your cluster environment. To use these procedures as shown, your $ORACLE_HOME environment variable must identify your successfully installed Oracle home.

2. Go to $ORACLE_HOME/oui/bin and run the addNode.sh script.
3. When the OUI displays the Node Selection Page, select the node or nodes to be added and click Next.

4. Verify the entries that the OUI displays on the Summary Page and click Next.

5. Run the root.sh script on the new node from $ORACLE_HOME when the OUI prompts you to do so.

6. On the new node, set the $ORACLE_HOME environment variable to the location of your Oracle home. Also add $ORACLE_HOME/lib to the environment variable, LD_LIBRARY_PATH.

7. On the new node, run the Oracle Net Configuration Assistant (NETCA) to add a Listener.

8. Use the DBCA to add an instance as described under the heading “Adding and Deleting Instances Using the DBCA Silent Mode” on page 10-8.

### Adding an Oracle Home to a New Node Using the OUI in Silent Mode

1. Ensure that you have successfully installed Oracle with the RAC software on at least one node in your cluster environment. To use these procedures as shown, your $ORACLE_HOME environment variable must identify your successfully installed Oracle home and the node to be added is named node2.

2. Go to $ORACLE_HOME/oui/bin and run the addNode.sh script using the following syntax:

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

3. Perform the OUI-related steps 5-8 as in the previous procedure, “Adding an Oracle Home to a New Node Using the Oracle Universal Installer”.

### Deleting an Oracle Home with RAC from a Node

The procedures for deleting an Oracle home assume that you have successfully installed RAC on the node from which you want to delete the Oracle home. You can use either of the following procedures to delete an Oracle home from a node:

- Deleting an Oracle Home with RAC from a Node Using the Oracle Universal Installer
- Deleting an Oracle Home from a Node Using the OUI in Silent Mode

---

**Note:** When you remove the Oracle Clusterware and RAC database software, you must first remove the Oracle database software and then remove the Oracle Clusterware from the nodes that you are deleting. In other words, you remove the software components from the nodes that you are deleting in the reverse order that you originally installed the clusterware and Oracle database software components.

---

### Deleting an Oracle Home with RAC from a Node Using the Oracle Universal Installer

1. From a node that is to remain a part of your RAC database environment, run the DBCA to remove any database instances from the node that you want to delete if instances exist on that node. To do this, refer to the procedure titled Adding and Deleting Instances Using the DBCA Silent Mode on page 10-8.
2. If you use ASM and this is the home from which the ASM instance runs, then perform the ASM clean up procedures as described under "ASM Instance Clean-Up Procedures for Node Deletion" on page 10-25.

3. If this is the home from which the node-specific Listener runs, then use NETCA to remove the Listener and its Oracle Clusterware resources. If necessary, recreate this Listener in another home on a node that is going to remain part of your cluster database.

4. On the node to be deleted, ensure that the $ORACLE_HOME environment variable is set to the Oracle home then run the following command from $ORACLE_HOME/oui/bin where node_name is the name of the node to be deleted:

   ./runInstaller -updateNodeList ORACLE_HOME=$ORACLE_HOME "CLUSTER_NODES={node_name}" -local

5. Depending on whether you have a shared or non-shared Oracle home, complete one of the two following procedures:
   - For a shared home, run the following command on each of the nodes that are to be deleted, using the shared home location for the $ORACLE_HOME environment variable on each node:

     ./runInstaller -detachHome -local ORACLE_HOME=$ORACLE_HOME

   - For a non-shared home, on each node that you are deleting, invoke OUI from the home and deinstall this home. Make sure that you choose the home to be removed and not just the products under that home.

6. On any node other than the node that you are deleting, make sure that the $ORACLE_HOME environment variable is defined for the Oracle home that you are deleting. Run the following command from $ORACLE_HOME/oui/bin where remaining_nodes_list is a comma-separated list of the nodes that are going to remain as part of your RAC database:

   ./runInstaller -updateNodeList ORACLE_HOME=$ORACLE_HOME "CLUSTER_NODES={remaining_nodes_list}"
Deleting an Oracle Clusterware Home from a Node

The procedures for deleting an Oracle Clusterware home assume that you have successfully installed the clusterware on the node from which you want to delete the Oracle Clusterware home. You can use either of the following procedures to delete an Oracle Clusterware home from a node:

- Deleting an Oracle Clusterware Home from a Node Using the Oracle Universal Installer
- Deleting an Oracle Clusterware Home from a Node Using Silent Mode

Deleting an Oracle Clusterware Home from a Node Using the Oracle Universal Installer

1. Perform the delete node operation for database homes as described in the section titled "Deleting an Oracle Home with RAC from a Node Using the Oracle Universal Installer" on page 10-5 and ensure that the $CRS_HOME environment variable is defined to identify the appropriate Oracle Clusterware home on each node.

2. If you ran the OIFCFG with the -global flag during the installation, then skip this step. Otherwise, from a node that is going to remain in your cluster, from the $CRS_HOME/bin directory, run the following command where node2 is the name of the node you are deleting:

   ```
   ./oifcfg delif –node node2
   ```

3. Obtain the remote port number, which you will use in the next step, using the following command from the $CRS_HOME/opmn/conf directory:

   ```
   cat ons.config
   ```

4. From $CRS_HOME/bin on a node that is going to remain in the cluster, run the RACGONS utility as in the following example where remote_port is the number of the remote port that you obtained in the previous step and node2 is the name of the node that you are deleting:

   ```
   ./racgons remove_config node2:remote_port
   ```

5. On the node to be deleted, run rootdelete.sh as the root user from the $CRS_HOME/install directory. If you are deleting more than one node, then perform this step on all of the other nodes that you are deleting.

6. From any node that you are not deleting, run the following command from the $CRS_HOME/install directory as the root user where node2,node2-number represent the node and the node number that you want to delete:

   ```
   ./rootdeletenode.sh node2,node2-number
   ```

   If necessary, identify the node number using the following command on the node that you are deleting:

   ```
   $CRS_HOME/bin/olsnodes -n
   ```

7. On the node or nodes to be deleted, run the following command from the $CRS_HOME/oui/bin directory where node_to_be_deleted is the name of the node that you are deleting:

   ```
   ./runInstaller -updateNodeList ORACLE_HOME=$CRS_HOME "CLUSTER_NODES={node_to_be_deleted}" CRS=TRUE -local
   ```
Quick-Start Procedures for Adding and Deleting Nodes and Instances on UNIX-Based Systems

8. On the node that you are deleting, run the OUI using the `runInstaller` command from the `$CRS_HOME/oui/bin` directory. Depending on whether you have a shared or non-shared Oracle home, complete one of the following procedures:

   - If you have a shared home, then on any node other than the node to be deleted, run the following command from the `$CRS_HOME/oui/bin` directory:
     
     ```
     ./runInstaller -detachHome -silent ORACLE_HOME=$CRS_HOME
     ```

   - If you have a non-shared home, then de-install the Oracle Clusterware using the OUI by running the `runInstaller` command from the `$CRS_HOME/oui/bin` directory.

9. On any node other than the node that you are deleting, run the following command from the `$CRS_HOME/oui/bin` directory where `remaining_nodes_list` is a comma-separated list of the nodes that are going to remain part of your RAC database:

   ```
   ./runInstaller -updateNodeList ORACLE_HOME=$CRS_HOME "CLUSTER_NODES={remaining_nodes_list}" CRS=TRUE
   ```

Deleting an Oracle Clusterware Home from a Node Using Silent Mode

1. Perform steps 1-7 from the previous procedure, "Deleting an Oracle Clusterware Home from a Node Using the Oracle Universal Installer" and ensure that the `$CRS_HOME` environment variable is defined to identify the appropriate Oracle Clusterware home on each node.

2. Depending on whether you have a shared or non-shared Oracle home, complete one of the two following procedures:

   - For shared homes, do not perform a de-install operation. Instead, perform a detach home operation on the node that you are deleting. To do this, run the following command from `$CRS_HOME/oui/bin`:
     
     ```
     ./runInstaller -detachHome ORACLE_HOME=$CRS_HOME
     ```

   - For a non-shared home, de-install the Oracle home from the node that you are deleting using the OUI as follows by running the following command from the `$ORACLE_HOME/oui/bin` directory, where `crs_home_name` is the name defined for the Oracle Clusterware home:
     
     ```
     ./runInstaller -deinstall -silent "REMOVE_HOMES={$CRS_HOME}" 
     ```

3. Perform step 9 from the previous procedure, "Deleting an Oracle Clusterware Home from a Node Using the Oracle Universal Installer".

   **Note:** Oracle recommends that you back up your voting disk and OCR files after you complete the node deletion process.

Adding and Deleting Instances Using the DBCA Silent Mode

You can use the Database Configuration Assistant (DBCA) in silent mode to add instances to nodes onto which you have extended an Oracle Clusterware home and an Oracle home. Use the following syntax to perform this operation, where `node` is the node onto which you want to add the instance, `gdbname` is the global database name,
Overview of Node Addition Procedures

Adding and Deleting Nodes and Instances

`instname` is the name of the new instance, `sysdba` is the name of an Oracle user with SYSDBA privileges, and `password` is the password for the user name in `sysdba`:

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

Note that you only need to provide an instance name if you want to override the Oracle naming convention for RAC instance names.

Use the DBCA to delete a database instance from a node as follows, where the variables are the same as those in the preceding add instance command:

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

Note that you only need to provide a node name if you are deleting an instance from a node other than the one on which you are running DBCA.

Detailed Procedures for Adding and Deleting Nodes and Instances on UNIX-Based Systems

This section describes the detailed procedures for adding and deleting nodes and instances in RAC environments on UNIX-based systems. The topics in this chapter are:

- Overview of Node Addition Procedures
- Step 1: Connecting New Nodes to the Cluster
- Step 2: Extending Clusterware and Oracle Software to New Nodes
- Step 3: Preparing Storage for RAC on New Nodes
- Step 4: Adding Nodes at the Oracle RAC Database Layer
- Step 5: Adding Database Instances to New Nodes
- Adding Nodes that Already Have Clusterware and Oracle Software to a Cluster
- Deleting Instances from Real Application Clusters Databases
- Deleting Nodes from Real Application Clusters Databases

Overview of Node Addition Procedures

This section provides detailed procedures for adding and deleting nodes to clusters in detail. Before adding nodes, configure the new nodes to be part of your cluster at the network level. Then extend the Oracle Clusterware home from an existing Oracle Clusterware home to the new nodes and extend the Oracle database software with RAC components to the new nodes. Finally, make the new nodes members of the existing cluster database.

Note: You can add nodes on some UNIX-based platforms without stopping existing nodes if your clusterware supports this. Refer to your vendor-specific clusterware documentation for more information.

If the nodes that you are adding to your cluster do not have clusterware or Oracle software installed on them, then you must complete the following five steps. The
Step 1: Connecting New Nodes to the Cluster

Complete the following procedures to connect the new nodes to the cluster and to prepare them to support your cluster database:

- Making Physical Connections
- Installing the Operating System
- Creating Oracle Users
- Verify the Installation with the Cluster Verification Utility

Making Physical Connections

Connect the new nodes' hardware to the network infrastructure of your cluster. This includes establishing electrical connections, configuring network interconnects, configuring shared disk subsystem connections, and so on. Refer to your hardware vendor documentation for details about this step.

Installing the Operating System

Install a cloned image of the operating system that matches the operating system on the other nodes in your cluster. This includes installing required service patches and drivers. Refer to your hardware vendor documentation for details about this process.

Creating Oracle Users

As root user on UNIX-based systems, create the Oracle users and groups using the same user ID and group ID as on the existing nodes.

Verify the Installation with the Cluster Verification Utility

Verify your installation using the Cluster Verification Utility (CVU) as in the following steps:

1. Run the cluvfy command to verify your installation at the post hardware installation stage as shown in the following example, where node_list is a comma-separated list of nodes you want in your cluster:
cluvfy stage -post hwos -n node_list all [-verbose]

This command causes CVU to verify your hardware and operating system environment after at the post-hardware setup stage. After you have configured the hardware and operating systems on the new nodes, you can use this command to verify node reachability, for example, to all of the nodes from the local node. You can also use this command to verify user equivalence to all given nodes the local node, node connectivity among all of the given nodes, accessibility to shared storage from all of the given nodes, and so on.

---

**Note:** You can only use the `all` option with the `-n` argument if you have set the `CV_NODELIST` variable to represent the list of nodes on which you want to perform the CVU operation.

---

**See Also:** "Using the Cluster Verification Utility" on page A-11 for more information about enabling and using the CVU

2. Run the following command to obtain a detailed comparison of the properties of the reference node with all of the other nodes that are part of your current cluster environment:

   cluvfy comp peer [-refnode ref_node] -n node_list
   [-orainv orainventory_group] [-osdba osdba_group] [-verbose]

   where `ref_node` is a node in your existing cluster against which you want CVU to compare, for example, the newly added nodes that you specify with the comma-separated list in `node_list` for the `-n` option, `orainventory_group` is the name of the Oracle inventory group, and `osdba_group` is the name of the OSDBA group.

### Checking the Installation

To verify that your installation is configured correctly, perform the following steps:

1. Ensure that the new nodes can access the private interconnect. This interconnect must be properly configured before you can complete the procedures in "Step 2: Extending Clusterware and Oracle Software to New Nodes" on page 10-12.

2. If you are not using a cluster file system, then determine the location on which your cluster software was installed on the existing nodes. Make sure that you have at least 250MB of free space on the same location on each of the new nodes to install the Oracle Clusterware. In addition, ensure you have enough free space on each new node to install the Oracle binaries.

3. Verify user equivalence to and from an existing node to the new nodes using `rsh` or `ssh`.

After completing the procedures in this section, your new nodes are connected to the cluster and configured with the required software to make them visible to the clusterware. Configure the new nodes as members of the cluster by extending the cluster software to the new nodes as described in Step 2.

---

**Note:** Do not change a hostname after the Oracle Clusterware installation. This includes adding or deleting a domain qualification.
Step 2: Extending Clusterware and Oracle Software to New Nodes

The following topics describe how to add new nodes to the clusterware and to the Oracle database software layers using the OUI:

- Adding Nodes at the Vendor Clusterware Layer
- Adding Nodes at the Oracle Clusterware Layer

Adding Nodes at the Vendor Clusterware Layer

Add the new nodes at the clusterware layer according to the vendor clusterware documentation. For systems using shared storage for the Oracle Clusterware home, ensure that the existing clusterware is accessible by the new nodes. Also ensure that the new nodes can be brought online as part of the existing cluster. Proceed to the next section to add the nodes at the clusterware layer.

Adding Nodes at the Oracle Clusterware Layer

Before beginning this procedure, ensure that your existing nodes have the $CRS_HOME environment variable set correctly. The OUI requires access to the private interconnect that you checked as part of the installation validation in Step 1. If it cannot make required connections, you will not be able to complete the following steps to add nodes:

1. On one of the *existing* nodes go to the $CRS_HOME/oui/bin directory and run the addNode.sh script to start the OUI.
2. The OUI runs in the add node mode and the OUI Welcome page appears. Click Next and the Specify Cluster Nodes for Node Addition page appears.
3. The upper table on the Specify Cluster Nodes for Node Addition page shows the existing nodes, the private nodes names and the virtual IP (VIP) addresses associated with the Oracle Clusterware home from which you launched the OUI. Use the lower table to enter the public, private node names and the virtual hostnames of the new nodes.
4. If you are using vendor clusterware, then the public node names automatically appear in the lower table. Click Next and the OUI verifies connectivity on the existing nodes and on the new nodes. The verifications that the OUI performs include determining whether:
   - The nodes are up
   - The nodes and private node names are accessible by way of the network

**Note:** Instead of performing the first six steps of this procedure, you can alternatively run addNode.sh in silent mode as described at the end of this section.
Note: If any of the existing nodes are down, you can proceed with the procedure. However, once the nodes are up, you must run the following command on each of those nodes:

```
runInstaller -updateNodeList -local "CLUSTER_NODES={available_node_list}" ORACLE_HOME=$CRS_HOME
```

This operation should be run from the $CRS_HOME/oui/bin directory and the available_node_list values is comma-separated list of all of nodes currently in the cluster and $CRS_HOME defines the Oracle Clusterware home directory.

- The virtual hostnames are not valid nodes on the network.

5. If any verifications fail, then the OUI re-displays the Specify Cluster Nodes for Node Addition page with a Status column in both tables indicating errors. Correct the errors or deselect the nodes that have errors and proceed. However, you cannot deselect existing nodes; you must correct problems on nodes that are already part of your cluster before you can proceed with node addition. If all the checks succeed, then the OUI displays the Node Addition Summary page.

    Note: Oracle strongly recommends that you install Oracle Clusterware on every node in the cluster that has vendor clusterware installed.

6. The Node Addition Summary page displays the following information showing the products that are installed in the Oracle Clusterware home that you are extending to the new nodes:

- The source for the add node process, which in this case is the Oracle Clusterware home
- The private node names that you entered for the new nodes
- The new nodes that you entered
- The required and available space on the new nodes
- The installed products listing the products that are already installed on the existing Oracle Clusterware home

Click Next and the OUI displays the Cluster Node Addition Progress page.

7. The Cluster Node Addition Progress page shows the status of the cluster node addition process. The table on this page has two columns showing the four phases of the node addition process and the phases’ status as follows:

- Instantiate Root Scripts—Instantiates rootaddnode.sh with the public nodes, private node names, and virtual hostnames that you entered on the Cluster Node Addition page.
- Copy the Oracle Clusterware home to the New Nodes—Copies the Oracle Clusterware home to the new nodes unless the Oracle Clusterware home is on a cluster file system.
- Save Cluster Inventory—Updates the node list associated with the CRS home and its inventory.
- Run rootaddnode.sh and root.sh—Displays a dialog prompting you to run rootaddnode.sh from the local node (the node on which you are running
the OUI) and to run root.sh on the new nodes. If OUI detects that the new nodes do not have an inventory location, then it instructs you to run orainstRoot.sh on those nodes. The central inventory location is the same as that of the local node. The addNodeActiontimestamp.log file, where timestamp shows the session start date and time, contains information about which scripts you need to run and on which nodes you need to run them.

The Cluster Node Addition Progress page's Status column displays In Progress while the phase is in progress, Suspended when the phase is pending execution, and Succeeded after the phase completes. On completion, click Exit to end the OUI session. After the OUI displays the End of Node Addition page, click Exit to end the OUI session.

8. Obtain the remote port number, which you will use in the next step, using the following command from the $CRS_HOME/opmn/conf directory:

   cat ons.config

9. Run the racgons utility from the bin subdirectory of the Oracle Clusterware home to configure the Oracle Notification Services (ONS) port number. Use the following command, supplying the name of the node you are adding for new_node_name and the remote port number obtained in the previous step:

   racgons add_config new_node_name:remote_port

10. Check that your cluster is integrated and that the cluster is not divided into partitions by completing the following operations:

    ■ Run the following CVU command to obtain detailed output for verifying cluster manager integrity on all of the nodes that are part of your RAC environment:

        cluvfy comp clumgr -n all [-verbose]

    ■ Use the following CVU command to obtain detailed output for verifying cluster integrity on all of the nodes that are part of your RAC environment:

        cluvfy comp clu [-verbose]

    ■ Use the following command to perform an integrated validation of the Oracle Clusterware setup on all of the configured nodes, both the pre-existing nodes and the nodes that you have added:

        cluvfy comp stage -post crinst -n all [-verbose]

   See Also: "Using the Cluster Verification Utility" on page A-11" for more information about enabling and using the CVU

As mentioned earlier in this procedure, you can optionally run addNode.sh in silent mode, replacing steps 1-6, as follows where nodeI, nodeI+1, and so on are the new nodes that you are adding:

   addNode.sh -silent "CLUSTER_NEW_NODES={nodeI, nodeI+1, ... nodeI+n}" "CLUSTER_NEW_PRIVATE_NODE_NAMES={pnI, pnI+1, ... pnI+n}" "CLUSTER_NEW_VIRTUAL_HOSTNAMES={vipI, vipI+1, ... vipI+n}"

   You can alternatively specify the variable=value entries in a response file and invoke addNode as follows:

   addNode.sh -silent -responseFile <filename> OR addNode.bat -silent -responseFile <filename>
Refer to the *Oracle Universal Installer and OPatch User’s Guide* for details about how to configure command line response files. Command line values always override response file values.

Run `rootaddnode.sh` on the local node, or the node on which you are performing this procedure, and run `root.sh` on the new nodes. If the OUI detects that the new nodes do not have an inventory location, then the OUI instructs you to run `orainstRoot.sh` on those nodes. The central inventory location is the same as that of the local node. The `addNodeActionstimestamp.log` file, where `timestamp` shows the session start date and time, contains the information about which scripts you need to run and on which nodes you need to run them.

After you have completed the procedures in this section for adding nodes at the Oracle Clusterware layer, you have successfully extended the Oracle Clusterware home from your existing the Oracle Clusterware home to the new nodes. Proceed to Step 3 to prepare storage for RAC on the new nodes.

### Step 3: Preparing Storage for RAC on New Nodes

To extend an existing RAC database to your new nodes, configure storage for the new nodes so that the storage is the same as on the existing nodes. For example, the Oracle Cluster Registry (OCR) and the voting disk must be accessible by the new nodes using the same path as the other nodes use. In addition, the OCR and voting disk devices must have the same permissions as on the existing nodes. Prepare the same type of storage on the new nodes as you are using on the other nodes in the RAC environment that you want to extend as follows:

- **Automatic Storage Management (ASM)**
  
  If you are using ASM, then make sure that the new nodes can access the ASM disks with the same permissions as the existing nodes.

- **Oracle Cluster File System (OCFS)**
  
  If you are using Oracle Cluster File Systems, then make sure that the new nodes can access the cluster file systems in the same way that the other nodes access them.

- **Vendor Cluster File Systems**
  
  If your cluster database uses vendor cluster file systems, then configure the new nodes to use the vendor cluster file systems. Refer to the vendor clusterware documentation for the pre-installation steps for your platform.

- **Raw Device Storage**
  
  If your cluster database uses raw devices, then prepare the new raw devices by following the procedures described in the following section.

**See Also:** Your platform-specific Oracle Database Oracle Clusterware and Oracle Real Application Clusters installation guide for more information about the Oracle Cluster File System

Run the following command to verify your cluster file system and obtain detailed output where `nodelist` includes both the pre-existing nodes and the newly added nodes and `filesystem` is the name of the file system that you used for the Oracle Cluster File System:

```
cluvfy comp cfs -n nodelist -f filesystem [-verbose]
```
Raw Device Storage Preparation for New Nodes

To prepare raw device storage on the new nodes, you need at least two new disk partitions to accommodate the redo logs for each new instance. Make these disk partitions the same size as the redo log partitions that you configured for the existing nodes’ instances. Also create an additional logical partition for the undo tablespace for automatic undo management.

On applicable operating systems, you can create symbolic links to your raw devices. Optionally, on all platforms you can create a raw device mapping file and set the DBCA_RAW_CONFIG environment variable so that it points to the raw device mapping file.

See Also: Your platform-specific Oracle Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide for more information about configuring raw partitions and using raw device mapping files.

Use your vendor-supplied tools to configure the required raw storage.

Run the following command to verify that the prepared raw device storage is accessible from all of the configured cluster nodes where node_list includes both the pre-existing nodes and the newly added nodes and storageID_list is a comma-separated list of storage identifiers:

```
cluvfy comp ssa [-n node_list] [-s storageID_list] [-verbose]
```

See Also: “Using the Cluster Verification Utility” on page A-11” for more information about enabling and using the CVU

After completing the procedures in this section, you have configured your cluster storage so that the new nodes can access the Oracle software. Additionally, the existing nodes can access the new nodes and instances. Use the OUI as described in the procedures in Step 4 to configure the new nodes at the RAC database layer.

Step 4: Adding Nodes at the Oracle RAC Database Layer

To add nodes at the Oracle RAC database later, run the OUI in add node mode to configure your new nodes. Before beginning this procedure, ensure that your existing nodes have the $ORACLE_HOME environment variable set correctly. If you have multiple Oracle homes, then perform the following steps for each Oracle home that you want to include on the new nodes:

Note: Instead of performing the first six steps of this procedure, you can alternatively run addNode.sh in silent mode as described at the end of this section.

1. On an existing node from the $ORACLE_HOME/oui/bin directory, run the addNode.sh script. This starts the OUI in the add node mode and displays the OUI Welcome page. Click Next on the Welcome page and the OUI displays the Specify Cluster Nodes for Node Addition page.
2. The Specify Cluster Nodes for Node Addition page has a table showing the existing nodes associated with the Oracle home from which you launched the OUI. A node selection table appears on the bottom of this page showing the nodes that are available for addition. Select the nodes that you want to add and click Next.

3. The OUI verifies connectivity and performs availability checks on the existing nodes and on the nodes that you want to add. Some of checks performed determine whether:
   - The nodes are up
   - The nodes are accessible by way of the network

4. If any of the checks fail, then fix the problem and proceed or deselect the node that has the error and proceed. You cannot deselect existing nodes; you must correct problems on the existing nodes before proceeding with node addition. If all of the checks succeed then the OUI displays the Node Addition Summary page.

5. The Node Addition Summary page has the following information about the products that are installed in the Oracle home that you are going to extend to the new nodes:
   - The source for the add node process, which in this case is the Oracle home
   - The existing nodes and new nodes
   - The new nodes that you selected
   - The required and available space on the new nodes
   - The installed products listing all the products that are already installed in the existing Oracle home

Click Finish and the OUI displays the Cluster Node Addition Progress page.

6. The Cluster Node Addition Progress page shows the status of the cluster node addition process. The table on this page has two columns showing the four phases of the node addition process and the phase's status as follows:
   - Copy the Oracle Home to the New Nodes—Copies the entire Oracle home from the local node to the new nodes unless the Oracle home is on a cluster file system
   - Save Cluster Inventory—Updates the node list associated with the Oracle home and its inventory
   - Run root.sh—Displays the dialog prompting you to run root.sh on the new nodes

The Cluster Node Addition Progress page's Status column displays Succeeded if the phase completes, In Progress if the phase is in progress, and Suspended
Step 5: Adding Database Instances to New Nodes

when the phase is pending execution. After the OUI displays the End of Node Addition page, click Exit to end the OUI session.

7. Run the root.sh script on all of the new nodes.

The Oracle Clusterware issues the oifcfg command as in the following example:

```
oifcfg setif -global eth0/146.56.76.0:public eth1/192.0.0.0:cluster_interconnect
```

This sets both networks to global. Therefore, you do not need to run the oifcfg command manually after you add a node unless the network interfaces differ.

8. Add a listener to the new node by running the Net Configuration Assistant (NetCA).

9. Assuming that the Oracle Clusterware software is running at this time, you can use the following commands to verify the OCR and the Oracle Clusterware components:

```
cluvfy comp ocr -n all [-verbose]
cluvfy comp crs -n all [-verbose]
```

See Also: "Using the Cluster Verification Utility" on page A-11" for more information about enabling and using the CVU.

As mentioned earlier in this procedure, you can optionally run addNode.sh in silent mode, replacing steps 1-6, as follows where nodeI, nodeI+1, and so on are the new nodes that you are adding:

```
addNode.sh -silent "CLUSTER_NEW_NODES={nodeI, nodeI+1, ... nodeI+n}"
```

The above variable=value entries can also be specified in a response file, filename, and addNode can be invoked as:

```
addNode.sh -silent -responseFile filename
```

Refer to the OUI Concepts Guide for details on how command line response file should be specified. Command line values always override response file values.

After completing the procedures in this section, you have defined the new nodes at the cluster database layer. You can now add database instances to the new nodes as described in Step 5.

Step 5: Adding Database Instances to New Nodes

You can use either Enterprise Manager or the Database Configuration Assistant (DBCA) to add database instances to new nodes. Before beginning this procedure, ensure that your existing nodes have the $ORACLE_HOME environment variable set correctly. To add a database instance to a new node with Enterprise Manager, perform the following procedure for each new node to add instances:

1. From the Cluster Database Home page, click the Maintenance tab.

2. Under the Deployments section, click Add Instance. This initiates a wizard to guide you through the instance addition process.

3. Perform the following tasks for each wizard step:
   - Provide host and ASM credentials.
Step 5: Adding Database Instances to New Nodes

- Specify the hosts for which you want to add instances.
- Review and submit the job.

After you submit the job, the wizard provides a summary page that shows whether the job succeeded and the wizard also provides other detailed information about the job, such as the elapsed time and an output log.

To add a database instance to a new node with DBCA, perform the following procedure for each new node to add instances:

1. Start the Database Configuration Assistant (DBCA) by entering `dbca` at the system prompt from the `bin` directory in the `$ORACLE_HOME`.
   
The DBCA displays the Welcome page for RAC. Click Help on any DBCA page for additional information.

2. Select Real Application Clusters database, click Next, and the DBCA displays the Operations page.

3. Select Instance Management, click Next, and the DBCA displays the Instance Management page.

4. Select Add Instance and click Next. The DBCA displays the List of Cluster Databases page that shows the databases and their current status, such as ACTIVE, or INACTIVE.

5. From the List of Cluster Databases page, select the active RAC database to which you want to add an instance. If your user ID is not operating-system authenticated, then the DBCA prompts you for a user ID and password for a database user that has SYSDBA privileges. If the DBCA prompts you, then enter a valid user ID and password and click Next. The DBCA displays the List of Cluster Database Instances page showing the names of the existing instances for the RAC database that you selected.

6. Click Next to add a new instance and the DBCA displays the Adding an Instance page.

7. On the Adding an Instance page, enter the instance name in the field at the top of this page if the instance name that the DBCA provides does not match your existing instance naming scheme. Then select the new node name from the list, click Next, and the DBCA displays the Services Page.

8. Enter the services information for the new node's instance, click Next, and the DBCA displays the Instance Storage page.

9. 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 datafile name to the raw device name for the tablespace.

10. 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, the DBCA displays another dialog box. Enter the raw device name that you created in the section "Raw Device Storage Preparation for New Nodes" on page 10-16 in the File Name field.

11. 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 10 for all of the other redo log groups, click Finish, and the DBCA displays a Summary dialog.
12. Review the information on the Summary dialog and click OK or click Cancel to end the instance addition operation. The DBCA displays a progress dialog showing the DBCA performing the instance addition operation. When the DBCA completes the instance addition operation, the DBCA displays a dialog asking whether you want to perform another operation.

13. After you terminate your DBCA session, run the following command to verify the administrative privileges on the new node and obtain detailed information about these privileges where nodelist consists of the newly added nodes:

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

After you have completed the procedures in this section, the DBCA has successfully added the new instance to the new node and completed the following steps:

- Created and started an ASM instance on each new node if the existing instances were using ASM
- Created a new database instance on each new node
- Created and configured high availability components
- Configured and started node applications for the GSD, Oracle Net Services listener, and Enterprise Manager agent
- Created the Oracle Net configuration
- Started the new instance
- Created and started services if you entered services information on the Services Configuration page

After adding the instances to the new nodes using the steps described in this section, perform any needed service configuration procedures as described in Chapter 6, "Introduction to Workload Management".

---

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

---

**Adding Nodes that Already Have Clusterware and Oracle Software to a Cluster**

Before beginning this procedure, ensure that your existing nodes have the $CRS_HOME and $ORACLE_HOME environment variables set correctly. To add nodes to a cluster that already have clusterware and Oracle software installed on them, you must configure the new nodes with the Oracle software that is on the existing nodes of the cluster. To do this, you must run two versions of an OUI process: one for the clusterware and one for the database layer as described in the following procedures:

1. Add new nodes at the Oracle clusterware layer by running the OUI from the Oracle Clusterware home on an existing node, using the following command:

   $CRS_HOME/oui/bin/addNode.sh -noCopy

2. Add new nodes at the Oracle software layer by running the OUI from the Oracle home as follows:

   $ORACLE_HOME/oui/bin/addNode.sh -noCopy

In the -noCopy mode, the OUI performs all add node operations except for the copying of software to the new nodes.
Deleting Instances from Real Application Clusters Databases

The procedures in this section explain how to use Enterprise Manager and the DBCA to delete an instance from a RAC database.

To delete an instance with Enterprise Manager:

1. From the Cluster Database Home page, click the Maintenance tab.
2. Under the Deployments section, click Delete Instance. This action initiates a wizard to guide you through the deletion process.
3. Perform the following tasks for each wizard step:
   - Provide host and ASM credentials.
   - Specify hosts from which to delete instances.
   - Review and submit the job.

After you submit the job, the wizard provides a summary page that shows whether the job succeeded and the wizard also provides other detailed information about the job, such as elapsed time and an output log.

To delete an instance with DBCA:

1. Start the DBCA on a node other than the node that hosts the instance that you want to delete. On the DBCA Welcome page select Oracle Real Application Clusters Database, click Next, and the DBCA displays the Operations page.
2. On the DBCA Operations page, select Instance Management, click Next, and the DBCA displays the Instance Management page.
3. On the Instance Management page, select Delete Instance, click Next, and the DBCA displays the List of Cluster Databases page.
4. Select a RAC database from which to delete an instance. If your user ID is not operating-system authenticated, then the DBCA also prompts you for a user ID and password for a database user that has SYSDBA privileges. If the DBCA prompts you for this, then enter a valid user ID and password. Click Next and the DBCA displays the List of Cluster Database Instances page. The List of Cluster Database Instances page shows the instances associated with the RAC database that you selected and the status of each instance.
5. Select a remote instance to delete and click Finish.
6. If you have services assigned to this instance, then the DBCA Services Management page appears. Use this feature to reassign services from this instance to other instances in the cluster database.
7. Review the information about the instance deletion operation on the Summary page and click OK. Otherwise, click Cancel to cancel the instance deletion operation. If you click OK, then the DBCA displays a Confirmation dialog.
8. Click OK on the Confirmation dialog to proceed with the instance deletion operation and the DBCA displays a progress dialog showing that the DBCA is performing the instance deletion operation. During this operation, the DBCA removes the instance and the instance’s Oracle Net configuration.

Note: Oracle recommends that you back up your voting disk and OCR files after you complete the node addition process.
completes this operation, the DBCA displays a dialog asking whether you want to perform another operation.

9. Click **No** and exit the DBCA or click **Yes** to perform another operation. If you click **Yes**, then the DBCA displays the Operations page.

At this point, you have accomplished the following:

- De-registered 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.

**Note:** Oracle recommends that you back up your voting disk and OCR after you complete the instance deletion process.

---

### Deleting Nodes from Real Application Clusters Databases

Before beginning these procedures, ensure that your existing nodes have the `$CRS_HOME` and `$ORACLE_HOME` environment variables set correctly. Use the following procedures to delete nodes from Oracle clusters on UNIX-based systems:

**Note:** You can perform some of the steps in this procedure in silent mode as described at the end of this section.

1. If there are instances on the node that you want to delete, then execute the procedures in the section titled "Deleting Instances from Real Application Clusters Databases" on page 10-21 before executing these procedures. If you are deleting more than one node, then delete the instances from all the nodes that you are going to delete.

2. If you use ASM, then perform the procedures in the following section, "ASM Instance Clean-Up Procedures for Node Deletion" on page 10-25.

3. If this is the Oracle home from which the node-specific listener named `LISTENER_nodename` runs, then use NetCA to remove this listener and its Oracle Clusterware resources. If necessary, re-create this listener in another home.

4. Depending on whether you have a shared or non-shared Oracle home, complete one of the two following procedures:

   - For a shared home, run the following command on each of the nodes that are to be deleted:
     
     ```bash
     runInstaller -detachHome -local ORACLE_HOME=$ORACLE_HOME
     ```

   - For a non-shared home, on each node that you are deleting, execute the following two steps:
     
     1. Run the following command:
        
        ```bash
        runInstaller -updateNodeList ORACLE_HOME=$ORACLE_HOME
        CLUSTER_NODES="" -local
        ```

Deleting Nodes from Real Application Clusters Databases

The `runInstaller` command is located in the directory `$ORACLE_HOME/oui/bin`. Executing this command does not launch an installer GUI.

- Invoke OUI from the home and deinstall this home. Make sure that you choose the home to be removed and not just the products under that home.

5. From an existing node, run the following command:

```bash
runInstaller -updateNodeList ORACLE_HOME=$ORACLE_HOME "CLUSTER_NODES={nodelist}"
```

where `nodelist` is a comma-separated list of nodes that remain in the cluster.

6. Run the following commands to remove node-specific interface configurations where `nodename` is the name of the node and `remote_port` is port on the deleted node:

```bash
rac bons remove_config nodename:remote_port
oifcfg delif -node nodename
```

7. On the node that you are deleting, run the command `$CRS_HOME/install/rootdelete.sh` to disable the Oracle Clusterware applications that are on the node. Only run this command once and use the `nosharedhome` argument if you are using a local file system. The default for this command is `sharedhome` which prevents you from updating the permissions of local files such that they can be removed by the `oracle` user.

   If the `ocr.loc` file is on a shared file system, then execute the command `$CRS_HOME/install/rootdelete.sh remote sharedvar`. If the `ocr.loc` file is not on a shared file system, then execute the `$CRS_HOME/install/rootdelete.sh remote nosharedvar` command.

   If you are deleting more than one node from your cluster, then repeat this step on each node that you are deleting.

8. Run `$CRS_HOME/install/rootdeletenode.sh` on any remaining node in the cluster to delete the nodes from the Oracle cluster and to update the Oracle Cluster Registry (OCR). If you are deleting multiple nodes, then execute the command `$CRS_HOME/install/rootdeletenode.sh node1,node1-number,node2,node2-number,... nodeN,nodeN-number` where `node1` through `nodeN` is a list of the nodes that you want to delete, and `node1-number` through `nodeN-number` represents the node number. To determine the node number of any node, execute the command `$CRS_HOME/bin/olsnodes -n`. To delete only one node, enter the node name and number of the node that you want to delete with the command `$CRS_HOME/install/rootdeletenode.sh node1,node1-number`.

9. Depending on whether you have a shared or non-shared Oracle Clusterware home, complete one of the two following procedures:

   - For a shared home, run the following command on each of the nodes that are to be deleted, providing the shared home location for `$CRS_HOME`:
     ```bash
     runInstaller -detachHome -local ORACLE_HOME=$CRS_HOME
     ```
   - For a non-shared home, on each node that you are deleting, execute the following two steps:
     - Run the following command:
runInstaller -updateNodeList ORACLE_HOME=$CRS_HOME
CLUSTER_NODES="" -local CRS=true

The `runInstaller` command is located in the directory `$CRS_HOME/oui/bin`. Executing this command does not launch an installer GUI.

- Invoke OUI from the home and deinstall this home. Make sure that you choose the home to be removed and not just the products under that home.

10. From an existing node, run the following command:

```
runInstaller -updateNodeList ORACLE_HOME=$CRS_HOME "CLUSTER_NODES={nodelist}"
```

where `nodelist` is a comma-separated list of nodes that remain in the cluster.

11. Run the following command to verify that the node is no longer a member of the cluster and to verify that the Oracle Clusterware components have been removed from this node:

```
cluvfy comp crs -n all [-verbose]
```

The response from this command should not contain any information about the node that you deleted; the deleted node should no longer have the Oracle Clusterware components on it. This verifies that you have deleted the node from the cluster.

See Also: "Using the Cluster Verification Utility" on page A-11 for more information about enabling and using the CVU

As mentioned earlier in this procedure, you can optionally delete nodes from Real Application Clusters databases in silent mode by completing the following steps:

1. Complete steps 1, 2, and 3 of the procedure described at the start of this section.

2. Depending on whether you have a shared or non-shared Oracle home, complete one of the following procedures:
   - For a shared home, run the following command on each of the nodes that are to be deleted:
     ```
     ./runInstaller -detachHome -local ORACLE_HOME=$ORACLE_HOME
     ```
   - For a non-shared home, de-install the Oracle home from the node that you are deleting by running the following command from the `$ORACLE_HOME/oui/bin` directory:
     ```
     ./runInstaller -deinstall -silent "REMOVE_HOMES={$ORACLE_HOME}"
     ```

3. Complete steps 5, 6, 7, and 8 of the procedure described at the start of this section.

4. Depending on whether you have a shared or non-shared Oracle Clusterware home, complete one of the following procedures:
   - For shared homes, do not perform a de-install operation. Instead, perform a detach home operation on the node that you are deleting. To do this, run the following command from `$CRS_HOME/oui/bin`:
     ```
     ./runInstaller -detachHome ORACLE_HOME=$CRS_HOME
     ```
   - For a non-shared home, on each node that you are deleting, execute the following step from the `$CRS_HOME/oui/bin` directory:
./runInstaller -deinstall -silent "REMOVE_HOMES={CRS_HOME}"*

where crs_home_name is the name given to the Oracle Clusterware home you are deleting.

5. Complete steps 10 and 11 of the procedure described at the start of this section.

ASM Instance Clean-Up Procedures for Node Deletion

Perform the following procedure to remove the ASM instances as follows:

1. If this is the Oracle home which from which the ASM instance runs, then remove the ASM configuration by completing the following steps. Run the command `srvctl stop asm -n node` for all nodes on which this Oracle home exists. Run the command `srvctl remove asm -n node` for all nodes on which this Oracle home exists.

2. If you are using a cluster file system for your ASM Oracle home, then on the local node, ensure that your local node has the `$ORACLE_BASE` and `$ORACLE_HOME` environment variables set correctly and run the following commands:

```
rm -r $ORACLE_BASE/admin/+ASM
rm -f $ORACLE_HOME/dbs/*ASM*
```

3. If you are not using a cluster file system for your ASM Oracle home, then run the `rm` or `delete` commands mentioned in the previous step on each node on which the Oracle home exists.

4. Remove oratab entries beginning with `+ASM`.

---

**Note:** Oracle recommends that you back up your voting disk and OCR files after you complete the node deletion process.
This chapter briefly describes database design and deployment techniques for Oracle Real Application Clusters (RAC) environments. It also describes general high availability topics such as how the Oracle Clusterware manages services within RAC. The topics in this chapter are:

- Service Configuration Recommendations for High Availability
- General Database Deployment Topics for Real Application Clusters

Service Configuration Recommendations for High Availability

This section describes the following high availability service configuration recommendations:

- Service Topologies and Workload Management in Real Application Clusters
- Recommended Real Application Clusters Service Configurations
- Automatic Workload Repository
- Setting Service Levels and Thresholds

Service Topologies and Workload Management in Real Application Clusters

Services are the basis for workload management in RAC. Clients and mid-tier applications make connection requests by specifying a global service name. Because RAC can reallocate services among instances in response to planned and unplanned outages, services greatly extend the availability and scalability of RAC environments.

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

Recommended Real Application Clusters Service Configurations

The recommended service configuration is to uniformly distribute service assignments across all available nodes. This simplifies your configuration and provides optimal high availability. Another approach is to non-uniformly configure services. In other words, workload sharing configurations can resemble many different topologies.

For example, assume that you have a five-node cluster with two instances, A and B, serving as the preferred instances for Customer Relationship Management (CRM). This same cluster could have instances C, D, and E as the preferred instances for Accounts Payable (AP). Instances A and B are the available instances for AP if one or more of AP’s preferred instances become unavailable. Instances C, D, and E are the
available instances for CRM if one or more of the CRM preferred instances becomes unavailable.

This configuration enables each service to use a group of instances that acts as both the preferred instances and as the instances that are available in case of a failure. After an outage, a client recovers its connections on another instance in the same group.

In this configuration, during normal operations RAC routes application sessions by service to separate groups of instances. If a preferred instance becomes unavailable, then the Oracle Clusterware relocates connections among the remaining RAC instances that offer that service.

Workload managed configurations achieve the highest availability and performance by transparently maintaining affinity based on service. Planned and unplanned outages on one domain can be isolated from other domains and the affected service is recovered or upgraded in isolation.

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. This enables 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 schedule job classes by service and to assign priorities to consumer groups. If necessary, you can rebalance workloads manually with the `DBMS_SERVICE.DISCONNECT_SESSION` PL/SQL procedure. You can also use this procedure to disconnect a series of sessions and 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 packages.

Setting Service Levels and Thresholds

Enterprise Manager and local Listeners subscribe to events that indicate changes in service levels. You can set service level metric thresholds with either Enterprise Manager or with Oracle packages.

You can see historical values for metrics in the `V$SERVICEMETRIC_HISTORY` view. Information about a service from the application level is available in the `V$SESSION` and `V$SQL` views. Service levels, or thresholds, are the baseline operational levels, and events indicate violations of these baselines. You can also examine `GV$SVCMETRIC` for timings such as resource consumption. Use the `V$ACTIVE_SERVICES` and `GV$ACTIVE_SERVICES` views to identify which services are running on which instances.
How the Oracle Clusterware Manages Service Relocation

When an instance goes offline due to a planned outage or failure, the Oracle Clusterware relocates the service to another available instances and re-establishes the connection without service interruption. This occurs as long as the underlying service components on which the relocation relies are enabled for relocation and restart.

General Database Deployment Topics for Real Application Clusters

This section describes a few topics to consider when deploying RAC databases. Your 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 a RAC database. This section contains the following topics:

- Tablespace Use in Real Application Clusters
- Object Creation and Performance in Real Application Clusters
- Node Addition and Deletion and the SYSAUX Tablespace in Real Application Clusters
- Distributed Transactions and Real Application Clusters

Tablespace Use in Real Application Clusters

In addition to using locally managed tablespaces, you can further simplify space administration by using automatic segment-space management and automatic undo management.

Automatic segment-space management distributes instance workloads among each instance’s subset of blocks for inserts. This improves RAC performance because it minimizes block transfers. To deploy automatic undo management in a RAC environment, each instance must have its own undo tablespace.

Object Creation and Performance in Real Application Clusters

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 Real Application Clusters

If you add nodes to your 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: our Oracle Real Application Clusters installation and configuration guide for guidelines about sizing the SYSAUX tablespace for multiple instances.
Distributed Transactions and Real Application Clusters

When you use Oracle Distributed Transaction Processing (DTP) in RAC, which includes both XA and distributed SQL transactions, all tightly coupled branches of a distributed transaction must be hosted on the same instance. To ensure this, create multiple DTP services, with one or more on each RAC instance. Each DTP service is a singleton service that is available on one and only one 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 RAC" on page 6-15 for more details about enabling services and distributed transactions and Oracle Database Application Developer’s Guide - Fundamentals for more information about distributed transactions in RAC
This chapter describes how to monitor Oracle Real Application Clusters (RAC) performance. This chapter explains how to verify the interconnect settings, how to use the performance views for RAC, and descriptions of RAC-specific statistics and how to monitor them. This chapter describes Automatic Workload Repository (AWR), how to analyze RAC statistics, and how to analyze RAC wait events. This chapter also explains how to monitor performance in RAC databases with Enterprise Manager. This chapter contains these topics:

- Overview of Monitoring Real Application Clusters Databases
- Verifying the Interconnect Settings for Real Application Clusters
- Performance Views in Real Application Clusters
- Real Application Clusters Performance Statistics
- Automatic Workload Repository in Real Application Clusters Environments
- Monitoring Real Application Clusters Statistics and Events
- Monitoring Performance with Oracle Enterprise Manager

Overview of Monitoring Real Application Clusters Databases
All single-instance tuning practices for Oracle Database apply to applications running on RAC databases. Therefore, implement the single-instance tuning methodologies described in Oracle Database Performance Tuning Guide.

Verifying the Interconnect Settings for Real Application Clusters
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.

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 command or 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.
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 “Administering System and Network Interfaces with `OIFCFG`” on page 9-7 to assign private network or private IP addresses.

---

**Performance Views in Real Application Clusters**

Each instance has a set of instance-specific views. 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$. A global view contains all columns from its respective instance-specific view as well as the INST_ID column. The instance number is also appended to the names of the archived redo log threads to create a unique identifier for each instance’s archived redo logs.

**See Also:** Oracle Database Reference for restrictions on GV$ views and complete descriptions of related parameters and views

**Creating Real Application Clusters Data Dictionary Views with CATCLUST.SQL**

If you did not create your RAC database with the Database Configuration Assistant (DBCA), then you must run the `CATCLUST.SQL` script to create RAC-related views and tables. You must have SYSDBA privileges to run this script.

**See Also:** our Oracle Real Application Clusters installation and configuration guide for more information about creating your RAC database

---

**Real Application Clusters Performance Statistics**

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

**The Content of Real Application Clusters Statistics**

RAC-specific 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 Real Application Clusters Environments

In RAC environments, each Automatic Workload Repository (AWR) snapshot captures data from all active instances within the cluster. The data for each snapshot set that is captured for all active instances is from the same point in time. In addition, the data for each instance is stored separately and is identified with an instance identifier. 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.

Monitoring Real Application Clusters Statistics and Events

This section explains wait events and statistics specific to 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.

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

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 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 RAC-related sections appear later in the report:

- Global enqueue statistics
- Global CR statistics
- Global CURRENT served statistics
- Global cache transfer statistics.

Real Application Clusters Wait Events

Analyzing and interpreting what sessions are waiting for is an important method to determine where time is spent. In 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:
Monitoring Real Application Clusters Statistics and Events

- 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 RAC convey information valuable for performance analysis. They are used in Automatic Database Diagnostic Monitor (ADDM) to enable precise diagnostics of the impact 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 Cache Fusion Impact in Real Application Clusters
- Analyzing Performance Using GCS and GES Statistics

Analyzing Cache Fusion Impact in Real Application Clusters

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 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.

**Note:** You must run Statspack at level 7 to collect statistics related to block contention and segment block waits.

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.
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 increases and a high proportion of time waited is for global cache (gc), then the cause should be determined. Most reports include a breakdown of events sorted by percentage of the total time.

It is useful to start with an ADDM report, which would analyze the routinely collected performance statistics with respect to their impact and point to the objects and SQL contributing most to the time waited, and then move on to the more detailed reports produced by AWR and Statspack. The most important wait events for RAC include various categories, such as:

- **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
- **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 remote instance received the block after a remote instance processing delay. In most cases, this is due to a log flush. The existence of these wait events does not necessarily characterize high concurrency for the blocks. High concurrency is instead evidenced by the existence of the gc buffer busy event. This event indicates that the block was pinned or held up by a session on a remote instance. It can also indicate that a session on the same instance has already requested the block, which in either case is in transition between instances and the current session needs to wait behind it.

These events are usually the most frequent in the absence of block contention and the length of the wait is determined by the time it takes on the physical network, the time to process the request in the serving instances and the time it takes for the requesting process to wake up after the block arrives.
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.

Monitoring Performance with Oracle Enterprise Manager

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 RAC database performance from your office, home, or a remote site, as long as you have access to a Web browser.

See Also: Oracle Enterprise Manager Concepts and the Enterprise Manager Online Help for more information about Enterprise Manager

Overview of Enterprise Manager Monitoring

You can install, configure, and monitor a RAC database from a single location using either Oracle Enterprise Manager Database Control or Oracle Enterprise Manager Grid Control as follows:

Oracle Enterprise Manager Database Control is designed to manage a single Oracle RAC10g Database with its instance, Listener, host, cluster, and Automatic Storage Management (ASM) targets. The Database Configuration Assistant (DBCA) automatically configures Database Control when you create a database.

Oracle Enterprise Manager Grid Control enables you to centrally manage your entire computing enterprise. You can install Grid Control separately onto servers either inside or outside of your cluster to manage multiple RAC databases and multiple cluster targets. Grid Control consists of the Grid Control console and one...
or more Oracle Management Services, the Oracle Management Repository, and remote Oracle Management Agents installed on each monitored host.

Both Database Control and Grid Control are cluster-aware and provide a central console to manage your cluster database. From the Cluster Database pages, you can:

- View the overall system status, such as the number of nodes in the cluster database and their current statuses. 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 alerts aggregated across all of the instances with drilldowns to the source of each alert. An alert 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.
- Set thresholds for alert generation on a cluster database-wide basis. Thresholds are values that indicate whether a warning or critical alert should be initiated.
- Monitor performance metrics aggregated across all of the instances or displayed side by side so you can readily compare instances.
- Monitor cluster cache coherency statistics to help you identify processing trends and optimize performance for your RAC environment.
- Perform cluster database-wide operations, including initiating back up and recovery operations, starting and stopping instances, and so on.

Enterprise Manager accumulates data over specified periods of time, called collection-based data. Enterprise Manager also provides current data, known as real-time data. The following sections explain how to monitor both types of data.

### Collection-Based Monitoring

Enterprise Manager responds to metrics from a RAC database and publishes alerts when thresholds are exceeded. The following sections explain metrics, alerts, and thresholds.

#### Metrics

Metrics are units of measurement used to determine the health of a target. It is through the use of metrics and associated thresholds that Enterprise Manager sends out alerts notifying you of problems with the target. A complete list of all the metrics for a target are available by clicking All Metrics on the target’s home page. From the displayed All Metrics page, you can view all of the metrics for the target and then drill down to any particular metric for additional information. Examples of RAC database metric categories are Tablespaces Full, Data Guard Status, and Database Cardinality. Examples of RAC database instance metric categories are Global Cache Statistics, Instance Throughput, and Alert Log.

Enterprise Manager monitors metrics at the database and instance levels, and any available node can monitor database metrics. However, only one node at a time monitors the entire database while each node monitors metrics for its local instance.

#### Alerts

An alert indicates a potential problem. Either a warning or critical threshold for a monitored metric has been crossed, or the target is no longer available. You can receive critical alerts when a monitored metric has crossed its critical threshold, or warning alerts when a monitored metric has crossed its warning threshold. Also, you can receive alerts on various target availability states, such as when the:
- Target is down
- Target is blacked out
- Oracle Management Agent that is monitoring the target is unreachable

An alert can also signal that a warning or critical threshold has been crossed. You can define a response action to be executed if a metric reaches the level for warnings or critical severity alerts once a threshold is crossed.

When you receive an alert, you can use Enterprise Manager to obtain more information to help you decide how to address the issue. For example, using the Warning Alerts page, a message for a target host could state that the CPU Utilization is 93.925667%. From this description, you can decide how to best redistribute the CPU load.

**Thresholds**

Thresholds are boundary values against which monitored metric values are compared. You can specify a warning threshold so that when a monitored metric value crosses that threshold, a warning alert is generated. This alert then notifies you about impending problems that you can address in a timely manner.

If a threshold is exceeded for a metric, then an alert appears on the Cluster Database Home page. You can also copy thresholds from one instance to other instances in Database Control and Grid Control, or you can copy thresholds between different RAC databases in Grid Control. Database Control does not manage multiple RAC databases.

**Real-Time Performance Monitoring**

Enterprise Manager provides performance pages to monitor all levels of a cluster environment, including the cluster, the cluster database, and the cluster database instances. Enterprise Manager uses data from the Automatic Workload Repository (AWR) to display performance information and initiate database alerts. Statistics collected by AWR are aggregated for all instances in a RAC database and displayed on a summary Enterprise Manager Performance page.

Oracle Enterprise Manager provides several real-time performance charts and drilldowns for the targets that you manage. Enterprise Manager displays both aggregate and instance-specific performance statistics using color-coded charts for easier viewing. To help you identify the source of a problem and resolve it, you can click a legend link next to a chart to display a detail page that provides comprehensive information.

The Cluster Database Home page is your entry point for monitoring Enterprise Manager. From that page, you can access the following RAC-specific performance pages that show performance trends across one or more cluster databases:

- Cluster Database Performance Page
- Cluster Database Instance Performance Page
- Cluster Performance Page

The following sections describe each of these pages as well as other related pages.
Using the Cluster Database Performance Page

The Cluster Database Performance page provides a quick glimpse of the performance statistics for a database. Statistics are rolled up across all of the instances in the cluster. Using links that are next to the charts at the bottom of the page, you can drill down to:

- 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

Figure 12–1 shows the Cluster Database Performance page that you can access by clicking the Performance tab from the Cluster Database Home page.
The following sections provide a description for several important charts, performance monitoring pages, and diagnostic pages.
Cluster Host Load Average Chart
The Cluster Host Load Average chart in the Cluster Database Performance page shows potential problems that are outside of the database. The chart shows maximum, average, and minimum values for available hosts for the previous hour. These values appear at the same level for a cluster that has an evenly balanced load. For Windows-based systems, CPU usage is displayed.

If the load average is higher than the average of the total number of CPUs across all of the hosts in the cluster, then too many processes are waiting for CPU resources. SQL statements that are not tuned often cause high CPU usage. Compare the load average values with the values displayed for CPU Used in the Average Active Sessions chart. If the sessions value is low and the load average value is high, then this indicates that something else on the host, other than your database, is consuming the CPU.

Global Cache Block Access Latency Chart
The Global Cache Block Access Latency summary chart in the Cluster Database Performance page shows the end-to-end elapsed time or latency for a block request.

If accessing a database block of any class does not locate a buffered copy in the local cache, a global cache operation is initiated. Before reading a block from disk, an attempt is made to find the block in the buffer cache of another instance. If the block is in another instance, a version of the block may be shipped. Two different types of blocks are distinguished: current and consistent read blocks.

Long latencies can be caused by:

- A high number of requests caused by SQL statements that are not tuned.
- A large number of processes in the run queue waiting for CPU or scheduling delays.
- Platform-specific operating system parameter settings that affect IPC buffering or process scheduling.
- Slow, busy, or faulty interconnects. In these cases, look for dropped packets, retransmits, or cyclic redundancy check (CRC) errors. Ensure that the network is private and that inter-instance traffic is not routed through a public network. You can access the Cluster Interconnects page from the Cluster Cache Coherency page to monitor the transfer and related statistics for interconnects.

Concurrent read and write activity on shared data in a cluster is a frequently occurring activity. Depending on the service requirements, this activity does not normally cause performance problems. However, when global cache requests cause a performance problem, optimizing SQL plans and the schema to achieve effective local cache hit efficiency and minimize I/O is a successful strategy for performance tuning. If the value for CR and current block request time reaches 10 ms, then your first step in resolving the problem should be to drill down to the Cluster Cache Coherency page for more detailed information.

To access the Cluster Cache Coherency page, click Cluster Cache Coherency in the Additional Monitoring Links section of the Performance page. You can alternatively click either of the legends to the right of the Global Cache Block Access Latency chart. The Cluster Cache Coherency page appears, as shown in Figure 12–2, which contains summary charts for cache coherency metrics for the cluster.
Table 12-1 provides a description of the Cluster Cache Coherency charts and their drilldowns for accessing more comprehensive information for problem resolution.

For additional information about interpreting these charts and metrics, click Help in the Cluster Cache Coherency page.
Average Active Sessions Chart

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 waiting for 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.

The chart displays the load on the instance and identifies performance bottlenecks. At the cluster database level, this chart shows the aggregate wait class statistics across all of the instances.

The Session Count is computed by categorizing the time that all active sessions consumed in the last sampling interval into different wait classes, summing the amount in each wait class, and dividing it by the sampling interval. For example, if there are 3 active sessions in the last sampling interval (15 seconds), and each session spent half of the time on CPU (7.5 seconds each) and half of the time idling, then 3 x 7.5 / 15 = 1.5 active sessions on CPU appear for that sampling interval.

Compare the peaks on the Average Active Sessions chart with those on the Database Throughput charts. If the Average Active Sessions chart displays a large number of sessions waiting, indicating internal contention, but throughput is high, then the situation may be acceptable. The database is probably also performing efficiently if internal contention is low but throughput is high. However, if internal contention is high but throughput is low, then consider tuning the database.

The wait class legends beside the Average Active Sessions chart enable you to drill down to instance-level information stored in Active Sessions by Instance pages. These pages show the service times for up to four top instances. From these pages, you can select an instance of interest to see wait events, top SQL, top sessions, top services, top modules, and top actions if you need to diagnose and fix problems causing higher waits in a specific category.
Database Throughput Charts
The Database Throughput charts summarize any contentions that appear in the Average Active Sessions chart, and also show how much work the database is performing on behalf of the user. The Per Second view is for databases that handle SQL queries, shown as Physical Reads in the bottom chart. The Per Transaction view is for databases that handle transactions, shown as Transactions in the top chart and Redo Size in the bottom chart. Logons show how many users are logged onto the database per second.

You can also obtain information at the instance level by clicking one of the legends to the right of the charts to access the Database Throughput by Instance page. This page shows the breakdown for all active instances of the rolled-up Database Throughput chart on the Cluster Database Performance page. You can use this page to view the throughput for a particular instance, which may help you diagnose throughput problems.

You can drill down further to see the sessions of an instance consuming the greatest resources. Click an instance name legend just below the chart to go to the Top Sessions page. For more information on this page, see “Top Sessions Page” on page 12-15.

Top Consumers Page
The Top Consumers page provides access to several tabs that enable you to view real-time or collection-based data for the top database consumers (top services, top modules, top clients, and top actions) of system resources. You can access the Top Consumers page by clicking the Top Consumers link in the Additional Monitoring Links section of the Cluster Database Performance page.

By default, the Top Consumers page initially displays the Overview tab, which shows aggregated summary data for all top consumers. For instance-level information about a consumer, click the portion of a chart representing the consumer or click the link below the chart for that consumer. The page that appears shows the running instances that are serving the consumer. You can expand the names in the Module column to show data for individual instances.

Top Sessions Page
The Top Sessions page shows a real-time summary list of sessions based on aggregated data. You can see which sessions have consumed the greatest amount of system resources and then decide whether you want to abort the sessions. You can access the Top Sessions page by clicking the Top Sessions link in the Additional Monitoring Links section of the Cluster Database Performance page.

The summary list shows sessions for all instances. You can narrow the display to a particular instance by clicking the name in the Instance Name column. When you click the name, the Top Sessions page for the database instance appears and lists only sessions that are applicable to that instance.

You can see session details for a user by clicking a number assigned to the user name in the SID (Session ID) column. The Session Details page displays complete information about all aspects of a particular session.

Instance Activity Page
The Instance Activity page enables you to view instance activity for several metrics within general metric categories, such as cursors, transactions, sessions, logical I/O, physical I/O, and net I/O. You can access top sessions statistics for a particular metric by clicking a metric legend below the chart if in Graphic mode, or by clicking a name in the summary table if in Tabular mode. You can also use the Switch Database
Instance drop-down menu to toggle from one database instance to another. You can view data on a per-second or per-transaction basis. You can access this page by clicking the **Instance Activity** link in the Additional Monitoring Links section at the bottom of the Cluster Database Performance page.

**Top Segments Page**
Collecting and viewing segment-level statistics is an effective method for identifying hot tables or indexes in a database. The Top Segments page enables you to gather segment-level statistics to identify performance problems associated with individual segments. This page is particularly useful for RAC, because it also tracks the number of CR and current blocks received by an object. A high number of current blocks received plus a high number of buffer waits may indicate potential contention.

You can access the Top Segments page by clicking the **Top Segments** link in the Additional Monitoring Links section. You can view segments for all instances, or use a filter to see segments for a specific instance.

**Database Locks Page**
Use the Database Locks page to determine whether multiple instances are holding locks for the same object. The page shows blocking locks, user locks, or all database locks. You can use this information to abort a session that is unnecessarily locking an object. You can access the Database Locks page by clicking the **Database Locks** link in the Additional Monitoring Links section.

**Using the Cluster Database Instance Performance Page**
To access the Cluster Database Instance Performance page, click the **Performance** tab in the Cluster Database Home page, and then click an instance name at the bottom of the page.

**Page Characteristics**
Statistics are rolled up for the instance. You can drill down from links that appear alongside the charts, or at the bottom of the page, to perform the following tasks:

- Identify the cause of performance issues.
- Decide whether resources need to be added or redistributed to resolve performance issues.
- Tune your SQL plan and schema for better optimization.
- Resolve performance issues.

The Cluster Database Instance Performance page is essentially the same as the Cluster Database Performance page with the following exceptions:

- This page shows performance data for a single instance, rather than enabling you to access data for all instances in a cluster database.
- You can access the Top Activity page, which displays all wait classes and related statistics for Active Sessions, Top SQL, Top Sessions, Top Files, and Top Objects for an instance.
- An Instance Disk I/O chart shows the total requests that represent the rate at which the database instance is issuing read/write requests.
- You can run Automatic Database Diagnostic Monitor (ADDM) and Active Session History (ASH) reports for an instance.
Creating Reports
The Cluster Database Instance Performance page provides the means for running ADDM and ASH reports.

- **ADDM Report**—enables you to create a new AWR snapshot and run ADDM on this and the previous snapshot. After you run ADDM, you can drill down for a detailed performance analysis and report page.

- **ASH Report**—enables you to create a performance data report of the database based on session-sampling data over a specified period of time. This report is very useful for diagnosing small (five- to ten-minute) performance spikes that might be averaged out and consequently hidden or minimized by other issues in the 1-hour AWR report.

Using the Cluster Performance Page
To access the Cluster Performance page, click the **Cluster** link in the General section of the Cluster Database Performance page, and then click the **Performance** tab from the Cluster Home page.

The Cluster Performance Page displays utilization statistics, such as CPU, Memory, and Disk I/O, during the past hour for all hosts of a cluster, which is part of the greater Enterprise Manager environment. With this information, you can determine whether resources need to be added, suspended, or redistributed.

You can view the CPU, Memory, and Disk I/O charts for each host individually by clicking the host name in the legend to the right of the charts. The resulting Host page also shows statistics for CPU load, memory scan page rate, and longest service time.

Using the Cluster Interconnects Page
The Cluster Interconnects page enables you to monitor the interconnect interfaces, determine configuration issues, and identify transfer rate-related issues including excess traffic. This page helps determine the load added by individual instances and databases on the interconnect. Sometimes you can immediately identify interconnect delays that are due to applications that are outside Oracle Database.

To access this page, click the **Cluster** link in the General section of the Cluster Database Home page, then click the **Interconnects** tab. The Cluster Interconnects page appears, as shown in Figure 12–3.
The Cluster Interconnects page contains the following summary tables:

- Interfaces by Hosts
- Interfaces in Use by Cluster Databases

These tables enable you to perform the following tasks:

- View all interfaces that are configured across the cluster.
- View statistics for the interfaces, such as absolute transfer rates and errors.
- Determine the type of interfaces, such as private or public.
- Determine whether the instance is using a public or private network.
- Determine which database instance is currently using which interface.
- Determine how much the instance is contributing to the transfer rate on the interface.
Note: An incorrect configuration, such as using the public network instead of the private network for the interconnect, or low transfer rates, generates metric alerts.
This chapter explains how you can extend the high availability of the Oracle Clusterware framework to your applications. You do this by wrapping your applications with Oracle Clusterware commands. That is, you can use the same high availability mechanisms of the Oracle database and Real Application Clusters to make your custom applications highly available. You can use the Oracle Clusterware to monitor, relocate, and restart your applications as described in this chapter under the following topics:

- Overview of Managing Custom Applications with Oracle Clusterware Commands
- Creating Application Profiles
- Example of Using Oracle Clusterware Commands to Create Application Resources
- Oracle Clusterware Action Program Guidelines
- Using Oracle Clusterware Commands

See Also: Appendix B, "High Availability Oracle Clusterware Command-Line Reference and C API" for detailed information about Oracle Clusterware commands

Note: You can install the Oracle Clusterware high availability Application Programming Interface (API) from the Oracle Database 10g release 10.2 client installation media. You can also install a demo application that uses the API from the Oracle Database 10g release 10.2 companion installation media. As described in this chapter, you can use the API to integrate non-oracle applications with the Oracle Clusterware to make them highly available.

Overview of Using the Oracle Clusterware Commands to Enable High Availability

The Oracle Clusterware includes a high availability framework that provides an infrastructure to manage any application. The Oracle Clusterware ensures that applications that it manages start when the system starts. The Oracle Clusterware also monitors the applications to make sure that they are always available. For example, if a process fails, then the Oracle Clusterware attempts to restart the process based on scripts that you customize. If a node in the cluster fails, then you can program processes that normally run on the failed node to restart on another node. The
monitoring frequency, starting, and stopping of the applications and the application dependencies are configurable.

To make applications highly available, first create an application profile that identifies your application. The application profile uses a second component, an action program, that describes how the Oracle Clusterware should monitor your application and how the Oracle Clusterware should respond to changes in your application’s status. The definitions of an application profile, action program, and the other primary Oracle Clusterware high availability components are as follows:

- **Action Program**—A program that defines the location of your program, the monitoring frequency, and the start and stop actions that the Oracle Clusterware should perform on the application. The start action starts the application, the stop action stops the application, and the monitoring or check action checks the application’s status.

- **Application Profile**—An Oracle Clusterware resource file that describes the attributes of your application. An application profile influences your application’s behavior and it identifies other programs or scripts that the Oracle Clusterware should run to perform various actions.

- **Privileges**—Access and usage privileges that enable the Oracle Clusterware to control all of the components of your application for high availability operations, including the right to start processes under other user identities. The Oracle Clusterware must run as a privileged user to control applications with the correct start and stop processes. On UNIX-based platforms, this usually implies that the Oracle Clusterware must run as the root user and on Windows-based platforms the Oracle Clusterware must run as Administrator.

- **Resource**—An entity that the Oracle Clusterware manages for high availability such as your application.

  **Note:** A resource in the Oracle Clusterware context is not the same as a resource in the Oracle database sense, such as "Resource Manager". A resource that the Oracle Clusterware uses only refers to application programs.

- **Resource Dependency**—A relationship among resources or applications that implies an operational ordering. For example, during a start operation, parent resources are started before resources that have dependencies. During a stop, the most dependent resources are stopped first.

- **Oracle Cluster Registry (OCR)**—A mechanism that stores configuration information that the Oracle Clusterware and other RAC manageability systems use. The OCR uses a hierarchical namespace for key value pairs. Keys and subkeys have enforced user, group, and other permissions. The OCR is stored either on a shared raw partition or on a non-cached, shared filesystem file.

- **Template**—A user-created text file that contains the default values for application profile attributes. Template files contain default values for profile attributes.

  **See Also:** Appendix B, "High Availability Oracle Clusterware Command-Line Reference and C API" for more information about using Oracle Clusterware commands to make your applications highly available.
Overview of Managing Custom Applications with Oracle Clusterware Commands

You can use the Oracle Clusterware commands to start, stop, relocate, and check the status of your custom applications. Do this by defining your application with an application profile. The profile defines attributes that affect how the Oracle Clusterware manages your application. Then register your application information in the OCR using the `crs_register` command. Use the following steps to create an application profile:

1. Create an application profile by editing an ASCII file or by running the `crs_profile` command. Refer to Table 13–1 for a listing of required and optional entries for the Oracle Clusterware profiles.
2. Register the application profile using the `crs_register` command.
3. Run the `crs_start` command to initiate the application profile and then the Oracle Clusterware runs the action program command that you have included in the profile to start your application.
4. The Oracle Clusterware periodically runs the action program command to check an application’s status.
5. In the event of a check or node failure, the Oracle Clusterware recovers the applications either by restarting it on the current node or by relocating the application to another node.
6. If you run the `crs_stop` command to stop the application, then the Oracle Clusterware runs the action program command to stop it.

You can manage application availability as follows:
- Specify starting resources during cluster or node start up
- Restart applications that fail
- Relocate applications to other nodes if they cannot run in their current location

Full administrative privileges are not required when using the Oracle Clusterware. Any user can create resources or applications. However, the creator or owner must grant permission to other users or user groups in order for others to be able to use the Oracle Clusterware on those applications. Additionally, profiles that have privileges defined can only be modified by privileged users. The following sections provide further details about application profiles.

---

**Note:** Do not use the Oracle Clusterware commands `crs_register`, `crs_profile`, `crs_start` or `crs_stop` on resources with names beginning with the prefix `ora` unless either Oracle Support asks you to, or unless Oracle has certified you as described in [http://metalink.oracle.com](http://metalink.oracle.com). You can, however, create resources that depend on resources that Oracle has defined. In addition, you can use the Oracle Clusterware commands to inspect the configuration and status.

---

Creating Application Profiles

Application profiles have attributes that define how the Oracle Clusterware starts, manages, and monitors applications. One attribute is the location of the action program that the Oracle Clusterware uses to manipulate the application. The Oracle
Clusterware uses the action program to monitor or check the application status and to start and stop it. Oracle reads application profiles from files stored in specified locations and stores the information in the OCR. You use the Oracle Clusterware commands within profiles to designate resource dependencies and to determine what happens to an application or service when it loses access to a resource on which it depends.

The filenames of application profiles must be in the form `resource_name.cap` where `resource_name` is the name that you or the system assigns to an application and `cap` is the file suffix. The Oracle Clusterware commands in profiles refer to applications by name, such as `resource_name`, but not by the full filename. The following section describes this in more detail.

**Application Resource Profiles**

Attributes are defined by `name=value` entries in profile files and these entries can be in any order in the file. The following are some of the primary attributes of an application profile:

- Resources that are required by an application which are defined by settings for the `REQUIREDResources` parameter. The Oracle Clusterware relocates or stops an application if a required resource becomes unavailable. Required resources are defined for each node.

- Rules for choosing the node on which to start or restart an application are defined by settings for the `PLACEMENT` parameter. The application must be accessible by the nodes that you nominate for placement.

- A list of nodes to use in order of preference when the Oracle Clusterware starts or fails over an application which is defined by settings for the `HOSTING_MEMBERS` parameter. This list is used if the placement policy defined by the `PLACEMENT` parameter is favored or restricted.

**Required and Optional Profile Attributes**

Application profiles have optional and required profile attributes. Optional profile attributes may be left unspecified in the profile. Optional profile attributes that have default values are merged at registration time with the values that are stored in the template for that resource type and for the generic template. Default values are derived from the template.

Each resource type has a template file named `TYPE_resource_type.cap` that is stored in the template subdirectory under the `crs` directory of the Oracle Clusterware home. A generic template file for values that are used in all types of resources is stored in the same location in the file named `TYPE_generic.cap`.

**Application Profile Attributes**

Table 13–1 lists the Oracle Clusterware application profile attributes in alphabetical order. For each attribute, the table shows whether the attribute is required, its default value, and an attribute description.
### Table 13–1  Application Profile Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Required</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTION_SCRIPT</td>
<td>Yes</td>
<td>None</td>
<td>The resource-specific script for starting, stopping, and checking a resource. You may specify a full path for the action program file. Otherwise, the default paths are used: CRS_HOME/crs/script for privileged, and CRS_HOME/crs/public for public. You may also specify a relative path with this default path as the starting point.</td>
</tr>
<tr>
<td>ACTIVE_PLACEMENT</td>
<td>No</td>
<td>0</td>
<td>When set to 1, the Oracle Clusterware re-evaluates the placement of a resource during addition or restart of a cluster node.</td>
</tr>
<tr>
<td>AUTO_START</td>
<td>No</td>
<td>0</td>
<td>A flag indicating whether the Oracle Clusterware should automatically start a resource after a cluster restart, regardless of whether the resource was running prior to the cluster restart. When set to 0, the Oracle Clusterware starts the resource only if it had been running before the restart. When set to 1, the Oracle Clusterware always starts the resource after a restart.</td>
</tr>
<tr>
<td>CHECK_INTERVAL</td>
<td>No</td>
<td>60</td>
<td>The time interval, in seconds, between repeated executions of the check entry point of a resource's action program. There can be some overhead associated if you set the check interval to a low value and enable frequent checks.</td>
</tr>
<tr>
<td>DESCRIPTION</td>
<td>No</td>
<td>Name of the resource</td>
<td>A description of the resource.</td>
</tr>
<tr>
<td>FAILOVER_DELAY</td>
<td>No</td>
<td>0</td>
<td>The amount of time, in seconds, that the Oracle Clusterware waits before attempting to restart or fail over a resource.</td>
</tr>
<tr>
<td>FAILURE_INTERVAL</td>
<td>No</td>
<td>0</td>
<td>The interval, in seconds, during which the Oracle Clusterware applies the failure threshold. If the value is zero (0), then tracking of failures is disabled.</td>
</tr>
<tr>
<td>FAILURE_THRESHOLD</td>
<td>No</td>
<td>0</td>
<td>The number of failures detected within a specified FAILURE_INTERVAL before the Oracle Clusterware marks the resource as unavailable and no longer monitors it. If a resource's check script fails this number of times, then the resource is stopped and set offline. If the value is zero (0), then tracking of failures is disabled. The maximum value is 20.</td>
</tr>
<tr>
<td>HOSTING_MEMBERS</td>
<td>Sometimes</td>
<td>None</td>
<td>An ordered list of cluster nodes separated by blank spaces that can host the resource. This attribute is required only if PLACEMENT equals favored or restricted. This attribute must be empty if PLACEMENT equals balanced. Enter node names as values for the HOSTING_MEMBERS attribute, not virtual host names or physical host names. Use the node names that you used when you installed the Oracle Clusterware. The resources that you mention should contain the node name for the node on which they run. Run the olsnodes commands to see your node names. The HOSTING_MEMBERS Oracle Clusterware attribute is set automatically when these Oracle Clusterware resources are created; you do not need to take further action to ensure that the attribute is set. The node name is usually the same as the physical host name. However, it can be different. For example, when vendor clusterware is present, the Oracle Clusterware nodes are named the same as the vendor clusterware nodes. Not all vendor clusterware implementations use the physical node names as node names. Use the lsnodes command to display vendor clusterware node names. When there is no vendor clusterware, then the Oracle Clusterware node names must be the same as the physical hostname. Your Listener.ora file should contain one entry for the virtual internet protocol (VIP) address, using the VIP's name, and another entry for the physical host, by IP address not name.</td>
</tr>
</tbody>
</table>

Your Listener.ora file should contain one entry for the virtual internet protocol (VIP) address, using the VIP’s name, and another entry for the physical host, by IP address not name.
Creating Application Profiles

**Default Profile Locations**

Profiles may be located anywhere and need not be on a cluster-visible file system. RAC provides default locations for profiles as described in the next sub-section. The default location for profiles with root privileges on UNIX-based systems or Administrator privileges on Windows-based systems is the profile subdirectory under the crs directory of the Oracle Clusterware home. The default location for profiles with non-root or non-Administrator privileges is the public subdirectory under the crs directory of the Oracle Clusterware home.

---

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Required</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>Yes</td>
<td>None</td>
<td>The name of the application. The application name is a string that contains a combination of letters a-z or A-Z, and digits 0-9. The naming convention is to start with an alphanumeric prefix, such as ora, and complete the name with an identifier for to describe it. The name can contain any platform-supported characters except the exclamation point (!). However, the application name cannot begin with a period.</td>
</tr>
<tr>
<td>OPTIONAL_RESOURCES</td>
<td>No</td>
<td>None</td>
<td>An ordered list of resource names separated by blank spaces that this resource uses during placement decisions. Up to 58 user-defined resources can be listed.</td>
</tr>
<tr>
<td>PLACEMENT</td>
<td>No</td>
<td>balanced</td>
<td>The placement policy (balanced, favored, or restricted) specifies how the Oracle Clusterware chooses the cluster node on which to start the resource.</td>
</tr>
<tr>
<td>REQUIRED_RESOURCES</td>
<td>No</td>
<td>None</td>
<td>An ordered list of resource names separated by blank spaces that this resource depends on. Each resource to be used as a required resource in this profile must be registered with the Oracle Clusterware or the resource’s profile registration will fail.</td>
</tr>
<tr>
<td>RESTART_ATTEMPTS</td>
<td>No</td>
<td>1</td>
<td>The number of times that the Oracle Clusterware attempts to restart a resource on a single cluster node before attempting to relocate the resource. A value of 1 means that the Oracle Clusterware only attempts to restart the resource once on a node. A second failure causes an attempt to relocate the resource.</td>
</tr>
<tr>
<td>RESTART_COUNT</td>
<td></td>
<td></td>
<td>The counter maintained by the Oracle Clusterware daemon for the number of times that a resource had been restarted. It goes from zero to RESTART_ATTEMPTS. This is also written to OCR.</td>
</tr>
<tr>
<td>SCRIPT_TIMEOUT</td>
<td>No</td>
<td>60</td>
<td>The maximum time, in seconds, that an action program may require to complete before an error is returned.</td>
</tr>
<tr>
<td>SCRIPT_TIMEOUT</td>
<td></td>
<td></td>
<td>This attribute specifies the timeout value for the &quot;check&quot; action of the action script. If the check action does not return before this time, the Oracle Clusterware will consider the action failed.</td>
</tr>
<tr>
<td>START_TIMEOUT</td>
<td></td>
<td></td>
<td>This attribute specifies the timeout value for the &quot;start&quot; action of the action script. If the start action does not return before this time, the Oracle Clusterware will consider the action failed.</td>
</tr>
<tr>
<td>STOP_TIMEOUT</td>
<td></td>
<td></td>
<td>This attribute specifies the timeout value for the &quot;stop&quot; action of the action script. If the stop action does not return before this time, the Oracle Clusterware will consider the action failed.</td>
</tr>
<tr>
<td>TYPE</td>
<td>Yes</td>
<td>None</td>
<td>Must be APPLICATION.</td>
</tr>
<tr>
<td>UPTIME_THRESHOLD</td>
<td></td>
<td></td>
<td>The value for UPTIME_THRESHOLD represents the length of time that a resource must be up before the Oracle Clusterware considers the resource to be stable. By setting a value for the UPTIME_THRESHOLD attribute, you can indicate a resource's stability.</td>
</tr>
</tbody>
</table>
Example of Using Oracle Clusterware Commands to Create Application Resources

The example in this section creates an application named postman. The Oracle Clusterware uses the script /opt/email/bin/crs_postman to start, stop, and monitor whether the application is running (action_script). The Oracle Clusterware checks postman every five seconds as specified by the setting for the check_interval parameter. The Oracle Clusterware restarts postman no more than once if it fails. When deciding on which node to place the postman application, the Oracle Clusterware considers the value for the optional_resources parameter. If possible, the Oracle Clusterware places postman on the same node. Finally, in order for postman to run, the resource network1 must be running on the same node as specified by the setting for the required_resources parameter. If network1 fails or if it is relocated to another node, then the Oracle Clusterware stops or moves the postman application.

Note: Do not use the Oracle Clusterware commands crs_register, crs_profile, crs_start or crs_stop on resources with names beginning with the prefix ora unless either Oracle Support asks you to, or unless Oracle has certified you as described in http://metalink.oracle.com. You can, however, create resources that depend on resources that Oracle has defined. In addition, you can use the Oracle Clusterware commands to inspect the configuration and status.

Using crs_profile to Create An Application Resource Profile

The following example uses the crs_profile command to create an application profile for monitoring email.

```
# crs_profile -create postman -t application -B /opt/email/bin/crs_postman -d "Email Application" -r network1 -l application2 -a postman.scr -o ci=5,ft=2,fi=12,ra=2
```

The contents of the application profile file that the example creates are as follows:

```
NAME=postman
TYPE=application
ACTION_SCRIPT=/oracle/crs/script/postman.scr
ACTIVE_PLACEMENT=0
AUTO_START=0
CHECK_INTERVAL=5
DESCRIPTION=email app
FAILOVER_DELAY=0
FAILURE_INTERVAL=12
FAILURE_THRESHOLD=2
HOSTING_MEMBERS=
OPTIONAL_RESOURCES=application2
PLACEMENT=balanced
REQUIRED_RESOURCES=network1
RESTART_ATTEMPTS=2
SCRIPT_TIMEOUT=60
```

See Also: Appendix B, "High Availability Oracle Clusterware Command-Line Reference and C API" for an explanation of the command options.
The Oracle Clusterware Required Resources List

The Oracle Clusterware uses the required resources list, with the placement policy and hosting nodes list, to determine the cluster nodes that are eligible to host an application. Required resources must be **ONLINE** on the nodes on which the application is running or started.

The failure of a required resource on a hosting node causes the Oracle Clusterware to attempt to restart the application on the current node. If **RESTART_ATTEMPTS** is not set to 0, and if the application cannot start on the current node, then the Oracle Clusterware attempts to fail the application over to another node that provides the required resource. Alternatively, the Oracle Clusterware stops the application if there is no suitable node. In this case, the Oracle Clusterware posts a **not restarting** event notification.

You can also use Required resource lists to start, stop, and relocate a group of interdependent applications when the `crs_start`, `crs_stop`, or `crs_relocate` commands are run with the force (`-f`) option.

If your application is accessed by way of a network, then Oracle recommends that you create a virtual internet protocol address for the application as a dependent resource.

In the previous example, `network1` is an application VIP address. Create all application VIP addresses as follows:

```bash
crs_profile --create network1 -t application \ 
-a $CRS_HOME/bin/usrvip \ 
-o oi=eth0,ov=138.3.83.78,on=255.255.240.0
```

In this example, `$CRS_HOME` is the home directory for the Oracle Clusterware installation. In addition, `eth0` is the name of the public network adapter, `138.3.83.78` resolves by way of DNS to a new IP address that will be used to locate your application regardless of the node it is running on, and `255.255.240.0` is the netmask for the public IP address. As the **oracle** user register the VIP address with the Oracle Clusterware as follows:

```bash
crs_register network1
```

On UNIX-based operating systems, the application VIP address script must run as the **root** user. As the **root** user, change the owner of the resource as follows:

```bash
crs_setperm network1 -o root
```

As the **root** user, enable the **oracle** user to run this script:

```bash
crs_setperm network1 -u user:oracle:r-x
```

As the **oracle** user, start the VIP address as follows:

```bash
crs_start network1
```

Application Placement Policies

The placement policy specifies how the Oracle Clusterware selects a node on which to start an application and where to relocate the application after a node failure. Only cluster nodes on which all of the required resources are available, as listed in an application's profile, are eligible to be considered as hosting nodes for an application. The Oracle Clusterware supports the following placement policies:

- **balanced**—The Oracle Clusterware favors starting or restarting the application on the node that is currently running the fewest resources. A placement that is
based on optional resources is considered first. Next, the host with the fewest
resources running is chosen. If no node is favored by these criteria, then any
available node is chosen.

- **favored**—The Oracle Clusterware refers to the list of nodes in the HOSTING_ MEMBERS attribute of the application profile. Only cluster nodes that are in this list and that satisfy the resource requirements are eligible for placement consideration. Placement due to optional resources is considered first. If no node is eligible based on optional resources, then the order of the hosting nodes determines which node runs the application. If none of the nodes in the hosting node list are available, then the Oracle Clusterware places the application on any available node. This node may or may not be included in the HOSTING_MEMBERS list.

- **restricted**—Similar to favored except that if none of the nodes on the hosting list are available, then the Oracle Clusterware does not start or restart the application. A restricted placement policy ensures that the application never runs on a node that is not on the list, even if you manually relocate it to that node.

You must specify hosting nodes in the HOSTING_MEMBERS attribute to use a favored or restricted placement policy. Do not specify hosting nodes in the HOSTING_MEMBERS attribute with a balanced placement policy. Otherwise, the application will not validate and you cannot register it. If ACTIVE_PLACEMENT is set to 1, then the placement of the application is re-evaluated whenever you add a node to the cluster or if the cluster node restarts. This enables the Oracle Clusterware to relocate applications to a preferred node after the node recovers from a failure. To have the Oracle Clusterware relocate an application to a preferred node at a time other than when the node rejoins the cluster, use the REBALANCE attribute to specify a time at which placement can be re-evaluated.

### Optional Resources in Placement Decisions

The Oracle Clusterware uses optional resources to choose a hosting node based on the number of optional resources that are in an ONLINE state on the hosting node. If each node has an equal number of optional resources in an ONLINE state, then the Oracle Clusterware considers the order of the optional resources as follows:

- The Oracle Clusterware compares the state of the optional resources on each node starting at the first resource that you list in the application profile and then proceeds through the list.

- For each consecutive resource in your list, if the resource is ONLINE on one node, then any node that does not have the resource ONLINE is not considered.

- The Oracle Clusterware evaluates each resource in the list in this manner until only one node is available to host the resource.

- The maximum number of optional resources is 58.

If this algorithm results in multiple preferred nodes, then the resource is placed on one of these nodes chosen according to its placement policy.

### Oracle Clusterware Action Program Guidelines

This section provides the following guidelines for writing Oracle Clusterware action programs that interpret the Oracle Clusterware start, stop, and check commands:

- The Oracle Clusterware relies on a status code upon exiting from an action program to set the resource state to ONLINE or OFFLINE. On Windows-based
systems, the program should be non-blocking, which may imply a Windows service or a Windows resource that does not block during console interactions.

- Action programs must return a status code to indicate success or failure. For example, on UNIX-based systems if the script is written in Bourne Shell, then the program should issue `exit(1)` to indicate failure, and `exit(0)` to indicate success. Similarly, on Windows-based systems, the action program should return a status of (0) to indicate success and (1) to indicate failure.

- After application failure, the Oracle Clusterware calls the action program with a check parameter. The action program replies to the Oracle Clusterware with a status of (1) to indicate failure. After receiving this failure status, the Oracle Clusterware calls the action program with a stop parameter. It is important that the action program returns a status of (0) to indicate a successful stop of the application, even though the application was not running when the stop request was called.

- The Oracle Clusterware sets the resource state to `UNKNOWN` if an action program’s stop entry point fails to exit within the number of seconds in the `SCRIPT_TIMEOUT` value, or if the action program returns with a code that indicates failure. This may occur during a start, relocation, or stop operation. Ensure that the action program’s stop entry point exits with a value that indicates success if the resource is successfully stopped or if the resource is not running.

- When a daemon or service starts, it usually needs to start as a background process, depending on the platform. However, a resource started in this way always returns success during a start attempt. This means that the default scripts cannot detect failures caused by minor errors, such as misspelled command paths.

- On UNIX-based systems, if a resource does not move to the background immediately upon startup, then you can start the application in the background by adding an ampersand (`&`) to the end of the line that starts the application.

- On Windows-based systems, you can use `net start` to start a service that needs to start in the background.

- When using commands to start daemons or services in the background, interactively make test runs of the commands used in the script to eliminate errors before using the script with the Oracle Clusterware.

### How the Oracle Clusterware Runs Action Programs

This section describes how the Oracle Clusterware runs action programs. The first argument to an action program is the command `start`, `stop`, or `check` depending on which action the Oracle Clusterware is running. The second argument is the Oracle Clusterware resource name of the application. This enables a script to determine which instance of the resource that the Oracle Clusterware is starting, stopping, or checking.

An action program can retrieve any of its Oracle Clusterware resource attributes from the environment by using `$_CAA_attribute_name`. For example, `$_CAA_NAME` contains the application name, the second argument to the script, and `$_CAA_HOSTING_MEMBERS` contains its `HOSTING_MEMBERS` attribute.

The Oracle Clusterware supports user-defined attributes in the Oracle Clusterware applications, which are attributes having names that contain `USR`. User-defined attributes are stored as part of the Oracle Clusterware application profile for the application. You can reference them in an action program using `$_USR_attributename`. 
To add a user-defined attribute, add it to the file CRS home/crs/template/application.tdf using the following syntax:

```#
# an example user-defined attribute
#
#!===========================
attribute: USR_EXAMPLE
  type: string
  switch: -o example
  default:
  required: no
```

The attribute parameter contains the name of the new attribute. The type parameter defines the type of the user-defined attribute and can be one of the following:

- string
- boolean
- integer—a numeric attribute
- positive_integer—a numeric attribute with must be positive
- name string
- name_list—a comma-delimited list of names

The switch parameter describes how the attribute is specified for the crs_profile command. Set the required field to no for user-defined attributes.

---

**Note:** User-defined attribute names that begin with USR_ORA are reserved for use by Oracle.
---

### Windows crsuser Program

This section describes the Windows-based crsuser program. If you create an Oracle Clusterware application, then run the following command:

```crsuser add [domain]\username```

Provide the user's Windows password. This creates the service OracleCRSToken_user that the Oracle Clusterware needs for starting the Oracle Clusterware resources under the given user ID. You can also use the crsuser command remove [domain]\username to remove a token service and crsuser list to list a registered users.

### Using Oracle Clusterware Commands

This section describes how to use the Oracle Clusterware commands under the following topics:

- Registering Application Resources
- Starting Application Resources
- Relocating Applications and Application Resources
- Stopping Applications and Application Resources
- Unregistering Applications and Application Resources
Displaying Clusterware Application and Application Resource Status Information

Registering Application Resources

Each application that you manage with the Oracle Clusterware must have an application profile and the profile must be registered in the OCR. Use the `crs_register` command to register applications in the OCR. For example, enter the following command to register the mail monitoring application from the previous example:

```bash
# crs_register postman
```

If you modify a profile, then update the OCR by re-running the `crs_register -u` command.

Starting Application Resources

To start an application resource that is registered with the Oracle Clusterware, use the `crs_start` command. For example:

```bash
# crs_start postman
```

The following text is an example of the command output:

```
Attempting to start `postman` on node `rac1`
Start of `postman` on node `rac1` succeeded.
```

The application now runs on the node named `rac1`.

---

**Note:** The name of the application resource may or may not be the same as the name of the application.

---

**See Also:** Appendix B, "High Availability Oracle Clusterware Command-Line Reference and C API" for examples of Oracle Clusterware command output

The command waits for the amount of time specified by the setting for the `SCRIPT_TIMEOUT` parameter to receive a notification of success or failure from the action program each time the action program is called. Application resources can be started if they have stopped due to exceeding their failure threshold values. You must register a resource with `crs_register` before you can start it.

To start and stop resources, use the `crs_start` and `crs_stop` commands. Manual starts or stops outside of the Oracle Clusterware can invalidate the resource status. In addition, the Oracle Clusterware may attempt to restart a resource on which you perform a manual stop operation.

All required resources must be online on the node where you start the resource. If the resources that the `REQUIRED_RESOURCES` parameter identifies are offline, then the command `crs_start -f resource_name` will start the required resources before starting the resource. Without the `-f` option, `crs_start -f resource_name` will fail if the required resources are offline.

Running the `crs_start` command on a resource sets the resource target value to `ONLINE`. The Oracle Clusterware attempts to change the state to match the target by running the action program with the `start` parameter. When a resource is running, both the target state and current state are `ONLINE`. 
Starting an Application on an Unavailable Node

When starting an application on a cluster node that is unavailable, `crs_start` can give indeterminate results. In this scenario, the start section of the action program is run, but the cluster node fails before notification of the start is displayed on the command line. The `crs_start` command returns a failure with the error `Remote start for resource_name failed on node node_name`. The application is actually ONLINE but fails over to another node making the application appear as though it were started on the incorrect node.

If a cluster node fails while you are starting a resource on that node, then check the state of the resource on the cluster by using the `crs_stat` command to determine the state of that resource.

Relocating Applications and Application Resources

Use the `crs_relocate` command to relocate applications and application resources. For example, to relocate the mail monitoring application to the node known as `rac2`, enter the following command:

```
# crs_relocate postman -c rac2
```

Each time that the action program is called, the `crs_relocate` command waits for the duration identified by the value for the `SCRIPT_TIMEOUT` parameter to receive notification of success or failure from the action program. A relocation attempt fails if:

- The application has required resources that are ONLINE on the initial node
- Applications that require the specified resource are ONLINE on the initial node

To relocate an application and its required resources, use the `-f` option with the `crs_relocate` command. The Oracle Clusterware relocates or starts all resources that are required by the application regardless of their state.

Stopping Applications and Application Resources

To stop applications and application resources, use the `crs_stop` command. Immediately after the `crs_stop` command completes, the application status converts to OFFLINE. Because the Oracle Clusterware always attempts to match a resource's state to its target, the Oracle Clusterware subsystem stops the application. The following example stops the mail application from the example:

```
# crs_stop postman
```

The following text is an example of the command output:

```
Attempting to stop `postman` on node `rac1`
Stop of `postman` on node `rac1` succeeded.
```

You cannot stop an application if the application is a required resource for another online application unless you use the force (`-f`) option. If you use the `crs_stop -f resource_name` command on an application that is required by other resources and if those resources are online, then the Oracle Clusterware stops the application. In addition, all of the resources that require that application that are online are also stopped.

Note: The Oracle Clusterware can only stop applications and application resources. The Oracle Clusterware cannot stop network, tape, or media changer resources.
Managing Automatic Oracle Clusterware Resource Operations for Action Scripts

The following section explains additional information for controlling how the Oracle Clusterware manages restarts. You can prevent the Oracle Clusterware from automatically restarting a resource by setting several action program attributes. You can also control how the Oracle Clusterware manages the restart counters for your action programs. In addition, you can customize the timeout values for the start, stop, and check actions that the Oracle Clusterware performs on action scripts. These topics are described under the following headings:

- Preventing Automatic Database Instance Restarts
- Automatically Manage Restart Attempts Counter for Resources
- Implications of Restart and Timeout Features for Previous Releases

Preventing Automatic Database Instance Restarts

When a node stops and restarts, the Oracle Clusterware starts the instance as soon as the node starts. This may not be desirable because instance startup might fail if system components on which the instance depends, such as a volume manager or a file system, are not running. This is especially true if the Oracle Clusterware does not manage the system component resources on which the instance depends. To manage automatic restarts, you can use the AUTO_START attribute to specify whether the Oracle Clusterware should automatically start a resource when a node restarts. Valid AUTO_START values are:

- **always**—Causes the resource to restart when the node restarts regardless of the resource’s state when the node stopped.
- **restore**—Does not start the resource at restart time if it was in an offline state, such as STATE=OFFLINE, TARGET=OFFLINE, when the node stopped. The resource is restored to its state when the node went down. The resource is started only if it was online before and not otherwise.
- **never**—The Oracle Clusterware never restarts the resource regardless of the resource’s state when the node stopped.

---

**Note:** Oracle only supports lower-case values for **always**, **restore**, and **never**.

Automatically Manage Restart Attempts Counter for Resources

When a resource fails, the Oracle Clusterware restarts the resource for only the number of times specified in the profile attribute RESTART_ATTEMPTS regardless of how often the resource fails. The CRSD process maintains an internal counter to track how often a resource has been restarted. There is a mechanism by which the Oracle Clusterware can automatically manage the restart attempts counter based on the stability of a resource. Use the UPTIME_THRESHOLD attribute to indicate resource stability.

You can specify the time for the UPTIME_THRESHOLD attribute in different units of measure, such as seconds (s), minutes (m), hours (h), days (d) or weeks (w). Examples of valid values for this attribute are: 7d for seven days, 5h for five hours, 180m for 180 minutes, and so on. Specify the time period as a numeric value and the unit of measure as the last character, s, m, h, d, or w.

After the time period that you have indicated by the setting for UPTIME_THRESHOLD has elapsed, the Oracle Clusterware resets the value for RESTART_COUNTS to 0. The Oracle Clusterware can alert you when the value for RESTART_COUNT reaches the value that you have set for RESTART_ATTEMPTS.
RESTART_ATTEMPTS and RESTART_COUNT Failure Scenarios

Some of the failure scenarios for the RESTART_ATTEMPTS and RESTART_COUNT attributes are:

- When a resource keeps restarting—The resource does not meet the uptime threshold criteria and it will be stopped after restarting for the number of attempts set by the value for RESTART_ATTEMPTS.

- When a node fails or restarts—The Oracle Clusterware resets the value for RESTART_COUNTER to 0 either when the resource relocates or when it restarts on the same node.

- If the crsd process fails—Because both RESTART_COUNT and RESTART_ATTEMPTS are stored in OCR, the behavior is not affected.

Implications of Restart and Timeout Features for Previous Releases

If you have a pre-Oracle Database 10g release 2 (10.2) installation, add the release 2 (10.2) attributes to your profiles by doing one of the following:

- Modify your resources with the release 2 (10.2) attributes and re-register the resources. This causes the separate timeouts to be effective.

- Do not modify your resources. This retains the pre-release 2 (10.2) behavior of using the value in SCRIPT_TIMEOUT as the timeout for all start, stop, and check actions.

Unregistering Applications and Application Resources

To unregister an application or an application resource, use the crs_unregister command. You cannot unregister an application or resource that is ONLINE or required by another resource. The following example unregisters the mail application:

```
crs_unregister postman
```

The unregistration process frees space on the OCR. Additionally, run the crs_unregister command as a clean-up step when a resource is no longer managed by the Oracle Clusterware. Generally, you should unregister all permanently stopped applications.

Displaying Clusterware Application and Application Resource Status Information

To display status information about applications and resources that are on cluster nodes, use the crs_stat command. The following example displays the status information for the postman application:

```
crs_stat postman
```

Enter the following command to view information about all applications and resources in tabular format:
# crs_stat -t

The following text is an example of the command output:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Target</th>
<th>State</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>cluster_lockd</td>
<td>application</td>
<td>ONLINE</td>
<td>ONLINE</td>
<td>rac2</td>
</tr>
<tr>
<td>dhcp</td>
<td>application</td>
<td>OFFLINE</td>
<td>OFFLINE</td>
<td></td>
</tr>
</tbody>
</table>

Enter the following command to determine:

- How many times an application or application resource has been restarted
- How many times an application or resource has failed within the failure interval
- The maximum number of times that an application or resource can be restarted or fail
- The target state of the application or resource and the normal status information

# crs_stat -v

To view verbose content in tabular format, enter the following command:

# crs_stat -v -t

The following text is an example of the command output:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>R/RA</th>
<th>F/FT</th>
<th>Target</th>
<th>State</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>cluster_lockd</td>
<td>application</td>
<td>0/30</td>
<td>0/0</td>
<td>ONLINE</td>
<td>ONLINE</td>
<td>rac2</td>
</tr>
<tr>
<td>dhcp</td>
<td>application</td>
<td>0/1</td>
<td>0/0</td>
<td>OFFLINE</td>
<td>OFFLINE</td>
<td></td>
</tr>
<tr>
<td>named</td>
<td>application</td>
<td>0/1</td>
<td>0/0</td>
<td>OFFLINE</td>
<td>OFFLINE</td>
<td></td>
</tr>
<tr>
<td>network1</td>
<td>network</td>
<td>0/5</td>
<td>ONLINE</td>
<td>ONLINE</td>
<td>ONLINE</td>
<td>rac1</td>
</tr>
<tr>
<td>network1</td>
<td>network</td>
<td>0/5</td>
<td>ONLINE</td>
<td>ONLINE</td>
<td>ONLINE</td>
<td>rac1</td>
</tr>
</tbody>
</table>

Enter the following command to view the application profile information that is stored in the OCR:

# crs_stat -p

The following text is an example of the command output:

NAME=cluster_lockd
TYPE=application
ACTION_SCRIPT=cluster_lockd.scr
ACTIVEPlacement=0
AUTO_Start=1
CHECK_INTERVAL=5
DESCRIPTION=Cluster lockd/statd
FAILOVER_DELAY=30
FAILURE_INTERVAL=60
FAILURE_THRESHOLD=1
REBALANCE=
HOSTING_MEMBERS=
OPTIONAL/Resources=
PLACEMENT=balanced
REQUIRED(Resources=
RESTART_ATTEMPTS=2
SCRIPT_TIMEOUT=60 ...

Refer to the crs_stat command for more information.
See Also:  Appendix B, "High Availability Oracle Clusterware Command-Line Reference and C API" for detailed information about the Oracle Clusterware commands
Application-Specific Deployment Topics

This chapter discusses topics for deploying online (OLTP), data warehouse, and general purpose (hybrid) applications in Oracle Real Application Clusters (RAC) environments. The topics in this chapter are:

- General Deployment Strategies for Real Application Clusters-Based Applications
- Deploying OLTP Applications in Real Application Clusters
- Deploying Data Warehouse Applications with Real Application Clusters

General Deployment Strategies for Real Application Clusters-Based Applications

All single-instance application development and deployment techniques apply to RAC. If your applications run well on a single-instance Oracle database, then they will run well on a RAC database.

Deploying OLTP Applications in Real Application Clusters

Cache Fusion makes 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 and RAC can re-distribute and load balance workloads to surviving instances without interrupting processing. RAC also provides excellent scalability so that if you add or replace a node, then Oracle re-masters resources and re-distributes processing loads.

Flexible Implementation with Cache Fusion

To accommodate the frequently changing workloads of online transaction processing systems, RAC remains flexible and dynamic despite changes in system load and system availability. 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 Real Application Clusters

This section discusses how to deploy data warehouse systems in RAC environments by briefly describing the data warehouse features available in shared disk architectures. The topics in this section are:

- Speed-Up for Data Warehouse Applications on Real Application Clusters
- Parallel Execution in Data Warehouse Systems and RAC
- Using Parallel Instance Groups

Speed-Up for Data Warehouse Applications on Real Application Clusters

RAC is ideal for data warehouse applications because it augments the single instance benefits of Oracle. RAC does this by maximizing the processing available on all of the nodes that belong to a 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 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 selecting the execution plans that are optimized for your RAC configuration.

Parallel Execution in Data Warehouse Systems and RAC

Oracle'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 RAC databases.

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

Using Parallel Instance Groups

You can control the instances that allocate parallel execution server processes with instance groups. To do this, assign each active instance to at least one or more instance groups. Then dynamically control which instances spawn parallel processes by activating a particular group of instances.

Note: An instance can belong to one or more groups. You can enter several instance group names with the INSTANCE_GROUPS parameter using a comma as a separator.

Data Security Considerations in Real Application Clusters

Wallets used by RAC instances for Transparent Database Encryption may be a local copy of a common wallet shared by multiple nodes or a shared copy residing on a network file system that all of the nodes can access. A deployment with a single wallet on a shared disk requires no additional configuration to use Transparent Data
Encryption. Deployments where no shared storage exists require that each RAC node maintain its own local wallet. Details about creating and provisioning a wallet can be found in the Database Security Guide.

After you create and provision a wallet a single node, you must copy the wallet and make it available to all of the other nodes. For systems using Transparent Data Encryption with encrypted wallets, you can use any standard file transport protocol. For systems using Transparent Data Encryption with obfuscated wallets, file transport through a secured channel is recommended. The wallet must reside in the directory specified by the setting for the WALLET_LOCATION or ENCRYPTION_WALLET_LOCATION parameter in sqlnet.ora. 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 run the ALTER SYSTEM SET KEY statement at a database instance, you must again copy the wallet residing on that node and make it available to all of the other nodes. To avoid unnecessary administrative overhead, reserve re-keying for exceptional cases where you are certain that the server master key is compromised and that not re-keying it would cause a serious security problem.
This appendix explains how diagnose problems for the Oracle Clusterware and Oracle Real Application Clusters (RAC) components. The appendix explains how to do this by using dynamic debugging, enabling tracing, as well as by using trace and log file information. This appendix also includes information about the Cluster Verification Utility (CVU). The topics in this appendix are:

- Overview of Troubleshooting Real Application Clusters
- Diagnosing the Oracle Clusterware High Availability Components
- Diagnosing Oracle Real Application Clusters Components
- Using the Cluster Verification Utility

Overview of Troubleshooting Real Application Clusters

Both the Oracle Clusterware and Oracle RDBMS components have sub-components that you can troubleshoot as described in this chapter. You can enable dynamic debugging to troubleshoot Oracle Clusterware processing. You can also enable debugging and tracing for specific components and specific Oracle Clusterware resources to focus your troubleshooting efforts. A special clusterware health check command enables you to determine the status of several Oracle Clusterware components at one time.

The trace and log files that Oracle generates for RAC are available for both the Oracle Clusterware components and for the Oracle RDBMS with RAC components. For the Oracle Clusterware, Oracle stores these under a unified directory log structure. In addition to debugging and tracing tools, you can use the Cluster Verification Utility (CVU) and instance-specific alert files for troubleshooting.

Diagnosing the Oracle Clusterware High Availability Components

This section describes the diagnosability features for the Oracle Clusterware which include log files for the Oracle Clusterware daemon, as well as log files for the Event Manager daemon (EVM) and the RACGONS (Oracle Notification Service) under the following topics:

- Dynamic Debugging
- Component Level Debugging
- Oracle Clusterware Software Shutdown and Startup
- Diagnostics Collection Script
- The Oracle Clusterware Alerts
Dynamic Debugging

You can use crsctl commands to enable dynamic debugging for the Oracle Clusterware, the Event Manager (EVM), and their subcomponents. You can also dynamically change debugging levels using crsctl commands.

Debugging information remains in the Oracle Cluster Registry (OCR) for use during the next startup. You can also enable debugging for resources.

The crsctl syntax to enable debugging for the Oracle Clusterware is:

```
crsctl debug log crs "CRSRTI:1,CRSCOMM:2"
```

The crsctl syntax to enable debugging for EVM is:

```
EVM crsctl debug log evm "EVMCOMM:1"
```

The crsctl syntax to enable debugging for resources is:

```
crsctl debug log res "resname:1"
```

Component Level Debugging

You can enable debugging for the Oracle Clusterware, EVM, and their modules by setting environment variables or by running crsctl commands. Run the following command to enable debugging for all of the modules:

```
ORA_CRSDEBUG_ALL
```

Run the following command to enable tracing for a specific sub-module:

```
ORA_CRSDEBUG_modulename
```

Run the following command to obtain the module list where module is crs, evm, or css:

```
crsctl lsmodules module
```

The Oracle Clusterware modules for which you can enable tracing are:

- CRSUI
- CRSCOMM
- CRSRTI
- CRSMAIN
- CRSPLACE
- CRSAPP
- CRSRES
- CRSOCR
Diagnosing the Oracle Clusterware High Availability Components

The OCR modules for which you can enable tracing are:

- OCRAPI
- OCRCLI
- OCRUTL
- OCRCAC
- OCRMSG
- OCRSRV
- OCRMAS
- OCRRAW
- OCROSD

The EVM modules for which you can enable tracing are:

- EVMDMAIN
- EVMCOMM
- EVMEVT
- EVMAPP
- EVMAGENT
- CRSOCR
- CLUCLS

The following are the CSS modules:

- CSSD
- COMMCRS
- COMMNS

Oracle Clusterware Software Shutdown and Startup

You can use a crsctl command as follows to stop the Oracle Clusterware and its related resources on a specific node:

```bash
crsctl stop crs
```

You can use a crsctl command as follows to start the Oracle Clusterware and its related resources on a specific node:

```bash
crsctl start crs
```

Enabling and Disabling Oracle Clusterware Daemons

You can use crsctl commands as follows to enable and disable the startup of the Oracle Clusterware daemons. Run the following command to enable startup for all of the Oracle Clusterware daemons:

```bash
crsctl enable crs
```
Run the following command to disable the startup of all of the Oracle Clusterware daemons:

crsctl disable crs

Diagnostics Collection Script

Use the `diagcollection.pl` script to collect diagnostic information from an Oracle Clusterware installation. The diagnostics provide additional information so that Oracle Support can resolve problems. Run this script from the following location:

`CRS_HOME/bin/diagcollection.pl`

The Oracle Clusterware Alerts

The Oracle Clusterware component posts alert messages when important events occur. The following is an example of an alert from the CRSD process:

```
[NORMAL] CLSD-1201: CRSD started on host %s
[ERROR] CLSD-1202: CRSD aborted on host %s. Error [%s]. Details in %s.
[ERROR] CLSD-1203: Failover failed for the CRS resource %s. Details in %s.
[NORMAL] CLSD-1204: Recovering CRS resources for host %s
[ERROR] CLSD-1205: Auto-start failed for the CRS resource %s. Details in %s.
```

The location of this alert log on UNIX-based systems is `CRS_HOME/log/hostname/alerthostname.log` where `CRS_Home` is the name of the location of the Oracle Clusterware. Windows-based systems use the same path structure.

The following is an example of an EVMD alert:

```
[NORMAL] CLSD-1401: EVMD started on node %s
[ERROR] CLSD-1402: EVMD aborted on node %s. Error [%s]. Details in %s.
```

Resource Debugging

You can use `crsctl` commands to enable resource debugging using the following syntax:

```
crsctl debug log res "ora.vip.com:1"
```

Checking the Health of the Clusterware

Use the `crsctl check` command to determine the health of your clusterware as in the following example:

```
crsctl check crs
```

This command displays the status of the CSSD, EVMD, and CRSD processes. Run the following command to determine the health of individual daemons where `daemon` is `crsd`, `cssd` or `evmd`:

```
crsctl check daemon
```
Clusterware Log Files and the Unified Log Directory Structure

Oracle uses a unified log directory structure to consolidate the Oracle Clusterware component log files. This consolidated structure simplifies diagnostic information collection and assists during data retrieval and problem analysis.

Oracle retains a maximum of 10 log files for each type of file that it generates for the Oracle Clusterware processes, such as crsd, occsd, racg, and evmd. In addition, Oracle deletes the oldest log file for any log file group when the maximum storage limit for the group’s files exceeds 10MB. Alert files are stored in the following directory structures as described under the following headings:

- The Cluster Ready Services Daemon (crsd) Log Files
- Oracle Cluster Registry (OCR) Log Files
- Cluster Synchronization Services (CSS) Log Files
- Event Manager (EVM) Log Files
- RACG Log Files

Note: This section uses UNIX-based directory structure examples. The directory structure in all cases is identical for Windows-based platforms.

The Cluster Ready Services Daemon (crsd) Log Files
Log files for the CRSD process (crsd) can be found in the following directories:

CRS home/log/hostname/crsd

Oracle Cluster Registry (OCR) Log Files
The Oracle Cluster Registry (OCR) records log information in the following location:

CRS Home/log/hostname/client

Cluster Synchronization Services (CSS) Log Files
You can find CSS information that the OCSSD generates in log files in the following locations:

CRS Home/log/hostname/cssd

Event Manager (EVM) Log Files
Event Manager (EVM) information generated by evmd is recorded in log files in the following locations:

CRS Home/log/hostname/evmd

RACG Log Files
The Oracle RAC high availability trace files are located in the following two locations:

CRS home/log/hostname/racg
$ORACLE_HOME/log/hostname/racg
Core files are in the sub-directories of the log directories. Each RACG executable has a sub-directory assigned exclusively for that executable. The name of the RACG executable sub-directory is the same as the name of the executable.

Troubleshooting the Oracle Cluster Registry

This section explains how to troubleshoot the Oracle Cluster Registry (OCR) under the following topics:

- Using the OCRDUMP Utility to View Oracle Cluster Registry Content
- Using the OCRCHECK Utility
- Oracle Cluster Registry Troubleshooting

Using the OCRDUMP Utility to View Oracle Cluster Registry Content

This section explains how to use the OCRDUMP utility to view OCR content for troubleshooting. The OCRDUMP utility enables you to view the OCR contents by writing OCR content to a file or stdout in a readable format.

You can use a number of options for OCRDUMP. For example, you can limit the output to a key and its descendents. You can also write the contents to an XML-based file that you can view using a browser. OCRDUMP writes the OCR keys as ASCII strings and values in a datatype format. OCRDUMP retrieves header information based on a best effort basis. OCRDUMP also creates a log file in $ORA_CRS_Home/log/hostname/client. To change the amount of logging, edit the file $ORA_CRS_Home/srvm/admin/ocrlog.ini.

Note: Make sure that you have file creation privileges in the $ORA_CRS_Home/log/hostname/client directory before using the OCRDUMP utility.

OCRDUMP Utility Syntax and Options

This section describes the OCRDUMP utility command syntax and usage. Run the ocrdump command with the following syntax where filename is the name of a target file to which you want Oracle to write the OCR output and where keyname is the name of a key from which you want Oracle to write OCR subtree content:

```
```

Table A–1 describes the OCRDUMP utility options and option descriptions.

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>filename</code></td>
<td>Name of a file to which you want OCRDUMP to write output.</td>
</tr>
<tr>
<td><code>-stdout</code></td>
<td>The predefined output location that you can redirect with, for example, a filename.</td>
</tr>
<tr>
<td><code>-keyname</code></td>
<td>The name of an OCR key whose subtree is to be dumped.</td>
</tr>
<tr>
<td><code>-xml</code></td>
<td>Writes the output in XML format.</td>
</tr>
<tr>
<td><code>-noheader</code></td>
<td>Does not print the time at which you ran the command and when the OCR configuration occurred.</td>
</tr>
<tr>
<td><code>-backupfile</code></td>
<td>Option to identify a backup file.</td>
</tr>
</tbody>
</table>
OCR Utility Examples

The following ocrdump utility examples extract various types of OCR information and write it to various targets:

- `ocrdump`
  Writes the OCR content to a file called `OCRDUMPFILE` in the current directory.

- `ocrdump MYFILE`
  Writes the OCR content to a file called `MYFILE` in the current directory.

- `ocrdump -stdout -keyname SYSTEM`
  Writes the OCR content from the subtree of the key `SYSTEM` to `stdout`.

- `ocrdump -stdout -xml`
  Writes the OCR content to `stdout` in XML format.

Sample OCRDUMP Utility Output

The following OCRDUMP examples show the KEYNAME, VALUE TYPE, VALUE, permission set (user, group, world) and access rights for two sample runs of the ocrdump command. The following shows the output for the `SYSTEM.language` key that has a text value of `AMERICAN_AMERICA.WE8ASCII37`.

```plaintext
[SYSTEM.language]
ORATEXT : AMERICAN_AMERICA.WE8ASCII37
SECURITY : {USER_PERMISSION : PROCR_ALL_ACCESS, GROUP_PERMISSION : PROCR_READ, OTHER_PERMISSION : PROCR_READ, USER_NAME : user, GROUP_NAME : group }
```

The following shows the output for the `SYSTEM.version` key that has integer value 3:

```plaintext
[SYSTEM.version]
UB4 (10) : 3
SECURITY : {USER_PERMISSION : PROCR_ALL_ACCESS, GROUP_PERMISSION : PROCR_READ, OTHER_PERMISSION : PROCR_READ, USER_NAME : user, GROUP_NAME : group }
```

Using the OCRCHECK Utility

The OCRCHECK utility displays the version of the OCR’s block format, total space available and used space, OCRID, and the OCR locations that you have configured. OCRCHECK performs a block-by-block checksum operation for all of the blocks in all of the OCRs that you have configured. It also returns an individual status for each file as well as a result for the overall OCR integrity check. The following is a sample of the OCRCHECK output:

```
Status of Oracle Cluster Registry is as follows :
Version : 2
```
Total space (kbytes) : 262144
Used space (kbytes) : 16256
Available space (kbytes) : 245888
ID : 1918913332
Device/File Name : /tmp/data.ocr
Device/File integrity check succeeded
Device/File Name : /tmp/mirror.ocr
Device/File integrity check succeeded
Cluster registry integrity check succeeded

OCRCHECK creates a log file in the directory $ORA_CRS_Home/log/hostname/client. To change amount of logging, edit the file $ORA_CRS_Home/srvm/admin/ocrlog.ini.

**Oracle Cluster Registry Troubleshooting**

Table A–2 describes common OCR problems with corresponding resolution suggestions.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not currently using OCR mirroring and would like to.</td>
<td>Run the ocrconfig command with the -replace option as described on page 3-3.</td>
</tr>
<tr>
<td>An OCR failed and you need to replace it. Error messages in Enterprise Manager or OCR log file.</td>
<td>Run the ocrconfig command with the -replace option as described on page 3-4.</td>
</tr>
<tr>
<td>An OCR has a mis-configuration.</td>
<td>Run the ocrconfig command with the -repair option as described on page 3-4.</td>
</tr>
<tr>
<td>You are experiencing a severe performance effect from OCR processing or you want to remove an OCR for other reasons.</td>
<td>Run the ocrconfig command with the -replace option as described on page 3-3.</td>
</tr>
</tbody>
</table>

**Enabling Additional Tracing for Real Application Clusters High Availability**

Oracle Support may ask you to enable tracing to capture additional information. Because the procedures described in this section may affect performance, only perform these activities with the assistance of Oracle Support. This section includes the following topics:

- Generating Additional Trace Information for a Running Resource
- Verifying Event Manager Daemon Communications

**Generating Additional Trace Information for a Running Resource**

To generate additional trace information for a running resource, Oracle recommends that you use CRSCTL commands. For example, run the following command to turn on debugging for resources:

```
crsctl debug log res "resname:level"
```

Alternatively, you can set the resource attribute _USR_ORA_DEBUG to the value 1 using one of the following two methods:

- For an individual resource, modify the resource profile (a text file named `resource_name.cap`) that you can generate with the following command:

```
crs_stat -p resource_name > CRS Home/crs/profile/resource_name.cap
```
Only create .cap files in this directory for resources that are owned by the root user. Use the CRS Home/crs/public/ directory for other resources, which are resources that are owned by the oracle user. Then set the _USRORADEBUG value to 1 in this .cap file and issue the following command:

crs_register -u resource_name

- For all resources, add the following line to the script, racgwrap, in ORACLE_HOME/bin:

```bash
_USRORADEBUG=1
export _USRORADEBUG
```

Only node-level resources, such as the VIP, should have their .cap file created in the directory CRS Home/crs/profile.

Verifying Event Manager Daemon Communications

The event manager daemons (evmd) running on separate nodes communicate through specific ports. To determine whether the evmd for a node can send and receive messages, perform the test described in this section while running session 1 in the background.

On node 1, session 1 enter:

```bash
$ evmwatch -A -t "@timestamp @@"
```

On node 2, session 2 enter:

```bash
$ evmpost -u "hello" [-h nodename]
```

Session 1 should show output similar to the following:

```bash
$ 21-Aug-2002 08:04:26 hello
```

Ensure that each node can both send and receive messages by executing this test in several permutations.

Diagnosing Oracle Real Application Clusters Components

This section describes the diagnosability features for Oracle Real Application Clusters components. This section includes the following topics:

- Where to Find Files for Analyzing Errors
- Using Instance-Specific Alert Files in Real Application Clusters
- Enabling Tracing for Java-Based Tools and Utilities in Real Application Clusters
- Resolving Pending Shutdown Issues

Where to Find Files for Analyzing Errors

Oracle records information about important events that occur in your RAC environment in trace files. The trace files for 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. For troubleshooting, you may need to also provide files from the following bdump locations:

- ORACLE_HOME/admin/db_name/bdump on UNIX-based systems
Diagnosing Oracle Real Application Clusters Components

Background Thread Trace Files in Real Application Clusters
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.

Background thread trace files are created regardless of whether the BACKGROUND_DUMP_DEST parameter is set in the server parameter file (SPFILE). If you set BACKGROUND_DUMP_DEST, then the trace files are stored in the directory specified. If you do not set the parameter, then the trace files are stored as follows:

- ORACLE_BASE/admin/db_name/bdump on UNIX-based systems
- %ORACLE_BASE%/admin/db_name/bdump on Windows-based systems
- SID_dbwr_1202.trc
- SID_smon_4560.trc

Oracle creates a different trace file for each background thread. For both UNIX- and Windows-based systems, trace files for the background processes are named SID_process_name_process_identifier.trc, for example:

User Process Trace Files in Real Application Clusters
Trace files are created for user processes if you set the USER_DUMP_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 UNIX-based systems and thread_identifier is the thread identifier on Windows-based systems.

Using Instance-Specific Alert Files in Real Application Clusters
Each instance in a 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 BACKGROUND_DUMP_DEST parameter in the initdb_name.ora initialization parameter file. If you do not set the BACKGROUND_DUMP_DEST parameter, the alert_SID.log file is generated in the following locations:

- ORACLE_BASE/admin/db_name/bdump on UNIX-based systems.
- %ORACLE_BASE%/admin/db_name/bdump on Windows-based systems.

Some files may also be in the udump directory.

In addition, the directory cdmp_timestamp contains KST, or in-memory traces of RAC instance failure information. This directory is located in ORACLE_HOME/admin/db_name/bdump/cdmp_timestamp, where timestamp is the time at which the error occurred.

Trace dump files are stored under the cdmp directory. Oracle creates these files for each process that is connected to the instance. The naming convention for the trace dump files is same as for trace files, but with .trw as the file extension.
Enabling Tracing for Java-Based Tools and Utilities in Real Application Clusters

All Java-based tools and utilities that are available in RAC are invoked by executing scripts of the same name as the tool or utility. This includes the Database Configuration Assistant (DBCA), the Net Configuration Assistant (NetCA), the Virtual Internet Protocol Configuration Assistant (VIPCA), Server Control (SRVCTL), and the Global Services Daemon (GSD). For example, to run the DBCA, enter the command dbca.

To enable additional debugging information, edit the command file (such as dbca in this example) and add the following parameters to the $JRE_DIR/bin/jre command line:

-DTRACING.ENABLED=true -DTRACING.LEVEL=2

For example, the file ORACLE_HOME/bin/dbca contains the line:

$JRE_DIR/bin/jre -DORACLE_HOME=$OH -DJDBC_PROTOCOL=thin -mx64 -classpath $CLASSPATH oracle.sysman.dbca.Dbca $ARGUMENTS

Change this line to:

$JRE_DIR/bin/jre -DORACLE_HOME=$OH -DTRACING.ENABLED=true -DTRACING.LEVEL=2 -DJDBC_PROTOCOL=thin -mx64 -classpath $CLASSPATH oracle.sysman.dbca.Dbca $ARGUMENTS

When you run the DBCA, Oracle records trace information in the DBCA log file.

To debug SRVCTL, add the argument -D to the command line. For example, to generate tracing information for an add database operation, run the following command:

$ srvctl -D add database -d mydatabase

Resolving Pending Shutdown Issues

In some situations a SHUTDOWN IMMEDIATE may be pending and Oracle will not quickly respond to repeated shutdown requests. This is because the Oracle Clusterware may be processing a current shutdown request. In such cases, issue a SHUTDOWN ABORT using SQL*Plus for subsequent shutdown requests.

Using the Cluster Verification Utility

This section describes the Cluster Verification Utility (CVU) under the following topics:

- Cluster Verification Utility Requirements
- Understanding CVU Commands, Help, Output, and Nodelist Shortcuts
- Cluster Verification Utility Nodelist Shortcuts

See Also: Your platform-specific Oracle Clusterware and RAC installation guide for information about how to manually install CVU

The CVU can verify the primary cluster components during an operational phase or stage. A component can be basic, such as free disk space, or it can be complex, such as checking the Oracle Clusterware integrity. For example, CVU can verify multiple Oracle Clusterware subcomponents across the Oracle Clusterware layers. Additionally, CVU can check disk space, memory, processes, and other important
Using the Cluster Verification Utility

cluster components. A stage could be, for example, database installation, for which CVU can perform a pre-check to verify whether your system meets the criteria for a RAC installation. Other stages include the initial hardware setup and the establishing of system requirements through the fully operational cluster setup.

When verifying stages, CVU uses entry and exit criteria. In other words, each stage has entry criteria that define a specific set of verification tasks to be performed before initiating that stage. This pre-check prevents you from beginning a stage, such as installing the Oracle Clusterware, unless you meet the Oracle Clusterware stage's pre-requisites.

The exit criteria for a stage define another set of verification tasks that you need to perform after the completion of the stage. Post-checks ensure that the activities for that stage have been completed. Post-checks identify stage-specific problems before they propagate to subsequent stages.

The node list that you use with CVU commands should be a comma-delimited list of host names without a domain. The CVU ignores domains while processing node lists. If a CVU command entry has duplicate node entries after removing domain information, then CVU eliminates the duplicate node entries. Wherever supported, you can use the `-n all` option to verify all of your cluster nodes that are part of a specific RAC installation. You do not have to be the root user to use the CVU and the CVU assumes that the current user is the oracle user.

---

**Note:** The CVU only supports an English-based syntax and English online help.

---

For network connectivity verification, the CVU discovers all of the available network interfaces if you do not specify an interface on the CVU command line. For storage accessibility verification, the CVU discovers shared storage for all of the supported storage types if you do not specify a particular storage identification on the command line. The CVU also discovers the Oracle Clusterware home if one is available.

Run the CVU command line tool using the `cluvfy` command. Using `cluvfy` does not adversely affect your cluster environment or your installed software. You can run `cluvfy` commands at any time, even before the Oracle Clusterware installation. In fact, the CVU is designed to assist you as soon as your hardware and operating system are operational. If you run a command that requires Oracle Clusterware on a node, then the CVU reports an error if the Oracle Clusterware is not yet installed on that node.

You can enable tracing by setting the environment variable `SRVM_TRACE` to true. For example, in `tcsh` an entry such as `setenv SRVM_TRACE true` enables tracing. The CVU trace files are created in the `CV_HOME/cv/log` directory. Oracle automatically rotates the log files and the most recently created log file has the name `cvutrace.log.0`. You should remove unwanted log files or archive them to reclaim disk place if needed. The CVU does not generate trace files unless you enable tracing.

### Cluster Verification Utility Requirements

The CVU requirements are:

- At least 30MB free space for the CVU software on the node from which you run the CVU
- A location for the current JDK, Java 1.4.1 or later
- A work directory with at least 25MB free space on each node
Understanding CVU Commands, Help, Output, and Nodelist Shortcuts

This section describes the following Cluster Verification Utility topics:

- Using CVU Help
- Verbose Mode and UNKNOWN Output
- Cluster Verification Utility Nodelist Shortcuts

Using CVU Help

The `cluvfy` commands have context sensitive help that shows their usage based on the command line arguments that you enter. For example, if you enter `cluvfy`, then the CVU displays high-level generic usage text describing the stage and component syntax. If you enter `cluvfy comp -list`, then the CVU shows the valid components with brief descriptions about each of them. If you enter `cluvfy comp -help`, then the CVU shows detailed syntax for each of the valid component checks. Similarly, `cluvfy stage -list` and `cluvfy stage -help` display valid stages and their syntax for their checks respectively.

If you enter an invalid CVU command, then the CVU shows the correct usage for that command. For example, if you type `cluvfy stage -pre dbinst`, then CVU shows the correct syntax for the pre-check commands for the `dbinst` stage. Enter the `cluvfy -help` command to see detailed CVU command information.

Verbose Mode and UNKNOWN Output

Although by default the CVU reports in non-verbose mode by only reporting the summary of a test, you can obtain detailed output by using the `-verbose` argument. The `-verbose` argument produces detailed output of individual checks and where applicable shows results for each node in a tabular layout.

If a `cluvfy` command responds with `UNKNOWN` for a particular node, then this is because the CVU cannot determine whether a check passed or failed. The cause of this could be a loss of reachability or the failure of user equivalence to that node. The cause could also be any system problem that was occurring on that node at the time that CVU was performing a check.

If you run the CVU using the `-verbose` argument and the CVU responds with `UNKNOWN` for a particular node, then this is because the CVU cannot determine whether a check passed or failed. The following is a list of possible causes for an `UNKNOWN` response:

- The node is down
- Executables that the CVU requires are missing in `CRS_home/bin` or the `Oracle home` directory
- The user account that ran the CVU does not have privileges to run common operating system executables on the node

**Note:** When using the CVU, the VU attempts to copy any needed information to the CVU work directory. Make sure that the CVU work directory exists on all of the nodes in your cluster database and that the directory on each node has write permissions established for the CVU user. Set this directory using the `CV_DESTLOC` environment variable. If you do not set this variable, then the CVU uses `/tmp` as the work directory on UNIX-based systems, and `C:\temp` on Windows-based systems.
- The node is missing an operating system patch or a required package
- The node has exceeded the maximum number of processes or maximum number of open files, or there is a problem with IPC segments, such as shared memory or semaphores

**Cluster Verification Utility Nodelist Shortcuts**

You can use the following nodelist shortcuts:

To provide the CVU a list of all of the nodes of a cluster, enter `-n all`. CVU attempts to obtain the node list in the following order:

1. If vendor clusterware is available, then the CVU selects all of the configured nodes from the vendor clusterware using the `lsnodes` utility.
2. If the Oracle Clusterware is installed, then the CVU selects all of the configured nodes from the Oracle Clusterware using the `olsnodes` utility.
3. If neither the vendor nor the Oracle Clusterware is installed, then the CVU searches for a value for the `CV_NODE_ALL` key in the configuration file.
4. If vendor and Oracle Clusterware are not installed and no key named `CV_NODE_ALL` exists in the configuration file, then the CVU searches for a value for the `CV_NODE_ALL` environmental variable. If you have not set this variable, then the CVU reports an error.

To provide a partial node list, you can set an environmental variable and use it in the CVU command. For example, on UNIX-based systems you can enter:

```bash
setenv MYNODES node1,node3,node5
cluvfy comp nodecon -n $MYNODES [-verbose]
```

**Cluster Verification Utility Configuration File**

You can use CVU’s configuration file to define specific inputs for the execution of the CVU. The path for the configuration file is `CV_HOME/cv/admin/cvu_config`. You can modify this using a text editor. The inputs to the tool are defined in the form of key entries. You must follow these rules when modifying the CVU configuration file:

- Key entries have the syntax `name=value`
- Each key entry and the value assigned to the key only defines one property
- Lines beginning with the number sign (`#`) are comment lines and are ignored
- Lines that do not follow the syntax `name=value` are ignored

The following is the list of keys supported by CVU:

- `CV_NODE_ALL`—If set, it specifies the list of nodes that should be picked up when Oracle Clusterware is not installed and a `-n all` option has been used in the command line. By default, this entry is commented out.
- `CV_RAW_CHECK_ENABLED`—If set to `TRUE`, it enables the check for accessibility of shared disks on RedHat release 3.0. This shared disk accessibility check requires that you install a `cuvqdisk` rpm on all of the nodes. By default, this key is set to `TRUE` and shared disk check is enabled.
- `CV_XCHK_FOR_SSH_ENABLED`—If set to `TRUE`, it enables the X-Windows check for verifying user equivalence with ssh. By default, this entry is commented out and X-Windows check is disabled.
■ ORACLE_SRVM_REMOTESHELL—If set, it specifies the location for ssh/rsh command to override CVU’s default value. By default, this entry is commented out and the tool uses /usr/sbin/ssh and /usr/sbin/rsh.

■ ORACLE_SRVM_REMOTECOPY—If set, it specifies the location for the scp or rcp command to override the CVU default value. By default, this entry is commented out and CVU uses /usr/bin/scp and /usr/sbin/rcp.

If CVU does not find a key entry defined in the configuration file, then the CVU searches for the environment variable that matches the name of the key. If the environment variable is set, then the CVU uses its value, otherwise the CVU uses a default value for that entity.

Performing Various CVU Tests

You can perform the following tests using CVU as described under the following topics:

■ Cluster Verification Utility System Requirements Verifications
■ Cluster Verification Utility Storage Verifications
■ Cluster Verification Utility Connectivity Verifications
■ Cluster Verification Utility User and Permissions Verifications
■ Cluster Verification Utility Node Comparisons and Verifications
■ Cluster Verification Utility Installation Verifications
■ Cluster Verification Utility Oracle Clusterware Component Verifications
■ Cluster Verification Utility Cluster Integrity Verifications
■ Cluster Verification Utility Argument and Option Definitions

See Also: Table A–3 for details about the arguments and options used in the following CVU examples

Cluster Verification Utility System Requirements Verifications

To verify the minimal system requirements on the nodes prior to installing the Oracle Clusterware or RAC, use the sys component verification command as follows:

```bash
cluvfy comp sys [ -n node_list ] [ -p { crs | database } ] [ -r { 10gR1 | 10gR2 } ] [ -osdba osdba_group ] [ -orainv orainventory_group ] [-verbose]
```

To check the system requirements for installing RAC, use the -p database argument, and to check the system requirements for installing the Oracle Clusterware, use the -p crs argument. To check the system requirements for installing the Oracle Clusterware or RAC from Oracle Database 10g release 1 (10.1), use the -r 10gR1 argument. For example, verify the system requirements for installing the Oracle Clusterware on the cluster nodes known as node1, node2 and node3 by running the following command:

```bash
cluvfy comp sys -n node1,node2,node3 -p crs -verbose
```

Cluster Verification Utility Storage Verifications

To verify whether storage is shared among the nodes in your cluster database or to identify all of the storage that is available on the system and can be shared across the cluster nodes, use the component verification command ssa as follows:

```bash
cluvfy comp ssa [ -n node_list ] [ -s storageID_list ] [-verbose]
```
See Also: Refer to "Known Issues for the Cluster Verification Utility" on page A-19 for the types of storage that CVU supports.

For example, discover all of the shared storage systems available on your system by running the following command:
```
cluvfy comp ssa -n all -verbose
```

You can verify the accessibility of a specific storage location, such as `/dev/sda`, across the cluster nodes by running the following command:
```
cluvfy comp ssa -n all -s /dev/sda
```

To verify whether a certain amount of free space is available on a specific location in the nodes of your cluster database, use the component verification command `space`.
```
cluvfy comp space [-n node_list] -l storage_location -z disk_space {B|K|M|G} [-verbose]
```

For example, you can verify the availability of at least 2GB of free space at the location `/home/dbadmin/products` on all of the cluster nodes by running the following command:
```
cluvfy comp space -n all -l /home/dbadmin/products -z 2G -verbose
```

To verify the integrity of your Oracle Cluster File System, use the component verification command `cfs` as follows:
```
cluvfy comp cfs [-n node_list] -f file_system [-verbose]
```

For example, you can verify the integrity of the cluster file system `/oradbshare` on all of the nodes by running the following command:
```
cluvfy comp cfs -f /oradbshare -n all -verbose
```

Note: The sharedness check for the file system is supported for Oracle Cluster File System version 1.0.14 or higher.

Cluster Verification Utility Connectivity Verifications

To verify the reachability of the cluster nodes from the local node or from any other cluster node, use the component verification command `nodereach` as follows:
```
cluvfy comp nodereach -n node_list [ -srcnode node ] [-verbose]
```

To verify the connectivity between the cluster nodes through all of the available network interfaces or through specific network interfaces, use the component verification command `nodecon` as follows:
```
cluvfy comp nodecon -n node_list [ -i interface_list ] [-verbose]
```

Use the `nodecon` command without the `-i` option as follows to use CVU to:
- Discover all of the network interfaces that are available on the cluster nodes
- Review the interfaces' corresponding IP addresses and subnets
- Obtain the list of interfaces that are suitable for use as VIPs and the list of interfaces to private interconnects
- Verify the connectivity between all of the nodes through those interfaces

`cluvfy comp nodecon -n all [-verbose]`

You can run this command in verbose mode to identify the mappings between the interfaces, IP addresses, and subnets. To verify the connectivity between all of the nodes through specific network interfaces, use the `comp nodecon` command with the `-i` option. For example, you can verify the connectivity between the nodes `node1`, `node2`, and `node3`, through interface `eth0` by running the following command:

`cluvfy comp nodecon -n node1,node2,node3 -i eth0 -verbose`

**Cluster Verification Utility User and Permissions Verifications**

To verify user accounts and administrative permissions-related issues, use the component verification command `admprv` as follows:

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

| -o user_equiv [-sshonly] |
| -o crs_inst [-orainv orainventory_group ] |
| -o db_inst [-orainv orainventory_group ] [-osdba osdba_group ] |
| -o db_config -d oracle_home |

To verify whether user equivalence exists on specific nodes, use the `-o user_equiv` argument. On UNIX-based platforms, this command verifies user equivalence first using `ssh` and then using `rsh`, if the `ssh` check fails. To verify the equivalence only through `ssh`, use the `-sshonly` option. By default, the equivalence check does not verify X-Windows configurations, such as whether you have disabled X-forwarding, whether you have the proper setting for the `DISPLAY` environment variable, and so on.

To verify X-Windows aspects during user equivalence checks, set the `CV_XCHK_FOR_SSH_ENABLED` key to `TRUE` in the configuration file that resides in the path `CV_HOME/cv/admin/cvu_config` before you run the `admprv -o user_equiv` command. Use the `-o crs_inst` argument to verify whether you have permissions to install the Oracle Clusterware.

You can use the `-o db_inst` argument to verify the permissions that are required for installing RAC and the `-o db_config` argument to verify the permissions that are required for creating a RAC database or for modifying a RAC database's configuration. For example, you can verify user equivalence for all of the nodes by running the following command:

`cluvfy comp admprv -n all -o user_equiv -verbose`

On Linux and Unix platforms, this command verifies user equivalence by first using `ssh` and then using `rsh` if the `ssh` check fails. To verify the equivalence only through `ssh`, use the `-sshonly` option. By default, the equivalence check does not verify X-Windows configurations, such as when you have disabled X-forwarding with the setting of the `DISPLAY` environment variable. To verify X-Windows aspects during user equivalence checks, set the `CV_XCHK_FOR_SSH_ENABLED` key to `TRUE` in the configuration file `CV_HOME/cv/admin/cvu_config` before you run the `admprv -o user_equiv` command.

To verify the existence of node applications, namely VIP, ONS and GSD, on all of the nodes, use the component `nodeapp` command:

`cluvfy comp nodeapp [ -n node_list ] [ -verbose ]`
Using the Cluster Verification Utility

**Cluster Verification Utility Node Comparisons and Verifications**

Use the component verification `peer` command to compare the nodes as follows:

```
cluvfy comp peer [-refnode node] -n node_list [-r { 10gR1 | 10gR2 }] [ -orainv orainventory_group ] [ -osdba osdba_group ] [-verbose]
```

The following command lists the values of several pre-selected properties on different nodes from Oracle Database 10g release 2 (10.2):

```
cluvfy comp peer -n node_list [-r 10gR2] [-verbose]
```

You can also use the `comp peer` command with the `-refnode` argument to compare the properties of other nodes against the reference node.

**Cluster Verification Utility Installation Verifications**

To verify whether your system meets all of the criteria for an Oracle Clusterware installation, use the pre-check command for the Oracle Clusterware Installation stage as follows:

```
cluvfy stage -pre crsinst -n node_list [-c ocr_location ] [-r { 10gR1 | 10gR2 }] [-v voting_disk ] [-osdba osdba_group ] [-orainv orainventory_group ] [-verbose]
```

After you have completed phase one, verify that the Oracle Clusterware is functioning properly before proceeding with phase two of your RAC installation by running the `post-check` command for the Oracle Clusterware Installation stage or, `-post crsinst` as follows:

```
cluvfy stage -post crsinst -n node_list [-verbose]
```

To verify whether your system meets all of the criteria for a RAC installation, use the pre-check command for the Database Installation stage as follows:

```
cluvfy stage -pre dbinst -n node_list [-r { 10gR1 | 10gR2 }] [-osdba osdba_group ] [-orainv orainventory_group ] [-verbose]
```

To verify whether your system meets all of the criteria for creating a database or for making a database configuration change, use the pre-check command for the Database Configuration stage as follows:

```
cluvfy stage -pre dbcfg -n node_list -d oracle_home [-verbose]
```

**Cluster Verification Utility Oracle Clusterware Component Verifications**

To verify the integrity of all of the Oracle Clusterware components, use the component verification `crs` command as follows:

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

To verify the integrity of each individual Cluster Manager sub-component, use the component verification command `clumgr` as follows:

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

To verify the integrity of the Oracle Cluster Registry, use the component verification command `ocr` as follows:

```
cluvfy comp ocr [-n node_list ] [-verbose]
```
Cluster Verification Utility Cluster Integrity Verifications

To check the integrity of your entire cluster, which means to verify that all of the nodes in the cluster have the same view of the cluster configuration, use the component verification command `cluvfy` as follows:

```
cluvfy comp clu
```

Cluster Verification Utility Argument and Option Definitions

Table A–3 describes the CVU arguments and options used in the previous examples:

<table>
<thead>
<tr>
<th>Argument or Option</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-n node_list</code></td>
<td>The comma-separated list of non-domain qualified node names on which the test should be conducted. If <code>all</code> is specified, then all of the nodes in the cluster will be used for verification.</td>
</tr>
<tr>
<td><code>-i interface_list</code></td>
<td>The comma-separated list of interface names.</td>
</tr>
<tr>
<td><code>-f file_system</code></td>
<td>The name of the file system.</td>
</tr>
<tr>
<td><code>-s storageID_list</code></td>
<td>The comma-separated list of storage identifiers.</td>
</tr>
<tr>
<td><code>-l storage_location</code></td>
<td>The storage path.</td>
</tr>
<tr>
<td><code>-z disk_space</code></td>
<td>The required disk space, in units of bytes (B), kilobytes (K), megabytes (M), or gigabytes (G).</td>
</tr>
<tr>
<td><code>-osdba osdba_group</code></td>
<td>The name of the OSDBA group. The default is <code>dba</code>.</td>
</tr>
<tr>
<td><code>-orainv orainventory_group</code></td>
<td>The name of the Oracle inventory group. The default is <code>oinstall</code>.</td>
</tr>
<tr>
<td><code>-verbose</code></td>
<td>Makes CVU print detailed output.</td>
</tr>
<tr>
<td><code>-o user_equiv</code></td>
<td>Checks user equivalence between the nodes.</td>
</tr>
<tr>
<td><code>-sshonly</code></td>
<td>Check user equivalence for ssh setup only.</td>
</tr>
<tr>
<td><code>-o crs_inst</code></td>
<td>Checks administrative privileges for installing Oracle Clusterware.</td>
</tr>
<tr>
<td><code>-o db_inst</code></td>
<td>Checks administrative privileges for installing RAC.</td>
</tr>
<tr>
<td><code>-o db_config</code></td>
<td>Checks administrative privileges for creating or configuring a database.</td>
</tr>
<tr>
<td><code>-refnode</code></td>
<td>The node that will be used as a reference for checking compatibility with other nodes.</td>
</tr>
<tr>
<td><code>-srcnode</code></td>
<td>The node from which the reachability to other nodes should be checked.</td>
</tr>
<tr>
<td>`-r { 10gR1</td>
<td>10gR2 }`</td>
</tr>
</tbody>
</table>

Known Issues for the Cluster Verification Utility

This section describes the following known limitations for CVU:

- Database Versions Supported by Cluster Verification Utility
- Linux Shared Storage Accessibility (ssa) Check Reports Limitations
- Shared Disk Discovery on Red Hat Linux
Database Versions Supported by Cluster Verification Utility
The current CVU release supports only Oracle Database 10g RAC and the Oracle Clusterware and it is not backward compatible. In other words, CVU cannot check or verify pre-Oracle Database 10g products.

Linux Shared Storage Accessibility (ssa) Check Reports Limitations
The current release of cluvfy has the following limitations on Linux regarding shared storage accessibility check.
- Currently NAS storage (r/w, no attribute caching) and OCFS (version 1.0.14 or higher) are supported.
- For sharedness checks on NAS, cluvfy commands require you to have write permission on the specified path. If the cluvfy user does not have write permission, cluvfy reports the path as not shared.

Shared Disk Discovery on Red Hat Linux
To perform discovery and shared storage accessibility checks for SCSI disks on Red Hat Linux 3.0 and SUSE Linux Enterprise Server, CVU requires the CVUQDISK package. If you attempt to use CVU and the CVUQDISK package is not installed on all of the nodes in your RAC environment, then CVU responds with an error.

Perform the following procedure to install the CVUQDISK package:
1. Login as the root user.
2. Copy the rpm, cvuqdisk-1.0.1-1.rpm, to a local directory. You can find this rpm in the rpm sub-directory of the top-most directory in the Oracle Clusterware installation media. For example, you can find cvuqdisk-1.0.1-1.rpm in the directory /mountpoint/clusterware/rpm/ where mountpoint is the mounting point for the disk on which the directory is located.
3. Set the environment variable to a group that should own the CVUQDISK package binaries. If CVUQDISK_GRP is not set, then by default the oinstall group is the owner’s group.
4. Determine whether previous versions of the CVUQDISK package are installed by running the command rpm -q cvuqdisk. If you find previous versions of the CVUQDISK package, then remove them by running the command rpm -e cvuqdisk previous_version where previous_version is the identifier of the previous CVUQDISK version.
5. Install the latest CVUQDISK package by running the command rpm -iv cvuqdisk-1.0.1-1.rpm.
This appendix describes the Oracle Clusterware application program interface (API) command reference and includes the following topics:

- Using Oracle Clusterware Commands
- The Oracle Clusterware Commands
- C Application Programming Interface to the Oracle Clusterware
- Functions for Managing Resource Structures

See Also: Chapter 13, "Making Applications Highly Available Using the Oracle Clusterware" for conceptual and introductory information about using Oracle Clusterware commands

Using Oracle Clusterware Commands

This section explains how to use the Oracle Clusterware commands to manage applications and application resources under the Oracle Clusterware framework.

Application Profile Syntax

The examples in this appendix show the command syntaxes that you use in application profiles. Lines starting with a pound sign (#) are comment lines and are not processed as part of a profile. A backslash (\) at the end of a line indicates that the next line is a continuation of the previous line. Refer to the section titled "Using crs_profile to Create An Application Resource Profile" on page 13-7 to see a profile example.

Security and Permissions

The Oracle Clusterware uses UNIX-like security where permissions may be specified for the owner, nodes of a group, or holders of specific privileges. There may also be default permissions for other users. These permissions are read, write, and run on Windows-based systems, and rwx on UNIX-based systems. Default ownership and permissions are set when you create an application profile. Table B–1 shows the Oracle Clusterware commands and their owners as well as which commands require write permission and which commands require run permission.
Table B–1  Oracle Clusterware Command Owner and Permissions Matrix

<table>
<thead>
<tr>
<th>Owner</th>
<th>Read</th>
<th>Write Permission Required</th>
<th>Run/Execute Permission Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>crs_stat {any options}</td>
<td>crs_register {no arguments}</td>
<td>crs_start</td>
</tr>
<tr>
<td></td>
<td></td>
<td>crs_register -u</td>
<td>crs_unregister</td>
</tr>
<tr>
<td></td>
<td></td>
<td>crs_unregister</td>
<td>crs_stat {no arguments}</td>
</tr>
<tr>
<td>Group</td>
<td>crs_stat {any options}</td>
<td>crs_register {no arguments}</td>
<td>crs_start</td>
</tr>
<tr>
<td></td>
<td></td>
<td>crs_register -u</td>
<td>crs_unregister</td>
</tr>
<tr>
<td></td>
<td></td>
<td>crs_unregister</td>
<td>crs_stat {no arguments}</td>
</tr>
<tr>
<td>Other</td>
<td>crs_stat {any options}</td>
<td>crs_register {no arguments}</td>
<td>crs_start</td>
</tr>
<tr>
<td></td>
<td></td>
<td>crs_register -u</td>
<td>crs_unregister</td>
</tr>
<tr>
<td></td>
<td></td>
<td>crs_unregister</td>
<td>crs_stat {no arguments}</td>
</tr>
</tbody>
</table>

Note: on platforms that do not have a group concept, the group permissions are NULL or not set to anything.

Note: Do not use the CRS commands crs_register, crs_profile, crs_start or crs_stop on resources with names that begin with the prefix ora unless either Oracle Support asks you to, or Oracle has certified you as described in http://metalink.oracle.com. You can, however, create resources that depend on resources that Oracle has defined. In addition, you can use the Oracle Clusterware commands to inspect the configuration and status.

The Oracle Clusterware Commands

Table B–2 alphabetically lists the Oracle Clusterware commands that this appendix describes. Note that the -q option runs all commands in quiet mode. This means that no messages are displayed on the console.

Table B–2  Oracle Clusterware Commands Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crs_getperm</td>
<td>Inspects the permissions associated with a resource.</td>
</tr>
<tr>
<td>crs_profile</td>
<td>Creates, validates, deletes, and updates an Oracle Clusterware application profile</td>
</tr>
<tr>
<td>crs_register</td>
<td>Registers configuration information for an application with the OCR.</td>
</tr>
</tbody>
</table>
The Oracle Clusterware Commands

Table B–2 (Cont.) Oracle Clusterware Commands Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crs_relocate</td>
<td>Relocates an application profile to another node.</td>
</tr>
<tr>
<td>crs_setperm</td>
<td>Sets permissions associated with a resource.</td>
</tr>
<tr>
<td>crs_stat</td>
<td>Lists the status of an application profile.</td>
</tr>
<tr>
<td>crs_start</td>
<td>Starts applications that have been registered.</td>
</tr>
<tr>
<td>crs_stop</td>
<td>Stops an Oracle Clusterware application.</td>
</tr>
<tr>
<td>crs_unregister</td>
<td>Removes the configuration information for an application profile from the OCR.</td>
</tr>
</tbody>
</table>

Note: on platforms that do not have a group concept, the group permissions are NULL or not set to anything.

crs_getperm

Inspects the permissions associated with a resource.

Syntax and Options for crs_getperm

Use the crs_getperm command with the following syntax:

crs_getperm resource_name [-u user] [-g group]

To obtain the permissions associated with a resource, use the following syntax:

crs_getperm resource_name

Example of crs_getperm

To list the permissions associated with the postman application, use the following command:

crs_getperm postman

crs_profile

Creates, validates, deletes, and updates an Oracle Clusterware application profile. It works on the user’s copy of a profile. This command does not operate against the database that the Oracle Clusterware is using.

You can also use crs_profile to generate a template script. The crs_profile command creates new application profiles and validates, updates, deletes, or lists existing profiles. An application profile assigns values to attributes that define how a resource should be managed or monitored in a cluster. For the root user, profiles are written to the CRS home/crs/profile directory. For non-privileged users, the profile is written to the CRS home/crs/public directory. Values in the profile that are left blank are ignored and may be omitted if not required. Omitted profile variables that are required for a resource type can cause validation or registration to fail.

After you have created an application profile and registered the application with the Oracle Clusterware using the crs_register command, you can use other Oracle Clusterware commands, such as crs_stat, crs_start, crs_stop, crs_relocate, and crs_unregister, on the application. You must register applications using the crs_register command before the other Oracle Clusterware commands...
are available to manage the application. The `crs_profile` command may have other options for defining user-defined attributes in a profile.

**Syntax and Options for `crs_profile`**

Use the `crs_profile` command with the following syntax to create an application profile template:

```
```

To create an application profile from an application profile template:
```
crs_profile -create resource_name -I template_file [-f] [-q]
```

To validate the application profile syntax of a profile, enter the following command:
```
crs_profile -validate resource_name [-q]
```

To list one or more application profiles:
```
crs_profile -print [resource_name [...] [-q]
```

To create an application profile template from an existing application profile:
```
crs_profile -template resource_name [-O template_file] [-q]
```

To update an application profile:
```
crs_profile -update resource_name [option [...] [-q]
```

To delete an application profile and its associated action program:
```
crs_profile -delete resource_name [-q]
```

**Note:** The `crs_profile` `-delete` command deletes the resource profile file but does not delete the action script file.

---

**Table B–3 crs_profile Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-create resource_name</code></td>
<td>Creates a resource for the indicated application according to the specified options. A <code>resource_name</code> is a string containing a combination of characters `a-z, A-Z, 0-9, '.', '_']. The resource name may not start with a period (.), The maximum length for a resource name is 128 characters. Resource profiles are created in the directory that you specify.</td>
</tr>
<tr>
<td><code>-t application</code></td>
<td>Indicates an application or application resource type.</td>
</tr>
<tr>
<td><code>[-a action_script]</code></td>
<td>Specifies the action program for the application. An action program has start, stop, and check entry points that are called by the Oracle Clusterware. You can specify either a full path name for the script file or its filename.</td>
</tr>
<tr>
<td><code>[-B executable_pathname]</code></td>
<td>Specifies the location of the application executable and causes the <code>crs_profile</code> command to generate an action program file that contains the application executable path name.</td>
</tr>
<tr>
<td><code>[-d directory]</code></td>
<td>Specifies the file location.</td>
</tr>
</tbody>
</table>
The Oracle Clusterware Commands

**Table B–3 (Cont.) crs_profile Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[-d description]</td>
<td>Specifies the description of the application. Enclose the description in double quotation marks (&quot;&quot;&quot;) if it contains white space. If you do not specify an application description, then the Oracle Clusterware uses the application name.</td>
</tr>
<tr>
<td>[-p placement_policy]</td>
<td>Specifies the policy according to which the Oracle Clusterware selects the node on which to start or restart the application. You can specify either PREFERRED or UNAVAILABLE as a placement policy. All policies except for balanced require you to also specify the -h flag.</td>
</tr>
<tr>
<td>[-h hosting_nodes]</td>
<td>Specifies an ordered list of nodes, separated by white space, that can host the application. If there is more than one node, then the list must be enclosed in double quotation marks (&quot;&quot;&quot;). If you specify the -p option with a placement policy of preferred or unavailable, then you must also specify the -h option.</td>
</tr>
<tr>
<td>[-r required_resources]</td>
<td>Specifies an ordered list of resources, separated by spaces, on which the application depends. These resources must be active on any node on which the application is running. If there are multiple required resources, then the list must be enclosed in double quotation marks (&quot;&quot;&quot;). If you do not specify a required resources list, then the Oracle Clusterware imposes no required dependencies upon the application resource.</td>
</tr>
<tr>
<td>[-l optional_resources]</td>
<td>Specifies an ordered list of optional resources, separated by white space. If there are multiple optional resources specified, then the list must be enclosed in double quotation marks (&quot;&quot;&quot;).</td>
</tr>
<tr>
<td>[-o option[,...]]</td>
<td>Comma-separated list of attribute option names and their values. To see the list of attribute option names, run the crs_profile -help command.</td>
</tr>
<tr>
<td>-f</td>
<td>Forces relocation of the specified applications, all applications dependent on them and all applications that they are dependent upon. This option is required for relocating applications that require another application or one that is required by any ONLINE application.</td>
</tr>
</tbody>
</table>

**Table B–4 Options for the -o Flag**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>st=script_timeout</td>
<td>Sets the maximum time for an action program to run. The Oracle Clusterware commands and the Oracle Clusterware daemon return an error message and post an event to Event Manager (EVM) when they invoke action program entry points and the script does not complete its execution within this time. The default is 60 seconds.</td>
</tr>
<tr>
<td>ap=active_placement</td>
<td>When set to 1, it re-evaluates the placement policy when a new cluster node becomes available. With placement policies of FAVORIED and RESTRICTED, a highly available application relocates to a more highly favored cluster node. The default is 0.</td>
</tr>
<tr>
<td>as=auto_start</td>
<td>When set to 1, the application automatically starts after a cluster restart, regardless of whether it had been stopped or running before the restart. When set to 0, automatically starts the application only if it had been running before the restart. The default is 0.</td>
</tr>
<tr>
<td>ci=check_interval</td>
<td>Sets the time (in seconds) at which the check entry point of the application’s action program runs. The check interval is the maximum amount of time that an application can be unavailable to clients before the Oracle Clusterware attempts to restart it. The default check interval is 60 seconds.</td>
</tr>
<tr>
<td>ft=failure_threshold</td>
<td>Specifies the number of times that the Oracle Clusterware may detect an application or application resource failure within the failure interval before it marks the application or application resource as unavailable and stops monitoring it. The value must be in the range 0–20. Setting the value to 0 (zero) turns off failure threshold monitoring. If you do not specify a failure threshold, then the Oracle Clusterware uses a default failure threshold of 0 (zero).</td>
</tr>
</tbody>
</table>
The following is an example of using the `crs_profile` command to create the application profile for an application named `dtcalc`:

```
crs_profile -create dtcalc
```

The following is an example of using the `crs_profile` command to validate the application profile named `dtcalc`:

```
crs_profile -validate dtcalc
```

If you do not specify either the `-a` or `-B` options, then the profile is created without an action program value. You should create a readable, executable script and update the profile with this script's value before attempting to start the application. If you specify the `-a` option alone, then the action program specified must exist at the specified location or in the default directory if no path is specified. Otherwise, the command fails.

**Errors for crs_profile**

This corresponding text message is given if the command fails:

```
A failure has occurred.
```

**Examples of crs_profile**

The following is an example of using the `crs_profile` command to create the application profile for an application named `dtcalc`:

```
crs_profile -create dtcalc
```

The following is an example of using the `crs_profile` command to validate the application profile named `dtcalc`:

```
crs_profile -validate dtcalc
```

If you do not specify either the `-a` or `-B` options, then the profile is created without an action program value. You should create a readable, executable script and update the profile with this script's value before attempting to start the application. If you specify the `-a` option alone, then the action program specified must exist at the specified location or in the default directory if no path is specified. Otherwise, the command fails.

**Errors for crs_profile**

This corresponding text message is given if the command fails:

```
A failure has occurred.
```

**crs_register**

The `crs_register` command registers one or more applications specified with the `resource_name` parameter for each application. This command requires write access to the target application. This command only succeeds if the profile is found either in the default location or in the directory specified by the `-dir` option. An application must be registered in order for the Oracle Clusterware to monitor the resource or to start, restart, or relocate a highly available application associated with an application resource. An application registration must be updated for any changes to an application profile to take effect. This command requires full administrative privileges.

An application can be registered or updated only if the Oracle Clusterware daemon is active and an Oracle Clusterware application profile exists in the profile directory for this application. If fields are missing from an application profile, then the profile is

---

### Table B–4 (Cont.) Options for the -o Flag

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fi=failure_interval</td>
<td>Specify the time (in seconds) during which the failure threshold is calculated and applied. The default failure interval is 0 (zero). The Oracle Clusterware uses a default failure interval of 0, which turns off monitoring of failure threshold and failure intervals. Specifying a nonzero failure interval has no effect unless the failure threshold is also nonzero.</td>
</tr>
<tr>
<td>ra=restart_attempts</td>
<td>Specifies the number of times that the Oracle Clusterware attempts to restart the application or application resource on the current node before attempting to relocate it. The default number of restart attempts is 1.</td>
</tr>
<tr>
<td>fd=failover_delay</td>
<td>Specifies the number of seconds that the Oracle Clusterware waits before attempting to relocate the application resource. An application resource that was running on a cluster node that failed restarts immediately on that node if it becomes available again within the failover delay period. If application resources are dependent on each other due to required resources defined in application profiles, then all interdependent application resources wait for the largest failover delay that you have defined for any of the resources. The default failover delay is 0 (zero) seconds.</td>
</tr>
</tbody>
</table>
merged with the default profile template and Oracle uses the values in the default profile template.

The ownership and default permissions for the application are set during the registration process. You can register any .cap file as long as the permissions for the file permit read and write, for example, crs_profile and crs_register must be able to read the file. By default, the user who registers the application is the owner of the application. If the profile cannot be registered, then the Oracle Clusterware displays messages explaining why. Use the crs_stat command to verify that the application is registered.

Syntax and Options for crs_register
You can use the crs_register command to register and update applications. Use the following crs_register syntax to register an application:

```
crs_register resource_name [-dir directory_path] [...] [-u] [-f] [-q]
```

The resource_name [...] parameter can be the name of one or more application resources as specified in an application profile. If you do not specify any options, then the crs_relocate command relocates each specified application resource according to its placement policy and required resource lists. The Oracle Clusterware does not relocate a resource if there are interdependent application resources unless you specify the -f option. A profile must exist for the application that you are registering.

The -dir option specifies where the .cap file is if it is not in the default directory.

Use crs_register -u immediately after a crs_profile -update or a manual edit of an application profile to ensure that the changes take effect immediately.

### Table B–5  crs_register Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-c cluster_node</td>
<td>Relocates each indicated application or application resource to the specified node regardless of its placement policy. If required resources are not available on the destination node or if the application resource is restricted from that node, then the crs_relocate command fails and the application or application resource remains on the current node.</td>
</tr>
<tr>
<td>-s source_node</td>
<td>Relocates all running applications or application resources from the source_node. If you do not also specify the -c option, then the crs_relocate command relocates each resource according to its placement policy and required resource lists.</td>
</tr>
<tr>
<td>-f</td>
<td>Forces relocation of the specified applications, all applications dependent on them and all applications that they are dependent upon. This option is necessary for relocating any application that requires another application or one that is required by any ONLINE application.</td>
</tr>
</tbody>
</table>

Use the following crs_register command syntax to update a registered application:

```
crs_register resource_name -update [option ...] [-o option,...] [-q]
```

See Also: Table B–3 and Table B–4 for the listing of options and their definitions for using -update; the options for -update are the same as for the crs_profile command.

### Example of crs_register
The following example registers an application named postman for which an application profile exists in the CRS home/crs/profile directory:
The Oracle Clusterware Commands

CRS home/bin/crs_register postman

Note: The profile will be in the crs/profile directory if the profile is created by the root user. The profile will be in the crs/public directory if the profile is created by any other user.

Errors for crs_register
Oracle displays a corresponding text message for the following error conditions:

- No root privilege
- The Oracle Clusterware daemon is not running
- Resource specified on the command line does not match the resource specified in the profile
- Resource is already registered with the Oracle Clusterware daemon
- Resource type cannot be updated

Related Commands for crs_register
crs_stat, crs_profile, crs_relocate, crs_start, crs_stop, crs_unregister

crs_relocate
The crs_relocate command relocates applications and application resources as specified by the command options that you use and the entries in your application profile. The specified application or application resource must be registered and running under the Oracle Clusterware in the cluster environment before you can relocate it. The command displays a message if you specify a cluster node that is unavailable or if the attempt to relocate failed. You must have full administrative privileges to use this command. When you perform a crs_relocate command, the Oracle Clusterware first runs the stop entry point of the action program on the node on which it is currently running. The Oracle Clusterware then performs the start entry point of the action program to start it on a new node.

If the Oracle Clusterware fails to stop the application or application resource on the current node due to an action program error, then it marks it as UNKNOWN. You cannot run crs_relocate on a resource in this state. Instead, run a crs_stop -f command on the resource and restart it with crs_start to return it to the ONLINE state before you attempt to relocate it again. If the Oracle Clusterware fails to restart an application resource, then you may need to check the resource action program.

If the action program start entry point fails to run successfully, then the stop entry point is run. If the stop entry point fails to run successfully, then the state is marked as UNKNOWN and relocation attempts are aborted. If the stop entry point succeeds, then the state is set to OFFLINE. The target state remains ONLINE however, so subsequent cluster node failures or restarts can cause the Oracle Clusterware to attempt to restart the application. If you have not specified the node to which to relocate it and if there are available cluster nodes that satisfy the placement criteria, then the Oracle Clusterware attempts to start the application on one of these available nodes.

If one or more user-defined attributes have been defined for application resources, then you can specify values for these attributes when relocating an application with
the crs_relocate command. The specified value is passed to the action program as an environment variable with the attribute name.

The actions that the Oracle Clusterware takes while relocating an application resource are echoed on the command line. You can also monitor them using the Event Manager (EVM). Standard error and standard output from a resource action program that is started by the crs_relocate command are redirected to the standard error and standard output for crs_relocate. Note that if the Oracle Clusterware daemon starts an application, then standard error and standard output of the action program is lost. Within an action program, you can check for user invocation of the action program using reason codes.

**Syntax and Options for crs_relocate**

Use the crs_relocate command with the following syntax:

```
crs_relocate resource_name [...] [-c cluster_node] [-f] [-q]
```

```
crs_relocate resource_name [-c cluster_node] [-q]
```

```
crs_relocate [USR_attribute_name=value] [...] resource_name [-c cluster_node] [-q]
```

```
crs_relocate -s source_node [-c cluster_node] [-q]
```

**Table B–6 crs_relocate Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ls</td>
<td>Lists resources, owners, and permissions for all resources.</td>
</tr>
<tr>
<td>-ls resource_name</td>
<td>Lists resources, owners, and permissions for a specified resource.</td>
</tr>
<tr>
<td>-t</td>
<td>Lists information for all resources in a tabular form.</td>
</tr>
<tr>
<td>-v</td>
<td>Lists how many times a resource has been restarted or has failed within the resource failure interval; the maximum number of times that a resource can be restarted or can fail; and the target state of the application. Also lists normal status information.</td>
</tr>
</tbody>
</table>

**Example of crs_relocate**

The following example relocates an application resource to the node known as rac1:

```
crs_relocate postman -c rac1
Attempting to stop 'postman' on node 'rac2'
Stop of 'postman' on node 'rac1' succeeded
Attempting to start 'postman' on node 'rac1'
Start of 'postman' on node 'rac1' succeeded
```

The following example attempts to relocate all application resources from node rac2 to node rac1:

```
crs_relocate -s rac2 -c rac1
Attempting to stop 'postman' on node 'rac2'
Stop of 'postman' on node 'rac2' succeeded.
Attempting to start 'postman' on node 'rac1'
Start of 'postman' on node 'rac1' succeeded.
Attempting to stop 'calc' on node 'rac2'
Stop of 'calc' on node 'rac2' succeeded.
Attempting to start 'calc' on node 'rac1'
Start of 'calc' on node 'rac1' succeeded.
```
If a user-defined attribute `USR_DEBUG` has been defined, then the following example runs the stop and start entry point of the action program with the `USR_DEBUG` environment variable set to `FALSE`. This overrides any value set in the application profile. In the corresponding action program, if you add the following line to the appropriate section of the action program, then you can view the value:

```
echo $USR_DEBUG
```

Then run the following command:

```
# crs_relocate USR_DEBUG=false database
```

**Errors for crs_relocate**

Oracle displays a corresponding text message for the following error conditions:

- No root privilege
- Application is not running
- Cluster node intended for relocation is not available
- Relocation failed

**crs_setperm**

Modifies the permissions associated with a resource. This command is similar to the `chmod` command in UNIX-based systems or the Windows desktop options, in this order: File, Properties, Security, and Permissions.

**Syntax and Options for crs_setperm**

Use the `crs_setperm` command with the following syntax:

```
crs_setperm resource_name -u aclstring [-q]
crs_setperm resource_name -x aclstring [-q]
crs_setperm resource_name -o user_name [-q]
crs_setperm resource_name -g group_name [-q]
```

In the previous example syntax, `-u` updates the acl string, `-x` deletes the acl string, `-o` changes the owner of the resource, and `-g` changes the primary group of the resource, and `aclstring` is one of the following:

- `user:username:rwx`
- `group:groupname:r-x`
- `other::r--`

**Example of crs_setperm**

The following example modifies the permissions on the `admin1` user for the `postman` application:

```
crs_setperm postman -u user:admin1:r-x
```

**crs_stat**

The `crs_stat` command provides status information for resources on the cluster nodes. To query resources with the `crs_stat` command, resource files must have
read and execute permissions (r and x permissions on UNIX-based systems). An exception is the -g option, that anyone can use to verify whether a resource exists.

Resources are either ONLINE or OFFLINE as shown in the STATE attribute. An application resource in the ONLINE state is running successfully on a cluster node. This cluster node is shown indicating its state.

The TARGET value shows the state to which the Oracle Clusterware attempts to set the resource. If the TARGET value is ONLINE and a cluster node fails, then the Oracle Clusterware attempts to restart the application on another node if possible. If there is a condition forcing a resource STATE to be OFFLINE, such as a required resource that is OFFLINE, then the TARGET value remains ONLINE and the Oracle Clusterware attempts to start the application or application resource once the condition is corrected.

A TARGET value for all non-application resources should be ONLINE unless the resource has a failure count higher than the failure threshold, in which case the TARGET is changed to OFFLINE. The Oracle Clusterware then treats the resource as if its STATE were OFFLINE. If the STATE is ONLINE and the TARGET is OFFLINE, then you can reset the target value to ONLINE using the crs_start command.

The verbose status -v gives additional information that may be useful, especially for troubleshooting. The RESTART_COUNT value shows how many times an application resource has been restarted on a single cluster node. The maximum number of restarts before the Oracle Clusterware stops restarting the application is equal to RESTART_ATTEMPTS. FAILURE_COUNT shows the number of times that a resource has failed within the FAILURE_INTERVAL defined in the application profile. The maximum number of failures before the Oracle Clusterware stops restarting an application is equal to the value set for the FAILURE_THRESHOLD parameter. If a cluster node fails and applications are waiting to be relocated due to the profile FAILOVER_DELAY attribute, then the verbose status also shows the FAILOVER_STATUS field. The FAILOVER_STATUS field is not shown at any other time. The FAILOVER_STATUS field shows the node the application failed on and how much time is left waiting for that node to restart before restarting on another node.

Syntax and Options for crs_stat

Use the crs_stat command with the following syntax:

crs_stat [resource_name [ ... ]] [-v] [-l] [-q] [-c cluster_node]
crs_stat [resource_name [ ... ]] -t [-v] [-q] [-c cluster_node]
crs_stat -p [resource_name [ ... ]] [-q]
crs_stat [-a] resource_name -g
crs_stat [-a] resource_name -r [-c cluster_node]
crs_stat -f [resource_name [ ... ]] [-q] [-c cluster_node]
crs_stat -ls resource_name

To view information about all resources, enter the command with no arguments.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ls</td>
<td>Lists resources, owners, and permissions for all resources.</td>
</tr>
</tbody>
</table>
The Oracle Clusterware Commands

### Table B–7 (Cont.) crs_stat Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ls  resource_name</td>
<td>Lists resources, owners, and permissions for a specified resource.</td>
</tr>
<tr>
<td>-t</td>
<td>Lists information for all resources in a tabular format.</td>
</tr>
<tr>
<td>-v</td>
<td>Lists how many times a resource has been restarted or has failed within the resource failure interval, the maximum number of times that a resource can be restarted or can fail, and the target state of the application. Also lists normal status information.</td>
</tr>
</tbody>
</table>

- **-v**— Displays extended information about the status of resources. Extra attributes shown include `RESTART_COUNT` and `FAILURE_COUNT`. The `RESTART_COUNT` attribute shows the number of restarts of the application that have been attempted since it was started. The `FAILURE_COUNT` attribute shows how many failures have occurred during the last `FAILURE_THRESHOLD` in seconds. The attribute `FAILOVER_STATUS` shows the time at which an application resource started while waiting to relocate due to a cluster node failure if the resource has a `FAILOVER_DELAY` value greater than 0. It is not displayed otherwise.

- **-l**— Displays the status in a list (non-tabular) format. This is the default format.

- **-t**— Displays the status in tabular format.

- **-p resource_name [...]**— Displays the status of the in-memory profile for the specified resources. If no resources are specified, then the status of the in-memory profile for all registered resources is displayed. If a resource is not registered, its status is not displayed.

- **-a resource_name**— Used with the `-g` or `-r` option to verify whether the specified resource is registered or running under the Oracle Clusterware. Specify the name of the resource as listed in its application profile.

- **-f**— Displays all information about the resource including extended information (`-v`) and the in-memory profile (`-p`).

- **-g**— Returns 0 (zero) if the specified application or application resource is registered with the Oracle Clusterware; returns 1 if it is not. This option only works if a single resource is specified with the `-a` option.

- **-r**— Returns 0 (zero) if the specified application or application resource is running under the Oracle Clusterware; returns 1 if it is not. This option can only be successful if a single resource is specified with the `-a` option.

- **-c cluster_node**— Displays information about applications or application resources on the specified cluster node.

### Examples of crs_stat

The following example displays the status information for the `postman` application:

```
crs_stat postman
```

```
NAME=postman
TYPE=application
TARGET=ONLINE
STATE=ONLINE on rac2
```

The following example displays the status information in a tabular form:

```
crs_stat -t
```

```
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Target</th>
<th>State</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
cluster_lockd application ONLINE ONLINE rac2
dhcp     application OFFLINE OFFLINE

The following example shows the output for the command `crs_stat` with the `-v` option for the node `rac2`:

```
crs_stat -v
```

```
NAME=cluster_lockd
TYPE=application
RESTART_ATTEMPTS=30
RESTART_COUNT=0
FAILURE_THRESHOLD=0
FAILURE_COUNT=0
TARGET=ONLINE
STATE=ONLINE on rac2
```

**crs_start**

The `crs_start` command sets an application or application resource target state to `ONLINE` and attempts to start specified registered applications or application resources. The target state is the state that the Oracle Clusterware attempts to achieve. You must have full administrative privileges to use this command.

There are several user-defined attributes available for starting an application with `crs_start`. The specified value is passed to the action program as an environment variable with the attribute name.

If you do not specify one node on which to start the application and there are available cluster nodes that satisfy the placement criteria, then the Oracle Clusterware attempts to start the application on one of the available nodes. You must stop a resource that has failed before restarting it. You should also confirm why your action program is failing.

The `crs_start` command starts dependent resources as defined in the application profile `REQUIRED_RESOURCE` fields. After a registered resource is started, you must use the `crs_stop` command to end its execution.

The `crs_start` command displays status messages and if there are problems starting an application, then feedback is displayed. If network, tape, or media changer resources are defined as required resources for an application or as an application resource that is started, then messages stating that the Oracle Clusterware is attempting to start these non-application resources appear. The Oracle Clusterware is not actually attempting to start these resources, but is testing the state of the devices. You can ignore these messages.

Standard error and standard output from a resource action program invoked by `crs_start` are redirected to the standard error and standard output for `crs_start`. Note that if the Oracle Clusterware daemon starts an application, then standard error and standard output of the action program is lost. Within an action program, you can check for user invocation of the action program using reason codes.

**Syntax and Options for crs_start**

```
crs_start resource_name [...] [-c cluster_node] [-q] [-f]
crs_start -all [-q]
crs_start [USR_attribute_name=value] [...] resource_name [-c node_name] [-q]
```
For resource_name [...], the names are as specified in an application profile of one or more resources to be started. The resource must be registered with the Oracle Clusterware. You can use the following options:

- `-c node_name`—Starts each indicated resource on the specified node if the cluster node is enabled by the placement policy and resource dependencies. If the cluster node specified is not enabled by the placement policy and resource dependencies, then the `crs_start` command fails. If the specified node is not available, the command fails. If one of the specified resources fails to start, then the command still attempts to start other resources listed on the command line.

- `-all`—Starts all registered Oracle Clusterware applications or application resources on active cluster nodes, according to their placement policies and required resource lists.

- `-q`—Runs the `crs_start` command in quiet mode; no messages are displayed.

- `-f`—Forces starting of a specified application or application resource if all of the required resources are available or can be started. Any applications or application resources in the specified application profile's required resource list are started if not currently running or are relocated if they are running on another cluster node. Use this option with only one specified application resource at a time.

Examples of `crs_start`

The following example starts an application named `postman`:

```
crs_start postman
Attempting to start `postman` on node `rac1`
Start of `postman` on node `rac1` succeeded.
```

The following example starts the application on a specific cluster node `rac2`:

```
crs_start -c rac2 postman
Attempting to start `postman` on node `rac2`
Start of `postman` on node `rac2` succeeded.
```

Errors for `crs_start`

A failure has occurred. Oracle displays a corresponding text message for the following error conditions:

- No root privilege
- The Oracle Clusterware daemon is not running
- The application is running
- A start action program is not found
- The cluster node was specified and it was not available
- A timeout was reached before the startup was unsuccessful

**crs_stop**

Stops an application on the specified node. You can run the `crs_stat` command to verify that the application you specify is stopped.

**Syntax and Options for crs_stop**

Use the `crs_stop` command with the following syntax:
crs_stop resource_name [...] [-f] [-q]
crs_stop -c cluster_node [...] [-q]
crs_stop -all [-q]
crs_stop [USR_attribute_name=value] [...] resource_name [-q]
-c cluster_node [...]  

Specifies one or more active cluster nodes on which to stop all of the applications or application resources. You can use the following options:

- `-all`—Stops all applications or application resources on all active cluster nodes.
- `-f`—Forces termination of the indicated applications or application resources and all application resources dependent on the specified resources. This option is useful to stop all application resources when the crs_stop command fails because of application resources that are dependent on the specified resource or if the application is in the unknown state because of an unsuccessful stop. This option makes crs_stop ignore any errors that are reported by an action program.

**Examples of crs_stop**

To stop an application named dtcalc, enter the following command:

crs_stop dtcalc

The Oracle Clusterware displays status messages as follows:

Attempting to stop `dtcalc` on node `rac2`
Stop of `dtcalc` on node `rac2` succeeded.

---

crs_unregister

The crs_unregister command removes the registration information of Oracle Clusterware resources from the binary Oracle Clusterware registry database. The Oracle Clusterware will no longer acknowledge this resource. An application associated with a resource that is unregistered is no longer highly available. You must have full administrative privileges to use this command.

Upon successful completion of the crs_unregister command, the resource is removed from the online Oracle Clusterware environment. You cannot unregister a resource that is a required resource for another resource. You must stop the resource by using the crs_stop command before unregistering it.

**Syntax and Options for crs_unregister**

Use the crs_unregister command with the following syntax:

```
crs_unregister resource_name [...] [-q]
```

The only option available for this command is `-q`, that runs the crs_unregister command in quiet mode, which means no messages are displayed.

**Example of crs_unregister**

The following example unregisters a highly available application called postman:

```
crs_unregister postman
```
Errors for crs_unregister
Oracle displays a corresponding text message for the following error conditions:

- No root privilege
- CAA daemon is not running
- The application is running
- The application is not registered

See Also: Chapter 13, "Making Applications Highly Available Using the Oracle Clusterware" for information about using the Oracle Clusterware commands

C Application Programming Interface to the Oracle Clusterware

The APIs described in this section provide access to operational control of resources that the Oracle Clusterware manages. The API is used to register user applications to the Oracle Clusterware system so that they can be managed by the Oracle Clusterware and made highly available. When an application is registered, the application can be started and its state queried. If the application is no longer to be run, it can be stopped and unregistered from the Oracle Clusterware. The Oracle Clusterware services are provided by another process, such as the crsd process, running outside the database server. These APIs communicate with the crsd process using an IPC mechanism.

cIscrs_init_crs

Initializes a context for communications with the Oracle Clusterware.

Parameters

- **ctx**—Context to initialize, caller allocated storage
- **errf**—A Callback function for the error info, or NULL
- **errCtx**—Context for the error callback func, or NULL
- **flags**—CLSCRS_FLAG_TRACE, to enable tracing of lower layers
- **CLSCRS_FLAG_USETHREADS**—Enables using the clscrs context in multiple threads

Returns
Status enum.

Syntax

CLSCRS_STAT clscrs_init_crs(clscrs_ctx **ctx, clscrs_msgf errf, void *errCtx, ub4 flags);

cIscrs_term_crs

Releases a context for communications with the Oracle Clusterware.

Parameter

**ctx**—Context previously initialized with clscrs_init_crs.
Returns
status enum

Syntax
CLSCRS_STAT clscrs_term_crs( clscrs_ctx **ctx);

clscrs_getnodename
Returns the correct node name. If you issue this command within a configured cluster, then this is the name as known to the clusterware. If not in a cluster, then the name is as known through other clusterware means. Getting a node name may require a lot of resources in the first call, especially if CLSCRS was not initialized with a pre-existing CSS context.

Parameters
■ ctx—Context previously initialized with clscrs_init
■ node_name—Filled in node name

Syntax
CLSCRS_STAT clscrs_getnodename(clscrs_ctx *ctx, oratext *node_name);

clscrs_env_create
Creates a call environment describing the parameters to the Oracle Clusterware calls. The memory is owned by clscrs which is released by a call to clscrs_env_delete.

Parameters
■ context—Input context
■ env—Output environment pointer

Returns
status enum

Syntax
CLSCRS_STAT clscrs_env_create( clscrs_ctx *ctx, clscrs_env *env );

clscrs_env_set
Sets an attribute or value argument in an environment for the Oracle Clusterware calls; the value may be NULL.

Parameters
■ env—Previously created environment
■ attr—Attribute name
■ val—Attribute value, may be NULL pointer
Returns
environment status enum

Syntax
CLSCRS_ENVCODE clscrs_env_set(clscrs_env env, const oratext *attr, const oratext *val);

clscrs_env_delete
Deletes an environment for the Oracle Clusterware calls. The environment must be created again before use. You cannot delete an environment and then set new parameters.

Parameters
■ env—The environment whose contents are deleted

Returns
environment status enum.

Syntax
CLSCRS_ENVCODE clscrs_env_delete(clscrs_env env);

clscrs_env_format
Creates a callback with formatted environment lines.

Parameters
■ env—Env to dump
■ msgf—Callback function
■ msgp—Argument to msgp, uninterpreted

Returns
None.

Syntax
void clscrs_env_format( clscrs_env env, clscrs_msgf msgf, void *msgp );

clscrs_start_resource
Direct the Oracle Clusterware to start a named set of resources. If a host is specified, then start there, otherwise start according to the placement policy for the resources in question. If the flag is async, then msgf is never called and Oracle returns the OK status after initiating the call to the Oracle Clusterware, and the actual starts are performed asynchronously, and msgf will never be called. If flag is not async and msgf is not NULL, then msgf will be driven one line at a time with collected output from the start programs.
Parameters
- ctx—The context
- names—The resource(s) to start
- num_names—The number of resources in names[] to start
- host—The host name on which to start the resource, may be NULL
- env—Environment arguments to start
- msgf—User message callback, may be NULL
- msgp—User callback argument, may be NULL
- msgf—Callback function
- msgf—Callback function

Returns
status enum

Syntax
CLSCRS_STAT clscrs_start_resource( clscrs_ctx *ctx, const oratext *name[], sword num_names, const oratext *host, clscrs_env env, clscrs_msgf msgf, void *msgp, uword flag );

clscrs_stop_resource

Direct the Oracle Clusterware to stop a named set of resources. The flag enables async stop.

Parameters
- ctx—the context
- names—The resource(s) to stop
- num_names—Number of resources in names[] to stop
- flag—Async, force options

Returns
status enum

Syntax
clscrs_stop_resource( clscrs_ctx *ctx, const oratext *names[], sword num_names, clscrs_env env, clscrs_msgf msgf, void *msgarg, uword flag );

clscrs_check_resource

Direct the Oracle Clusterware to run the check actions for the identified resources. A delay can be specified to indicate a time to wait in seconds before the check action is performed on the server. This occurs after the call returns. This can be used to let something run before performing the check. For each resource identified by the insplist, a status will be returned in the op_status, indicating whether the resource exists and whether the caller has permission to invoke operations on the resource. A
resource that is explicitly named with no pattern matching will have a NOTFOUND error. A pattern that has no matches will return no error. There will be no valid attributes returned for the resources. There is no way of knowing exactly when the checks will actually be run or be completed. Returns SUCCESS if the operation status of all of the identified resources is SUCCESS, otherwise FAILURE and specific status will be found in the op_status entries. The caller must create the op_status list before the call and destroy it when done.

Parameters
- in_splist [in]—List of resources to check
- delay_secs [in]—Time to wait before the check is run, as appropriate to the resource
- flags [in]—None
- op_status [out]—Resource list structure holding the status of the check operation for each resource

Returns
status enum.

Syntax
clscrs_check_resource( clscrs_splist *in_splist, ub4 delay_seconds, uword flags, clscrs_reslist *op_status);

clscrs_register_resource

Register the resources in the input resource list. The attributes for the resources are encapsulated in the input resource list. The output op_status list contains the results of the register operation for each resource and contains no valid resource attributes.

The caller must create and populate the in_reslist and must create the op_status list. Both the lists must be destroyed by the caller. The op_status list cannot be reused with another API call, you must create and destroy it for each API call.

One or more attributes of an already registered resource can be modified by passing the CLSCRS_FLAG_REG_UPDATE flag. The flags apply to all resources in the input reslist.

Parameters
- in_reslist [in]—List of resources to be registered
- flags [in]—CLSCRS_FLAG_REG_UPDATE or 0
- op_status [out]—Resource list holding the status of the register operation for each resource

Returns
- CLSCRS_STAT_SUCCESS—If all input resources are successfully registered
- CLSCRS_STAT_FAIL—If atleast one resource could not be registered
- CLSCRS_STAT_CONNECTION—If there is a communication error with the crsd process
Syntax

clscrs_register_resource(clscrs_reslist *in_reslist, uword flags, clscrs_reslist *op_status);

clscrs_unregister_resource

Unregister the resources in the input list of resource names. The output op_status list contains the results of the unregister operation for each resource. The caller must create and populate the rqlist and must create the op_status list. Both the lists must be destroyed by the caller. The op_status list cannot be reused with another API call. You must create and destroy it for each API call.

Parameters

- rqlist [in] — List of resources names to be unregistered
- flags [in] — Option flags
- op_status [out] — Resource list holding the status of the unregister operation for each resource

Returns

- CLSCRS_STAT_SUCCESS — If all input resources are successfully unregistered
- CLSCRS_STAT_FAIL — If at least one resource could not be unregistered
- CLSCRS_STAT_CONNECTION — If there is a communication error with the crsd process

Syntax

clscrs_unregister_resource(clscrs_splist *rqlist, uword flags, clscrs_reslist *op_status);

clscrs_stat

Obtains information about the resources identified in rqlist. The resources can be restricted to a specific node given in the nodename parameter. The information is returned in the output out_reslist.

If the rqlist is NULL, information about all registered resources is returned. If the node name is not NULL, the result is restricted to resources running on the specific node. Otherwise, all nodes in the cluster are considered.

The res_flags are used to scope the resource domain that is matched. There are no values for this parameter. The res_attrs identifies the attributes to be returned in the out_reslist. The caller may create a specific list of attributes of interest or use a pre-defined list returned by one of the macros CLSCRS_ATTRS_NONE, CLSCRS_ATTRS_ALL, CLSCRS_ATTRS_STATE, CLSCRS_ATTRS_SHORT. The attribute list applies to all resources identified by the rqlist. Any errors for a resource are returned in the output out_reslist.

Parameters

- rqlist [in] — List of resource names to query
- nodename [in] — Query resources running only on the given node (optional, may be NULL)
Functions for Managing Resource Structures

- res_atts [in]—List to specify the attributes to be returned for each resource
- res_flags [in]—Flags to restrict the resources being matched
- out_reslist [out]—List holding the returned resource information

Returns
- CLSCRS_STAT_SUCCESS if information is obtained for all input resources
- CLSCRS_STAT_FAIL if no information is available for at least one resource
- CLSCRS_STAT_CONNECTION if there is an error communicating with the crsd process

Syntax
clscrs_stat(clscrs_splist *rqlist, oratext *nodename, uword res_flags, clscrs_splist *res_attrs, clscrs_reslist *out_reslist);

Functions for Managing Resource Structures

This section describes functions for creating, managing, and destroying resources and the attributes associated with resources. The following structures are used in this section:
- clscrs_sp—This is a stringpair (sp) structure. It contains a name and a value string. The value may be NULL. It is created and destroyed and its contents can be examined and the value replaced. An sp may be a member of exactly one stringpair list (splist).
- clscrs_splist—A stringpair list (splist) is a list of zero or more stringpairs used in various contexts. Sometimes the list contains names and values, sometimes the list contains only names. The list is created and destroyed; members can be retrieved by name, added, deleted, replaced, and the list can be traversed with first and next operations.
- clscrs_res—This represents a resource abstraction that contains the name of a resource and additional data about the context in which it is used. Sometimes it contains resource attribute data and other times it contains status and return message about an operation. A resource may be in exactly one resource list.
- clscrs_reslist—A resource list is an abstraction used to contain information about zero or more resources. A resource list is created, appended, iterated over, and destroyed.

Export Operations

This section describes the following export operations:
- Stringpair Operations
- Splist Operations
- Resource Operations
- Resource List Operations

Stringpair Operations
This section describes stringpair operations.
clsrcs_sp_set  Change the value part of a stringpair. The new value may be NULL. After the call returns, the memory of the value can be recycled.

Parameters
- sp [in]—Stringpair to set the value for
- value [in]—Value component of the stringpair (may be NULL)

Returns
- clsrcsretSUCC on success
- clsrcsretBADARG if the sp argument is NULL

Syntax
clsrcsret clsrcs_sp_set(clsrcs_sp *sp, const oratext *value);

clsrcs_sp_get  Get the name and value components of a stringpair.

Parameters
- sp [in]—Stringpair for which the name and value are being obtained
- name [out]—Name component of the stringpair
- value [out]—Value component of the stringpair (may be NULL)

Returns
- clsrcsretSUCC on success
- clsrcsretNOMEM if no memory can be allocated
- clsrcsretBADARG if the sp argument is NULL

Syntax
clsrcsret clsrcs_sp_get(clsrcs_sp *sp, oratext **name, oratext **value);

clsrcs_sp_get_value  Get the value component of a stringpair.

Parameters
- sp [in]—Stringpair for which the name and value are being obtained
- value [out]—Value component of the stringpair (may be NULL)

Returns
- clsrcsretSUCC on success
- clsrcsretBADARG if the sp argument is NULL

Syntax
clsrcsret clsrcs_sp_get_value(clsrcs_sp *sp, oratext **value);

Splists Operations
This section describes the splist operations.

clsrcs_splist_create  Create a new stringpair list. The memory for the splist is allocated by the function.

Parameters
Functions for Managing Resource Structures

- **ctx [in]**—clscrs context
- **splist [out]**—The new string list created

**Returns**
- **clscrsretSUCC** on success
- **clscrsretNOMEM** if no memory can be allocated
- **clscrsretBADCTX** if the context is NULL

**Syntax**
```
clscrsret clscrs_splist_create(clscrs_ctx *ctx, clscrs_splist **splist);
```

**clscrs_splist_create_and_set** Create a new string list (splist) and set the name and value for the first stringpair in the list. The memory for the splist is allocated by the function.

**Parameters**
- **ctx [in]**—clscrs context
- **name [in]**—Name component of the stringpair
- **value [in]**—Value component of the stringpair (may be NULL)
- **sp [out]**—The new stringpair created

**Returns**
- **clscrsretSUCC** on success
- **clscrsretNOMEM** if no memory can be allocated
- **clscrsretBADCTX** if the context is NULL
- **clscrsretBADARG** if the name argument is NULL

**Syntax**
```
clscrsret clscrs_splist_create_and_set(clscrs_ctx *ctx, const oratext *name, const oratext *value, clscrs_splist **splist);
```

**clscrs_splist_append** Add a new stringpair (sp) to a stringpair list (splist).

**Parameters**
- **splist [in]**—splist to add the new {name, value} stringpair.
- **name [in]**—Name component of the stringpair
- **value [in]**—Value component of the stringpair (may be NULL)

**Returns**
- **clscrsretSUCC** on success
- **clscrsretNOMEM** if no memory can be allocated
- **clscrsretBADARG** if the name argument is NULL

**Syntax**
```
clscrsret clscrs_splist_append(clscrs_splist *splist, const oratext *name, const oratext *value);
```
**clscrs_splist_first** Get the first stringpair (sp) from a stringpair list (splist).

Parameters
- name [in]—splist to get the first sp from
- sp [out]—The first sp from the given splist

Returns
- clscrsretSUCC on success
- clscrsretBADARG if the splist is NULL
- clscrsretEMPTY if there are no stringpair elements in the splist

Syntax
```
c1scrsret clscrs_splist_first(clscrs_splist *splist, clscrs_sp **sp);
```

**clscrs_splist_next** Get the current next stringpair (sp) from a stringpair list (splist). This function is called to iterate over the stringpairs in a splist.

Parameters
- name [in]—splist to get the next sp from
- sp [out]—The next sp {name, value} in the given splist

Returns
- clscrsretSUCC on success
- clscrsretBADARG if the splist is NULL
- clscrsretENDLIST if there are no more stringpair elements in the splist

Syntax
```
c1scrsret clscrs_splist_next(clscrs_splist *splist, clscrs_sp **sp);
```

**clscrs_splist_replace** Replace the value for a stringpair (sp) in a stringpair list (splist).

Parameters
- splist [in]—Splist to add the new {name, value} stringpair.
- name [in]—Name for which the value is being replaced.
- value [in]—Replacement value for the name (may be NULL)

Returns
- clscrsretSUCC on success
- clscrsretBADARG if the name argument is NULL
- clscrsretBADARG if the name argument is NULL
- clscrsretBADARG if the name argument is NULL

Syntax
```
c1scrsret clscrs_splist_replace(clscrs_splist *splist, const oratext *name, const oratext *value);
```

**clscrs_splist_delete_sp** Delete a stringpair (sp) from a stringpair list (splist).

Parameters
Functions for Managing Resource Structures

- **splist** [in]—Splist from which to delete the (name, value) stringpair.
- **name** [in]—Name to be deleted from the given splist.

Returns
- clscrsretSUCC on success
- clscrsretNONAME if there is no stringpair matching the given name
- clscrsretBADARG if the name argument is NULL

Syntax
```
clscrsret clscrs_splist_delete_sp(clscrs_splist *splist, const oratext *name);
```

**clscrs_splist_find** Find the value for a stringpair (sp) in a stringpair list (splist).

Parameters
- **splist** [in]—Splist to look into
- **name** [in]—Name for which the value is looked up
- **value** [out]—Value associated with the given name in the given splist

Returns
- clscrsretSUCC on success
- clscrsretNONAME if there is no stringpair matching the given name
- clscrsretBADARG if the name argument is NULL

Syntax
```
clscrsret clscrs_splist_find(clscrs_splist *splist, const oratext *name, oratext **value);
```

**clscrs_splist_count** Count the number of stringpairs (sp) in a stringpair list (splist).

Parameters
- **splist** [in]—Splist to count the number of stringpairs.
- **count** [out]—The number of stringpairs in the given splist.

Returns
- clscrsretSUCC on success
- clscrsretBADARG if the splist is NULL

Syntax
```
clscrsret clscrs_splist_count(clscrs_splist *splist, ub4 *count);
```

**clscrs_splist_destroy** Free the memory for a stringpair list (splist).

Parameter
- **splist** [in]—Splist to destroy

Returns
- clscrsretSUCC on success

Syntax
Functions for Managing Resource Structures

clscrs_splist_destroy(clscrs_splist **splist);

**clscrs_res_create** Create a new resource. The memory for the resource structure is allocated by the function. The memory is freed when a resource list (clscrs_reslist) is destroyed through clscrs_reslist_destroy().

Parameters
- ctx [in]—clscrs context
- resname [in]—Name of the resource
- res [out]—The new resource created

Returns
- clscrsretSUCC on success
- clscrsretNOMEM if no memory can be allocated
- clscrsretBADCTX if the context is NULL
- clscrsretBADARG if the resource name is NULL

Syntax
clscrsret clscrs_res_create(clscrs_ctx *ctx, const oratext *resname, clscrs_res **res);

**clscrs_res_get_name** Get the name of a resource.

Parameters
- res [in]—Resource for which the name is obtained
- name [out]—Name of the resource

Returns
- clscrsretSUCC on success
- clscrsretBADARG if the resource argument is NULL

Syntax
clscrsret clscrs_res_get_name(clscrs_res *res, oratext **name);

**clscrs_res_set_attr** Set a resource attribute for a resource.

Parameters
- res [in]—Resource for which the attribute is set
- attrname [in]—Name of the resource attribute
- value [in]—Value for the resource attribute (may be NULL)

Returns
- clscrsretSUCC on success
- clscrsretBADARG if the attribute name is NULL
- clscrsretNOMEM if no memory can be allocated

Syntax
Functions for Managing Resource Structures

```c
clscrsret clscrs_res_set_attr(clscrs_res *res, const oratext *attrname, const oratext *value);
```

**clscrs_res_get_attr** Get a resource attribute for a resource.

**Parameters**
- res [in]—Resource for which the attribute is obtained
- attrname [in]—Name of the resource attribute
- value [out]—Value for the resource attribute

**Returns**
- clscrsretSUCC on success
- clscrsretBADARG if the attribute name is NULL
- clscrsretNOMEM if no memory can be allocated

**Syntax**
```c
clscrsret clscrs_res_get_attr(clscrs_res *res, const oratext *attrname, oratext **value);
```

**clscrs_res_get_attr_list** Get the attribute list for a resource. The attribute list is a list of string pairs. The client does not allocate the memory for attribute list.

**Parameters**
- res [in]—Resource for which the attribute list
- attrlist [out]—List of attributes for the given resource

**Returns**
- clscrsretSUCC on success
- clscrsretBADARG if the resource is NULL
- clscrsretNOMEM if no memory can be allocated
- clscrsretNOATTRS if there are no attributes set for the resource

**Syntax**
```c
clscrsret clscrs_res_get_attr_list(clscrs_res *res, clscrs_splist **attrlist);
```

**clscrs_res_set_attr_list** Set the attribute list for a resource. The attribute list is a list of string pairs. The list is created from the clscrs_splist_create call.

**Parameters**
- res [in]—Resource for which the attribute list is set
- attrlist [in]—List of attributes to be set for the given resource

**Returns**
- clscrsretSUCC on success
- clscrsretBADARG if the resource is NULL

**Syntax**
```c
clscrsret clscrs_res_set_attr_list(clscrs_res *res, clscrs_splist *attrlist);
```
clscrs_res_attr_count

Get the number of attributes for a resource.

Parameters
- res [in] — Resource for which number of attributes is obtained
- count [out] — Number of attributes for the given resource

Returns
- clscrsretSUCC on success
- clscrsretBADARG if the resource is NULL

Syntax
clscrsret clscrs_res_attr_count(clscrs_res *res, ub4 *count);

clscrs_res_get_op_status

Get the status of an operation for a resource. The memory for the msg is allocated by the function.

Parameters
- res [in] — Resource for which the operation status is obtained
- status [out] — Status of an operation on the given resource
- msg [out] — Text message for the result of an operation on the resource

Returns
- clscrsretSUCC on success
- clscrsretNOMEM if no memory can be allocated
- clscrsretNOMSG if there is no msg available
- clscrsretBADARG if the resource is NULL

Syntax
clscrsret clscrs_res_get_op_status(clscrs_res *res, CLSCRS_STAT *status, oratext **msg);

clscrs_res_get_registered

Get the registration status of a resource.

Parameters
- res [in] — Resource for which the operation status is set
- registered [out] — Boolean indicating whether the resource is registered

Returns
- clscrsretSUCC on success
- clscrsretNOMEM if no memory can be allocated
- clscrsretBADARG if the resource is NULL

Syntax
clscrsret clscrs_res_get_registered(clscrs_res *res, boolean *registered);
**clscrs_res_get_node_list**  Get the nodelist currently hosting the resource, or NULL if there is no host for the resource or there are no attributes. The caller need not allocate memory for the nodelist.

**Parameters**
- `res [in]`—Resource for which the nodelist is obtained
- `nodelist [out]`—Splist holding the node(s)

**Returns**
- `clscrsretSUCC` on success
- `clscrsretBADARG` if the resource is NULL

**Syntax**
```
clscrsret clscrs_res_get_node_list(clscrs_res *res, clscrs_splist **nodelist);
```

**clscrs_res_destroy**  Free the memory for a resource.

**Parameter**
- `res [in]`—Resource for which the memory is freed

**Returns**
- `clscrsretSUCC` on success

**Syntax**
```
clscrsret clscrs_res_destroy(clscrs_res **res);
```

**clscrs_reslist_create**  Create a new resource list. The memory for the resource list is allocated by the function.

**Parameters**
- `ctx [in]`—clscrs context
- `reslist [out]`—Resource list (empty) that is created

**Returns**
- `clscrsretSUCC` on success
- `clscrsretNOMEM` if no memory can be allocated
- `clscrsretBADCTX` if the context is NULL

**Syntax**
```
clscrsret clscrs_reslist_create(clscrs_ctx *ctx, clscrs_reslist **reslist);
```

**clscrs_reslist_append**  Add a resource to a resource list.

**Parameters**
- `reslist [in]`—Resource list to add the resource to
- `res [in]`—Resource to add

**Returns**
- `clscrsretSUCC` on success
Functions for Managing Resource Structures

- `clscrssetBADARG` if reslist is NULL
- `clscrssetRESEXISTS` if the resource already exists in reslist

Syntax
```
clscrsset clscrs_reslist_append(clscrs_reslist *reslist, clscrs_res *res);
```

clscrs_reslist_first  Get the first resource on a resource list.

Parameters
- `reslist` [in]—Resource list for which the first resource is obtained
- `res` [out]—The first resource on the resource list

Returns
- `clscrssetSUCC` on success.
- `clscrssetBADARG` if reslist is NULL
- `clscrssetEMPTY` if there are no resources in the list

Syntax
```
clscrsset clscrs_reslist_first(clscrs_reslist *reslist, clscrs_res **res);
```

clscrs_reslist_next  Get the current next resource from the resource list. This function is called to iterate over the resources in a resource list.

Parameters
- `reslist` [in]—Resource list for which the first resource is obtained
- `res` [out]—The next resource on the resource list

Returns
- `clscrssetSUCC` on success
- `clscrssetBADARG` if reslist is NULL
- `clscrssetENDLIST` if there are no more resources in the list

Syntax
```
clscrsset clscrs_reslist_next(clscrs_reslist *reslist, clscrs_res **res);
```

clscrs_reslist_find  Find a resource in a resource list.

Parameters
- `reslist` [in]—Resource list (empty) that is created
- `name` [in]—Name of the resource that is being obtained
- `res` [out]—The resource corresponding to the given name

Returns
- `clscrssetSUCC` on success
- `clscrssetNORES` if the resource is not found
- `clscrssetBADARG` if reslist or name is NULL

Syntax
```
clscrsset clscrs_reslist_find(clscrs_reslist *reslist, clscrs_res **res);
```
clscrs_replist_find(clscrs_replist *reslist, const oratext *name, clscrs_res **res);  

clscrs_replist_count Count the number of resources in a resource list.
Parameters
■ reslist [in]—Resource list for which the count is obtained
■ count [out]—Number of resources in the resource list
Returns
■ clscrsretSUCC on success
■ clscrsretBADARG if reslist is NULL
Syntax
clscrsret clscrs_replist_count(clscrs_replist *reslist, ub4 *count);

clscrs_replist_delete_res Delete a resource from a resource list.
Parameters
■ reslist [in]—Resource list from which to delete the resource
■ name [in]—Name of the resource to delete
Returns
■ clscrsretSUCC on success
■ clscrsretBADARG if reslist or name is NULL
■ clscrsretNORES if the resource is not in reslist
Syntax
clscrsret clscrs_replist_delete_res(clscrs_replist *reslist, const oratext *name);

clscrs_replist_destroy Free the memory for a resource list.
Parameters
■ reslist [in]—Resource list for which the memory is to be freed
Returns
■ clscrsretSUCC on success
Syntax
clscrsret clscrs_replist_destroy(clscrs_replist **reslist);

clscrs_get_error_message Retrieve the error message corresponding to the return codes from a clscrs API. The memory for the error message is allocated by the caller. If the buffer is not large enough, the length is returned in msg_len.
Parameters
■ ctx [in]—clscrs context
Functions for Managing Resource Structures

- **err_code [in]** — Error code returned from the clscrs API
- **msg [out]** — Message corresponding to err_code
- **msg_len [inout]** — Length of the message buffer.

Returns
- **clscrsretSUCC** on success

Syntax

```c
clscrsret clscrs_get_error_message(clscrs_ctx *ctx, clscrsret err_code, oratext *msg, sb4 msg_len);
```

**clscrs_get_fixed_attrlist**

Get the list of attributes corresponding to an attribute group identifier.

Parameters
- **ctx [in]** — clscrs context
- **attrgrp [in]** — Attribute group that identifies a group of attributes
- **attrlist [out]** — List of attributes returned that corresponds to the attribute group

Returns
- **clscrsretSUCC** on success

Syntax

```c
clscrs_splist* clscrs_get_fixed_attrlist(clscrs_ctx *ctx, clscrs_attr_grp attrgrp);
```

**Resource Operations**

This section describes resource functions. The clscrs_res resource abstraction contains the name of a resource and additional data appropriate to the context in which it is used. Sometimes it carries status and error return information about an operations. Other times it contains attribute data as input to an operation. A resource may be in exactly one resource list. If so, its successor may be found with the NEXT operation.

**CLSCRSRET clscrs_res_create(clscrs_cts *ctx, const oratext *resname, clscrs_res **res)** Creates a single resource and fills in a handle to it. The resname must be provided, and cannot be NULL.

**CLSCRSRET clscrs_res_get_name (clscrs_res *res, oratext **name);** Returns a pointer to the name of a resource. The returned name pointer is only valid as long as the resource exists.

**CLSCRSRET clscrs_res_get_op_status (clscrs_res *res, CLSCRS_STAT *stat, oratext **msg)** If there is a valid operation error value in the resource, fill in the stat and the pointer to an error message and return SUCCESS. May fill in a NULL to the msg. If there is no valid operation status, returns INVALID.

**clscrs_splist *clscrs_res_get_node_list(clscrs_res *res);** Returns an splist holding the nodes currently hosting the resource, or NULL if there is no host for the resource or there are no attributes. The count of the list may obtained and the list iterated. The list
is owned by the resource and will be destroyed when the resource is destroyed. This operation is a special case, interpreting the semantics of the attribute or attributes that may hold the current hosting member list. There is no specified ordering of the list.

CLSCRSRET clscrs_res_get_attr(clscrs_res *res, const oratext *attrname, oratext **value) ; Fills in a pointer to the value of the attribute with the given name and returns SUCCESS if found, or FAILURE if the name is not present in the attribute set of the resource, and INVALID if there is no attribute list in the resource. May assert if given an invalid resource handle.

CLSCRSRET clscrs_res_set_attr(clscrs_res *res, oratext *attrname, oratext *value) ; Sets the value of the attribute with the given name and returns SUCCESS or FAILURE if the name is not present in the attribute set of the resource. If the attribute already exists, then its current value will be replaced. On return, the memory for the name and value may be recycled.

CLSCRSRET clscrs_res_attr_count(clscrs_res *res, ub4 *count ) Fill in the number of attributes that will be returned with a scan using an attrIter. Returns SUCCESS if there are attributes. If there are no attributes, returns INVALID, but still sets the count to zero.

CLSCRSRET clscrs_res_get_attr_list(clscrs_res *res, clscrs_splist **attrlist); Returns an splist for the attributes of the resource, allowing next() operations for a scan. The list is owned by the resource, which will destroy it when the resource is destroyed. Returns SUCCESS if there is an attribute list, or INVALID if there is not one in the resource. There is no specified ordering of the list.

Resource List Operations
This section describes the resource list operations. The clscrs_reslist resource list is an abstraction that contains information about zero or more resources. A list is created, appended, iterated over, and destroyed.

CLSCRSRET clscrs_reslist_create(clscrs_ctx *ctx, clscrs_reslist **reslist) Creates a resource list and fills a handle.

CLSCRSRET clscrs_reslist_count(clscrs_reslist *reslist, ub4 *count) Fills a count of the number of resources in a list.

CLSCRSRET clscrs_reslist_first(clscrs_reslist *reslist, clscrs_res **first) Fills a handle to the first resource in the list of resources. If the list is empty, fill in NULL.

CLSCRSRET clscrs_reslist_next(clscrs_reslist *reslist, clscrs_res **next) Fills a handle to the next resource in the list of resources. If the list is empty, fill in NULL.

CLSCRSRET clscrs_reslist_find(clscrs_reslist *reslist, const oratext *name, clscrs_res **res) Find a resource by name in a reslist, and fill in a handle to the one located. If the resource is not found, fill in NULL. Does an exact match lookup with no pattern matching.

clscrs_res *clscrs_reslist_destroy(clscrs_res *res) Delete a resource list and all of the resources that it currently contains.
This appendix describes the syntax of the Oracle Cluster Registry (OCR) tool, OCRCONFIG under the following topic:

- The OCR Configuration Tool Command Syntax and Options

The OCR Configuration Tool Command Syntax and Options

Use the `ocrconfig` command to perform OCR Configuration Tool operations with administrative privileges on UNIX-based systems or as a user with Administrator privileges on Windows-based systems. The `ocrconfig` command syntax is as follows where `options` is one of the verbs shown in the Option column of Table C-1:

`ocrconfig -option`

<table>
<thead>
<tr>
<th>Option</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>-backuploc</td>
<td>To change an OCR backup file location. For this entry, use a full path that is accessible by all of the nodes.</td>
</tr>
<tr>
<td>-downgrade</td>
<td>To downgrade an OCR to an earlier version.</td>
</tr>
<tr>
<td>-export</td>
<td>To export the contents of an OCR into a target file.</td>
</tr>
<tr>
<td>-help</td>
<td>To display help for the <code>ocrconfig</code> commands.</td>
</tr>
<tr>
<td>-import</td>
<td>To import the OCR contents from a previously exported OCR file.</td>
</tr>
<tr>
<td>-overwrite</td>
<td>To update an OCR configuration that is recorded on the OCR with the current OCR configuration information that is found on the node from which you are running this command.</td>
</tr>
<tr>
<td>-repair</td>
<td>To update an OCR configuration on the node from which you are running this command with the new configuration information specified by this command.</td>
</tr>
<tr>
<td>-replace</td>
<td>To add, replace, remove an OCR location.</td>
</tr>
<tr>
<td>-restore</td>
<td>To restore an OCR from an automatically created OCR backup file.</td>
</tr>
<tr>
<td>-showbackup</td>
<td>To display the location, timestamp, and the node name of origin of the backup files created in the past 4 hours, 8 hours, 12 hours, and in the last day and week. You do not have to be the root user to execute this option.</td>
</tr>
<tr>
<td>-upgrade</td>
<td>To upgrade an OCR to a later version.</td>
</tr>
</tbody>
</table>
For example, to export the OCR contents to a binary file, use the `ocrconfig` command with the following syntax where `file_name` is the file to which you want to export the OCR contents as follows:

`ocrconfig -export file_name`
This chapter describes how to administer Oracle Real Application Clusters (RAC) databases and instances using the Server Control Utility (SRVCTL). The topics in this chapter include:

- Overview of SRVCTL for Administering Real Application Clusters
- SRVCTL Command Syntax and Options
- SRVCTL General Cluster Database Administration Tasks
- SRVCTL Cluster Database Configuration Tasks
- SRVCTL Node-Level Tasks
- SRVCTL Command Reference
- SRVCTL Commands

See Also: Your platform-specific Oracle Real Application Clusters installation and configuration guide for information about using the DBCA

Overview of SRVCTL for Administering Real Application Clusters

The Server Control (SRVCTL) utility is installed on each node by default. You can use SRVCTL to start and stop the database and instances, manage configuration information, and to delete or move instances and services. SRVCTL also manages configuration information.

Some SRVCTL operations store configuration information 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.

To use SRVCTL, enter the `srvctl` command and its options in case sensitive syntax as described under the heading "SRVCTL Command Reference" on page D-3.

Guidelines for Using SRVCTL in Real Application Clusters

Guidelines for using SRVCTL are:

- To use SRVCTL to change your RAC database configuration, log in to the database as the `oracle` user. Members of the DBA group can start and stop the database.
- Only use the version of SRVCTL that is provided with Oracle Database 10g on RAC databases that are created or upgraded for Oracle Database 10g.
Always use SRVCTL from the ORACLE_HOME of the database that you are administering.

SRVCTL does not support concurrent executions of commands on the same object. Therefore, only run one SRVCTL command at a time for each database, service, or other object.

Obtaining Command-Line Help for SRVCTL

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
```

---

**Caution:** Although you may be able to cancel running SRVCTL commands by entering Control-C at the command line, you may corrupt your configuration data by doing this. You are strongly advised not to attempt to terminate SRVCTL in this manner.

---

SRVCTL Command Syntax and Options

SRVCTL commands, objects, and options are case sensitive. Database, instance, and service names are case insensitive and case preserving. SRVCTL interprets the following command syntax:

```
srvctl command object [options]
```

In SRVCTL syntax:

- `srvctl` is the command to start the SRVCTL utility.
- `command` is a verb such as `start`, `stop`, or `remove`.
- `object` is an object or target 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 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. SRVCTL prompts for user credentials when you use the `-q` option with any SRVCTL command.

---

**Note:** Enclose comma-delimited lists in double-quote ("...,...") symbols.

---

SRVCTL Cluster Database Configuration Tasks

The database configuration tasks are:

- 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.

**SRVCTL General Cluster Database Administration Tasks**

The general database administration tasks are:

■ 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

**SRVCTL Node-Level Tasks**

The node-level tasks are:

■ Adding and deleting node level applications.

■ Setting and unsetting the environment for node-level applications.

■ Administering node applications.

■ Administering ASM instances.

■ Starting and stopping a group of programs that includes virtual IP addresses, Listeners, Oracle Notification Services, and Oracle Enterprise Manager agents (for maintenance purposes).

**SRVCTL Command Reference**

This section presents the SRVCTL command reference:

■ srvctl add

■ srvctl config

■ srvctl enable

■ srvctl disable

■ srvctl start

■ srvctl stop

■ srvctl modify

■ srvctl relocate

■ srvctl status

■ srvctl getenv

■ srvctl setenv and unsetenv
SRVCTL Commands

This section summarizes the SRVCTL commands, objects, and options. Oracle recommends that you use the DBCA to create your RAC database as well as your initial service configurations. This is because the DBCA configures both the Oracle Clusterware resources and the Net Service entries for each service.

SRVCTL Commands Summary

Table D–1 summary the SRVCTL commands. Execute SRVCTL commands from the command line and specify one or more objects with the appropriate options for the command and its objects.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>srvctl add</td>
<td>Adds the node applications, database, database instance, ASM instance, or service.</td>
</tr>
<tr>
<td>srvctl remove</td>
<td>Removes the node applications, database, database instance, ASM instance, or service.</td>
</tr>
<tr>
<td>srvctl config</td>
<td>Lists the configuration for the node applications, database, ASM instance, or service.</td>
</tr>
<tr>
<td>srvctl enable</td>
<td>Enables the database, database instance, ASM instance, or service.</td>
</tr>
<tr>
<td>srvctl disable</td>
<td>Disables the database, database instance, ASM instance, or service.</td>
</tr>
<tr>
<td>srvctl start</td>
<td>Starts the node applications, database, database instance, ASM instance, or service.</td>
</tr>
<tr>
<td>srvctl stop</td>
<td>Stops the node applications, database, database instance, ASM instance, or service.</td>
</tr>
<tr>
<td>srvctl modify</td>
<td>Modifies the node applications, database, database instance, or service configuration.</td>
</tr>
<tr>
<td>srvctl relocate</td>
<td>Relocates the service from one instance to another.</td>
</tr>
<tr>
<td>srvctl status</td>
<td>Obtains the status of the node applications, database, database instance, ASM instance, or service.</td>
</tr>
<tr>
<td>srvctl getenv</td>
<td>Displays the environment variable in the configuration for the node applications, database, database instance, or service.</td>
</tr>
<tr>
<td>srvctl setenv and unsetenv</td>
<td>Sets and unsets the environment variable in the configuration for the node applications, database, database instance, or service.</td>
</tr>
</tbody>
</table>

SRVCTL Objects Summary

Table D–2 lists the SRVCTL objects for SRVCTL commands. Use the full name or the abbreviation for the purpose described.
srvctl 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 UNIX-based systems, or you must be logged on as a user with Administrator privileges on Windows-based 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 D–2  SRVCTL Objects (Nouns) and Abbreviations

<table>
<thead>
<tr>
<th>Object Noun Name</th>
<th>Abbreviation</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>asm</td>
<td>asm</td>
<td>To add, configure, enable, start, obtain the status of, stop, disable, and remove ASM instances.</td>
</tr>
<tr>
<td>database</td>
<td>db</td>
<td>To add, configure, modify, manage environment variables for, enable, start, obtain the status of, stop, disable, and remove databases.</td>
</tr>
<tr>
<td>instance</td>
<td>inst</td>
<td>To add, configure, modify, manage environment variables for, enable, start, obtain the status of, stop, and remove database instances.</td>
</tr>
<tr>
<td>nodeapps</td>
<td>no abbreviation</td>
<td>To add, configure, modify, manage environment variables for, start, obtain the status of, stop, and remove node applications.</td>
</tr>
<tr>
<td>service</td>
<td>serv</td>
<td>To add, configure, modify, manage environment variables for, enable, start, obtain the status of, relocate, disable, stop, and remove services from your cluster database.</td>
</tr>
</tbody>
</table>

### Table D–3  srvctl add Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>srvctl add database</td>
<td>Adds a database and configuration.</td>
</tr>
<tr>
<td>srvctl add instance</td>
<td>Adds one or more instances and configurations.</td>
</tr>
<tr>
<td>srvctl add service</td>
<td>Adds services.</td>
</tr>
<tr>
<td>srvctl add nodeapps</td>
<td>Adds node applications.</td>
</tr>
<tr>
<td>srvctl add asm</td>
<td>Adds ASM instances.</td>
</tr>
</tbody>
</table>

### 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 [-o domain_name] [-p spfile] [-A addr_str] [-r [PRIMARY | PHYSICAL_STANDBY | LOGICAL_STANDBY]] [-s start_options] [-n db_name] -y database_name [ automatic | manual ]
```
Table D–4  

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td>-o oracle_home</td>
<td>The Oracle home for the database.</td>
</tr>
<tr>
<td>-m domain_name</td>
<td>The Domain for the database.</td>
</tr>
<tr>
<td>-p spfile</td>
<td>The server parameter file for the database.</td>
</tr>
<tr>
<td>-A addr_str</td>
<td>The database cluster alias (name</td>
</tr>
<tr>
<td>-r {PRIMARY</td>
<td>PHYSICAL_STANDBY</td>
</tr>
<tr>
<td>-s start_options</td>
<td>Startup options for the database.</td>
</tr>
<tr>
<td>-n db_name</td>
<td>The name of the database, where it is different from the unique name given by the -d option.</td>
</tr>
</tbody>
</table>

**Example**  
An example of this command is:  
```
srvctl add database -d crm -o /ora/ora10
```

**srvctl add instance**  
Adds a configuration for an instance to your cluster database configuration.

**Syntax and Options**  
Use the `srvctl add instance` command with the following syntax:
```
srvctl add instance -d db_unique_name -i inst_name -n node_name
```

**Table D–5  srvctl add instance Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td>-i inst_name</td>
<td>The instance name.</td>
</tr>
<tr>
<td>-n node_name</td>
<td>The node name.</td>
</tr>
</tbody>
</table>

**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 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. Also, you can use the `srvctl add service` command to configure the Transparent Application Failover (TAF) policy for a service.

**See Also:**  
Chapter 6 for more information about TAF policies

**Syntax and Options**  
Use the `srvctl add service` command with the following syntax:
```
srvctl add service -d db_unique_name -s service_name -r preferred_list
```
Use the following syntax to add a new preferred or available instance to the service configuration:

```
srctl add service -d db_unique_name -s service_name -r preferred_list [-a available_list] [-P TAF_policy]
```

**Table D–6  srctl add service Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td>-s service_name</td>
<td>The service name.</td>
</tr>
<tr>
<td>-r preferred_list</td>
<td>The list of preferred instances.</td>
</tr>
<tr>
<td>-a available_list</td>
<td>The list of available instances</td>
</tr>
<tr>
<td>-P TAF_policy</td>
<td>The TAF policy (NONE, BASIC, or PRECONNECT).</td>
</tr>
</tbody>
</table>

**Examples** 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 basic failover for the available instances:

```
srctl add service -d db_unique_name -s service_name -r crm01,crm02 -a crm03
```

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:

```
srctl add service -d db_unique_name -s service_name -r crm01,crm02 -a crm03 -P Preconnect
```

**srctl add nodeapps**

Adds a node application configuration to the specified node.

```
srctl add nodeapps -n node_name -o oracle_home -A addr_str
```

**Note:** On UNIX-based systems, you must be logged in as root and on Windows-based systems, you must be logged in as a user with Administrator privileges to run this command.
**SRVCTL Commands**

---

<table>
<thead>
<tr>
<th>Table D–8</th>
<th>srvctl add nodeapps Options</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Option</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-n node_name</td>
<td>Node name.</td>
</tr>
<tr>
<td>-o oracle_home</td>
<td>Oracle home for the cluster database.</td>
</tr>
<tr>
<td>-A addr_str</td>
<td>The node level VIP address (name</td>
</tr>
</tbody>
</table>

**Example** An example of this command is:

```
srvctl add nodeapps -n crmnode1 -o /ora/ora10 -A 1.2.3.4/255.255.255.0
```

**srvctl add asm**

Adds a record for an ASM instance to the specified node.

**Syntax and Options** Use the `srvctl add asm` command with the following syntax:

```
srvctl add asm -n node_name -i asm_instance_name -o oracle_home
```

<table>
<thead>
<tr>
<th>Table D–9</th>
<th>srvctl add asm Options</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Option</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-n node_name</td>
<td>Node name.</td>
</tr>
<tr>
<td>-i asm_instance_name</td>
<td>The ASM instance name.</td>
</tr>
<tr>
<td>-o oracle_home</td>
<td>Oracle home for the cluster database.</td>
</tr>
</tbody>
</table>

**Example** An example of this command is:

```
srvctl add asm -n crmnode1 -i asm1 -o /ora/ora10
```

**srvctl config**

The SRVCTL `config` command displays the configuration stored in the OCR.

<table>
<thead>
<tr>
<th>Table D–10</th>
<th>srvctl config Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Command</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>srvctl config database on page D-8</td>
<td>Displays the configuration information of the cluster database.</td>
</tr>
<tr>
<td>srvctl config service on page D-9</td>
<td>Displays the configuration information for the services.</td>
</tr>
<tr>
<td>srvctl config nodeapps on page D-9</td>
<td>Displays the configuration information for the node applications.</td>
</tr>
<tr>
<td>srvctl config asm on page D-10</td>
<td>Displays the configuration for the ASM instances on the node.</td>
</tr>
<tr>
<td>srvctl config listener on page D-10</td>
<td>Displays a list of configured Listeners that are registered with the Oracle Clusterware on a given node.</td>
</tr>
</tbody>
</table>

**srvctl config database**

Displays the configuration for a RAC database or lists all configured databases.

**Syntax and Options** Use the `srvctl config database` command with the following syntax:

```
srvctl config database [-d db_unique_name [-a] [-t]]
```
SRVCTL Commands

Examples
An example of this command to list all database is:

```
svrctl config database
```

An example of this command to show sample TNS entries for a specific database is:

```
svrctl config database -d db_erp
```

svrctl config service
Displays the configuration for a service.

Syntax and Options
Use the `svrctl config service` command with the following syntax:

```
svrctl config service -d db_unique_name [-s service_name] [-a]
```

Example
An example of this command is:

```
svrctl config service -d crm -s crm
```

**Note:** If you do not specify `-s service`, then SRVCTL displays all services for the specified cluster database.

svrctl config nodeapps
Displays the configuration for node applications.

Syntax and Options
Use the `svrctl config nodeapps` command with the following syntax:

```
svrctl config nodeapps -n node_name [-a] [-g] [-s] [-l]
```

Table D–11  `svrctl config database` Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-d db_unique_name</code></td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td><code>-a</code></td>
<td>Additional attributes.</td>
</tr>
<tr>
<td><code>-t</code></td>
<td>Display sample TNS entries.</td>
</tr>
</tbody>
</table>

Table D–12  `svrctl config service` Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-d db_unique_name</code></td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td><code>-s service_name</code></td>
<td>Service name.</td>
</tr>
<tr>
<td><code>-a</code></td>
<td>Additional attributes.</td>
</tr>
</tbody>
</table>

Example
An example of this command is:

```
svrctl config service -d crm -s crm
```

Table D–13  `svrctl config nodeapps` Option

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-n node_name</code></td>
<td>Node name.</td>
</tr>
</tbody>
</table>
 srvctl enable

The SRVCTL enable command enables the named object so that it can run under the Oracle Clusterware for automatic startup, failover, or restart. The Oracle Clusterware application supporting the object may be up or down to use this function. Enable is the default value. If the object is already enabled, then the command is ignored. Enabled objects can be started, and disabled objects cannot be started.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>srvctl enable database</td>
<td>on page D-11 Enables the database.</td>
</tr>
</tbody>
</table>
**srvctl enable database**

Enables the Oracle Clusterware resources for a database and enables the database's instances if the database was previously disabled.

**Syntax and Options** Use the `srvctl enable database` command with the following syntax:

```
srvctl enable database -d db_unique_name
```

The only option available for this command is `-d` to specify the database name.

**Example** An example of this command is:

```
srvctl enable database -d crm
```

**srvctl enable instance**

Enables an instance for the Oracle Clusterware. If all instances are disabled, then enabling an instance also enables the database.

**Syntax and Options** Use the `srvctl enable instance` command with the following syntax:

```
srvctl enable instance -d db_unique_name -i inst_name_list
```

**Example** An example of this command is:

```
srvctl enable instance -d crm -i "crm1,crm2"
```

**srvctl enable service**

Enables a service for the 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
```

---

**Table D–14 (Cont.) srvctl enable Summary**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>srvctl enable instance on page D-11</td>
<td>Enables the instance.</td>
</tr>
<tr>
<td>srvctl enable service on page D-11</td>
<td>Enables a service.</td>
</tr>
<tr>
<td>srvctl enable asm on page D-12</td>
<td>Enables an ASM instance.</td>
</tr>
</tbody>
</table>

**Table D–15 srvctl enable instance Option**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database</td>
</tr>
<tr>
<td>-i inst_name_list</td>
<td>Comma-separated list of instance names.</td>
</tr>
</tbody>
</table>

**Example** An example of this command is:

```
srvctl enable instance -d crm -i "crm1,crm2"
```
srvctl enable service -d db_unique_name { -s service_name_list | -s service_name -i inst_name} 

**Table D–16  srvctl enable service Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td>-s service_name_list</td>
<td>Comma-separated list of service names.</td>
</tr>
<tr>
<td>-s service_name</td>
<td>Single service name.</td>
</tr>
<tr>
<td>-i inst_name</td>
<td>Instance name.</td>
</tr>
</tbody>
</table>

**Examples**  The following example globally enables a service:

```bash
srvctl enable service -d crm -s crm
```

The following example enables a service to use a preferred instance:

```bash
crvctl enable service -d crm -s crm -i crm1
```

**srvctl enable asm**

Enables an ASM instance.

**Syntax and Options**  Use the `srvctl enable asm` command with the following syntax:

```bash
crvctl enable asm -n node_name [-i asm_inst_name]
```

**Table D–17  srvctl enable asm Option**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-n node_name</td>
<td>Node name</td>
</tr>
<tr>
<td>-i inst_name</td>
<td>ASM instance name.</td>
</tr>
</tbody>
</table>

**Example**  An example of this command is:

```bash
crvctl enable asm -n crmmode1 -i asm1
```

**srvctl disable**

Disables a specified object (cluster database, database instance, ASM instance, or service). SRVCTL disable is intended to be used when an object is to be repaired or is down for maintenance to prevent inappropriate automatic restarts. When you issue the `disable` command, the object is disabled and unavailable to run under the Oracle Clusterware for automatic startup, failover, or restart. If you specify `-i instance_name`, then SRVCTL only disables the service from running on the specified instance.

**Table D–18  srvctl disable Summary**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>srvctl disable database</td>
<td>on page D-13</td>
</tr>
<tr>
<td>srvctl disable instance</td>
<td>on page D-13</td>
</tr>
<tr>
<td>srvctl disable service</td>
<td>on page D-13</td>
</tr>
</tbody>
</table>
**srvctl disable database**
Disables a cluster database and its instances.

**Syntax and Options** Use the `srvctl disable database` command with the following syntax:

```
srvctl disable database -d db_unique_name
```

The only option available for this command is `-d` to specify the database name.

**Example** An example of this command is:

```
srvctl disable database -d mybd1
```

**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.

**Syntax and Options** Use the `srvctl disable instance` command with the following syntax:

```
srvctl disable instance -d db_unique_name -i inst_name_list
```

**Table D–19 srvctl disable instance Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-d db_unique_name</code></td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td><code>-i inst_name_list</code></td>
<td>Comma-separated instance names.</td>
</tr>
</tbody>
</table>

**Example** An example of this command is:

```
srvctl disable instance -d crm -i "crm1,crm3"
```

**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, an `srvctl disable service` operation on the entire service affect all of the instances and disable 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 | -s service_name -i inst_name)
```

**Table D–18 (Cont.) srvctl disable Summary**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>srvctl disable asm</code></td>
<td>Disables an ASM instance</td>
</tr>
</tbody>
</table>

---

### Table D–18 (Cont.) srvctl disable Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>srvctl disable asm</code></td>
<td>Disables an ASM instance</td>
</tr>
</tbody>
</table>
**Table D–20**  **srvctl disable service Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d <code>db_unique_name</code></td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td>-s <code>service_name_list</code></td>
<td>Comma-separated service names.</td>
</tr>
<tr>
<td>-s <code>service_name</code></td>
<td>Single service name.</td>
</tr>
<tr>
<td>-i <code>inst_name</code></td>
<td>Instance name.</td>
</tr>
</tbody>
</table>

**Examples** The following example globally disables two services:

```bash
srvctl disable service -d crm -s crm,marketing
```

The following example disables a service running on the preferred named instance and results in running a service on one less instance:

```bash
srvctl disable service -d crm -s crm -i crml
```

**srvctl disable asm**

Disables an ASM instance.

**Syntax and Options** Use the `srvctl disable asm` command with the following syntax:

```bash
srvctl disable asm -n `node_name` [-i `asm_inst_name`]
```

**Example** An example of this command is:

```bash
srvctl disable asm -n crmnode1 -i asm1
```

**srvctl start**

Starts 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, then 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 be in the OSDBA group also.

**Table D–22**  **srvctl start Summary**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>srvctl start database</code></td>
<td>Starts the cluster database and its instances</td>
</tr>
<tr>
<td><code>srvctl start instance</code></td>
<td>Starts the instance</td>
</tr>
<tr>
<td><code>srvctl start service</code></td>
<td>Starts the service</td>
</tr>
<tr>
<td><code>srvctl start nodeapps</code></td>
<td>Starts the node applications</td>
</tr>
</tbody>
</table>
**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] [-c connect_str | -q]
```

**Example** An example of this command is:
```
srvctl start database -d crm -o open
```

**srvctl start instance**
Starts instances in the cluster database.

**Syntax and Options** Use the `srvctl start instance` command with the following syntax:
```
srvctl start instance -d db_unique_name -i inst_name_list [-o start_options] [-c connect_str | -q]
```

**Example** An example of this command is:
```
srvctl start instance -d crm -i "crm1,crm4"
```
**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 on page D-22 and on page D-25 respectively.

**Syntax and Options** Use the `srvctl start service` command with the following syntax:

```bash
srvctl start service -d db_unique_name [-s service_name_list [-i inst_name]] [-o start_options] [-c connect_str | -q]
```

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-d db_unique_name</code></td>
<td>Unique name for the database</td>
</tr>
<tr>
<td><code>-s service_name_list</code></td>
<td>Comma-separated service names; the service name list is optional and if not provided, the SRVCTL starts all of the database’s services</td>
</tr>
<tr>
<td><code>-i inst_name</code></td>
<td>Instance name</td>
</tr>
<tr>
<td><code>-o start_options</code></td>
<td>Options to startup command (for example: open, mount, or nomount)</td>
</tr>
<tr>
<td><code>-c connect_str</code></td>
<td>Connect string (default: / as sysdba)</td>
</tr>
<tr>
<td><code>-q</code></td>
<td>Query connect string from standard input</td>
</tr>
</tbody>
</table>

**Examples** The following example starts named service names. If the instances that support these services, including available instances that the service uses for failover, are not running but are enabled, then they are started:

```bash
srvctl start service -d crm -s crm
```

The following example starts a named service on a specified instance:

```bash
srvctl start service -d crm -s crm -i crm2
```

**srvctl start nodeapps**

Starts node-level applications on a particular node.

**Syntax and Options** Use the `srvctl start nodeapps` command with the following syntax:

```bash
srvctl start nodeapps -n node_name
```

The only option available for this command is `-n` to specify the node name.

**Example** An example of this command is:

```bash
srvctl start nodeapps -n mynode1
```

**srvctl start asm**

Starts an ASM instance.
Syntax and Options  Use the `srvctl start asm` command with the following syntax:

```
srvctl start asm -n node_name [-i asm_inst_name] [-o start_options] [-c connect_str | -q]
```

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-n node_name</td>
<td>Node name</td>
</tr>
<tr>
<td>-i inst_name</td>
<td>ASM instance name.</td>
</tr>
<tr>
<td>-o start_options</td>
<td>Options to startup command, for example open, mount, or nomount.</td>
</tr>
<tr>
<td>-c connect_string</td>
<td>Connect string where the default is forward slash (/).</td>
</tr>
<tr>
<td>-q</td>
<td>Query connect string from standard input.</td>
</tr>
<tr>
<td>-h</td>
<td>Display help.</td>
</tr>
</tbody>
</table>

**Examples**  An example of this command to start a single ASM instance is:

```
srvctl start asm -n crmnode1 -i asml
```

An example to start all ASM instances on a node is:

```
srvctl start asm -n crmnode2
```

**srvctl start listener**

Starts the default Listener known as `node_name`, or the Listeners represented in a given list of Listener names, that are registered with the 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]
```

**Example**  An example of this command is:

```
srvctl start listener -n mynode1
```

**srvctl stop**

Stops the Oracle Clusterware applications for the database, all or named instances, all or named service names, or node level applications. Only the Oracle Clusterware applications that are starting or running are stopped. Objects running outside of the 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. Refer to the SRVCTL disable command starting with `srvctl disable database` on page D-13.
**SRVCTL Commands**

---

**Note:** If the object is stopped and is not disabled, then it can restart as a result of another planned operation. That is, the object will not restart as a result of a failure. Oracle recommends that you disable an object that should remain stopped after you issue a stop command.

---

### Table D–27  
**srvctl stop Summary**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>srvctl stop database</td>
<td>StOPS the cluster database</td>
</tr>
<tr>
<td>srvctl stop instance</td>
<td>StOPS the instance</td>
</tr>
<tr>
<td>srvctl stop service</td>
<td>StOPS the service</td>
</tr>
<tr>
<td>srvctl stop nodeapps</td>
<td>StOPS the node-level applications</td>
</tr>
<tr>
<td>srvctl stop asm</td>
<td>StOPS ASM instances</td>
</tr>
<tr>
<td>srvctl stop listener</td>
<td>StOPS the specified Listener or Listeners.</td>
</tr>
</tbody>
</table>

### srvctl stop database

Stops a database, its instances, and its services.

**Syntax and Options** Use the `srvctl stop database` command with the following syntax:

```
srvtl stop database -d db_unique_name [-o stop_options] [-c connect_str | -q]
```

#### Table D–28  
**srvctl stop database Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database</td>
</tr>
<tr>
<td>-o stop_options</td>
<td>shutdown command options (for example: normal, transactional, immediate, or abort)</td>
</tr>
<tr>
<td>-c connect_str</td>
<td>Connect string (default: / as sysdba)</td>
</tr>
<tr>
<td>-q</td>
<td>Prompt for user credentials connect string from standard input</td>
</tr>
</tbody>
</table>

**Example** An example of this command is:

```
srvtl stop database -d crm
```

### srvctl stop instance

Stops instances and stops all enabled and non-running services that have these instances as either preferred or available instances.

**Syntax and Options** Use the `srvctl stop instance` command with the following syntax:

```
srvtl stop instance -d db_unique_name -i inst_name_list [-o stop_options] [-c connect_str | -q]
```
**Example** An example of this command is:

```
svrctl stop instance -d crm -i crm
```

**svrctl stop service**

Stops one or more services globally across the cluster database, or on the specified instance.

**Syntax and Options** Use the `svrctl stop service` command with the following syntax:

```
svrctl stop service -d db_unique_name [-s service_name_list [-i inst_name]] [-c connect_str] [-q] [-f]
```

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-d db_unique_name</code></td>
<td>Unique name for the database</td>
</tr>
<tr>
<td><code>-i inst_name_list</code></td>
<td>Comma-separated instance names</td>
</tr>
<tr>
<td><code>-o stop_options</code></td>
<td>Options for shutdown command (for example: normal, transactional, immediate, or abort)</td>
</tr>
<tr>
<td><code>-c connect_str</code></td>
<td>Connect string (default: / as sysdba)</td>
</tr>
<tr>
<td><code>-q</code></td>
<td>Query connect string from standard input</td>
</tr>
</tbody>
</table>

**Examples** The following example stops a service globally across a cluster database:

```
svrctl stop service -d crm -s crm
```

The following example stops a service on a specified instance:

```
svrctl stop service -d crm -s crm -i crm2
```

**svrctl stop nodeapps**

Stops node-level applications on a particular node.

**Syntax and Options** Use the `svrctl stop nodeapps` command with the following syntax:
srvctl stop nodeapps -n node_name

The only option for this command is the -n option, which specifies the node name.

**Example** An example of this command is:

```
srvctl stop nodeapps -n mynode1
```

**srvctl stop asm**

Stops an ASM instance.

**Syntax and Options** Use the `srvctl stop asm` command with the following syntax:

```
srvctl stop asm -n node_name [-i inst_name] [-o stop_options] [-c connect_str | -q]
```

**Table D–31 srvctl stop asm Option**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-n node_name</td>
<td>Node name.</td>
</tr>
<tr>
<td>-i inst_name</td>
<td>ASM instance name.</td>
</tr>
<tr>
<td>-o stop_options</td>
<td>Options for shutdown command, for example, normal, transactional, immediate, or abort.</td>
</tr>
<tr>
<td>-c connect_string</td>
<td>Connect string where the default is forward slash (/).</td>
</tr>
<tr>
<td>-q</td>
<td>Query connect string from standard input.</td>
</tr>
<tr>
<td>-h</td>
<td>Display help.</td>
</tr>
</tbody>
</table>

**Example** An example of this command is:

```
srvctl stop asm -n crmnode1 -i asm1
```

**srvctl stop listener**

Stops the default Listener known as `node_name`, or the Listeners represented in a given list of Listener names, that are registered with the Oracle Clusterware on the given node.

**Syntax and Options** Use the `srvctl stop listener` command with the following syntax:

```
srvctl stop listener -n node_name [-l listener_name_list]
```

**Example** An example of this command is:

```
srvctl stop listener -n mynode1
```

**srvctl modify**

Enables you to modify the instance configuration without removing and adding the 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.

### 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] [-m domain_name] [-p spfile] [-r {PRIMARY | PHYSICAL_STANDBY | LOGICAL_STANDBY}] [-s start_options] [-y {AUTOMATIC | MANUAL}]
```

### Table D–33 srvctl modify database Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-d db_unique_name</code></td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td><code>-n db_name</code></td>
<td>Name of the database, where it is different from the unique name given by <code>-d</code> option.</td>
</tr>
<tr>
<td><code>-o oracle_home</code></td>
<td>Oracle home for cluster database.</td>
</tr>
<tr>
<td><code>-m domain_name</code></td>
<td>Domain for cluster database.</td>
</tr>
<tr>
<td><code>-p spfile</code></td>
<td>Server parameter file for cluster database.</td>
</tr>
<tr>
<td><code>-r role</code></td>
<td>Role of the database (primary, physical standby, or logical standby)</td>
</tr>
<tr>
<td><code>-s start_options</code></td>
<td>Startup options for the database.</td>
</tr>
<tr>
<td><code>-y</code></td>
<td>Management policy for the database, either automatic or manual.</td>
</tr>
<tr>
<td><code>-h</code></td>
<td>Print usage.</td>
</tr>
</tbody>
</table>

**Example** The following example changes the role of a database to a logical standby:

```
srvctl modify database -d crm -r logical_standby
```

### srvctl modify instance
Modifies the configuration for a database instance from its current node to another node or changes the dependency between and ASM instance and a database instance.

**Syntax and Options** Use the `srvctl modify instance` command with the following syntax:

```
srvctl modify instance -d db_unique_name -i inst_name [-n node_name | -s asm_instance_name | -r]
```
**SRVCTL Commands**

---

### Examples

An example of this command to relocate a database instance is:

```
srvctl modify instance -d crm -i crm1 -n my_new_node
```

The following example of this command establishes a dependency between an ASM instance and a database instance:

```
 srvctl modify instance -d crm -i crm1 -s asm1
```

### 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, for example:

- 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 D-16.

- 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 the `srvctl modify service` command with the following syntax:

---
srvctl modify service -d db_unique_name -s service_name -i old_inst_name -t new_inst_name [-f]

Table D–35  srvctl modify service Options for an Available Instance

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td>-s service_name</td>
<td>Service name.</td>
</tr>
<tr>
<td>-i old_inst_name</td>
<td>Old instance name.</td>
</tr>
<tr>
<td>-t new_inst_name</td>
<td>New instance name.</td>
</tr>
<tr>
<td>-f</td>
<td>Disconnect all sessions during stop or relocate service operations.</td>
</tr>
</tbody>
</table>

You can also use the `srvctl modify service` command to change an available instance to a preferred instance as follows:

srvctl modify service -d db_unique_name -s service_name -i avail_inst_name -r [-f]

Table D–36  srvctl modify service Options for Changing an Available Instance to Preferred

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td>-s service_name</td>
<td>Service name.</td>
</tr>
<tr>
<td>-i avail_inst_name</td>
<td>Instance name.</td>
</tr>
<tr>
<td>-r</td>
<td>Upgrade instance to preferred.</td>
</tr>
<tr>
<td>-f</td>
<td>Disconnect all sessions during stop or relocate service operations.</td>
</tr>
</tbody>
</table>

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 crml -r

To change the status of multiple instances, you can use the `srvctl modify service` command to list which instances are to be the preferred and which are to be the available instances for a service as follows:

srvctl modify service -d db_unique_name -s service_name -n -i pref_inst_list [-a avail_inst_list] [-f]

Table D–37  srvctl modify service Options for Changing Instances between Preferred and Available status

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td>-s service_name</td>
<td>Service name.</td>
</tr>
</tbody>
</table>
Table D–37 (Cont.) srvctl modify service Options for Changing Instances between Preferred and Available status

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-n</td>
<td>Uses only the instances named for this service (unnamed instances already assigned to the service are removed).</td>
</tr>
<tr>
<td>-i avail_inst_name</td>
<td>List of preferred instances.</td>
</tr>
<tr>
<td>-a avail_inst_list</td>
<td>List of available instances.</td>
</tr>
<tr>
<td>-f</td>
<td>Disconnect all sessions during stop or relocate service operations.</td>
</tr>
</tbody>
</table>

Example An example of this command to exchange a preferred and available instance is:

```
srvctl modify service -d crm -s crm -n -i crm1 -a crm2
```

**srvctl modify nodeapps**

Applies a new Oracle home or virtual IP address to nodeapps.

**Syntax and Options** Use the `srvctl modify nodeapps` command with the following syntax:

```
srvctl modify nodeapps -n node_name [-o oracle_home] [-A new_vip_address]
```

Table D–38 srvctl modify nodeapps Option

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-n node_name</td>
<td>Node name.</td>
</tr>
<tr>
<td>-o oracle_home</td>
<td>Oracle home for the cluster database.</td>
</tr>
<tr>
<td>-A new_vip_address</td>
<td>The node level VIP address (name</td>
</tr>
</tbody>
</table>

Example An example of this command is:

```
srvctl modify nodeapps -n mynode1 -A 100.200.300.40/255.255.255.0/eth0
```

**srvctl 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 relocated service is temporary until you modify the configuration. The `srvctl modify` command described on page D-24 permanently changes the service configuration.

Table D–39 srvctl relocate Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>srvctl relocate service</td>
<td>Relocates the named service names from one named instance to another named instance</td>
</tr>
</tbody>
</table>
**srvctl relocate service**
Temporarily relocates a service member to run on another instance.

**Syntax and Options** Use the `srvctl relocate service` command with the following syntax:
```
srvctl relocate service -d db_unique_name -s service_name -i old_inst_name -t new_inst_name [-f]
```

**Example** To temporarily relocate a named service member from `crm1` to `crm3`:
```
srvctl relocate service -d crm -s crm -i crm1 -t crm3
```

**srvctl status**
Displays the current state of a named database, instances, services, or node applications.

**Table D–41  srvctl status Summary**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>srvctl status database</td>
<td>Obtains the status of a database.</td>
</tr>
<tr>
<td>srvctl status instance</td>
<td>Obtains the status of an instance.</td>
</tr>
<tr>
<td>srvctl status service</td>
<td>Obtains the status of services.</td>
</tr>
<tr>
<td>srvctl status nodeapps</td>
<td>Obtains the status of node applications.</td>
</tr>
<tr>
<td>srvctl status asm</td>
<td>Obtains the status of ASM instances.</td>
</tr>
</tbody>
</table>

**srvctl status database**
Obtains 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 D–42  srvctl status database Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database</td>
</tr>
<tr>
<td>-f</td>
<td>Include disabled applications</td>
</tr>
</tbody>
</table>
Example An example of this command is:

```
srvctl status database -d crm -v
```

srvctl status instance
Obtains the status of instances.

Syntax and Options Use the `srvctl status instance` command with the following syntax:

```
srvctl status instance -d db_unique_name -i inst_name_list [-f] [-v]
```

Table D–43  `srvctl status instance` Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-d db_unique_name</code></td>
<td>Unique name for the database</td>
</tr>
<tr>
<td><code>-i inst_name_list</code></td>
<td>Comma-separated list of instance names</td>
</tr>
<tr>
<td><code>-f</code></td>
<td>Include disabled applications</td>
</tr>
<tr>
<td><code>-v</code></td>
<td>Verbose output</td>
</tr>
</tbody>
</table>

Example An example of this command is:

```
srvctl status instance -d crm -i "crm1,crm2" -v
```

srvctl status service
Obtains 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 D–44  `srvctl status service` Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-d db_unique_name</code></td>
<td>Unique name for the database</td>
</tr>
<tr>
<td><code>-s service_name_list</code></td>
<td>Comma-separated list of service names</td>
</tr>
<tr>
<td><code>-f</code></td>
<td>Include disabled applications</td>
</tr>
<tr>
<td><code>-v</code></td>
<td>Verbose output</td>
</tr>
</tbody>
</table>

Example The following example obtains the status of a named service globally across the database:

```
srvctl status service -d crm -s crm -v
```
**srvctl status nodeapps**
Obtains the status of node applications on a particular node.

**Syntax and Options** Use the `srvctl status nodeapps` command with the following syntax:

```
srvctl status nodeapps -n node_name
```

The only option available for this command is `-n` to specify the node name.

**Example** An example of this command to obtain the status of all nodes supporting database applications is:

```
srvctl status nodeapps -n mynode1
```

**srvctl status asm**
Obtains the status of an ASM instance.

**Syntax and Options** Use the `srvctl status asm` command with the following syntax:

```
srvctl status asm -n node_name
```

The only option available for this command is `-n` to specify the node name.

**Example** An example of this command is:

```
srvctl status asm -n crmnode1
```

**srvctl getenv**
Gets and displays values for the environment from the configuration file. Use SRVCTL with the `set`, `get`, and `unset` environment configuration verbs to administer the environment configurations for databases, instances, services, and node applications.

**Table D–45  srvctl getenv Summary**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>srvctl getenv database</code></td>
<td>Gets the cluster database environment values.</td>
</tr>
<tr>
<td><code>srvctl getenv instance</code></td>
<td>Gets the instance environment values.</td>
</tr>
<tr>
<td><code>srvctl getenv service</code></td>
<td>Gets the service environment values.</td>
</tr>
<tr>
<td><code>srvctl getenv nodeapps</code></td>
<td>Gets the node application environment values.</td>
</tr>
</tbody>
</table>

**srvctl getenv database**
Displays the cluster database environment values.

**Syntax and Options** Use the `srvctl getenv database` command with the following syntax:

```
srvctl getenv database -d db_unique_name [-t name_list]
```
**SRVCTL Commands**

---

**Example**  The following example gets the environment configuration for a cluster database:

```
srvctl getenv database -d crm
```

**srvctl getenv instance**

Gets the values for an instance environment configuration.

**Syntax and Options** Use the `srvctl getenv instance` command with the following syntax:

```
srvctl getenv instance -d db_unique_name -i inst_name [-t name_list]
```

**Example**  The following example sets the environment configuration for an instance:

```
srvctl getenv instance -d -crm -i instance1
```

**srvctl getenv service**

Gets the values for a service environment configuration.

**Syntax and Options** Use the `srvctl getenv service` command with the following syntax:

```
srvctl getenv service -d db_unique_name -s service_name [-t name_list]
```

**Table D–46  srvctl getenv database Options**

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database</td>
</tr>
<tr>
<td>-t name_list</td>
<td>Names of environment variables</td>
</tr>
</tbody>
</table>

**Table D–47  srvctl getenv database Options**

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database</td>
</tr>
<tr>
<td>-i inst_name</td>
<td>Instance name</td>
</tr>
<tr>
<td>-t name_list</td>
<td>Names of environment variables</td>
</tr>
</tbody>
</table>

**Example**  The following example sets the environment configuration for an instance:

```
srvctl getenv instance -d -crm -i instancel
```

**srvctl getenv service**

Gets the values for a service environment configuration.

**Syntax and Options** Use the `srvctl getenv service` command with the following syntax:

```
srvctl getenv service -d db_unique_name -s service_name [-t name_list]
```

**Table D–48  srvctl getenv service Options**

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database</td>
</tr>
<tr>
<td>-s service_name</td>
<td>Service name</td>
</tr>
<tr>
<td>-t name_list</td>
<td>Names of environment variables</td>
</tr>
</tbody>
</table>

**Example**  The following example lists all environment variables for a service:

```
srvctl getenv service -d crm -s crm
```
**srvctl getenv nodeapps**

Gets the environment variables for the node application configurations.

**Syntax and Options** Use the `srvctl getenv nodeapps` command with the following syntax:
```
srvctl getenv nodeapps -n node_name [-t name_list]
```

**Example** The following example lists all environment variables for the node applications:
```
srvctl getenv nodeapps -n crmnode1
```

---

**srvctl setenv and unsetenv**

The `setenv` command sets values for the environment in the configuration file. The `unsetenv` command unsets values for the environment in the configuration file.

**Table D–50 srvctl setenv and unsetenv Summary**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>srvctl setenv database</code></td>
<td>on page D-29 Administers cluster database environment configurations</td>
</tr>
<tr>
<td><code>srvctl setenv instance</code></td>
<td>on page D-30 Administers instance environment configurations</td>
</tr>
<tr>
<td><code>srvctl setenv service</code></td>
<td>on page D-30 Administers service environment configurations</td>
</tr>
<tr>
<td><code>srvctl setenv nodeapps</code></td>
<td>on page D-31 Administers node application environment configurations</td>
</tr>
<tr>
<td><code>srvctl unsetenv database</code></td>
<td>on page D-31 Unsets the cluster database environment configuration</td>
</tr>
<tr>
<td><code>srvctl unsetenv instance</code></td>
<td>on page D-31 Unsets instance environment configurations</td>
</tr>
<tr>
<td><code>srvctl unsetenv service</code></td>
<td>on page D-32 Unsets service environment configurations</td>
</tr>
<tr>
<td><code>srvctl unsetenv nodeapps</code></td>
<td>on page D-32 Unsets node application environment configurations</td>
</tr>
</tbody>
</table>

---

**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 D–51 srvctl setenv database Options**

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td>-t =,...name=val</td>
<td>Names and values of environment variables.</td>
</tr>
</tbody>
</table>
**Example** The following example sets the language environment configuration for a cluster database:

```
srvctl setenv database -d crm -t LANG=en
```

**srvctl setenv instance**

Administers instance environment configurations.

**Syntax and Options** Use the `srvctl setenv instance` command with the following syntax:

```
srvctl setenv instance -d [db_unique_name] [-i inst_name] { -t name=val, name=val, ... } | -T name=val
```

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td>-i inst_name</td>
<td>Instance name.</td>
</tr>
<tr>
<td>-t name...</td>
<td>Names and values of environment variables.</td>
</tr>
<tr>
<td>-T name=val</td>
<td>Enables single environment variable to be set to a value that contains commas or other special characters.</td>
</tr>
</tbody>
</table>

**Example** The following example sets the environment configuration for an instance:

```
srvctl setenv instance -d -crm -i instance1 -t LANG=EN
```

**srvctl setenv service**

Administers service environment configurations.

**Syntax and Options** Use the `srvctl setenv service` command with the following syntax:

```
srvctl setenv service -d [db_unique_name] [-s service_name] { -t name=val, name=val, ... } | -T name=val
```

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td>-s service_name</td>
<td>Service name.</td>
</tr>
<tr>
<td>-t name...</td>
<td>Names and values of environment variables.</td>
</tr>
<tr>
<td>-T name=val</td>
<td>Enables single environment variable to be set to a value that contains commas or other special characters.</td>
</tr>
</tbody>
</table>

---

**Table D–51 (Cont.) srvctl setenv database Options**

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-T name=val</td>
<td>Enables single environment variable to be set to a value that contains commas or other special characters.</td>
</tr>
</tbody>
</table>
Example  To set all environment variables for a service:

```
srvctl setenv service -d crm -s crm -t CLASSPATH=/usr/local/jdk/jre/rt.jar
```

**srvctl setenv nodeapps**

Sets the environment variables for the node application configurations.

**Syntax and Options** Use the `srvctl setenv nodeapps` command as follows:

```
srvctl setenv nodeapps -n node_name {-t name=val[,name=value,...] | -T name=value}
```

**Example** To set an environment variable for a node application:

```
srvctl setenv nodeapps -n crmnode1 -t CLASSPATH=/usr/local/jdk/jre/rt.jar
```

**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
```

**Example** The following example unsets the environment configuration for a cluster database environment variable:

```
srvctl unsetenv database -d crm -t CLASSPATH
```

**srvctl unsetenv instance**

Unsets instance environment configurations.

**Syntax and Options** Use the `srvctl unsetenv instance` command as follows:

```
srvctl unsetenv instance -d db_unique_name [-i inst_name] -t name_list
```
Example The following example unsets the environment configuration for an instance:
```
svrctl unsetenv instance -d -crm -i instance1 -t CLASSPATH
```

**svrctl unsetenv service**

Unsets service environment configurations.

**Syntax and Options** Use the `svrctl unsetenv service` command as follows:
```
svrctl unsetenv service -d db_unique_name [-s service_name] -t name_list
```

**Example** To unset an environment variables for a service:
```
svrctl unsetenv service -d crm -s crm -t CLASSPATH
```

**svrctl unsetenv nodeapps**

Unsets the environment configuration for the node application configurations.

**Syntax and Options** Use the `svrctl unsetenv nodeapps` command as follows:
```
svrctl unsetenv nodeapps -n node_name -t name_list
```

**Table D–58  `svrctl unsetenv nodeapps` Options**

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-n node_name</td>
<td>Node name.</td>
</tr>
<tr>
<td>-t name=val</td>
<td>Names and values of environment variables.</td>
</tr>
</tbody>
</table>

**Example** The following example unsets the environment configuration for a node's node applications:
```
svrctl unsetenv nodeapps -n crmnode1 -t name_list
```
**srvctl remove**

Removes the configuration, the Oracle Clusterware applications for the node (including the virtual IP address, the Oracle Enterprise Manager agent, the GSD, and the Listeners), the database, named instances, or the named services from the cluster database. Environment settings for the object are also removed.

If you do not use the force flag (-f), then Oracle 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 -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. The srvctl stop command is described on page D-17.

**Table D–59  srvctl remove Summary**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>srvctl remove database</td>
<td>Removes a database and configuration.</td>
</tr>
<tr>
<td>srvctl remove instance</td>
<td>Removes one or more instances and configurations.</td>
</tr>
<tr>
<td>srvctl remove service</td>
<td>Removes services.</td>
</tr>
<tr>
<td>srvctl remove nodeapps</td>
<td>Removes node applications.</td>
</tr>
<tr>
<td>srvctl remove asm</td>
<td>Removes ASM instances.</td>
</tr>
</tbody>
</table>

**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]
```

**Table D–60  srvctl remove database Options**

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td>-f</td>
<td>Force remove.</td>
</tr>
</tbody>
</table>

**Example** An example of this command is:

```
srvctl remove database -d crm
```

**srvctl remove instance**

Removes the configurations for an instance.

**Syntax and Options** Use the srvctl remove instance command with the following syntax:
srvctl remove instance -d db_unique_name -i inst_name [-f]

**Table D–61  srvctl remove instance Options**

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td>-i inst_name</td>
<td>Instance name.</td>
</tr>
<tr>
<td>-f</td>
<td>Force remove.</td>
</tr>
</tbody>
</table>

**Example** An example of this command is:
srvctl remove instance -d crm -i crm01

**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 inst_name] [-f]

**Table D–62  srvctl remove service Options**

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d db_unique_name</td>
<td>Unique name for the database.</td>
</tr>
<tr>
<td>-s service_name</td>
<td>Service name.</td>
</tr>
<tr>
<td>-i inst_name</td>
<td>Instance name.</td>
</tr>
<tr>
<td>-f</td>
<td>Force remove.</td>
</tr>
</tbody>
</table>

**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 nodeapps**
Removes the node application configuration from the specified node. You must have full administrative privileges to run this command. On UNIX-based systems, you must be logged in as `root` and on Windows-based systems, you must be logged in as a user with Administrator privileges.

**Syntax** Use the `srvctl remove nodeapps` command as follows:
srvctl remove nodeapps -n node_name_list [-f]

**Table D–63  srvctl remove nodeapps Options**

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-n node_name_list</td>
<td>Node name or a comma-delimited list of node names.</td>
</tr>
</tbody>
</table>
Example  An example of this command is:

crctl remove nodeapps -n "mynode1,mynode2,mynode3"

srvctl remove asm
Removes an ASM instance.

Syntax and Options  Use the `srvctl remove asm` command with the following syntax:

crctl remove asm -n node_name [-i asm_inst_name]

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-n node_name</code></td>
<td>Node name</td>
</tr>
<tr>
<td><code>-i asm_inst_name</code></td>
<td>ASM instance name.</td>
</tr>
</tbody>
</table>

Example  An example of this command is:

crctl remove asm -n crmnode1 -i asm1
This appendix describes the Oracle Real Application Clusters (RAC) management tools messages. The messages in this appendix appear alphabetically by message prefix in the following groups:

- PRKA—Cluster Node Applications Messages
- PRKC—Cluster Command Messages
- PRKD—Global Services Daemon Messages
- PRKE—Global Services Daemon Controller Utility Messages
- PRKH—Server Manager (SRVM) Messages
- PRKI—Cluster Pre-Install Messages
- PRKN—Server Manager (SRVM) System Library Messages
- PRKO—Server Control (SRVCTL) Utility Messages
- PRKP—Cluster Database Management Messages
- PRKR—Cluster Registry Messages
- PRKS—Automatic Storage Management Messages
- PRKU—Command-Line Parser Utility Messages
- PRKV—Virtual IP Configuration Assistant Messages

See Also: Oracle Database Platform Guide for Microsoft Windows (32-Bit) for Windows-based messages and for all other messages refer search online at http://tahiti.oracle.com

Overview of Real Application Clusters-Specific Messages

Within Oracle Database 10g, RAC messages use the same ORA prefix and reside in the same e*.msg files as non-RAC or single-instance messages. Additionally, some RAC messages are issued by the single-instance Oracle database. These messages appear online by way of a Tahiti error message search as described in the Oracle database documentation in the Preface of this book under the heading "Related Documents" on page -xix.
Prefixes and Message Codes for RAC-Specific Messages

Message prefixes indicate where to find information about the messages in this chapter. In addition, the prefixes indicate which RAC component issued the messages.

Types of Real Application Clusters Messages and Related Files

The RAC high availability and cluster database management tools use the following syntax for internal messages:

\[ \text{PRKx-0000: } \text{Cause description} \ast \text{variable} \ast \text{text} \ast \{\text{variable}\} \ast \text{text} \ast \{\text{variable}\} \]

A message might appear as follows:

\[ \text{PRRC-1022: } \text{"Could not get "node name" for node } \{0\} \text{ in } \{1\}" \]

The variables in the previous example represent numbers, names, and character strings for objects in the physical or logical cluster and the cluster database. You can ignore empty variables.

PRKA—Cluster Node Applications Messages

**PRKA-2001: GSD already exists**

**Cause:** An attempt was made to create the Global Services Daemon (GSD) application while there is a running GSD application.

**Action:** Stop and remove the running GSD and create the GSD application again using `srvctl add nodeapps`.

**PRKA-2002: Listener already exists**

**Cause:** An attempt was made to create a Listener application while there is a running Listener application.

**Action:** Stop and remove the running Listener and create the Listener application again using `srvctl add nodeapps`.

**PRKA-2003: EM Agent already exists**

**Cause:** An attempt was made to create an Agent application while there is a running Agent application.

**Action:** Stop and remove the running Agent and create the Agent application again using `srvctl add nodeapps`.

**PRKA-2004: No database found for the cluster -- not used?**

**Cause:**

**Action:**

**PRKA-2005: Netmasks are different for starting and ending IP addresses**

**Cause:** Invalid range of VIP addresses specified for the cluster.

**Action:** Check if the starting and ending addresses in the VIP range belong to the same netmask.

**PRKA-2006: Invalid VIP address range**

**Cause:** Invalid range of VIP addresses specified for the cluster.

**Action:** Check if the starting address is less than ending address.

**PRKA-2007: Empty VIP address range**

**Cause:** Invalid range of VIP addresses specified for the cluster.
**Action:** Enter valid non-null VIP address range.

**PRKA-2008: No VIP address is available in the VIP pool**
- **Cause:** There are IP address available for use in the range of IP addresses specified earlier.
- **Action:** Check VIP pool to ensure valid VIP addresses.

**PRKA-2009: VIP address is not found**
- **Cause:** IP address is not found.
- **Action:** Enter valid VIP address.

**PRKA-2010: VIP address already exists**
- **Cause:** An attempt was made to create a Virtual IP while there is a running Virtual IP application.
- **Action:** Stop and remove the running Virtual IP application and create the Virtual IP application using 'srvctl add nodeapps'

**PRKA-2011: Remote command failed - not used?**
- **Cause:** The command failed to execute on remote node.
- **Action:** Check the command for syntax and semantics and reissue the command.

**PRKA-2012: Failed to get the VIP address range configuration from the OCR**
- **Cause:** There could no IP address ranges in OCR, or there is an OCR error occurred while retrieving VIP address range
- **Action:** Refer to OCR logs for detailed error information.

**PRKA-2013: Failed to add the VIP address range configuration to the OCR**
- **Cause:** An OCR error occurred while adding VIP address range into OCR.
- **Action:** Refer to OCR logs for detailed error information.

**PRKA-2014: Failed to remove the VIP address range configuration from the OCR**
- **Cause:** An OCR error occurred while adding VIP address range into OCR.
- **Action:** Refer to OCR logs for detailed error information.

**PRKA-2015: Invalid node name**
- **Cause:** The node name entered does not belong to cluster.
- **Action:** Enter a node that belong to the cluster.

**PRKA-2016: Invalid listener name**
- **Cause:** The listener name entered is invalid.
- **Action:** Correct the listener name.

**PRKA-2017: There are no IP addresses in the specified range. Make sure to specify the correct range for the IP addresses.**
- **Cause:**
- **Action:** Specify valid range for IP addresses.

**PRKA-2018: Virtual IP can only be created or removed by the system privilege user.**
- **Cause:** A user who has administrative privileges can only configure Virtual IPs on the system.
- **Action:** Check privileges of user attempting to create or remove VIP.
Action: PRKA-2019: Error executing command \"{0}\". File is missing.

**PRKC—Cluster Command Messages**

**PRKC-1023:** Invalid IP address format: {0}
- **Cause:** Given IP address does not adhere to the IP standard.
- **Action:** Use correct IP address format.

**PRKC-1024:** Invalid netmask: {0}
- **Cause:** Given netmask does not adhere to the standard.
- **Action:** Change netmask to a valid netmask.

**PRKC-1025:** Failed to create a file in directory {0}
- **Cause:** Might be due to insufficient permission or due to insufficient disk space.
- **Action:** Check the permission on the directory and available space.

**PRKC-1026:** Failed doing I/O to file {0}
- **Cause:** Might be due to insufficient permission or due to insufficient disk space or network issue.
- **Action:** Check the permission on the directory and available space. Make sure the storage medium is accessible.

**PRKC-1027:** Error checking existence of file {0} on {1}
- **Cause:** Might be due to improper user equivalence setup or due to insufficient permission on the parent directory.
- **Action:** Check user equivalence is setup correctly and the directory has read and execute permission.

**PRKC-1028:** Error checking write permission for directory {0} on {1}
- **Cause:** Might be due to improper user equivalence setup or due to insufficient permission on the parent directory.
- **Action:** Check user equivalence is setup correctly and the directory has read and execute permission.

**PRKC-1029:** Failed to get modification time for file {0} on node {1}, {2}
- **Cause:** Might be due to improper user equivalence setup or due to insufficient permission on the parent directory.
- **Action:** Check user equivalence is setup correctly and the directory has read and execute permission.

**PRKC-1030:** Error checking accessibility for node {0}, {1}
- **Cause:** Might be due to network issue or improper user equivalence setup.
- **Action:** Check node reachability between the local and the remote nodes and make sure user equivalence is setup correctly.

**PRKC-1031:** Error checking free space for {0} on {1}
- **Cause:** Might be due to network issue or improper user equivalence setup.
- **Action:** Check node reachability between the local and the remote nodes and make sure user equivalence is setup correctly.

**PRKC-1032:** Directory {0} does not exist
- **Cause:** The specified directory does not exist.
**Action:** Make sure the specified directory exists. Use absolute path for the directory.

**PRKC-1033: Executable {0} does not exist**
- **Cause:** The specified executable does not exist.
- **Action:** Make sure the existence of executable before retrying.

**PRKC-1034: No local node name found for host {0}**
- **Cause:** The local node may not be part of cluster or there was an error occurred while retrieving local node name.
- **Action:** Check CRS logs in ORA_CRS_HOME/log/<hostname> and OCR logs in ORA_CRS_HOME/srvm/log

**PRKC-1035: Node names for this cluster could not be retrieved**
- **Cause:** An error occurred while retrieving node names from the cluster.
- **Action:** Check CRS logs in ORA_CRS_HOME/log/<hostname> and OCR logs in ORA_CRS_HOME/srvm/log.

**PRKC-1036: CRS_HOME name passed to the method was null**
- **Cause:** The CRS home name cannot be null.
- **Action:** Pass non-null CRS_HOME to the method.

**PRKC-1037: Error removing files listed in {0} from node {1}**
- **Cause:** Might be due to improper user equivalence setup or due to insufficient permission.
- **Action:** Check user equivalence is setup correctly and the mentioned files have correct permission.

**PRKC-1038: Error copying files listed in {0} to node {1}**
- **Cause:** Might be due to improper user equivalence setup or due to insufficient permission.
- **Action:** Check user equivalence is setup correctly and the mentioned files have correct permission.

**PRKC-1039: Error creating directories listed in {0} on node {1}**
- **Cause:** Might be due to improper user equivalence setup or due to insufficient permission.
- **Action:** Check user equivalence is setup correctly and the mentioned directories have correct permission.

**PRKC-1040: Remote Shell is not known yet.**
- **Cause:** User equivalence is not yet set.
- **Action:** Setup user equivalence.

**PRKC-1041: Remote Copy command is not known yet.**
- **Cause:** User equivalence is not yet set.
- **Action:** Setup user equivalence.

**PRKC-1042: The Remote Shell {0} requested by client is not recognized**
- **Cause:** The system does not recognize the given remote shell command.
- **Action:** Use ssh or rsh as the remote shell.
PRKC-1043: The Remote Copy command {0} requested by client is not recognized
   Cause: The system does not recognize the given remote copy command.
   Action: Use scp or rcp for remote copy.

PRKC-1044: Failed to check remote command execution setup for node {0} using
   shells {1} and {2}
   Cause: Could not find the remote shell commands {1} and {2} on node {0}.
   Action: Setup user equivalence. Use environment variables to pass remote shell
   commands location.

PRKC-1045: Node name is null
   Cause: Node name received is null.
   Action: Pass non-null node name to method/command.

PRKC-1046: The file list passed to the method was null
   Cause: The list of files obtained is null.
   Action: Pass non-null file list to method.

PRKC-1047: The directory list passed to the method was null
   Cause: The list of directory obtained is null.
   Action: Pass non-null directory list to method.

PRKC-1048: {0} is not supported on Windows platform
   Cause: Unsupported operation.
   Action: Do not attempt {0} on Windows.

PRKC-1049: CRS is not supported in version {0}. It is supported from version {1}
   onwards
   Cause: Unsupported operation.
   Action: Upgrade to version {1} before attempting CRS commands.

PRKC-1050: EVM is not supported in version {0}. It is supported from version {1}
   onwards
   Cause: Unsupported operation.
   Action: Upgrade to version {1} before attempting to use EVM.

PRKC-1051: Invalid data to set for registry key {0}.
   Cause: Data validation failed for registry operation.
   Action: Check data passed as value for registry key {0}

PRKC-1052: Invalid data type for registry key {0}.
   Cause: Data validation failed for registry key {0}.
   Action: Check data type of value for registry key {0}.

PRKC-1053: Error returned from node {0} is "{1}" 
   Cause: 
   Action: Contact your customer support representative.

PRKC-1054: Node {0} is not accessible
   Cause: Might be due to network issue or improper user equivalence setup.
Action: Contact network/system administrator to check on connectivity/health of node {0}

PRKC-1055: Directory name passed was null
    Cause: 
    Action: Ensure non-null directory name is passed to command

PRKC-1056: Failed to get the hostname for node {0}
    Cause: Unable to fetch the hostname.
    Action: Check hostname of node {0}.

PRKC-1057: The computername and hostname do not match for node {0}. Make sure that they match.
    Cause: 
    Action: Contact network/computer administrator to reconfig computername and/or hostname before reattempting operation

PRKC-1058: The hostname registry key {0} has an empty value on node {1}
    Cause: Null value was passed for the registry key {0}
    Action: Contact your customer service representative.

PRKC-1059: The computername registry key {0} has an empty value on node {1}
    Cause: Null value was passed for the registry key {0}
    Action: Contact your customer service representative.

PRKC-1060: Cluster misconfigured on node "{0}" ", "{1}" property in {2} does not match with configured data in Oracle Cluster Registry
    Cause: Improper setup for the cluster.
    Action: Contact your customer service representative.

PRKC-1061: Failed to list contents of the directory {0}
    Cause: Might be due to insufficient permission.
    Action: Check the permission and the contents of directory {0}.

PRKC-1062: Node list is null
    Cause: The node list is null.
    Action: Pass non-null node list to method/command.

PRKC-1063: Node list {0} contained {1} nodes, minimum number of nodes required is {2}
    Cause: 
    Action: Modify nodelist to contain appropriate number of nodes before retrying method/command

PRKC-1064: Node list {0} contained {1} nodes whereas the cluster has {2} nodes
    Cause: The node list configured for the cluster does not match with the given node list.
    Action: Modify nodelist to contain {2} nodes.

PRKC-1065: Nodes {0} do not belong to this cluster
    Cause: The provided nodes are not configured for the cluster.
    Action: Remove invalid nodes that do not belong to cluster.
PRKC-1066: Failed to retrieve node number for node "{0}\n
Cause: Cluster was not properly setup.
Action: Check CRS logs for detailed error information.

PRKC-1067: File name is null
Cause: 
Action: Change file name to non-null value and retry method invocation.

PRKC-1068: Failed to get private interconnect node name for node {0}, {1}
Cause: Cluster was not properly setup.
Action: Check logs for detailed error information.

PRKC-1069: Failed to get sub keys of "{0}" registry key, {1}
Cause: Either the parent or the sub-key does not exist.
Action: Contact your customer service representative.

PRKC-1070: Failed to get sub keys of "{0}" registry key on node "{1}\n, {2}
Cause: Either the parent or the sub-key does not exist.
Action: Contact your customer service representative.

PRKC-1071: Nodes "{0}" did not respond to ping in "{1}" seconds, {2}
Cause: The node might be down or there could be a network outage.
Action: Contact network administrator to check connectivity between specified nodes.

PRKC-1072: Invalid node "{0}" specified for checking the result of operation
Cause: Node name validation for the operation failed.
Action: Check value of node {0}. Specify valid values for nodes.

PRKC-1073: Failed to transfer directory "{0} to any of the given nodes "{1}". {2}
Cause: Could not copy the directory "{0}" to any of the given nodes "{1}".
Action: Check user equivalence is setup properly. Check the log files for detail.

PRKC-1074: Error on node {0}:{1}
Cause: 
Action: Contact your customer support representative.

PRKC-1075: User equivalence does not exist with nodes "{0}". {1}
Cause: User equivalence was not setup.
Action: Contact system administrator to check for user equivalence between specified nodes.

PRKC-1076: Oracle Home is null
Cause: 
Action: Contact your customer support representative.

PRKC-1077: List file {0} has the following invalid files:{1}
Cause: Filename validation failed for the given files.
Action: Remove invalid entries from list file.
PRKC-1078: Failed to transfer directories listed in "\{0\}" to any of the given nodes "\{1\}". (2)
   Cause: Could not copy the directory "\{0\}" to any of the given nodes "\{1\}".
   Action: Check user equivalence is setup properly. Check the log files for detail.

PRKC-1079: {0} is not contained within an Oracle Home
   Cause:
   Action: Use {0} contained within an Oracle home.

PRKC-1080: Failed to transfer file "\{0\}" to any of the given nodes "\{1\}". (2)
   Cause: Could not copy the file "\{0\}" to any of the given nodes "\{1\}".
   Action: Check user equivalence is setup properly. Check the log files for detail.

PRKC-1081: Failed to transfer listed files in "\{0\}" to any of the given nodes "\{1\}". (2)
   Cause: Could not copy the files "\{0\}" to any of the given nodes "\{1\}".
   Action: Check user equivalence is setup properly. Check the log files for detail.

PRKC-1082: Failed to create listed directory in "\{0\}" to any of the given nodes "\{1\}". (2)
   Cause: Could not create the directory "\{0\}" to any of the given nodes "\{1\}".
   Action: Check user equivalence is setup properly. Check permissions for the parent directory on the given nodes.

PRKC-1083: Failed to remove listed directory in "\{0\}" to any of the given nodes "\{1\}". (2)
   Cause: Could not remove the directory "\{0\}" to any of the given nodes "\{1\}".
   Action: Check user equivalence is setup properly. Check permissions for the parent directory on the given nodes.

PRKC-1084: Failed to create path "\{0\}" in any of the given nodes "\{1\}". (2)
   Cause: Could not create the path "\{0\}" to any of the given nodes "\{1\}".
   Action: Check user equivalence is setup properly. Check permissions for the parent directory on the given nodes.

PRKC-1085: Failed to retrieve Oracle private name for node "\{0\}".
   Cause: Improper setup.
   Action: Check if node {0} is part of cluster. If so, contact customer support representative. If not, correct node name.

PRKC-1086: Unknown path type "\{0\}" specified for path "\{1\}" existence check
   Cause: Unknown path type.
   Action: Specify valid path type for path existence check.

PRKC-1087: Failed to retrieve virtual IP for node "\{0\}".
   Cause: Cluster is not setup properly.
   Action: Check if node is part of cluster. If so contact customer support representative. If not, correct node name.

PRKC-1088: Failed to recursively list files from the directories specified in file: {0}
   Cause: File access issue.
Action: Check the files exist and make sure the permissions are correct.

PRKC-1089: List file \{0\} is empty
  Cause: An empty list file was passed to a transfer/copy command.
  Action: Fix the contents of the list file before reattempting command.

PRKC-1090: Failed to update environment on nodes \"\{0\}\", \{1\}
  Cause: Might be due to improper privilege.
  Action: Check user privilege on nodes \"\{0\}\".

PRKC-1091: The include list file passed is null
  Cause:
  Action: Pass non-null include file list to retry command/method.

PRKC-1092: Failed to retrieve the location of votedisks: \{0\}
  Cause: Node name passed to command is not part of cluster.
  Action: Invalid node name passed to command. Use node name that is part of cluster.

PRKC-1093: Failed to retrieve the version of crs software on node \"\{0\}\" : \{1\}
  Cause: Cluster is not setup properly.
  Action: Contact your customer support representative.

PRKC-1094: Failed to retrieve the active version of crs: \{0\}
  Cause: Cluster is not setup properly.
  Action: Contact your customer support representative.

PRKC-1095: OLE initialization or OCX load error while registering OCX library on nodes \"\{0\}\", \{1\}
  Cause: Cluster is not setup properly.
  Action: Contact your customer support representative.

PRKC-1096: Failed to do node reboot on nodes \"\{0\}\", \{1\}
  Cause: Cluster is not setup properly
  Action: Contact your customer support representative.

PRKC-1097: PRKC- The file, \"\{0\}\" is not an OCX
  Cause: Cluster is not setup properly.
  Action: Contact your customer support representative.

PRKC-1098: Failed to create new listfile with proper UNC path from the path specified in file: \{0\}
  Cause: Internal error.
  Action: Contact your customer support representative.

PRKC-1099: The host names or IP addresses passed as an argument are null
  Cause:
  Action: Pass non-null arguments for hostnames or IP addresses.

PRKC-1100: The network port \{0\} is already in use
  Cause: Preoccupied network port \{0\}.
Action: Free port [0] for use by CRS infrastructure.

**PRKC-1101: The network port [0] is not available for use.**
Cause: The port [0] might be used by another application.
Action: Make port [0] available for use.

**PRKC-1102: CSS is not configured with local-only OCR on node "[0]\"**
Cause: CSS is not configured to run in a stand alone environment.
Action: Follow the documentation to setup CSS properly.

**PRKC-1103: Failed to check CSS status for any of the given nodes "[0]", [1]**
Cause: CSS might be down. User equivalence might not be set or there could be a network issue.
Action: Check the logs to see details.

**PRKC-1104: Property, "[0]\" is not found in file "[1]\"**
Cause: File "[1]\" does not contain the property "[0]\".
Action: Check the file '[1]'.

**PRKC-1105: Failed to retrieve the local CSS home for node "[0]\"**
Cause: Cluster is not setup properly.
Action: Contact your customer support representative.

**PRKC-1106: Failed to remove file "[0]\" on node "[1]", [2]**
Cause: Could not remove the file "[0]\" on node "[1]".
Action: Check user equivalence is setup properly. Check permissions for the parent directory as well as the file on the given nodes.

**PRKC-1107: Failed to create symbolic link from file "[0]\" to "[1]\" on node "[2]\", [3]**
Cause: Could not create symbolic link.
Action: Check user equivalence is setup properly. Check permissions for the parent directory as well as the file on the given nodes.

**PRKC-1108: Failed to copy file "[0]\" on node "[1]\" to file "[2]\" on node "[3]\", [4]**
Cause: Could not copy the file "[0]\" on node "[1]\".
Action: Check user equivalence is setup properly. Check permissions for the parent directory as well as the file on the given nodes.

**PRKC-1109: Failed to move file "[0]\" to file "[1]\" on node "[2]\", [3]**
Cause: Could not move the file "[0]\" to file "[1]\" on node "[2]\".
Action: Check user equivalence is setup properly. Check permissions for the parent directory as well as the file on the given nodes.

**PRKC-1110: Failed to create directory "[0]\" on node "[1]\", [2]**
Cause: Could not create the directory "[0]\" on node "[1]\".
Action: Check user equivalence is setup properly. Check the log files for detail.

**PRKC-1111: Failed to remove directory "[0]\" on node "[1]\", [2]**
Cause: Could not remove the directory "[0]\" on node "[1]\".
Action: Check user equivalence is setup properly. Check the log files for detail.
PRKC-1112: Failed to list contents of directory "{0}" on node "{1}", [2]
   Cause: Could not read the directory "{0}" on node "{1}".
   Action: Check user equivalence is setup properly. Check the log files for detail.

PRKC-1113: Directory "{0}" does not exist on node "{1}"
   Cause: The mentioned directory does not exist.
   Action: Check for existence of directory {0}. Use absolute path for the directory.

PRKC-1114: Registry key name is null
   Cause:
   Action: Correct registry key name passed to command.

PRKC-1115: Registry subkey name is null
   Cause:
   Action: Correct registry subkey name passed to command.

PRKC-1116: Failed to create registry subkey "{0}" under key "{1}" on node "{2}", [3]
   Cause: Registry operation failed.
   Action: See logs for detail.

PRKC-1117: Failed to delete registry subkey "{0}" under key "{1}" on node "{2}", [3]
   Cause: Registry operation failed.
   Action: See logs for detail.

PRKC-1118: Failed to check existence of registry key "{0}" on node "{1}", [2]
   Cause: Registry operation failed.
   Action: See logs for detail.

PRKC-1119: Failed to set data for value "{0}" of registry key "{1}" on node "{2}", [3]
   Cause: Registry operation failed.
   Action: See logs for detail.

PRKC-1120: Failed to retrieve data for value "{0}" of registry key "{1}" on node "{2}", [3]
   Cause: Registry operation failed.
   Action: See logs for detail.

PRKC-1121: Failed to delete value "{0}" of registry key "{1}" on node "{2}", [3]
   Cause: Registry operation failed.
   Action: See logs for detail.

PRKC-1122: Service name is null
   Cause: Empty service name provided.
   Action: Reattempts the command with a non-null service name.

PRKC-1123: Failed to create service "{0}" on node "{1}", [2]
   Cause:
   Action: Please refer to logs for detailed error information.
PRKC-1124: Failed to create service dependency between services "{0}" and "{1}" on node "{2}", [3]
  Cause: 
  Action: Please refer to logs for detailed error information.

PRKC-1125: Failed to start service "{0}" on node "{1}", [2]
  Cause: 
  Action: Please refer to logs for detailed error information.

PRKC-1126: Failed to stop service "{0}" on node "{1}", [2]
  Cause: 
  Action: Please refer to logs for detailed error information.

PRKC-1127: Failed to delete service "{0}" on node "{1}", [2]
  Cause: 
  Action: Please refer to logs for detailed error information.

PRKC-1128: Invalid Interface type {0} is specified as an argument.
  Cause: 
  Action: Correct interface type before retrying command.

PRKC-1129: Invalid IP address type {0} is specified as an argument.
  Cause: 
  Action: Change IP address type before reattempting command.

PRKC-1130: Version argument is null.
  Cause: 
  Action: Correct version argument and retry method invocation.

PRKC-1131: Failed to set user permissions on path {0} on nodes {1}, [{2}]
  Cause: Might be due to improper user equivalence setup.
  Action: Check user equivalence is set correctly.

PRKC-1132: Failed to set administrator permissions on path {0} on nodes {1}, [{2}]
  Cause: Might be due to improper user equivalence setup.
  Action: Check user equivalence is set correctly.

PRKC-1133: Names of services on which this service "{0}" is dependent on is null.
  Cause: 
  Action: Invalid dependency list passed to method. Correct dependency list and retry command.

PRKC-1134: Failed to remove service dependency between services "{0}" and "{1}" on node "{2}", [3]
  Cause: 
  Action: Refer to logs for detailed error information.

PRKC-1135: Registry key name {0} is not valid, it must be fully defined registry key path
  Cause: Invalid registry key format.
  Action: Provide valid registry key path.
PRKD—Global Services Daemon Messages

PRKD-3000: Failed to initialize and register with clusterware
Cause: This can occur if cluster synchronization services was not functioning properly.
Action: Check the state of your clusterware by running olsnodes from ORA_CRS_HOME/bin. It should list the nodes in the cluster.

PRKD-3001: Ready to receive client requests
Cause: Global Services Daemon is ready.
Action: No action needed at this time.

PRKE—Global Services Daemon Controller Utility Messages

PRKE-1008: Failed to get list of active nodes from clusterware
Cause: Not operating in cluster mode, failure to load srvm libraries, failure to initialize clusterware context.
Action: Check for the health of clusterware using “lsnodes” utility to get the list of active nodes in the cluster. Also, check for the presence of srvm libraries in the load library path. If these two checks get cleared, please contact your customer support representative.

PRKE-1009: Failed to start GSD on local node
Cause: GSD is already running on the local node, or GSD could not be started because of some other reason.
Action: Run “srvctl status” to check if a daemon is already running on the local node. If a daemon is not running on local node, please contact your customer support representative to check for the other reasons for failure.

PRKE-1010: Failed to stop GSD on local node
Cause: GSD is not running on the local node, or GSD could not shutdown gracefully.
Action: Run “srvctl status” to check if a GSD is running on the local node. If a GSD is running, please contact your customer support representative to check for other reasons for failure.

PRKE-1011: Failed to get status of GSD on local node
Cause: Failure to get the list of active nodes from the cluster, or failure to query the list of live daemons from the clusterware.
Action: Run “lsnodes” to check the health of the clusterware. If the clusterware is working fine, check for the presence of srvm libraries in the load library path. If the libraries are present, please contact your customer support representative for further assistance.

PRKH—Server Manager (SRVM) Messages

PRKH-1000: Unable to load the SRVM HAS shared library
Cause: The system cannot find, or load the srvm has shared library.
Action: Check your system’s library load path. Verify that the library exists, and that the libraries it depends on are readable and are in the load path.

PRKH-1001: HASContext Internal Error
Cause: An unexpected internal error has occurred while attempting to communicate with CRS.
Action: Contact Oracle support

PRKH-1002: Internal HASContext Error: JNI Native Call Failure
Cause: An unexpected internal error has occurred while executing native code from java.
Action: Contact Oracle support

PRKH-1003: Failed to allocate memory in native layer:
Cause: The SRVM framework was unable to allocate memory.
Action: Your system is running low on memory. Check the memory and swap space, resolve the problem and retry.

PRKH-1004: Failed to execute remote join cluster alias {0} for nodes:
Cause: Unable to execute a join cluster alias operation on another node.
Action: Check the failed nodes. Verify that the system can execute remote commands on those nodes. Check the cluster alias and it’s related functionality.

PRKH-1005: Failed to execute remote leave cluster alias {0} for nodes:
Cause: Unable to execute a leave cluster alias operation on another node.
Action: Check the failed nodes. Verify that the system can execute remote commands on those nodes. Check the cluster alias and it’s related functionality.

PRKH-1006: The following nodes are either not active or not configured:
Cause: The system was unable to find a node requested.
Action: Please check the node list supplied and verify that the node is available and configured.

PRKH-1007: Exception Caused by:
Cause: The current exception was caused by another earlier exception.
Action: Examine all of the nested exceptions to determine the root cause of the error.

PRKH-1008: Internal HASContext Error: Argument {0} must be set.
Cause: Internal software error.
Action: Contact Oracle support.

PRKH-1009: CRS HOME must be defined in the environment or in the Oracle Cluster Registry
Cause: The system is unable to determine the ORA_CRS_HOME for this CRS installation.
Action: Check that the Oracle Cluster Registry is properly configured and available to this node.

PRKH-1010: Unable to communicate with CRS services
Cause: The system is unable to communicate with the CRS services.
Action: Check that all the CRS daemons are running and are properly configured. Verify that the current program is attempting to communicate with the correct CRS daemons.
PRKH-1011: Process does not have sufficient privileges to perform the requested operation.  [0]

Cause: The user running the current operation is not permitted to execute a given operation.

Action: Check other messages in the error stack to determine which operation failed. Verify that you are running the operation as the correct user.

PRKH-1012: Unable to find or resolve user [0]

Cause: The system was unable to find the username supplied.

Action: Check the username and try again.

PRKH-1013: The oracle home [0] does not exist

Cause: The system was unable to find the oracle home supplied.

Action: Check the full path of the oracle home and try again.

PRKI—Cluster Pre-Install Messages

PRKI-2059 Unable to copy files to the nodes

Cause: Lack of permission to create destination file or copy file to the destination location.

Action: Verify permission to create the destination file. Also, please make sure that the destination file is not currently in use.

PRKI-2060 Unable to create service on the nodes

Cause: Lack of permission to create the service, or service already exists.

Action: Verify permission to create service on destination node or nodes. Also, verify that the service is not already existing.

PRKI-2061 Unable to start service on the nodes

Cause: Service is not created, service is already running, service marked for deletion, service could not be started properly.

Action: Try to start the service manually from the service control panel and check the Windows Error message that it gives.

PRKI-2064 Unable to update registry entries on the nodes

Cause: Lack of permission to modify registry entries on the nodes, error while updating the registries.

Action: Verify permission to modify registry entries.

PRKI-2066 Unable to find an Oracle disk partition. Please exit from the wizard, create Oracle partitions and try again.

Cause: Absence of Oracle disk partition, failure to load OLM dll.

Action: Verify that OLM dll (oobjlib.dll) is present in the load path. Also, verify that Oracle disk partitions are present by going to the Disk Administrator and looking for partitions.

PRKI-2114 Cannot collect and verify hardware information for all the nodes. Press abort to exit the wizard or ignore to continue.

Cause: Failure to connect to the remote nodes while adding a node to a cluster. Failure to collect VIA information if VIA was chosen.

Action: If you are trying to add a node to an already existing cluster, try to map a drive letter to a drive on each of the remote node (using commands like "net use").
Also, verify that we have permission to start temporary service on remote nodes that will collect VIA information, in case VIA was chosen.

PRKI-2116 Cannot connect to remote drive on node {0}
   Cause: Failure to connect to remote node.
   Action: Try to map a driver letter on local node to the remote node using commands like "net use".

PRKI-2119 Unable to delete files on {0}. Install may not succeed if files are not deleted
   Cause: Files we are trying to delete do not exist, or are currently being used.
   Action: If the files we are trying to delete do not exist, ignore this error. If the files we are trying to delete are currently being used, please end the processes that are using the files and then retry.

PRKI-2120 Unable to delete service {0} on {1}. Install may not succeed if services are not deleted
   Cause: Service does not exist, or service could not be deleted.
   Action: Check the service control panel to see if the service exists. If the service does not exist, ignore this error. If the service exists, try to delete it from the command line using utilities like 'net' and 'sc'. If this fails, check the Windows error message returned by these commands.

PRKN—Server Manager (SRVM) System Library Messages

PRKN-1008: Unable to load the shared library "{0}" or a dependent library, from
   [1]=2 [3]
   Cause: The SRVM framework was unable to load a shared library to perform native operations.
   Action: Make sure that the library in question is installed and in a location where it can be loaded. Check the environment setting that determines shared library loading on your platform.

PRKN-1009: Native System Internal Error
   Cause: An unexpected internal error occurred while attempting to execute a native operation from Java.
   Action: Contact Oracle Support.

PRKN-1010: This system is not properly configured as a cluster
   Cause: SRVM was unable to find the system libraries which are required in order to use the system as a cluster.
   Action: Verify that the CRS installation was done properly, try to execute 'srvctl status' command to get more diagnostic information.

PRKN-1011: Failed to retrieve value for "{0}" under registry key "[1]" on node "{2}
   Cause: There was a problem accessing the Windows registry key.
   Action: Open Windows registry editor on the node, check if the registry key exists and it contains the subkey.
PRKO—Server Control (SRVCTL) Utility Messages

PRKO-2001: "Invalid command line syntax"

Cause: An invalid SRVCTL command line was entered.
Action: Use -h SRVCTL command line option to find out the correct command line syntax and re-enter the command.

PRKO-2002: "Invalid command line option: "

Cause: An invalid SRVCTL command line option was entered.
Action: Use -h SRVCTL command line option to find out the correct command line syntax and re-enter the command.

PRKO-2003: "Invalid command line option value: "

Cause: An invalid SRVCTL command line option value was entered.
Action: Use -h SRVCTL command line option to find out the correct command line syntax and re-enter the command.

PRKO-2004: "Repetition of command line option: "

Cause: Duplicate SRVCTL command line option was entered.
Action: Eliminate the duplicate and re-enter the command.

PRKO-2005: "Application error: Failure in getting Cluster Database Configuration for: "

Cause: An error occurred when getting Cluster Database configuration for the named database.
Action: Make sure that the database has been configured in Cluster Database Configuration Repository; make sure that GSDs are running on each node in the cluster.

PRKO-2006: "Invalid node name: "

Cause: An invalid node name was entered.
Action: Use the correct node name. A valid node name should match the output from 'lsnodes' and must not contain domain name.

PRKO-2007: "Invalid instance name: "

Cause: An invalid instance name was entered.
Action: Use the correct instance name for the database. Run 'srvctl config database -d <db_name>' command to find out all instances of a database in the Cluster Database Configuration.

PRKO-2008: "Invalid connect string: "

Cause: An invalid connect string was entered.
Action: Use the correct connect string syntax: <user>/<password>[ as <role>]

PRKO-2009: "Invalid name/value string: "

Cause: An invalid environment name/value pair was entered during SRVCTL setenv command.
Action: Make sure that the correct name/value string format is used: <name>=<value>

PRKO-2010: "Error in adding instance to node: "
Cause: An error occurred when adding instance to Cluster Database configuration.

Action: Use 'srvctl config database -d <db_name>' command to check if the database has been configured in Cluster Database Configuration Repository; make sure that GSDs are running on each node in the cluster.

PRKO-2011: "Error in removing instance: "
Cause: An error occurred when removing an instance from Cluster Database Configuration.

Action: Use 'srvctl config database -d <db_name>' command to check if the database and instance have been configured in Cluster Database Configuration Repository; make sure that GSDs are running on each node in the cluster.

PRKO-2012: "Error in moving instance to node: "
Cause: An error occurred when changing the instance and node mapping in Cluster Database Configuration.

Action: Use 'srvctl config database -d <db_name>' command to check if the database and instance have been configured in Cluster Database Configuration Repository; make sure that GSDs are running on each node in the cluster.

PRKO-2013: "Error in setting env: "
Cause: An error occurred when setting environment variables for a database or an instance.

Action: Use 'srvctl config database -d <db_name>' command to check if the database and/or the instance have been configured in Cluster Database Configuration Repository; make sure that GSDs are running on each node in the cluster.

PRKO-2014: "Error in unsetting env: "
Cause: An error occurred when unsetting environment variables for a database or an instance.

Action: Use 'srvctl getenv' command to check if the environment variable exists for the database or instance; make sure that the database and/or the instance have been configured in Cluster Database Configuration Repository; make sure that GSDs are running on each node in the cluster.

PRKO-2015: "Error in checking condition of instance on node: "
Cause: Could not get status information of an instance.

Action: Use 'srvctl config database -d <db_name>' command to check if the instance has been configured in the Oracle database; make sure that GSDs are running on each node in the cluster.

PRKO-2016: "Error in checking condition of listener on node: "
Cause: Could not get status information of a listener.

Action: Check if the listener has been configured in the listener configuration file and if the database instance on the node is listed in SID_NAME in the listener configuration; make sure that GSDs are running on each node in the cluster.

PRKO-2017: "Service {0} is not supported on instance {1}."
Cause: The named instance is not configured in the service configuration.

Action: In service related operations, make sure you operate on the instances that are configured for the service.
PRKO-2018: "Mutual exclusive command line options -c and -q cannot be used at the same time."
   
   **Cause:** Both command line option -c and -q are specified on the command line.
   **Action:** Use either one of the options, but not both.

PRKO-2019: Cannot change management policy when database is disabled. Either remove -y option or enable database before before running this command
   
   **Cause:** The database resource attributes cannot be changed when it is in online state.
   **Action:** Stop the database using command `srvctl stop database` and then rerun the command to change policy.

PRKO-2120: "The internal database service {0} cannot be managed with srvctl."
   
   **Cause:** The internal database service is not supposed to be a HA service.
   **Action:** Do not operate on the internal database service.

PRKO-2101: "Instance is disabled:"
   
   **Cause:** The instance disabled and cannot be started.
   **Action:** Enable the instance before starting.

PRKO-2102: "Service {0} is not supported on instance {1}.
   
   **Cause:** The named instance is not configured in the service configuration.
   **Action:** In service related operations, make sure you operate on the instances that are configured for the service.

PRKO-2104: "Error in checking condition of service:"
   
   **Cause:** There is a problem with the CRS daemon. Or, service resource is in some strange state.
   **Action:** Check if the daemon is up and listening to the port srvctl is talking to. Check the service resource status.

PRKO-2105: "Error in checking condition of VIP on node:
   
   **Cause:** There is a problem with the CRS daemon. Or, VIP resource is in some strange state.
   **Action:** Check if the daemon is up and listening to the port srvctl is talking to. Check the VIP resource status.

PRKO-2106: "Error in checking condition of GSD on node:
   
   **Cause:** There is a problem with the CRS daemon. Or, GSD resource is in some strange state.
   **Action:** Check if the daemon is up and listening to the port srvctl is talking to. Check the GSD resource status.

PRKO-2108: "Node applications are still running on node:
   
   **Cause:** Node applications are running on the node.
   **Action:** Stop the node applications before removing them.

PRKO-2109: "Invalid address string:
   
   **Cause:** Invalid address string.
   **Action:** Use the correct format for address string.

PRKO-2112: "Some or all node applications are not removed successfully on
node: "
  Cause: There was some problem when removing node applications.
  Action: Details of the problem are given in the text.

PRKO-2113: "Instance {0} is already a preferred instance for service {1}."
  Cause: The instance is configured as a preferred instance for the service.
  Action: Pick another instance to configure.

PRKO-2114: "Instance {0} is already an available instance for service {1}."
  Cause: The instance is configured as an available instance for the service.
  Action: Pick another instance to configure.

PRKO-2115: "Cannot remove node-level applications on node {0}, because a
  listener application exists. Remove the listener application first, then
  re-run this command."
  Cause: A listener application exists.
  Action: Remove the listener application first, then re-run this command.

PRKO-2116: "Error in checking condition of ONS daemon on node: "
  Cause: There is a problem with the CRS daemon. Or, ONS resource is in some
  strange state.
  Action: Check if the daemon is up and listening to the port srvctl is talking to.
  Check the ONS resource status.

PRKO-2117: "This command should be executed as the system privilege user."
  Cause: User did not login as the system privilege user.
  Action: Login as system privilege user and re-try.

PRKO-2118: "Error in checking condition of ASM instance {0} on node {1}."
  Cause: There is a problem with the CRS daemon. Or, ASM resource is in some
  strange state.
  Action: Check if the daemon is up and listening to the port srvctl is talking to.
  Check the ASM resource status.

PRKO-2119: "Error in checking enable/disable status of ASM instance {0} on
  node {1}."
  Cause: There was some problem when checking enable status of the ASM
  instance.
  Action: Make sure that OCR is functioning and that CRS daemon is running.

PRKP—Cluster Database Management Messages

PRKP-1000 "Cannot retrieve configuration for cluster database {0}" 
  Cause: The cluster database configuration cannot be retrieved from the
  repository. This can occur either because the database was never registered, or
  because the repository itself has not been created.
  Action: Check if the database has been configured by printing a list of all cluster
  databases using 'srvctl config'. If the repository has not been created, use 'srvconfig
  -init' to create it.

PRKP-1001 "Error starting instance {0} on node {1}"
Cause: The instance could not be started using the SQL*Plus startup command.
Action: Try starting the named instance manually using SQL*Plus to see why it failed.

PRKP-1002 "Error stopping instance {0} on node {1}"
Cause: The SQL*Plus shutdown command returned an error while stopping the instance.
Action: Try stopping the named instance manually using SQL*Plus to see why it failed.

PRKP-1003 "Startup operation partially failed"
Cause: Some components of the cluster database could not be started.
Action: See earlier error message for details

PRKP-1004 "Shutdown operation partially failed"
Cause: Some components of the cluster database reported errors while being stopped
Action: See earlier error messages for details

PRKP-1005 "Failed to start up cluster database {0}"
Cause: The cluster database could not be started.
Action: See earlier error messages for details.

PRKP-1006 "Failed to shut down cluster database {0}"
Cause: The cluster database reported errors while being shut down
Action: See earlier error messages for details

PRKP-1007 "Failed to start all the listeners associated with all the instances of cluster database {0}"
Cause:
Action: Contact your customer support representative.

PRKP-1008 "Failed to start listeners associated with instance {0} on node {1}"
Cause:
Action: Contact your customer support representative

PRKP-1009 "Failed to stop all the listeners associated with all the instances of cluster database {0}"
Cause: Either the listener name associated with an instance could not be determined, or "lsnrctl stop" failed for a listener.
Action: Verify that listener.ora contains a SID_LIST entry for each instance of the named database, and that the lsnrctl stop command succeeds for those listeners

PRKP-1010 "Failed to stop all the listeners associated with instance {0} on node{1}"
Cause: Either the listener name associated with an instance could not be determined, or "lsnrctl stop" failed for a listener.
Action: Verify that listener.ora contains a SID_LIST entry for each instance of the named database, and that the lsnrctl stop command succeeds for those listeners

PRKP-1011 "Failed to get all the listeners associated with instance {0} on node{1}"
Cause: The listener name associated with an instance could not be determined.
Action: Ensure that listener.ora contains a SID_LIST entry for the named instance.
PRKP-1012 "Invalid environment variable [0] setting for cluster database [1]"

Cause: The argument to the -t option is not of the form <name>=<value> or it contains special characters.

Action: Ensure that the -t option has an argument of the form <name>=<value>. Enclose the argument to the -t flag in quotes.

PRKP-1013 "[0]: undefined environment variable for cluster database [1]"

Cause: The named environment variable is not defined for the named cluster database

Action: Set a value for the variable with "srvctl set env"

PRKP-1014 "[0]: undefined environment variable for instance [1] of cluster database [2]"

Cause: The named environment variable is not defined for the given instance

Action: Set a value for the variable with "srvctl set env"

PRKP-1015 "[0]: undefined environment variable"

Cause: The named environment variable is not defined

Action: Set a value for the named environment variable with "srvctl set env"

PRKP-1016 "Database [0] already enabled"

Cause: An attempt was made to enable a database that is already enabled.

Action: No action required.

PRKP-1017 "Instance [0] already enabled."

Cause: An attempt was made to enable an instance that is already enabled.

Action: No action required.

PRKP-1018 "Service [0] already enabled."

Cause: An attempt was made to enable a service that is already enabled.

Action: No action required.

PRKP-1019 "Database [0] already disabled."

Cause: An attempt was made to disable a database that is already disabled.

Action: No action required.

PRKP-1020 "Instance [0] already disabled."

Cause: An attempt was made to disable an instance that is already disabled.

Action: No action required.

PRKP-1021 "Service [0] already disabled."

Cause: An attempt was made to disable a service that is already disabled.

Action: No action required.

PRKP-1022 "The database [0] is still running."

Cause: An attempt was made to delete a database that is still running.

Action: Stop the database using 'srvctl stop database' before deleting the database.

PRKP-1023 "The instance [0] is still running."

Cause: An attempt was made to delete an instance that is still running.
Action: Stop the instance using 'srvctl stop instance' before deleting the instance.

PRKP-1024 "The service {0} is still running."
  Cause: An attempt was made to delete a service that is still running.
  Action: Stop the service using 'srvctl stop service' before deleting the service.

PRKP-1025 "The service {0} does not exist."
  Cause: An attempt was made to operate on a non-configured service.
  Action: Check if the service is configured through 'srvctl status service'.

PRKP-1026 "No instance found for database {0}."
  Cause: An attempt was made to operate on a non-configured instance.
  Action:

PRKP-1027 "Instance {0} is not found for database {1}."
  Cause: An attempt was made to operate on a non-configured instance.
  Action: Check if the instance is configured through 'srvctl config instance'.

PRKP-1028 "No preferred instance(s) for service {0}."
  Cause: An attempt was made to create a service without preferred instances.
  Action: Supply preferred instances for the service through 'srvctl create service'.

PRKP-1029 "Failed to register the service {0}."
  Cause: Internal error.
  Action: Contact support.

PRKP-1030 "Failed to start the service {0}."
  Cause: Internal error.
  Action: Contact support.

PRKP-1031 "Failed to stop the service {0}."
  Cause: Internal error.
  Action: Contact support.

PRKP-1032 "Cannot start the disabled service {0}."
  Cause: Internal error.
  Action: Contact support.

PRKP-1033 "Cannot relocate service {0} from instance {1} to instance {2}."
  Cause: Internal error.
  Action: Contact support.

PRKP-1034 "{0}: undefined environment variable for node {1}."
  Cause: Internal error.
  Action: Contact support.

PRKP-1035 "Invalid environment variable {0} setting for node {1}."
  Cause:
  Action: Contact support.

PRKP-1036 "Failed to unregister HA resource {0}"
Cause: Contact support.
Action: Contact support.

PRKP-1037 "Failed to create cluster database {0}.
Cause: Contact support.
Action: Contact support.

PRKP-1038 "Invalid instance {0} specified for the Service {1}.
Cause: Contact support.
Action: Contact support.

PRKP-1039 "operation result is null"
Cause: Contact support.
Action: Contact support.

PRKP-1041: "Failed to reload all the listeners associated with cluster database {0}"
Cause: Refer to CRS logs for detailed error information.
Action: Refer to CRS logs for detailed error information.

PRKP-1042: "Failed to reload all the listeners associated with instance {0} on node{1}"
Cause: Refer to CRS logs for detailed error information.
Action: Refer to CRS logs for detailed error information.

PRKP-1043: "Failed to reload listeners on node {0}"
Cause: Refer to CRS logs for detailed error information.
Action: Refer to CRS logs for detailed error information.

PRKP-1044 "Failed to enable the database {0}.
Cause: Contact support.
Action: Contact support.

PRKP-1045 "Failed to disable the database {0}.
Cause: Contact support.
Action: Contact support.

PRKP-1046 "Failed to enable the instance {0}.
Cause: Contact support.
Action: Contact support.

PRKP-1048 "Failed to change configuration for service {0}.
Cause: Contact support.
Action: Contact support.

PRKP-1049 "[0]: undefined environment variable for service {1} of cluster database {2}"
Cause: The environment variable is not defined for the service.
Action: No action required.

PRKP-1050 "Failed to remove the service {0}.
Cause: There was a problem while executing the crs_unregister command.
Action: Verify whether the crs_unregister command succeeds in unregistering a CRS resource.

PRKP-1051 "Failed to remove the service \{0\} on instance \{1\}."
  Cause: There was a problem while executing the crs_unregister command.
  Action: Verify whether the crs_unregister command succeeds in unregistering a CRS resource.

PRKP-1052 "Failed to enable the service \{0\}."
  Cause: There was a problem while executing the crs_register command.
  Action: Verify whether the crs_register command succeeds in registering a CRS resource.

PRKP-1053 "Failed to disable the service \{0\}."
  Cause: There was a problem while executing the crs_register command.
  Action: Verify whether the crs_register command succeeds in registering a CRS resource.

PRKP-1054 "Failed to enable the service \{0\} on instance \{1\}."
  Cause: There was a problem while executing the crs_register command.
  Action: Verify whether the crs_register command succeeds in registering a CRS resource.

PRKP-1055 "Failed to disable the service \{0\} on instance \{1\}."
  Cause: There was a problem while executing the crs_register command.
  Action: Verify whether the crs_register command succeeds in registering a CRS resource.

PRKP-1056 "Failed to get the status of the resource \{0\}."
  Cause: There was a problem while executing the crs_stat command.
  Action: Verify whether the crs_stat command gives the status of the CRS resources registered.

PRKP-1057 "Failed to set the environment for service \{0\}."
  Cause: There was a problem while accessing the OCR configuration registry.
  Action: Check whether the OCR registry is accessible by executing a srvctl config command.

PRKP-1058 "Failed to unset the environment for service \{0\}."
  Cause: There was a problem while accessing the OCR configuration registry.
  Action: Check whether the OCR registry is accessible by executing a srvctl config command.

PRKP-1059 "Failed to get the environment for service \{0\}."
  Cause: There was a problem while accessing the OCR configuration registry.
  Action: Check whether the OCR registry is accessible by executing a srvctl config command.

PRKP-1060 "Failed to get CRS home."
  Cause: Internal error.
  Action: Contact support.
PRKP-1061 "Failed to modify the database {0}."
  Cause: Internal error.
  Action: Contact support.

PRKP-1062 "Service {0} is already running."
  Cause: Attempt to start a service that is already running.
  Action: None required.

PRKP-1063 "Service {0} is already stopped."
  Cause: Attempt to stop a service that is already stopped.
  Action: None required.

PRKP-1064 "Service {0} is already running on instance {1}."
  Cause: Attempt to start a service on an instance where it is already running.
  Action: None required.

PRKP-1065 "Service {0} is already stopped on instance {1}."
  Cause: Attempt to stop a service on an instance where it is already stopped.
  Action: None required.

PRKP-1066 "Instance {0} is not an available instance for service {1}."
  Cause: 
  Action: Contact support.

PRKP-1067 "Instance {0} is the last available instance for service {1}. Try modify service instead."
  Cause: 
  Action: Contact support.

PRKP-1068 "Cannot stop the critical instance {0} in critical standby database {1} because it would result in a shutdown of the primary database."
  Cause: Attempt to stop the critical instance in critical standby database while the primary database is running.
  Action: Do not stop the critical instance in critical standby database while the primary database is running.

PRKP-1069 "Failed to change domain of the database {0} to {1}, because this domain name is already used by service {2} configured under the database."
  Cause: Attempt to change the database domain when there are services configured with this domain.
  Action: Do not change the database domain when there are services configured with that domain.

PRKP-1070 "Service name {0} contains illegal characters."
  Cause: Invalid characters have been specified in the service name given.
  Action: Supply a name for the service with the character set [a-zA-Z0-9_.].

PRKP-1071: "Database unique name or instance name {0} contains illegal characters {1}."
  Cause: 
  Action: Correct db/instance name before retrying method/command.
PRKP-1072 "Failed to create service {0} for database {1}, because the specified service
domain name is the same as the database domain {2}.”
    Cause:
    Action:

PRKP-1073 "Cannot create database {0} because a database named {1} already exists.”
    Cause: Attempt to create a database that already exists.
    Action: Choose a different name for the database being created.

PRKP-1074 "Failed to relocate a service resource to instance {0} during modifying
    service configuration for service {1}.”
    Cause: Internal error.
    Action: Contact support.

PRKP-1075 "Instance {0} is the last preferred instance for service {1}.”
    Cause: Internal error.
    Action: Contact support

PRKP-1076: Instance name passed to the method is null
    Cause: Oracle instance SID passed this API cannot be null
    Action: Make sure Oracle instance is configure before this call. Check client
    specific error message for details

PRKP-1077: Oracle home is null for the cluster database {0}
    Cause: Oracle home specified cannot be null
    Action: Make sure Oracle home is properly set. Check client specific error
    message for more details

PRKP-1078: Failed to retrieve "\{0\}" attribute value from "\{1\}" file, {2}
    Cause: An error occurred while retrieving CRS attribute value because of some
    CRS/OCR error.
    Action: Refer to CRS or OCR logs in CRS home/log/<hostname> for detailed
    error information

PRKP-1079: Cannot start service {0} on disabled database {1}
    Cause: Service cannot be started on a database which is already disabled
    Action: Enable the database using 'srvctl enable database' before re-attempting
    command

PRKP-1080: "Cannot start service {0} on disabled instance {1}.”
    Cause: Service cannot be started in an instance which is already disabled
    Action: Enable the instance using 'srvctl enable instance' before attempting to
    start the service

PRKP-1081: Database name passed to the method is null
    Cause: The database name cannot be null to perform any operation.
    Action: Make sure the database name you are using and retry the command

PRKP-1082: Instance "\{0\}" does not exist in database "\{1\}"
    Cause: The configuration information for this instance may not exist in OCR, or
    instance is never configured
Action: Make sure the instance name you are using before retrying the command. Alternatively you can verify the instances configured with database by running command 'srvctl config database -d <database>'

PRKP-1083: The service {0} already exists.
Cause: You cannot have multiple database services with same name.
Action: Make sure you use unique service names for each database.

PRKP-1084: Database "{0}" cannot have more than "{1}" services
Cause: The database supports only 115 user services. You cannot configure services more than this limit.
Action: If you must add this service then remove some unused services using command 'srvctl removes services' before adding any new services

PRKR—Cluster Registry Messages

PRKR-1001 "cluster database {0} does not exist"
Cause: The cluster database was never configured in the cluster registry.
Action: Check if the database has been configured by printing a list of all cluster databases using 'srvctl config'.

PRKR-1002 "cluster database {0} already exists"
Cause: An attempt was made to configure a cluster database which was already existing in the cluster registry.
Action: Check if the database has already been configured by printing a list of all cluster databases using 'srvctl config'.

PRKR-1003 "instance {0} does not exist"
Cause: The named instance is not configured in the cluster registry.
Action: Use srvctl options to check if the instance was configured in the cluster registry.

PRKR-1004 "instance {0} already exists"
Cause: The named instance is already configured in the cluster registry.
Action: Use srvctl options to check if the instance has already been configured in the cluster registry.

PRKR-1005 "adding of cluster database {0} configuration failed, {1}" 
Cause: An error occurred while attempting to add cluster database configuration to the cluster registry.
Action: Check if the cluster registry is accessible by using 'ocrcheck' or 'srvctl config'

PRKR-1006 "deleting of cluster database {0} configuration failed, {1}" 
Cause: Same as PRKR-1005.
Action: See earlier error messages.

PRKR-1007 "getting of cluster database {0} configuration failed, {1}" 
Cause: Same as PRKR-1005.
Action: See earlier error messages.

PRKR-1008 "adding of instance {0} on node {1} to cluster database {2} failed, {3}"
Cause: Same as PRKR-1005.
Action: See earlier error messages.

PRKR-1009 "deleting of instance {0} from cluster database {1} failed, {2}"
Cause: Same as PRKR-1005.
Action: See earlier error messages.

PRKR-1010 "moving of instance {0} to node {1} of cluster database {2} failed, {3}"
Cause: Same as PRKR-1005.
Action: See earlier error messages.

PRKR-1011 "renaming of instance {0} to instance {1} of cluster database {2} failed, {3}"
Cause: Same as PRKR-1005.
Action: See earlier error messages.

PRKR-1016 "reading of cluster database {0} configuration failed, {1}, {2}"
Cause: Same as PRKR-1005.
Action: See earlier error messages.

PRKR-1017 "writing of cluster database {0} configuration failed, {1}, {2}"
Cause: Same as PRKR-1005.
Action: See earlier error messages.

PRKR-1018 "reading of directory failed, {0}, {1}"
Cause: Internal error.
Action: Contact your customer support representative.

PRKR-1019 "writing of directory failed, {0}, {1}"
Cause: Internal error.
Action: Contact your customer support representative.

PRKR-1020 "reading of version information failed, {0}, {1}"
Cause: Same as PRKR-1005.
Action: See earlier error messages.

PRKR-1021 "writing of version information failed, {0}, {1}"
Cause: Same as PRKR-1005.
Action: See earlier error messages.

PRKR-1022 "raw device {0} contains incompatible version, {1} != {2}"
Cause: An attempt was made to use an incompatible version of the cluster registry.
Action: Contact your customer support representative.

PRKR-1023 "file {0} does not exist"
Cause: The named file did not exist.
Action: Check if the file exists.

PRKR-1024 "file {0} does not have {1} permissions"
Cause: The named file did not have the specified permission.
Action: Try changing the permission on the file to the specified permission.
PRKR-1025 "file {0} does not contain property {1}"
  Cause: The file did not contain the specified property.
  Action: Contact your customer support representative.

PRKR-1026 "property {0} not set in file {1}"
  Cause: The file did not contain the specified property.
  Action: Contact your customer support representative.

PRKR-1027 "failed to retrieve list of cluster databases"
  Cause: Same as PRKR-1005.
  Action: See earlier error messages.

PRKR-1028 "raw device {0} is invalid\n[HINT: initialize raw device by using
"srvconfig" tool]"
  Cause: The cluster registry was never initialized.
  Action: Use srvconfig -init to initialize the cluster registry.

PRKR-1038 "invalid argument {0} specified to -init option"
  Cause: 
  Action: 

PRKR-1039 "invalid option {0} specified"
  Cause: The specified option was invalid.
  Action: Check usage.

PRKR-1040 "missing <file> argument for {0} option"
  Cause: The specified option was invalid for srvconfig
  Action: Check usage for details.

PRKR-1045 "raw device version "(0)"
  Cause: Attempted to retrieve the version of the cluster registry.
  Action: No action is required.

PRKR-1046 "srvconfig detected valid raw device "(0)"
  Cause: A valid cluster registry was detected.
  Action: No action is required.

PRKR-1047 "raw device {0} is in use by daemon(s) on node(s) {1}"
  Cause: An attempt was made to initialize the cluster registry while the Global
  Services Daemons were up on one or more nodes in the cluster.
  Action: Stop all Global Services Daemons on all nodes in the cluster by using
  'srvctl stop' command on every node. Try the 'srvconfig -init' operation again.

PRKR-1050 "file {0} creation in {1} directory failed, check permissions, etc."
  Cause: Attempted to create a file in a directory which did not exist or which did not
  have the right permissions.
  Action: Create the directory if it did not exist or change the permission of the
  directory.

PRKR-1051 "file {0} does not contain an entry for dbname {1}"
Cause: Internal error.
Action: Contact your customer support representative.

PRKR-1052 "file name {0} is not of <cluster database name>.conf form"
Cause: An attempt was made to register a cluster database in the cluster registry and the file argument passed was not of <cluster database name>.conf form.
Action: See usage of srvconfig for more information.

PRKR-1053 "invalid range {0} specified in node_list = {1}"
Cause: 
Action: Contact your customer support representative.

PRKR-1054 "invalid parameter {0} specified in inst_oracle_sid = {1}"
Cause: Extra number of arguments provided to srvconfig.
Action: See usage of srvconfig for details.

PRKR-1055 "invalid extra arguments {0} specified to {1} option"
Cause: Provided invalid arguments to srvconfig.
Action: See usage of srvconfig for details.

PRKR-1056 "invalid registry entry {0} found, should be of form {1}"
Cause: Detected an invalid registry entry while attempting to add a 8.1.7 or earlier version of Oracle Parallel Server in the cluster registry.
Action: Contact your customer support representative.

PRKR-1057 "environment variable does not exist"
Cause: Attempted to retrieve non existing environment variable.
Action: Set the environment variable.

PRKR-1058 "Service {0} does not exist in cluster database {1}.
Cause: The named service is not configured in the cluster registry.
Action: Use srvctl options to check if the service was configured in the cluster registry.

PRKR-1059 "Node {0} does not exist.
Cause: Node applications for the named node are not configured in the cluster registry.
Action: Use srvctl options to check if the node applications were configured for the given node in the cluster registry.

PRKR-1060 "Failed to add configuration for node {0}"
Cause: 
Action: Contact your customer support representative.

PRKR-1061 "Failed to run remote command to get node configuration for node {0}"
Cause: An internal error occurred while retrieving configuration for the given node.
Action: Contact your customer support representative.

PRKR-1062 "Failed to find configuration for node {0}"
Cause:
Action: Contact your customer support representative.

PRKR-1063 "VIP {0} is already existing"
Cause:
Action: Contact your customer support representative.

PRKR-1064 "SRVM configuration operation failed due to Oracle Cluster Registry error:"
Cause: Error occurred while accessing the Oracle Cluster Registry.
Action: Contact your customer support representative.

PRKR-1066 "cluster database domain does not match"
Cause:
Action: Contact your customer support representative.

PRKR-1067 "Failed to get environment for cluster database {0}, {1}"
Cause: Unable to retrieve the environment configuration for the given cluster database from the cluster registry.
Action: Use ocrdump to check if environment was configured for the given cluster database in the cluster registry.

PRKR-1068 "Failed to get environment for instance {1} of cluster database {0}, {2}"
Cause: Unable to retrieve the environment configuration for the given instance from the cluster registry.
Action: Use ocrdump to check if environment was configured for the given instance in the cluster registry.

PRKR-1069 "Failed to set environment for cluster database {0}, {1}"
Cause: Unable to update the environment configuration for the given cluster database to the cluster registry.
Action: Contact your customer support representative.

PRKR-1070 "Failed to set environment for instance {1} of cluster database {0}, {2}"
Cause: Unable to update the environment configuration for the given instance to the cluster registry.
Action: Contact your customer support representative.

PRKR-1071 "Failed to unset environment for cluster database {0}, {1}"
Cause: Unable to update the environment configuration for the given cluster database to the cluster registry.
Action: Contact your customer support representative.

PRKR-1072 "Failed to unset environment for instance {1} of cluster database {0}, {2}"
Cause: Unable to update the environment configuration for the given instance to the cluster registry.
Action: Contact your customer support representative.

PRKR-1073 "\n
########## Configuration of nodeapps follows ##########\n"
Cause:
Action: Contact your customer support representative.

PRKR-1074 "\n
########## Configuration of vip_range follows ##########\n"
PRKS—Automatic Storage Management Messages

PRKR-1075 "Insufficient privileges for doing this operation"

Cause: User did not have sufficient privileges when running this command.
Action: Execute this command as a privileged user.

PRKR-1076 "This command cannot run when the RAC daemons (crsd, evmd, ocssd) are running. Make sure the daemons are not running before invoking this command"

Cause: The RAC daemons were running when this command was invoked.
Action: Make sure that the RAC daemons have been stopped before running this command.

PRKR-1077 "One or more arguments passed to the function are not valid"

Cause: One or more invalid arguments were passed to the given method.
Action: Contact your customer support representative.

PRKR-1078 "Database {0} cannot be administered using current version of srvctl. Instead run srvctl from {1}"

Cause: Using incorrect SRVCTL version.
Action: Contact your customer support representative.

PRKR-1079 "Failed to initialize the Oracle Cluster Registry"

Cause: Failed to initialize the Oracle Cluster Registry.
Action: Contact your customer support representative.

PRKS—Automatic Storage Management Messages

PRKS-1000: "ASM instance "]0]\" already exists on node "]1]\"

Cause: An attempt was made to add configuration for ASM instance on the node where it already exists.
Action: Check if ASM instance was configured on the node using ’srvctl config asm -n <node>' before adding configuration for it.

PRKS-1001: ASM instance "]0]\" does not exist on node "]1]\"

Cause: The configuration for ASM instance doesn’t exist on the node.
Action: Check if ASM instance was configured on the node using ’srvctl config asm -n <node>' before performing the operation.

PRKS-1002: Failed to create CRS profile for ASM instance "]0]\" on node "]1]\", 

Cause: ’crs_stat -p’ command failed for the ASM instance resource on the node.
Action: Contact Oracle Support.

PRKS-1003: Failed to register CRS resource for ASM instance "]0]\" on node "]1]\", 

Cause: crs_register command failed for ASM instance resource on the node.
Action: Contact Oracle Support.

PRKS-1004: Failed to unregister CRS resource for ASM instance "]0]\" on node "]1]\", 


Cause: crs_unregister command failed for the ASM instance resource on the node.
Action: Check if there is a database instance dependent upon the ASM instance on the node using crs_stat -p ora.<db>.<inst> and see if ASM instance resource name appears in the required resources for the database instance. Remove the database instance's required resource dependency on ASM instance using 'srvctl modify asm' command before retrying this operation.

PRKS-1005: Failed to create CRS resource for ASM instance "[0]" on node "[1]", [2]
Cause: crs_register command failed for the ASM instance resource on the node.
Action: Contact Oracle Support.

PRKS-1006: ASM instance "[0]" is already running on node "[1]".
Cause: An attempt was made to start a running ASM instance on the node.
Action: None.

PRKS-1007: ASM instance "[0]" is still running on node "[1]".
Cause: An attempt was made to remove the configuration for a running ASM instance on the node.
Action: Stop the ASM instance using 'srvctl stop asm -n <node> -i <inst>' command before performing the remove operation.

PRKS-1008: ASM instance "[0]" is not running on node "[1]".
Cause: An attempt was made to stop a non-running ASM instance on the node.
Action: None.

PRKS-1009: Failed to start ASM instance "[0]" on node "[1]", [2]
Cause: crs_start failed for the ASM instance resource on the node, may be due to invalid startup credentials or missing parameter file.
Action: Check if VIP resource is online on the node and then try to startup the ASM instance using SQL*Plus to get more diagnostic information.

PRKS-1010: Failed to start ASM instances "[0]" on node "[1]", [2]
Cause: See above.
Action: See above.

PRKS-1011: Failed to check status of ASM instance "[0]" on node "[1]", [2]
Cause: crs_stat failed for the ASM instance resource on the node.
Action: Contact Oracle Support.

PRKS-1012: Failed to stop ASM instance "[0]" on node "[1]", [2]
Cause: crs_stop failed for the ASM instances on the node, may be due to CRS resource dependency or invalid stop ASM instance credentials.
Action: Check if there are database instances dependent upon the ASM instance on the node using crs_stat -p ora.<db>.<inst> and see if ASM instance resource names appear in the required resources for the database instance. Stop the database instances first using 'srvctl stop instance' command before retrying ASM instances stop operation or try to stop the ASM instance using SQL*Plus to get more diagnostic information.

PRKS-1013: Failed to stop ASM instances "[0]" on node "[1]", [2]
Cause: See above.
Action: See above.

PRKS-1014: Failed to enable CRS resource for ASM instance "(0)" on node "(1)", [(2)]
Caused: Failed to set the enable flag of ASM instance configuration in cluster registry.
Action: Check if cluster registry is accessible by executing 'srvctl config ' command.

PRKS-1015: Failed to enable CRS resources for ASM instances "(0)" on node "(1)", [(2)]
Caused: See above for each ASM instance.
Action: Check if cluster registry is accessible by executing 'srvctl config ' command.

PRKS-1016: Failed to disable CRS resource for ASM instance "(0)" on node "(1)", [(2)]
Caused: Failed to reset enable flag of ASM instance configuration in cluster registry.
Action: Check if cluster registry is accessible by executing 'srvctl config ' command.

PRKS-1017: Failed to disable CRS resources for ASM instances "(0)" on node "(1)", [(2)]
Caused: See above for each ASM instance.
Action: Check if cluster registry is accessible by executing 'srvctl config ' command.

PRKS-1019: Cannot create CRS dependency between database instance "(0)" configured on node "(1)" and ASM instance "(2)" configured on node "(3)".
Caused: An attempt was made to create CRS dependency between database instance and ASM instance configured on different nodes.
Action: Make sure that database instance and ASM instance are configured on the same node for creating CRS dependency between them.

PRKS-1020: Failed to create CRS dependency between database instance "(0)" and ASM instance "(1)", [(2)]
Caused: 'crs_register -u' failed to create CRS dependency between the database instance and the ASM instance on the node.
Action: Contact Oracle support.

PRKS-1021: Failed to remove CRS dependency between database instance "(0)" and ASM instance "(1)", [(2)]
Caused: 'crs_register -u' command failed to remove CRS dependency between the database instance and the ASM instance on the node.
Action: Contact Oracle Support.

PRKS-1022: Failed to remove CRS dependency between database instance "(0)" and ASM instances "(1)", [(2)]
Caused: See above.
Action: See above.

PRKS-1023: Failed to remove CRS resource for ASM instance "(0)" on node "(1)", [(2)]
Cause: crs_unregister command failed to unregister CRS resource for the ASM instances on the node.
Action: Contact Oracle Support.

PRKS-1026: ASM Configuration for node "(0)" does not exist in cluster registry.
Cause: An attempt was made to retrieve names of ASM instances configured on the node without configuring any ASM instance on it.
Action: First configure ASM instance using 'srvctl add asm' on the node command before executing the get configuration operation.

PRKS-1027: Configuration for ASM instance "(0)" does not exist in cluster registry.
Cause: An attempt was made to retrieve node name for a given ASM instance name which is not configured on any cluster nodes.
Action: First configure ASM instance using 'srvctl add asm' command before executing the get configuration operation.

PRKS-1028: Configuration for ASM instance "(0)" on node "(1)" does not exist in cluster registry.
Cause: An attempt was made to retrieve configuration for the ASM instance on the node name where it was not configured.
Action: Use 'srvctl config asm -n <node>' command to determine the ASM instance names configured on it and then pass one of these names as "-i <inst>" argument to 'srvctl config asm' command.

PRKS-1029: Client version "(0)" is not compatible with ASM instance configuration version "(1)" in cluster registry.
Cause: Version of the client that tried to retrieve ASM instance configuration is not compatible with ASM instance configuration version.
Action: Make sure client version is compatible with ASM instance configuration version before accessing it.

PRKS-1030: Failed to add configuration for ASM instance "(0)" on node "(1)" in cluster registry, [2]
Cause: Failed to add configuration for the ASM instance in cluster registry.
Action: Check if cluster registry is accessible by executing 'srvctl config' command.

PRKS-1031: Failed to retrieve configuration for ASM instance "(0)" on node "(1)" from cluster registry, [2]
Cause: Failed to retrieve configuration for the ASM instance from cluster registry.
Action: Check if cluster registry is accessible by executing 'srvctl config' command.

PRKS-1032: Failed to modify configuration for ASM instance "(0)" on node "(1)" in cluster registry, [2]
Cause: Failed to modify configuration for the ASM instance on the node in cluster registry.
Action: Check if cluster registry is accessible by executing 'srvctl config' command.

PRKS-1033: Failed to remove configuration for ASM instance "(0)" on node "(1)" from cluster registry, [2]
Cause: Failed to remove configuration for the ASM instance from cluster registry.
PRKU—Command-Line Parser Utility Messages

**PRKU-1000:** "Exception Caused by: "
*Cause:* The current exception was caused by another earlier exception.
*Action:* Examine all of the nested exceptions to determine the root cause of the error.

**PRKU-1001:** "The parameter {0} is required."
*Cause:* The parameter specified is required by a command line utility.
*Action:* Supply the parameter specified.

**PRKU-1002:** "Unexpected argument {0}."
*Cause:* The argument specified is not needed or not understood by this utility.
*Action:* Verify the arguments on the command you are executing.

**PRKU-1003:** "The parameter {0} requires an argument."
*Cause:* The parameter specified must be accompanied by an argument.
*Action:* Verify the arguments on the command you are executing.

PRKU—Command-Line Parser Utility Messages

**PRKU-1013** "A network interface is required"
*Cause:* A network interface is not provided for configuring virtual IP address.
*Action:* when running VIPCA in GUI mode, select a suitable network interface from the list. When running VIPCA in silent mode, provide a suitable interface using '-interfaces' option.

PRKV—Virtual IP Configuration Assistant Messages

**PRKV-1013** "A network interface is required"
*Cause:* A network interface is not provided for configuring virtual IP address.
*Action:* when running VIPCA in GUI mode, select a suitable network interface from the list. When running VIPCA in silent mode, provide a suitable interface using '-interfaces' option.
PRKV-1014 "Enter a valid IP address for the node "{0}".
Cause: An IP address to be configured as virtual IP is not entered for the node.
Action: Enter an unused IP address to be used for configuring virtual IP address.

PRKV-1015 "Enter a valid subnet mask for the IP address "{0}".
Cause: An IP netmask for the IP address is not entered.
Action: Enter netmask for the IP address.

PRKV-1016 "IP address "{0}" is invalid. Enter a valid IP address."
Cause: IP address entered is invalid.
Action: Enter a valid IP address in standard IEEE format.

PRKV-1017 "IP address "{0}" does not exist. Enter a valid IP address."
Cause: IP address cannot be resolved in the network.
Action: Add an IP address in DNS or /etc/hosts so it can be resolved.

PRKV-1018 "IP address "{0}" has invalid format. Enter a valid IP address."
Cause: IP address is not in standard IEEE format.
Action: Enter a valid IP address in IEEE format.

PRKV-1019 "Netmask entered "{0}" has invalid format. Enter a valid netmask."
Cause: Netmask entered in not in standard IEEE format.
Action: Enter a valid IP netmask in IEEE format.

PRKV-1039 "IP address "{0}" has already been used. Enter an unused IP address."
Cause: IP address entered is already used by another node as virtual IP address.
Action: Enter an unused IP address.

PRKV-1059: "Invalid node name "{0}" entered in an input argument."
Cause: IP address entered is already used by another node as virtual IP address.
Action: Enter an unused IP address.

PRKV-1060: "Virtual IP address is not entered for node "{0}". Enter a valid virtual IP address."
Cause:
Action: Enter valid Vitual IP address for node {0} before reattempting command.

PRKV-1061 "Invalid IP address "{0}" entered in an input argument."
Cause:
Action: Enter valid IP address in place of {0} before reattempting command.

PRKV-1062 "Invalid netmask "{0}" entered in an invalid argument."
Cause:
Action: Enter valid netmask in place of {0} before reattempting command.

PRKV-1063 "Insufficient privileges." Cause:
Action: Contact your system administrator to grant you privileges to perform required operations.

PRKV-1064 "Failed to obtain handle to CRS."
Cause:
Action: Refer to CRS logs for detailed error information.

PRKV-1065 "Host IP address "{0}" cannot be used as virtual IP for the node "{1}". Enter a different IP address.
Cause:
Action: Change IP address {0} before reattempting command.

PRKV-1066 "Host IP address "{0}" of cluster node "{1}" cannot be used as virtual IP for the node "{2}". Enter a different IP address.
Cause:
Action: Change IP address {0} before reattempting command.

PRKV-1070: Different interface names entered as an input for each node. Specify same set of interfaces for each node in the cluster.
Cause: The nodes in CRS cluster should use same set of public interfaces for configuring Virtual IP.
Action: Make sure you have public interfaces on each node of the cluster and they match in name, and subnet.

PRKV-1072: Invalid syntax used for option "nodevips". Check usage (vipca -help) for proper syntax.
Cause: The syntax argument value specified for option “nodevips” is invalid.
Action: Run “vipca -help” to see the syntax in which nodevips option value should be specified.

PRKV-1073: "Node names "{0}" are duplicated in an input argument."
Cause: The node names entered an input VIPCA command line should be unique
Action: Remove duplicate node names from VIPCA command line and rerun the command.

PRKV-1074: The given interface(s), "{0}" is not public. Public interfaces should be used to configure virtual IPs.
Cause: The interface specified as an input is not public interface. CRS cluster uses public interfaces for configuring VIPs.
Action: Find a public interface on your system that matches with name and subnet on all nodes of the cluster and specify it as an argument to VIPCA
Automatic Workload Repository (AWR)
A built-in repository that exists in every Oracle Database. At regular intervals, the 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 Real Application Clusters that provides copies of blocks directly from a holding instance’s memory cache to a requesting instance’s memory cache.

cluster
A set of interconnected instances that cooperates to perform the same task.

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 Real Application Clusters database.

CRSD - Oracle Clusterware Daemon
The primary Oracle Clusterware process.

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 Verification Utility (CVU)
A tool that verifies a wide range of RAC-specific components such as shared storage devices, networking configurations, system requirements, Oracle Clusterware, groups, and users.

CRSD
A UNIX-based 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.

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.

Enterprise Manager Configuration Assistant (EMCA)
A graphical user interface-based configuration assistant that you can use to configure Enterprise Manager features.

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 UNIX-based event manager daemon that starts the racgevt process to manage callouts.

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 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 gives the client a connection to an active instance that provides the requested database service.

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.

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.
**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. RAC provides for up to 10 Global Cache Service Processes. The number of LMSn varies depending on the amount of messaging traffic among nodes in the cluster.

**Global Cache Service (GCS) resources**
Global resources that coordinate access to data blocks in the buffer caches of multiple 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.ORACLE.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. GSD is installed on the nodes by default.

**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).

**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.
**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 a RAC database, each node within a cluster usually has one instance of the running Oracle software that references the database. When a database is started, Oracle allocates a memory area called the System Global Area (SGA) and starts one or more Oracle processes. This combination of the SGA and the Oracle 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 RAC guaranteeing that all cluster members are functional or active. IMR 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 UNIX-based subsystems for online disk storage management.

**Inter-Process 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 machine. 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 machine on which an instance resides.

Oracle Cluster File System (OCFS)
The Oracle proprietary cluster file system software that is available for Linux and Windows-based platforms.

Oracle Cluster Registry (OCR)
The 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.

Object Link Manager (OLM)
The Oracle interface that maps symbolic links to logical drives and displays them in the OLM graphical user interface.

OCSSD
A UNIX-based process that manages the Cluster Synchronization Services (CSS) daemon. Manages cluster node membership and runs as oracle user; failure of this process results in cluster restart.

Oracle Interface Configuration Tool (OIFCFG)
A command-line tool for both single-instance Oracle databases and 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 (OUI) also uses OIFCFG to identify and display available interfaces.

Oracle Notification Services (ONS)
A publish and subscribe service for communicating information about all FAN events.

OPROCD
A UNIX-based process monitor for a cluster. Note that this process will only appear on platforms that do not use vendor clusterware with Oracle Clusterware.

Oracle Clusterware
Oracle-provided clusterware that manages cluster database processing including node membership, group services, global resource management, and high availability functions.

Oracle Universal Installer (OUI)
A tool to install the Oracle Clusterware, the Oracle relational database software, and the Oracle Real Application Clusters 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 Real Application Clusters since 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 datafiles, control files, and archived redo logs. It is included with the Oracle server and does not require separate installation. You can invoke RMAN as a command line utility from the operating system (O/S) prompt or use the GUI-based Enterprise Manager Backup Manager.

**Runtime Connection Load Balancing**
Connection pool load balancing enables Oracle 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 three client environments to provide connection pool 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 machine and to move files from one machine to another. SSH uses strong authentication and secure communications over insecure channels.

**Server Control (SRVCTL) Utility**
Server Management (SRVM) comprises the components required to operate Oracle Enterprise Manager in Real Application Clusters. 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.

**services**
Entities that you can define in 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.

**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 a Real Application Clusters database, each node within the cluster has an instance referencing the database.

thread
Each Oracle instance has its own set of online redo log groups. These groups are called a thread of online redo. In non-Real Application Clusters environments, each database has only one thread that belongs to the instance accessing it. In Real Application Clusters environments, each instance has a separate thread, that is, each instance has its own online redo log. Each thread has its own current log member.

thread number
The number of the redo thread to be used by an instance as specified by the THREAD initialization parameter or the THREAD clause in the ALTER DATABASE ADD LOGFILE statement. You can use any available redo thread number but an instance cannot use the same thread number as another instance.

transparent application failover (TAF)
A runtime failover for high-availability environments, such as Real Application Clusters and Oracle Real Application Clusters 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 (OCI) library.

voting disk
A file that manages information about node membership.
adding nodes
quick-start, 10-3
Additional Real Application Clusters
documentation, 2-1
ADDM
see Automatic Database Diagnostic Monitor
administering
services, 6-17
administering instances
with Server Management, 9-1
administering services
DBCA, 6-19
Enterprise Manager, 6-19
PL/SQL, 6-19
SRVCTL, 6-19
administration
overview, 2-2
tools, 2-2
administrative tools
overview and concepts, 1-12
affinity
awareness, 8-5
alert administration
Enterprise Manager, 9-5
alert blackouts
Enterprise Manager, 9-5
alert log
managing, A-9
alert logs, A-10
alerts
performance monitoring, 12-8
ALTER SYSTEM ARCHIVE LOG CURRENT
statement, 9-5
ALTER SYSTEM ARCHIVE LOG statement, 9-5
INSTANCE option, 9-5
ALTER SYSTEM CHECKPOINT LOCAL
statement, 9-5
ALTER SYSTEM CHECKPOINT LOCAL
statement
background process
crsd, 1-4
evmd, 1-4
ocssd, 1-4
oprocd, 1-4
background processes
SMON, 5-3, 8-3
architecture
of RAC, 1-5
ARCHIVE LOG command, 9-6
archive logs
file format and destination, 7-4
file naming, 7-3
archiver process
monitor, 7-8
archiving mode
changing, 7-8
archiving redo log files
identified in control file, 4-2
log sequence number, 7-4
ASM
archiving scenario, 7-5
installation
Real Application Clusters, introduction to, 2-3
see Automatic Storage Management
asm
as SRVCTL noun, D-5
Automatic Database Diagnostic Monitor, 12-4, 12-5, 12-6
automatic segment-space management, 2-7, 11-3
Automatic Storage Management, 2-7, 4-1
see ASM
Automatic Storage Management (ASM), 1-5
Automatic Storage Management instance, 4-3
automatic undo management, 2-7
Automatic Workload Repository, 2-7, 6-24, 11-2, 12-3, 12-5
snapshots, 12-3
AVAILBLE instances
and services, 6-2
Average Active Sessions chart
Enterprise Manager, 12-14
AWR
see Automatic Workload Repository

B
background process
crsd, 1-4
evmd, 1-4
ocssd, 1-4
oprocd, 1-4
background thread trace files, A-10
BACKGROUND_DUMP_DEST parameter, A-10
backups
  server parameter file, 5-13
bandwidth
  interconnect, 12-1
block mode conversions
  statistics for, 12-2
blocks
  associated with instance, 8-3
buffer cache, 1-6
  instance recovery, 8-3
buffer sizes
  IPC, adjusting for Real Application Clusters, 12-1

C
Cache Fusion, 1-6
  and e-commerce applications, 14-1
  function, 2-6
  overview, 1-14
CATCLUST.SQL script
  using to create views for Real Application
  Clusters, 12-2
CLB_GOAL_LONG, 6-3
CLB_GOAL_SHORT, 6-3
client side
  load balancing, 6-3
cloning, 10-1
CLSD-1009 message
  resolving, 3-8
CLSD-1011 message
  resolving, 3-8
cluster
  definition of, 1-1
Cluster Database Home Page
  Enterprise Manager, 9-3
Cluster Database Instance page
  Enterprise Manager, 9-3
Cluster Database Instance Performance page
  Enterprise Manager, 12-16
Cluster Database Performance page
  Enterprise Manager, 12-17
cluster file system
  archiving parameter settings, 7-6
  archiving scenario, 7-5
  overview, 2-6
  restore, 8-2
  storage in Real Application Clusters, 4-1
Cluster Home page
  Enterprise Manager, 9-4
Cluster Host Load Average chart
  Enterprise Manager, 12-12
Cluster Interconnects page
  Enterprise Manager, 12-17
Cluster Managed Database Services Detail Page
  Enterprise Manager, 6-20
Cluster Managed Database Services Page
  Enterprise Manager, 6-19
Cluster Manager (CSS)
  log files, A-5
Cluster Ready Services, 2-8
Cluster Ready Services Control
  see CRSCTL
Cluster Synchronization Service (CSS), 1-2
Cluster Synchronization Services (CSS)
  purpose, 1-3
Cluster Verification Utility
  cluster integrity verifications, A-19
  connectivity verifications, A-16
  installation requirements, A-12
  installation verifications, A-18
  known issues, A-19
  node comparisons and verifications, A-18
  nodelist shortcuts, A-14
  Oracle Clusterware component
  verifications, A-18
  performing various tests with, A-15
  see CVU
  storage verifications, A-15
  system requirements verifications, A-15
  user and permissions verifications, A-17
  using, A-11
  using online help for, A-13
  verbose mode, A-13
CLUSTER_INTERCONNECTS
  parameter, 12-2
committed data
  instance failure, 8-3
communication protocols
  verifying settings for, 12-1
configuration problems
  reinitializing the OCR, 3-8
CONNECT command, 5-2, 9-6
CONNECT SYS
  example of, 5-5
  connecting
    to instances, 5-2
  connection load balancing
    concepts, 1-9
    long method, 6-3
    short method, 6-3
  connection pools
    and FAN, 6-10
  consistent blocks, 1-6
  control files
    MAXLOGHISTORY, 4-2
CREATE DATABASE statement
  MAXLOGHISTORY clause, 4-2
Create Services Page
  Enterprise Manager, 6-20
creating
  server parameter files, 5-7, 5-13
  crs_getperm, B-3
  CRS_HOME, 1-9
  crs_profile, B-3
  crs_register, B-6
  crs_relocate, B-8
  crs_setperm, B-10
  crs_start, B-13
Index-3

crs_stat, B-10
crs_stop, B-14
crs_unregister, B-15
CRSCTL
  overview and concepts, 1-13
crsd, 1-4
  purpose, 1-4
current blocks, 1-6
CVU
  concepts, 1-10
  overview and concepts, 1-12

data dictionary
  querying views, 12-2
Data Guard, 2-9
data security
  wallet, 14-2
data warehouse
  deploying applications for in Real Application Clusters, 14-2
database
  as SRVCTL noun, D-5
  creation, 1-10
  number of archived log files, 4-2
  quiescing, 9-6
Database Configuration Assistant
  Database Storage page, 10-19
  Instance Management page, 10-19
  List of Cluster Database Instances page, 10-19
  List of Cluster Databases page, 10-19
  Operations page, 10-19
  see DBCA
  Summary dialog, 10-19
  Welcome page, 10-19
Database Configuration Assistant (DBCA)
  creating views for Real Application Clusters, 12-2
Database Locks page
  Enterprise Manager, 12-16
Database Storage page, 10-19
Database Throughput chart
  Enterprise Manager, 12-15
Databases Overview page
  Enterprise Manager, 9-4
Databases Summary page
  Enterprise Manager, 9-2
DBCA, 1-10
  adding and deleting instances in silent mode, 10-8
  service administration, 6-21
DBMS_SERVICE, 11-2
degree of parallelism (DOP), 14-2
deleting nodes
  quick-start, 10-3
design
  Real Application Clusters environments, 1-12
DISCONNECT command, 5-2
disconnecting from instances, 5-2
Distributed Transaction Processing, 11-4
distributed transactions, 6-15, 11-4
documentation
  Real Application Clusters, 2-1
DTP, 11-4
DTP/XA transactions, 6-16
dynamic performance views
  creating, 12-2
  for performance monitoring, 12-2
dynamic resource allocation
  overview, 1-14
E
e-commerce
  applications in Real Application Clusters, 14-1
Enterprise Manager, 5-1, 9-1
  administering services, 6-19
  alert administration, 9-5
  alert blackouts, 9-5
  Average Active Sessions chart, 12-14
  Cluster Database Home Page, 9-3
  Cluster Database Instance page, 9-3, 12-17
  Cluster Database Instance Performance page, 12-16
  Cluster Home page, 9-4
  Cluster Host Load Average chart, 12-12
  Cluster Interconnects page, 12-17
  Database Locks page, 12-16
  Database Throughput chart, 12-15
  Databases Overview page, 9-4
  Databases Summary page, 9-2
  Global Cache Block Access Latency chart, 12-12
  Global Cache Coherency chart, 12-14
  Instance Activity page, 12-15
  Instance page, 9-3
  job administration, 9-4
  overview, 2-4
  overview and concepts, 1-12
  performance monitoring, 12-7
  services pages, accessing, 6-20
  Top Consumers page, 12-15
  Top Segments page, 12-16
  Top Sessions page, 12-15
Enterprise Manager Database Control
  overview, 2-4
Enterprise Manager Grid Control
  instance discovery, 9-2
  node discovery, 9-2
  overview, 2-5
error messages
  for management tools, E-1
  PRKA, E-2
  PRKC, E-4
  PRKD, E-14
  PRKE, E-14
  PRKH, E-14
  PRKI, E-16
  PRKN, E-17
  PRKO, E-18
  PRKP, E-21
PRKR, E-29
PRKS, E-34
PRKU, E-38
PRKV, E-38
Event Management (EVM), 1-2
    purpose, 1-3
Event Manager (EVM)
    log files, A-5
events
    load balancing advisory, 6-5
evmd, 1-4
    purpose, 1-4

F
failover
    and Real Application Clusters, 2-8
failure
    instance, 8-3
    multiple node, 8-4
    node, 8-3
FAN
    callouts, 6-7
    clients integrated for, 6-10
Fast Application Notification
    see FAN
Fast Application Notification (FAN), 6-4
Fast Connection Failover (FCF)
    and services, 6-2
features
    taking advantage of, 2-5
features, new, xxiii
files
    archiving redo log, 7-4
    control file, 4-2
    redo log, 7-4

G
Global Cache Block Access Latency chart
    Enterprise Manager, 12-12
Global Cache Coherency chart
    Enterprise Manager, 12-14
Global Cache Service (GCS), 1-6
Global Cache Service statistics, 12-3, 12-4
GLOBAL clause
    forcing a checkpoint, 9-5
Global Enqueue Service (GES), 1-6
Global Enqueue Service statistics, 12-3
Global Resource Directory (GRD), 1-6
global service name, 11-1
GV$ACTIVE_SERVICES, 11-2
GV$SVC_METRIC, 11-2

H
H.A.R.D.
    implementing for the OCR, 3-9
hash partitioning
    with Real Application Clusters, 1-12
high availability
    and Oracle Clusterware, 1-2
    and Real Application Clusters, 2-8
high availability framework, 6-1
    concepts, 1-8
HOST command, 9-6

initdb_name.ora file
    BACKGROUND_DUMP_DEST parameter, A-10
    USER_DUMP_DEST parameter, A-10
initialization parameters
    cluster database issues regarding, 5-10
    CLUSTER_INTERCONNECTIONS, 12-2
    RECOVERY_PARALLELISM, 8-7
    that must be identical on all instances, 5-9
    that must be unique on all instances, 5-9
INST_ID column, 12-2
installation
    introduction, 1-9
    Oracle Clusterware, 1-9
    Real Application Clusters, 1-10
instance
    as SRVCTL noun, D-5
Instance Activity page
    Enterprise Manager, 12-15
instance addition
    quick-start, 10-3
instance deletion
    quick-start, 10-3
instance discovery
    Enterprise Manager Grid Control, 9-2
Instance Management page, 10-19
INSTANCE option, 9-5
Instance page
    Enterprise Manager, 9-3
INSTANCE_NUMBER parameter, 5-9
instances
    failure, 8-4
    failures, recovery from, 8-3
    maximum number for RAC, 1-5
    memory structures, 1-6
    recovery, 5-3, 8-3
    recovery, abnormal shutdown, 5-3
    recovery, multiple failures, 8-4
    Server Management, 9-1
    shutting down, 5-3
    verifying, 9-6
interconnect
    and performance, 12-1
    and the RAC architecture, 1-5
    bandwidth, 12-1
    protocols for Real Application Clusters, 12-1
    verifying settings for, 12-1
IP address
    installation requirements, 1-10
IPCs
    buffer sizes, adjusting, 12-1
J
java-based tools
   - tracing, A-11
JDBC
   - and services, 6-2
   - clients, 6-12
job administration
   - Enterprise Manager, 9-4

L
LCK0 Instance Enqueue Process, 1-7
List of Cluster Database Instances page, 10-19
List of Cluster Databases page, 10-19
LMD Global Enqueue Service Daemon, 1-7
LMOM Global Enqueue Service Monitor, 1-7
LMS Global Cache Service Process, 1-7
load balancing
   - by services, 1-5
   - server side, 6-3
Load Balancing Advisory, 6-17
load balancing advisory, 6-1, 6-8
   - concepts, 1-9
   - configuring, 6-8
   - events, 6-5
   - FAN events, 6-9
local archiving scenario
   - RMAN, 7-6
Local Area Network (LAN), 1-5
LOCAL clause
   - forcing a checkpoint, 9-5
local file system
   - archiving parameter settings, 7-7
   - restore, 8-2
log files
   - for CSS, A-5
   - for EVM, A-5
   - for OCR, A-5
   - for Oracle Clusterware, A-5
log sequence numbers, 7-4
log switches, 4-2
LOG_ARCHIVE_FORMAT parameter, 7-4

M
mass deployment
   - cloning, 10-1
MAXLOGHISTORY option, 4-2
media failures
   - recovery from, 8-6
memorystructures
   - in RAC, 1-6
message request counters, 12-2
messages
   - errors for management tools, E-1
metrics
   - performance monitoring, 12-8
mirroring
Oracle Cluster Registry, 3-3
modified data
   - instance recovery, 8-3
monitoring
   - archiver process, 7-8
   - overview and concepts, 1-13
   - statistics for Real Application Clusters, 12-2
multiple node failures, 8-4
multiple nodes
   - starting from one node, 5-4
multiplexed redo log files, 4-2
multiplexing
   - OCR, 1-7

N
Net Configuration Assistant
   see NetCA
NetCA, 1-10
Network Attached Storage (NAS), 1-7
network file system, 1-5
network interfaces
   - defining with OIFCFG, 9-7
new features, xxiii
node
   - failure and VIPs, 1-6
node addition
   - quick-start, 10-3
node deletion
   - quick-start, 10-3
node discovery
   - Enterprise Manager Grid Control, 9-2
nodeapps
   as SRVCTL noun, D-5
nodes
   - affinity awareness, 8-5
   - failure of, 8-3

O
objects
   - creation of and effect on performance, 11-3
OCI
   - and services, 6-2
OCR
   - installation, 1-10
   - redundancy, 1-10
OCR Configuration Tool, C-1
   see OCRCONFIG
OCRCHECK
   - diagnosing OCR problems with, 3-7
   - using, 3-7
OCRCONFIG
   - overview and concepts, 1-13
   - ocrconfig command, C-1
   - ocrconfig -export, C-2
   - ocrconfig -option, C-1
OCRDUMP
   - diagnosing OCR problems with, 3-7
   - using, 3-7
using to view OCR content, A-6
ocrdump command, A-7
ocssd, 1-4
    purpose, 1-4
ODP.NET
    and FAN events, 6-13
    and FCF, 6-13
    and services, 6-2
    load balancing advisory events, 6-14
OIFCFG, 9-7
    commands, 9-8, 9-9
    defining network interfaces, 9-7
    if_name element, 9-8
    if_type element, 9-8
    nodename element, 9-8
    overview and concepts, 1-13
    subnet element, 9-8
    syntax, 9-8
online recovery, 8-3
online transaction processing
    applications in Real Application Clusters, 14-1
operating system-specific Oracle documentation
    archived redo log name, 7-4
Operations page, 10-19
oprocd, 1-4
    purpose, 1-4
Oracle Cluster File System (OCFS), 1-5
Oracle Cluster Registry
    adding, 3-3
    contents, 3-2
    data loss protection mechanism, overriding, 3-7
    diagnosing problems with OCRDUMP and
    OCRCHECK, 3-7
    downgrading, 3-9
    exporting, 3-8
    implementing H.A.R.D., 3-9
    importing, 3-8
    importing on UNIX-based systems, 3-8
    importing on Windows-based systems, 3-9
    managing backups, 3-5
    recovering using OCR backup files, 3-5
    removing, 3-3, 3-5
    repairing, 3-3, 3-4
    replacing, 3-3, 3-4
    restoring on UNIX-based systems, 3-6
    restoring on Windows-based systems, 3-6
    restoring using automatically generated OCR
    backups, 3-5
    see OCR
    troubleshooting, A-6
    upgrading, 3-9
Oracle Cluster Registry (OCR), 1-7
    log files, A-5
    managing, 3-2
    purpose, 1-2
Oracle Clusterware
    architecture, 1-2
    cloning, 10-1
    command examples, 13-7
    commands, B-2
    component processing, 1-2
    enabling and disabling startup using crsctl
    commands, A-3
    event manager alerts, A-4
    high availability framework, 1-8
    installation overview, 1-9
    introduction, 2-4, 13-1
    introduction and concepts of, 1-1
    log files and their locations, A-5
    overview, 13-3
    startup and shutdown using crsctl
    commands, A-3
    UNIX processes, 1-4
    working with Oracle Clusterware
    commands, 13-11
Oracle Clusterware home
    adding to a new node, 10-3
    adding, silent mode, 10-4
    deleting in silent mode, 10-8
    deleting manually, 10-7
Oracle Clusterware management
    overview and concepts, 1-13
Oracle home
    adding in silent mode, 10-5
    adding manually, 10-4
    cloning, 10-1
    deleting, 10-5
    deleting manually, 10-5, 10-6
Oracle Interface Configuration Tool
    see OIFCFG
Oracle Interface Configuration tool
    see OIFCFG
Oracle Net Services
    and services, 6-2
Oracle Notification Service (ONS)
    purpose, 1-3
Oracle Real Application Clusters
    overview of administration, 1-1, 2-1
Oracle Universal Installer
    see OUI
ORACLE_HOME, 1-10
ORACLE_SID parameter, 5-9
OracleCRService
    purpose, 1-4
OracleCSService
    purpose, 1-5
OracleEVMService
    purpose, 1-5
OraFenceService
    purpose, 1-5
OU
    Real Application Clusters installation, 1-10
OUI
    database installation, 1-10
    Oracle Clusterware installation, 1-9
outages
    planned, 6-6
    unplanned, 6-6
parallel instance groups, 14-2
parallel recovery, 8-7
PARALLEL_MAX_SERVERS parameter, 8-7
parallelism
  in Real Application Clusters, 14-2
  parallel-aware query optimization, 14-2
parameter file
  overview, 5-7
parameters
  that must be identical on all instances, 5-9
  that must be unique on all instances, 5-9
performance
  primary components affecting, 12-1
performance evaluation
  overview and concepts, 1-14
performance monitoring
  alerts, 12-8
  Enterprise Manager, 12-7
  metrics, 12-8
  thresholds, 12-9
PL/SQL
  administering services, 6-22
PREFERRED
  instances and services, 6-2
private interconnect
  and Oracle Clusterware, 1-2
PRKA messages, E-2
PRKC messages, E-4
PRKD messages, E-14
PRKE messages, E-14
PRKH messages, E-14
PRKI messages, E-16
PRKN messages, E-17
PRKO messages, E-18
PRKP messages, E-21
PRKR messages, E-29
PRKS messages, E-34
PRKU messages, E-38
PRKV messages, E-38
Process Monitor Daemon (OPROCD)
  purpose, 1-3
quiesce database
  in Real Application Clusters, 9-6
RAC
  cloning, 10-1
  software components, 1-6
RACG
  purpose, 1-3
  raw devices, 1-5
Real Application Clusters
  and e-commerce, 14-1
  installation overview, 1-10
RECOVER command, 9-6
recovery
  after SHUTDOWN ABORT, 5-3
  from multiple node failure, 8-4
  from single-node failure, 8-3
  instance, 5-3
  media failures, 8-6
  online, 8-3
  parallel, 8-7
  PARALLEL_MAX_SERVERS parameter, 8-7
  Recovery Manager
    see RMAN
  RECOVERY_PARALLELISM parameter, 8-7
  redo log files
    identified in control file, 4-2
    instance recovery, 8-3
    log sequence number, 7-4
    using, 4-2
  redo log groups, 4-2
  redo logs
    format and destination specifications, 7-4
  Resource Manager
    and services, 6-2
    resource manager, 11-2
    resource profiles
      attributes, 13-4
      resources
        releasing, 8-3
  restore scenarios
    RMAN, 8-1
  restore scheme
    cluster file system, 8-2
    local file system, 8-2
  RMAN
    in Real Application Clusters, 2-9
    local archiving scenario, 7-6
    overview
    restore scenarios, 8-1
    rolling back
      instance recovery, 8-3
      runtime connection load balancing, 6-2
scalability
  adding nodes and instances, 10-9
  adding nodes and instances, quick-start
    format, 10-3
security
  wallet, 14-2
sequence numbers
  with Real Application Clusters, 1-12
sequences
  log sequence number, 7-4
Server Control
  see SRVCTL
Server Control Utility (SRVCTL), 1-7
Server Management
  administration of instances, 9-1
server parameter file, 2-7
  backing up, 5-13
Index-8

setting values in, 5-7
server parameter files
  creating, 5-7, 5-13
service
  as SRVCTL noun, D-5
  DTP property, 6-15
  levels, 6-26
SERVICE TIME, 6-9
SERVICE_NAMES
  parameter, 6-22
services, 2-7, 6-1, 11-1
  administering, 6-17
  administering with DBCA, 6-19
  administering with Enterprise Manager, 6-19
  administering with PL/SQL, 6-19
  administering with SRVCTL, 6-19
  and distributed transactions, 6-15
  basic concepts about, 6-2
  concepts, 1-9
  configurations for, 11-1
  default, 6-3
  introduction to, 1-5
  names, limitations for, 6-21
  using, 6-2
sessions
  multiple, 5-2
  setting instances, 5-2
  shared everything, 1-5
SHOW INSTANCE command, 9-6
SHOW PARAMETER command, 9-6
SHOW PARAMETERS command, 9-6
SHOW SGA command, 9-6
SHUTDOWN ABORT command, 5-3
SHUTDOWN command, 9-6
  ABORT option, 5-3
SHUTDOWN TRANSACTIONAL, 5-3
shutting down
  instances, 5-3
  shutting down an instance
    abnormal shutdown, 5-3
  shutting down instances, 5-3
sidalrt.log file, A-10
siddbwr.trc file, A-10
sidsmon.trc file, A-10
single system image, 1-6
single-instance Oracle databases
  how they differ from RAC, 1-5
singleton services, 6-16
SMON process
  instance recovery, 8-3, 8-4
  recovery after SHUTDOWN ABORT, 5-3
snapshot control file, 7-1
SPFILE
  restore with Enterprise Manager, 8-2
  restore with RMAN, 8-2
split brain syndrome, 1-2
SQL statements
  instance-specific, 9-5
SQL*Plus, 5-2, 9-5
SRVCTL
  add asm command, D-8
  add database command, D-5
  add instance command, D-6
  add nodeapps command, D-7
  add service command, D-7
  add, usage description, D-5
  cluster database configuration tasks, D-2
  cluster database tasks, D-3
  concurrent commands, D-2
  config asm command, D-10
  config database command, D-8
  config nodeapps command, D-9
  config service command, D-9
  config, usage description, D-8
  disable asm command, D-14
  disable database command, D-13
  disable instance command, D-13
  disable service command, 6-24, D-13
  disable, usage description, D-12
  enable asm command, D-12
  enable database command, D-11
  enable instance command, D-11
  enable service command, 6-24, D-12
  enable, usage description, D-10
  getenv database command, D-28
  getenv instance command, D-28
  getenv nodeapps command, D-29
  getenv service command, D-28
  getenv, usage description, D-27
  modify database command, D-21
  modify instance command, D-21
  modify nodeapps command, D-24
  modify service command, D-23
  modify, usage description, D-20
  node-level tasks, D-3
overview, 5-3
overview and concepts, 1-13
relocate service command, D-25
relocate, usage description, D-24
remove asm command, D-35
remove database command, D-33
remove instance command, D-34
remove nodeapps command, D-34
remove service command, D-34
remove, usage description, D-33
setenv database command, D-30
setenv instance command, D-30
setenv nodeapps command, D-31
setenv service command, D-30
setenv, usage description, D-29
start asm command, D-17
start database command, 5-6, D-15
start instance command, 5-5, D-15
start listener command, D-17
start nodeapps command, D-16
start service command, 6-24, D-16
start, usage description, D-14
status asm command, D-27
status database command, D-25
status instance command, D-26
status nodeapps command, D-27
status service command, D-26
status, usage description, D-25
stop asm command, D-20
stop database command, 5-6, D-18
stop instance command, 5-5, D-18
stop listener command, D-20
stop nodeapps command, D-20
stop service command, D-19
stop, usage description, D-17
unsetenv database command, D-31
unsetenv instance command, D-31
unsetenv nodeapps command, D-32
unsetenv service command, D-32
unsetenv, usage description, D-29
Standard Edition
ASM
standby databases, 2-9
STARTUP command, 9-6
statistics
  contents of, 12-2
  monitoring for Real Application Clusters, 12-2
Statspack, 12-3
storage, 2-7
  basic RAC requirements for, 1-5
  Real Application Clusters, introduction to, 2-3
  storage
    cluster file system, 4-1
Summary dialog, 10-19
SYSAUX tablespace, 11-3
SYSDBA
  privilege for connecting, 5-2
SYSOPER privilege
  for connecting, 5-2
system change, 7-4
System Global Area (SGA), 1-6
  size requirements, 1-6
T
TAF
  and services, 6-2
TCP/IP, 1-6
THREAD parameter, 5-13
thresholds
  performance monitoring, 12-9
  services, 6-26
THROUGHPUT, 6-9
timed statistics, 12-2
timeout
  messages, avoiding, 1-6
Top Consumers page
  Enterprise Manager, 12-15
Top Segments page
  Enterprise Manager, 12-16
Top Sessions page
  Enterprise Manager, 12-15
trace files, A-10
  background thread trace files, A-10
  managing, A-9
  sidalrt.log, A-10
  Siddbwr.trc file, A-10
  Sidssmon.trc file, A-10
tracing
  enabling for RAC high availability, A-8
  java-based tools, A-11
transactions
  instance failure, 8-3
  rolling back, 8-3
  waiting for recovery, 8-3
Transparent Database Encryption, 14-2
troubleshooting, A-9
tuning
  overview, 1-14
U
UNDO_TABLESPACE parameter, 5-9
unified log directory structure, A-5
Unified Scheduler, 11-2
User Datagram Protocol (UDP)
  and the private interconnect, 1-2
user trace files, A-10
USER_DUMP_DEST parameter, A-10
V
V$ACTIVE_INSTANCES, 9-6
V$ACTIVE_SERVICES, 11-2
V$SERVICEMETRIC_HISTORY, 11-2
V$SESSION, 11-2
V$SQL, 11-2
vendor clusterware, 1-1
verification
  datafiles, online files, 4-2
views
  creating for Real Application Clusters, 12-2
  for performance monitoring, 12-2
VIP addresses, 1-6
  configuring, 1-6
virtual host name, 1-6
virtual IP address, 1-10
volume manager
  alternate storage method for RAC, 1-5
Voting Disk, 1-7
voting disk
  administering, 3-1
  backing up, 3-1
  dynamically adding, 3-2
  installation, 1-10
  purpose, 1-2
  recovering, 3-2
  redundancy, 1-10
voting disks
  multiple, 1-7
  redundancy, 1-7
W
wait events, 12-3

Index-9
wallet
  data security, 14-2
Welcome page, 10-19
Workload Management
  concepts, 1-8
workload management, 11-1
  basic concepts about, 6-2
  manual rebalancing, 11-2
workloads
  and services, 6-2

XA transactions, 11-4