Oracle® Real Application Clusters

Real Application Clusters Administration and Deployment Guide

26ai
G43034-01
October 2025

Oracle Real Application Clusters Real Application Clusters Administration and Deployment Guide, 26ai

G43034-01

Copyright © 1999, 2025, Oracle and/or its affiliates.

Primary Author: Subhash Chandra

Contributing Authors: Janet Stern, Eric Belden, Mark Bauer, Troy Anthony, Carol Colrain, Anil Nair

Contributors: Ram Avudaiappan, Prasad Bagal, Anand Beldalker, Gajanan Bhat, David Brower, George Claborn, Maria Colgan, Jonathan Creighton, Rajesh Dasari, Mark Dilman, Richard Frank, GP Prabhaker Gongloor, Wei Hu, Yong Hu, Dominique Jeunot, Sameer Joshi, Raj K. Kammend, Ankita Khandelwal, Sana Karam, Roland Knapp, Karen Li, Barb Lundhild, Venkat Maddali, Bill Manry, John McHugh, Saar Maoz, Markus Michalewicz, Philip Newlan, Michael Nowak, Muthu Olagappan, Bharat Paliwal, Hanlin Qian, Hairong Qin, Mark Ramacher, Sampath Ravindhran, Kevin Reardon, Kathy Rich, Dipak Saggi, Daniel Semler, Ara Shakian, Kesavan Srinivasan, Leo Tominna, Peter Wahl, Tak Wang, Richard Wessman, Douglas Williams, Michael Zoll, Aseem Kashyap

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

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

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

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

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface

Audience i
Set Up Java Access Bridge to Implement Java Accessibility i
Command Syntax ii
Conventions ii

1 Introduction to Oracle RAC

Overview of Oracle RAC 1
Overview of Oracle Multitenant with Oracle RAC 4
Overview of Installing Oracle RAC 4
- Understanding Compatibility in Oracle RAC Environments 5
- Oracle RAC Database Installation 5
- Oracle RAC Database Creation 6
- Overview of Extending Oracle RAC Clusters 7

Overview of Oracle Real Application Clusters One Node 8
Overview of Oracle Clusterware for Oracle RAC 9
- Guidelines for Using Oracle Clusterware 9
- Overview of Temporary Tablespaces 10
  - Parallel Execution Support for Cursor-Duration Temporary Tablespaces 11
  - Temporary Tablespace Organization 11
  - Temporary Tablespace Hierarchy 12
  - Temporary Tablespace Features 12
  - Metadata Management of Temporary Files 13
  - DDL Support for Temporary Tablespaces 13
  - Temporary Tablespace for Users 14
  - Atomicity Requirement for Commands 15
  - Temporary Tablespace and Dictionary Views 15

Overview of Oracle RAC Architecture and Processing 16
- Understanding Cluster-Aware Storage Solutions 17
- Oracle RAC and Network Connectivity 17
- Overview of Using Dynamic Database Services to Connect to Oracle AI Database 18
- Overview of Virtual IP Addresses 18
- Restricted Service Registration in Oracle RAC 19

Copyright © 1999, 2025, Oracle and/or its affiliates.

About Oracle RAC Software Components 20
About Oracle RAC Background Processes 20
Overview of Automatic Workload Management with Dynamic Database Services 22
Overview of Blocker Resolver 24
Overview of Database In-Memory and Oracle RAC 25
Overview of Managing Oracle RAC Environments 25
About Designing and Deploying Oracle RAC Environments 26
About Administrative Tools for Oracle RAC Environments 26
About Monitoring Oracle RAC Environments 28
About Evaluating Performance in Oracle RAC Environments 29

2 Administering Storage in Oracle RAC

About Oracle ASM 1
Overview of Storage Management for Oracle RAC 2
Data File Access in Oracle RAC 3
NFS Server for Storage 3
Redo Log File Storage in Oracle RAC 3
Automatic Undo Management in Oracle RAC 4
Oracle Automatic Storage Management with Oracle RAC 4
Storage Management in Oracle RAC 5
Modifying Disk Group Configurations for Oracle ASM 6
Oracle ASM Disk Group Management 6
Configuring Preferred Mirror Read Disks in Extended Distance Clusters 7
Converting Nonclustered Oracle ASM to Clustered Oracle ASM 7
Administering Oracle ASM Instances with SRVCTL in Oracle RAC 7

3 Administering Database Instances and Cluster Databases

Overview of Oracle RAC Database Administration 2
Required Privileges for Oracle RAC Database Administration 2
Oracle RAC Database Deployment Models 3
Tools for Administering Oracle RAC 4
Administering Oracle RAC with SRVCTL 4
Administering Oracle RAC with Oracle Enterprise Manager 5
Administering Oracle RAC with SQL*Plus 5
How SQL*Plus Commands Affect Instances 6
Starting and Stopping Instances and Oracle RAC Databases 7
Starting One or More Instances and Oracle RAC Databases Using SRVCTL 8
Stopping One or More Instances and Oracle RAC Databases Using SRVCTL 9
Stopping All Databases and Instances Using CRSCTL 11
Starting and Stopping Individual Instances Using SQL*Plus 11

Copyright © 1999, 2025, Oracle and/or its affiliates.

Starting and Stopping PDBs in Oracle RAC 12
Local Rolling Database Maintenance 14
About Local Rolling Database Maintenance 15
Requirements for Using Local Rolling Maintenance 15
Patching Oracle RAC Database in Local Rolling Mode 16
How to Recover from a Failed Transfer in Local Rolling Mode 17
Pluggable Database Rank 18
Pluggable Database Placement 19
Example of Creating a Pluggable Database with Cardinality and Rank 19
Reducing Downtime During Database and Instance Outages 20
Oracle RAC High Availability Best Practices 21
Oracle RAC Two-Stage Rolling Updates 21
Smooth Reconfiguration of Oracle RAC Instances 22
Ordered Sequence Optimizations in Oracle RAC 22
Verifying That Instances are Running 23
Using SRVCTL to Verify That Instances are Running 23
Using SQL*Plus to Verify That Instances are Running 24
Terminating Sessions On a Specific Cluster Instance 24
Overview of Initialization Parameter Files in Oracle RAC 27
About Creating an SPFILE for Oracle RAC 27
Setting SPFILE Parameter Values for Oracle RAC 28
Parameter File Search Order in Oracle RAC 29
Backing Up the Server Parameter File 30
Initialization Parameter Use in Oracle RAC 30
Initialization Parameters Specific to Oracle RAC 31
Parameters That Must Have Identical Settings on All Instances 33
Parameters That Should Have Identical Settings on All Instances 34
Quiescing Oracle RAC Databases 35
Administering Multiple Cluster Interconnects on Linux and UNIX Platforms 36
Use Cases for Setting the CLUSTER_INTERCONNECTS Parameter 36
Customizing How Oracle Clusterware Manages Oracle RAC Databases 38
Advanced Oracle Enterprise Manager Administration 39
Using Oracle Enterprise Manager Cloud Control to Discover Nodes and Instances 40
Other Oracle Enterprise Manager Capabilities 40
Administering Jobs and Alerts in Oracle RAC 41
Administering Jobs in Oracle RAC 41
Administering Alerts in Oracle RAC with Oracle Enterprise Manager 42
Using Defined Suspensions in Oracle Enterprise Manager 42

4 Administering Oracle RAC One Node

Creating an Oracle RAC One Node Database 1

Copyright © 1999, 2025, Oracle and/or its affiliates.

Converting Databases 2
Converting a Database from Oracle RAC to Oracle RAC One Node 3
Converting a Database from Oracle RAC One Node to Oracle RAC 3
Online Database Relocation 4

5 Workload Management with Dynamic Database Services

Connection Load-Balancing 2
About Connection Load-Balancing 2
Server-Side Load Balancing 3
Generic Database Clients 3
Client-Side Connection Configuration for Older Clients 4
About Client-Side Connection Configuration for Older Clients 4
JDBC Thin Clients 5
OCI Clients 5
Client-Side Load Balancing 5
Smart Connection Rebalance 6
Load Balancing Advisory 7
Overview of the Load Balancing Advisory 7
Configuring Your Environment to Use the Load Balancing Advisory 8
Load Balancing Advisory FAN Events 9
Monitoring Load Balancing Advisory FAN Events 10
Enabling Clients for Oracle RAC 10
Overview of Oracle Integrated Clients and FAN 11
Enabling JDBC-Thin Clients for Fast Connection Failover 12
About Fast Connection Failover and JDBC-Thin Clients 12
Oracle Notification Service for JDBC-Thin Clients 13
Configuring FCF for JDBC/OCI and JDBC-Thin Driver Clients 13
Enabling JDBC Clients for Run-time Connection Load Balancing 14
Configuring JDBC-Thin Clients for Application Continuity for Java 15
Configuring JDBC-Thin Clients for Transaction Guard 16
Enabling OCI Clients for Fast Connection Failover 16
Enabling OCI Clients for Run-time Connection Load Balancing 17
Configuring OCI Clients to use Transaction Guard 18
Enabling ODP.NET Clients to Receive FAN High Availability Events 18
Enabling ODP.NET Clients to Receive FAN Load Balancing Advisory Events 19
Configuring ODP.NET Clients to use Transaction Guard 20
Distributed Transaction Processing in Oracle RAC 20
Overview of XA Transactions and Oracle RAC 21
Using Global Transactions and XA Affinity for XA Transactions 22
Using Services with XA Transactions on Oracle RAC 23
Configuring Services for XA Applications 23

Copyright © 1999, 2025, Oracle and/or its affiliates.

Relocating Services in Administrator-Managed Databases 24
Oracle RAC Sharding 24
Automatic Workload Repository 25
Measuring Performance by Service Using the Automatic Workload Repository 25
Automatic Workload Repository Service Thresholds and Alerts 27
About Automatic Workload Repository Service Thresholds and Alerts 27
Example of Services and Thresholds Alerts 28
Enable Service, Module, and Action Monitoring 28
Using Oracle Services 29
Service Deployment Options 29
Service Usage in an Oracle RAC Database 30
Oracle Clusterware Resources for a Service 30
Database Resource Manager Consumer Group Mappings for Services 30
Performance Monitoring by Service with AWR 31
Parallel Operations and Services 31
Oracle GoldenGate and Oracle RAC 31
Service Characteristics 31
Service Name 32
Service Edition 32
Service Management Policy 33
Database Role for a Service 33
Instance Preference 34
Service Co-location 34
Load Balancing Advisory Goal for Run-time Connection Load Balancing 35
Connection Load Balancing Goal 35
Distributed Transaction Processing 35
Default Service Connections 36
Restricted Service Registration 36
Administering Services 37
Overview of Service Administration 38
Administering Services with Oracle Enterprise Manager 39
Administering Services with SRVCTL 40
Creating Services with SRVCTL 41
Creating Services for Application Continuity and Transaction Guard 41
Starting and Stopping Services with SRVCTL 42
Enabling and Disabling Services with SRVCTL 43
Relocating Services with SRVCTL 43
Obtaining the Status of Services with SRVCTL 43
Obtaining the Configuration of Services with SRVCTL 44
Global Services 44
Service-Oriented Buffer Cache Access 45

Copyright © 1999, 2025, Oracle and/or its affiliates.

Connecting to a Service: An Example 45

6 Ensuring Application Continuity

Understanding Application Continuity 2
About Application Continuity 2
Key Concepts for Application Continuity 3
How Application Continuity Works for Applications 5
Support for Oracle Application Continuity and Transparent Application Continuity 7
Restrictions and Other Considerations for Application Continuity 9
Application Continuity for Various Applications 11
Applications That Use Containers with Request Boundaries 11
Applications that are Database Agnostic 12
Transparent Application Continuity 12
About Transparent Application Continuity 12
Deciding on Transparent Application Continuity 14
Using Transparent Application Continuity in Oracle Cloud Environments 15
Configuring Application Continuity 16
Overview of Application Continuity Configuration Tasks 17
Configuring Connections for High Availability and Application Continuity 20
Configuring Oracle AI Database for Application Continuity 21
Establishing the Initial State Before Application Continuity Replays 22
Checking Initial States for Application Continuity 23
FAILOVER_RESTORE 23
States Restored with FAILOVER_RESTORE 24
Full FAILOVER_RESTORE with Database Templates 24
FAILOVER_RESTORE with Database Templates 25
Configuring a Keystore using WALLET_ROOT for FAILOVER_RESTORE 26
Configuring a Keystore using SQLNET.ORA for FAILOVER_RESTORE 29
FAILOVER_RESTORE = NONE and No Callback 31
Connection Labeling 31
Connection Initialization Callback 32
Potential Side Effects of Application Continuity 32
Administering Application Continuity Operation and Usage 36
Using Application Continuity for Planned Maintenance 37
Administering Restoring Original Function Results 37
Restoring Original Oracle Function Values and Application Continuity 38
Checking Your Keep Permissions 40
Granting and Revoking Keep Permissions for Restoring Original Oracle Function Values 41
Granting Permission to Keep Restoring Original Oracle Function Values for Oracle Sequences 41
Rules for Grants on Restoring Original Oracle Function Values 42

Copyright © 1999, 2025, Oracle and/or its affiliates.

Protection-Level Statistics 42
Session State Consistency 43
About Session State Consistency 43
Auto Session State Consistency 44
Dynamic Session State Consistency 45
Application Continuity Statistics 46
Application Continuity Protection Check 47
About Application Continuity Protection Check 47
Enabling and Disabling Application Continuity Protection Check 48
Generating an Application Continuity Protection Check Report 49
Filtering Statistics and Events 52
Cleaning Up ACCHK Collected Information 53
Delaying the Reconnection in Application Continuity 53
Understanding How to Wait for the Service to be Ready for Application Continuity 53
Creating Services on Oracle RAC with Application Continuity 54
Modifying Services on Single-instance Databases to use Application Continuity 54
Running Without Application Continuity 55
Disabling Replay in Application Continuity 56
Understanding Enabling and Disabling Replay in Application Continuity 56
Application Calls Autonomous Transactions, External PL/SQL, or Java Actions that Should Not Be Repeated 57
Application Synchronizes Independent Sessions 57
Application Uses Time at the Middle Tier in the Processing Logic 58
Application Assumes that ROWIds Do Not Change 58
Application Assumes that Location Values Do Not Change 58
Terminating or Disconnecting a Session Without Replay 59
Fast Application Notification (FAN) 59
Overview of Fast Application Notification (FAN) 60
The Importance of Using Fast Application Notification 61
How FAN is Used with Oracle AI Database and Applications 61
Requirements for Using FAN 63
FAN Callouts 63
Fast Application Notification High Availability Events 64
Subscription to High Availability Events 67
Using Fast Application Notification Callouts 67
Configure for Unplanned Outages 68
Managing Planned Maintenance 69
About Planned Maintenance Management 69
Planned Maintenance Without User Interruption 70
Managing a Group of Services for Maintenance 72
Stopping a Group of Services Example 73
Starting Services 73

Copyright © 1999, 2025, Oracle and/or its affiliates.

Pluggable Database-Level Operations 74
Relocating Services 74
Stopping Services 75
Server Draining Ahead of Planned Maintenance 76
Planned Failover with Application Continuity 83
Transaction Guard for Improving Client Failover 83
About Transaction Guard 84
Database Native Transaction Guard 85
Transaction Guard Support During Major Database Version Upgrades 85
Transaction Guard Configuration Checklist 86
Configuring Services for Transaction Guard 87
Application Continuity During Major Database Version Upgrades with DBMS_ROLLING 88
Reset Database Session State 88

7 Configuring Recovery Manager and Archiving

Overview of Configuring RMAN for Oracle RAC 1
Archiving Mode in Oracle RAC 2
Configuring the RMAN Snapshot Control File Location 2
Configuring RMAN to Automatically Backup the Control File and SPFILE 3
Crosschecking on Multiple Oracle RAC Nodes 3
Configuring Channels for RMAN in Oracle RAC 4
Configuring Channels to Use Automatic Load Balancing 4
Managing Archived Redo Logs Using RMAN in Oracle RAC 4
Archived Redo Log File Conventions in Oracle RAC 5
RMAN Archiving Configuration Scenarios 6
Oracle Advanced Cluster File System Archiving Scheme 6
Advantages of the Cluster File System Archiving Scheme 7
Initialization Parameter Settings for the Cluster File System Archiving Scheme 7
Location of Archived Logs for the Cluster File System Archiving Scheme 8
Noncluster File System Local Archiving Scheme 8
Considerations for Using Noncluster File System Local Archiving 8
Initialization Parameter Settings for Non-Cluster File System Local Archiving 9
Location of Archived Logs for Noncluster File System Local Archiving 9
File System Configuration for Noncluster File System Local Archiving 10
Monitoring the Archiver Processes 10

8 Managing Backup and Recovery

Managing Backup and Recovery in Clusters 1
RMAN Backup Scenario for Noncluster File System Backups 1
RMAN Restore Scenarios for Oracle RAC 2

Copyright © 1999, 2025, Oracle and/or its affiliates.

Restoring Backups from a Cluster File System 2
Restoring Backups from a Noncluster File System 3
Using RMAN or Oracle Enterprise Manager to Restore the Server Parameter File (SPFILE) 3
Instance Recovery in Oracle RAC 3
Single Node Failure in Oracle RAC 4
Multiple-Node Failures in Oracle RAC 4
Using RMAN to Create Backups in Oracle RAC 4
Channel Connections to Cluster Instances with RMAN 5
Node Affinity Awareness of Fast Connections 6
Deleting Archived Redo Logs after a Successful Backup 6
Autolocation for Backup and Restore Commands 7
Media Recovery in Oracle RAC 7
Parallel Recovery in Oracle RAC 7
Parallel Recovery with RMAN 8
Disabling Parallel Recovery 8
Disabling Instance and Crash Recovery Parallelism 8
Disabling Media Recovery Parallelism 8
Using a Fast Recovery Area in Oracle RAC 8

9 Cloning Oracle RAC to Nodes in a New Cluster

Introduction to Cloning Oracle RAC 1
Preparing to Clone Oracle RAC 2
Deploying Oracle RAC Clones to Nodes in a Cluster 3
Locating and Viewing Log Files Generated During Cloning 7

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

About Adding Nodes Using Cloning in Oracle RAC Environments 1
Cloning Local Oracle Homes on Linux and UNIX Systems 2
Cloning Shared Oracle Homes on Linux and UNIX Systems 3
Cloning Oracle Homes on Windows Systems 3

11 Adding and Deleting Oracle RAC from Nodes on Linux and Unix Systems

About Adding and Deleting Nodes 1
Adding Oracle RAC to Nodes with Oracle Grid Infrastructure Installed 2
Adding Administrator-Managed Oracle RAC Database Instances to Target Nodes 4
About Adding Administrator-Managed Oracle RAC Database Instances 4
Using Oracle DBCA in Interactive Mode to Add Database Instances to Target Nodes 5
Using Oracle DBCA in Silent Mode to Add Database Instances to Target Nodes 6
Deleting Oracle RAC from a Cluster Node 6

Copyright © 1999, 2025, Oracle and/or its affiliates.

Deleting Instances from Oracle RAC Databases 7
Using Oracle DBCA in Interactive Mode to Delete Instances from Nodes 7
Using Oracle DBCA in Silent Mode to Delete Instances from Nodes 8
Removing Oracle RAC 9
Deleting Nodes from A Cluster 9

12 Adding and Deleting Oracle RAC from Nodes on Windows Systems

Adding Oracle RAC to Nodes with Oracle Grid Infrastructure Installed 1
Adding Administrator-Managed Oracle RAC Database Instances to Target Nodes 3
About Using Oracle DBCA to Add Oracle RAC Instances 3
Using Oracle DBCA in Interactive Mode to Add Database Instances to Target Nodes 4
Using Oracle DBCA in Silent Mode to Add Database Instances to Target Nodes 5
Deleting Oracle RAC from a Cluster Node 5
Deleting Instances from Administrator-Managed Databases 6
Using Oracle DBCA in Silent Mode to Delete Instances from Nodes 6
Using Oracle DBCA in Interactive Mode to Delete Instances from Nodes 7
Removing Oracle RAC 8
Deleting Nodes from the Cluster 8

13 Design and Deployment Techniques

Deploying Oracle RAC for High Availability 1
About Designing High Availability Systems 1
Best Practices for Deploying Oracle RAC in High Availability Environments 2
Consolidating Multiple Applications in Cluster Databases 3
Managing Capacity During Consolidation 4
Managing the Global Cache Service Processes During Consolidation 4
Using Oracle AI Database Cloud for Consolidation 4
Scalability of Oracle RAC 5
General Design Considerations for Oracle RAC 6
General Database Deployment Topics for Oracle RAC 7
Tablespace Use in Oracle RAC 7
Object Creation and Performance in Oracle RAC 7
Node Addition and Deletion and the SYSAUX Tablespace in Oracle RAC 7
Distributed Transactions and Oracle RAC 8
Deploying OLTP Applications in Oracle RAC 8
Flexible Implementation with Cache Fusion 8
Deploying Data Warehouse Applications in Oracle RAC 9
Parallelism for Data Warehouse Applications on Oracle RAC 9
Parallel Processing in Data Warehouse Systems and Oracle RAC 9
Data Security Considerations in Oracle RAC 10

Copyright © 1999, 2025, Oracle and/or its affiliates.

Transparent Data Encryption and Keystores 10
Windows Firewall Considerations 10
Securely Run ONS Clients Using Wallets 11
Introduction to Blocker Resolver 12
Blocker Resolver Architecture 12
Optional Configuration for Blocker Resolver 13
Blocker Resolver Diagnostics and Logging 14

14 Monitoring Performance

Monitoring and Tuning Oracle RAC Databases 1
Overview of Monitoring Oracle RAC and Oracle Clusterware 2
Monitoring Oracle RAC and Oracle Clusterware with Oracle Enterprise Manager 2
The Cluster Database Home Page 3
The Interconnects Page 3
The Cluster Database Performance Page 4
Tuning Oracle RAC Databases 5
Database Reliability Framework 5
Verifying the Interconnect Settings for Oracle RAC 5
Influencing Interconnect Processing 6
Performance Views in Oracle RAC 7
Creating Oracle RAC Data Dictionary Views with CATCLUST.SQL 7
Oracle RAC Performance Statistics 7
Automatic Workload Repository in Oracle RAC Environments 8
Active Session History Reports for Oracle RAC 8
Overview of ASH Reports for Oracle RAC 9
ASH Report for Oracle RAC: Top Cluster Events 9
ASH Report for Oracle RAC: Top Remote Instance 9
Monitoring Oracle RAC Statistics and Wait Events 9
Oracle RAC Statistics and Events in AWR and Statspack Reports 10
Oracle RAC Wait Events 10
Monitoring Performance by Analyzing GCS and GES Statistics 11
Analyzing the Effect of Cache Fusion in Oracle RAC 11
Analyzing Performance Using GCS and GES Statistics 12
Analyzing Cache Fusion Transfer Impact Using GCS Statistics 12
Analyzing Response Times Based on Wait Events 13
Understanding Normal and Problem Wait Event Response Times 13
Block-Related Wait Events 14
Message-Related Wait Events 14
Contention-Related Wait Events 14
Load-Related Wait Events 15

Copyright © 1999, 2025, Oracle and/or its affiliates.

15 Converting Single-Instance Oracle AI Databases to Oracle RAC and Oracle RAC One Node

Converting to Oracle RAC and Oracle RAC One Node Using Oracle DBCA 1
- Overview of Converting Databases to Oracle RAC Using Oracle DBCA 2
- Converting Oracle AI Database Installations to Oracle RAC Using Oracle DBCA 3
  - Use Oracle DBCA to Create an Image of the Single-Instance Database 3
  - Perform the Oracle Grid Infrastructure Installation 4
  - Validate the Cluster 4
  - Copy the Preconfigured Database Image 5
  - Install the New Oracle AI Database Software with Oracle RAC 5
- Converting Single Instance on a Cluster to Oracle RAC One Node Using Oracle DBCA 5
- Converting Single Instance on a Cluster to Oracle RAC Using Oracle DBCA 6
  - Scenarios for Converting Single Instance on a Cluster to Oracle RAC 6
  - Single-Instance Database on a Cluster Running from an Oracle RAC-Enabled Home 6
  - Single-Instance Database on a Cluster Running from an Oracle RAC-Disabled Home 9
- Converting Single Instance on a Single Server to Oracle RAC One Node Using Oracle DBCA 10

Preparing to Convert with rconfig and Oracle Enterprise Manager 11
- Prerequisites for Converting to Oracle RAC Databases 12
- Configuration Changes During Oracle RAC Conversion Using rconfig 12
- Converting Databases to Oracle RAC Using rconfig or Oracle Enterprise Manager 13
- Converting Databases to Oracle RAC Using Oracle Enterprise Manager 14

Converting Databases to Oracle RAC Using rconfig 15

Example of rconfig XML Input Files for ConvertToRAC 16

Postconversion Steps 18

A Server Control Utility Reference

SRVCTL Usage Information A-1

Specifying Command Parameters as Keywords Instead of Single Letters A-2

Character Set and Case Sensitivity of SRVCTL Object Values A-3

Summary of Tasks for Which SRVCTL Is Used A-4

Using SRVCTL Help A-5

SRVCTL Privileges and Security A-6

Additional SRVCTL Topics A-7

Deprecated SRVCTL Subprograms or Commands A-7
- Single Character Parameters Deprecated for all SRVCTL Commands A-7
- Miscellaneous SRVCTL Commands and Parameters A-14

SRVCTL Command Reference A-14
- About Using SRVCTL Commands A-15
- database Commands A-17

Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl add database A-18
srvctl config database A-21
srvctl convert database A-22
srvctl disable database A-23
srvctl downgrade database A-24
srvctl enable database A-24
srvctl getenv database A-25
srvctl modify database A-25
srvctl predict database A-29
srvctl relocate database A-29
srvctl remove database A-31
srvctl setenv database A-31
srvctl start database A-32
srvctl status database A-34
srvctl stop database A-35
srvctl unsetenv database A-36
srvctl update database A-37
srvctl upgrade database A-37
diskgroup Commands A-37
srvctl disable diskgroup A-38
srvctl enable diskgroup A-38
srvctl predict diskgroup A-39
srvctl remove diskgroup A-39
srvctl start diskgroup A-40
srvctl status diskgroup A-40
srvctl stop diskgroup A-41
home Commands A-41
srvctl start home A-42
srvctl status home A-42
srvctl stop home A-43
instance Commands A-44
srvctl add instance A-44
srvctl disable instance A-45
srvctl enable instance A-46
srvctl modify instance A-46
srvctl remove instance A-47
srvctl start instance A-48
srvctl status instance A-49
srvctl stop instance A-50
srvctl transfer instance A-51
srvctl update instance A-52
listener Commands A-53

Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl add listener A-53
srvctl config listener A-55
srvctl disable listener A-56
srvctl enable listener A-56
srvctl getenv listener A-57
srvctl modify listener A-58
srvctl predict listener A-59
srvctl remove listener A-59
srvctl setenv listener A-60
srvctl start listener A-61
srvctl status listener A-61
srvctl stop listener A-62
srvctl unsetenv listener A-62
srvctl update listener A-63
network Commands A-63
srvctl add network A-64
srvctl config network A-65
srvctl modify network A-65
srvctl predict network A-67
srvctl remove network A-67
nodeapps Commands A-68
srvctl add nodeapps A-69
srvctl config nodeapps A-70
srvctl disable nodeapps A-71
srvctl enable nodeapps A-71
srvctl getenv nodeapps A-72
srvctl modify nodeapps A-73
srvctl remove nodeapps A-74
srvctl setenv nodeapps A-75
srvctl start nodeapps A-76
srvctl status nodeapps A-76
srvctl stop nodeapps A-76
srvctl unsetenv nodeapps A-77
ons Commands A-78
srvctl add ons A-78
srvctl config ons A-79
srvctl disable ons A-80
srvctl enable ons A-80
srvctl export ons A-80
srvctl modify ons A-80
srvctl remove ons A-81
srvctl start ons A-82

Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl status ons A-82
srvctl stop ons A-82
pdb Commands A-83
srvctl add pdb A-83
srvctl config pdb A-85
srvctl disable pdb A-86
srvctl enable pdb A-86
srvctl modify pdb A-87
srvctl remove pdb A-88
srvctl start pdb A-89
srvctl status pdb A-90
srvctl stop pdb A-91
scan Commands A-91
srvctl add scan A-92
srvctl config scan A-93
srvctl disable scan A-93
srvctl enable scan A-94
srvctl modify scan A-94
srvctl predict scan A-95
srvctl relocate scan A-95
srvctl remove scan A-96
srvctl start scan A-97
srvctl status scan A-97
srvctl stop scan A-98
scan_listener Commands A-98
srvctl add scan_listener A-99
srvctl config scan_listener A-100
srvctl disable scan_listener A-101
srvctl enable scan_listener A-101
srvctl export scan_listener A-102
srvctl modify scan_listener A-102
srvctl predict scan_listener A-103
srvctl relocate scan_listener A-104
srvctl remove scan_listener A-104
srvctl start scan_listener A-105
srvctl status scan_listener A-105
srvctl stop scan_listener A-106
srvctl update scan_listener A-107
service Commands A-107
srvctl add service A-108
srvctl config service A-116
srvctl disable service A-117

Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl enable service A-118
srvctl modify service A-119
srvctl predict service A-126
srvctl relocate service A-127
srvctl remove service A-129
srvctl start service A-129
srvctl status service A-131
srvctl stop service A-132
vip Commands A-134
srvctl add vip A-135
srvctl config vip A-136
srvctl disable vip A-136
srvctl enable vip A-137
srvctl getenv vip A-137
srvctl modify vip A-138
srvctl predict vip A-138
srvctl relocate vip A-139
srvctl remove vip A-139
srvctl setenv vip A-140
srvctl start vip A-141
srvctl status vip A-141
srvctl stop vip A-142
srvctl unsetenv vip A-143
volume Commands A-143
srvctl config volume A-144
srvctl disable volume A-145
srvctl enable volume A-146
srvctl remove volume A-146
srvctl start volume A-147
srvctl status volume A-148
srvctl stop volume A-149

B Troubleshooting Oracle RAC

Where to Find Files for Analyzing Errors B-1
Managing Diagnostic Data in Oracle RAC B-2
Using Instance-Specific Alert Files in Oracle RAC B-2
Enabling Tracing for Java-Based Tools and Utilities in Oracle RAC B-3
Resolving Pending Shutdown Issues B-3
How to Determine If Oracle RAC Instances Are Using the Private Network B-3

Copyright © 1999, 2025, Oracle and/or its affiliates.

List of Tables

3-1	How SQL*Plus Commands Affect Instances	7
3-2	Descriptions of V$ACTIVE_INSTANCES Columns	24
3-3	Parameters That Should Have Identical Settings on All Instances	34
5-1	Load Balancing Advisory FAN Events	9
6-1	Example Treatment of Restoring Original Oracle Function Values by Products During Replay	38
6-2	Event Parameter Name-Value Pairs and Descriptions	65
6-3	FAN Parameters and Matching Session Information	67
6-4	Standard Connection Tests for Some Common Application Servers	81
7-1	Archived Redo Log File Name Format Parameters	5
7-2	UNIX/NFS Location Log Examples, Noncluster File System Local Archiving	10
7-3	UNIX/NFS Configuration for Shared Read Local Archiving Examples	10
9-1	clone.pl Script Parameters	5
9-2	Environment Variables Passed to the clone.pl Script	6
9-3	Cloning Parameters Passed to the clone.pl Script.	6
9-4	Finding the Location of the Oracle Inventory Directory	8
11-1	Variables in the Oracle DBCA Silent Mode Syntax	6
12-1	Variables in the Oracle DBCA Silent Mode Syntax	7
A-1	String Restrictions for SRVCTL Object Names	A-3
A-2	Deprecated Single-Character Parameters for SRVCTL Commands	A-8
A-3	Deprecated Commands and Parameters for SRVCTL	A-14
A-4	Object Keywords and Abbreviations	A-16
A-5	srvctl add database Command Parameters	A-18
A-6	srvctl config database Command Parameters	A-21
A-7	srvctl convert database Command Parameters	A-22
A-8	srvctl disable database Command Parameters	A-23
A-9	srvctl downgrade database Command Parameters	A-24
A-10	srvctl enable database Command Parameters	A-24
A-11	srvctl getenv database Command Parameters	A-25
A-12	srvctl modify database Command Parameters	A-26
A-13	srvctl relocate database Command Parameters	A-30
A-14	srvctl remove database Command Parameters	A-31
A-15	srvctl setenv database Command Parameters	A-32
A-16	srvctl start database Command Parameters	A-32
A-17	srvctl status database Parameters	A-34
A-18	srvctl stop database Command Parameters	A-35
A-19	srvctl unsetenv database Command Parameters	A-36

Copyright © 1999, 2025, Oracle and/or its affiliates.

A-20 srvctl upgrade database Command Parameters A-37
A-21 srvctl disable diskgroup Command Parameters A-38
A-22 srvctl enable diskgroup Command Parameters A-39
A-23 srvctl start diskgroup Command Parameters A-40
A-24 srvctl status diskgroup Command Parameters A-40
A-25 srvctl stop diskgroup Command Parameters A-41
A-26 srvctl start home Command Parameters A-42
A-27 srvctl status home Command Parameters A-42
A-28 srvctl stop home Command Parameters A-43
A-29 srvctl add instance Command Parameters A-44
A-30 srvctl disable instance Command Parameters A-45
A-31 srvctl enable instance Command Parameters A-46
A-32 srvctl modify instance Command Parameters A-47
A-33 srvctl remove instance Command Parameters A-48
A-34 srvctl start instance Parameters A-49
A-35 srvctl status instance Command Parameters A-50
A-36 srvctl stop instance Command Parameters A-51
A-37 srvctl transfer instance Command Parameters A-51
A-38 srvctl add listener Command Parameters A-54
A-39 srvctl config listener Command Parameters A-56
A-40 srvctl disable listener Command Parameters A-56
A-41 srvctl enable listener Command Parameters A-57
A-42 srvctl getenv listener Command Parameters A-57
A-43 srvctl modify listener Command Parameters A-58
A-44 srvctl setenv listener Command Parameters A-60
A-45 srvctl start listener Command Parameters A-61
A-46 srvctl status listener Command Parameters A-61
A-47 srvctl stop listener Command Parameters A-62
A-48 srvctl unsetenv listener Command Parameters A-63
A-49 srvctl update listener Command Parameters A-63
A-50 srvctl add network Command Parameters A-64
A-51 srvctl modify network Command Parameters A-65
A-52 srvctl remove network Command Parameters A-67
A-53 srvctl add nodeapps Command Parameters A-69
A-54 srvctl disable nodeapps Command Parameters A-71
A-55 srvctl enable nodeapps Command Parameters A-71
A-56 srvctl getenv nodeapps Command Parameters A-72

Copyright © 1999, 2025, Oracle and/or its affiliates.

A-57 srvctl modify nodeapps Command Parameters A-73
A-58 srvctl remove nodeapps Command Parameters A-75
A-59 srvctl setenv nodeapps Command Parameters A-75
A-60 srvctl start nodeapps Command Parameters A-76
A-61 srvctl stop nodeapps Command Parameters A-77
A-62 srvctl unsetenv nodeapps Command Parameters A-77
A-63 srvctl add ons Command Parameters A-79
A-64 srvctl export ons Command Parameters A-80
A-65 srvctl modify ons Command Parameters A-81
A-66 srvctl add pdb Command Parameters A-84
A-67 srvctl config pdb Command Parameters A-85
A-68 srvctl disable pdb Command Parameters A-86
A-69 srvctl enable pdb Command Parameters A-87
A-70 srvctl modify pdb Command Parameters A-87
A-71 srvctl remove pdb Command Parameters A-89
A-72 srvctl start pdb Command Parameters A-89
A-73 srvctl status pdb Parameters A-90
A-74 srvctl stop pdb Command Parameters A-91
A-75 srvctl add scan Command Parameters A-92
A-76 srvctl config scan Command Parameters A-93
A-77 srvctl modify scan Command Parameters A-95
A-78 srvctl relocate scan Command Parameters A-96
A-79 srvctl remove scan Command Parameters A-96
A-80 srvctl start scan Command Parameters A-97
A-81 srvctl add scan_listener Command Parameters A-99
A-82 srvctl config scan_listener Command Parameters A-100
A-83 srvctl disable scan_listener Command Parameters A-101
A-84 srvctl enable scan_listener Command Parameters A-101
A-85 srvctl export scan_listener Command Parameters A-102
A-86 srvctl modify scan_listener Command Parameters A-103
A-87 srvctl remove scan_listener Command Parameters A-104
A-88 srvctl start scan_listener Command Parameters A-105
A-89 srvctl status scan_listener Command Parameters A-106
A-90 srvctl stop scan_listener Command Parameters A-106
A-91 srvctl add service Command Parameters A-109
A-92 Service Attribute Dependency Validation A-115
A-93 srvctl config service Command Parameters A-116

Copyright © 1999, 2025, Oracle and/or its affiliates.

A-94 srvctl disable service Command Parameters A-117
A-95 srvctl enable service Command Parameters A-118
A-96 srvctl modify service Parameters for Moving a Service A-119
A-97 srvctl modify service Parameters for Changing to a Preferred Instance A-120
A-98 srvctl modify service Parameters for Changing Status of Multiple Instances A-121
A-99 srvctl modify service Parameters A-121
A-100 srvctl predict service Command Parameters A-126
A-101 srvctl relocate service Command Parameters A-127
A-102 srvctl remove service Command Parameters A-129
A-103 srvctl status service Command Parameters A-132
A-104 srvctl stop service Command Parameters A-133
A-105 srvctl add vip Command Parameters A-135
A-106 srvctl config vip Command Parameters A-136
A-107 srvctl getenv vip Command Parameters A-137
A-108 srvctl modify vip Command Parameters A-138
A-109 srvctl relocate vip Command Parameters A-139
A-110 srvctl remove vip Command Parameters A-140
A-111 srvctl setenv vip Command Parameters A-140
A-112 srvctl start vip Command Parameters A-141
A-113 srvctl status vip Command Parameters A-141
A-114 srvctl stop vip Command Parameters A-142
A-115 srvctl unsetenv vip Command Parameters A-143
A-116 srvctl config volume Command Parameters A-144
A-117 srvctl disable volume Command Parameters A-145
A-118 srvctl enable volume Command Parameters A-146
A-119 srvctl remove volume Command Parameters A-147
A-120 srvctl start volume Command Parameters A-147
A-121 srvctl status volume Command Parameters A-148
A-122 srvctl stop volume Command Parameters A-149

Copyright © 1999, 2025, Oracle and/or its affiliates.

Preface

Oracle Real Application Clusters Administration and Deployment Guide describes the Oracle Real Application Clusters (Oracle RAC) architecture.

This publication provides an overview of the product, including Oracle Real Application Clusters One Node (Oracle RAC One Node). This publication also describes administrative and deployment topics for Oracle RAC.

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

- Audience
- Set Up Java Access Bridge to Implement Java Accessibility
  Install Java Access Bridge so that assistive technologies on Microsoft Windows systems can use the Java Accessibility API.
- Command Syntax
  Refer to these command syntax conventions to understand command examples in this guide.
- Conventions

Audience

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

- Install and configure an Oracle RAC database
- Administer and manage Oracle RAC databases
- Manage and troubleshoot clusters and networks that use Oracle RAC

Set Up Java Access Bridge to Implement Java Accessibility

Install Java Access Bridge so that assistive technologies on Microsoft Windows systems can use the Java Accessibility API.

Java Access Bridge is a technology that enables Java applications and applets that implement the Java Accessibility API to be visible to assistive technologies on Microsoft Windows systems.

Refer to *Java Platform, Standard Edition Accessibility Guide* for information about the minimum supported versions of assistive technologies required to use Java Access Bridge. Also refer to this guide to obtain installation and testing instructions, and instructions for how to use Java Access Bridge.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Related Topics

- *Java Platform, Standard Edition Java Accessibility Guide*

Command Syntax

Refer to these command syntax conventions to understand command examples in this guide.

Convention	Description
$	Bourne or BASH shell prompt in a command example. Do not enter the prompt as part of the command.
%	C Shell prompt in a command example. Do not enter the prompt as part of the command.
#	Superuser (root) prompt in a command example. Do not enter the prompt as part of the command.
`monospace`	UNIX command syntax
backslash \	A backslash is the UNIX and Linux command continuation character. It is used in command examples that are too long to fit on a single line. Enter the command as displayed (with a backslash) or enter it on a single line without a backslash: `dd if=/dev/rdsk/c0t1d0s6 of=/dev/rst0 bs=10b \  count=10000`
braces { }	Braces indicate required items: `.DEFINE {macro1}`
brackets [ ]	Brackets indicate optional items: `cvtcrt termname [outfile]`
ellipses ...	Ellipses indicate an arbitrary number of similar items: `CHKVAL fieldname value1 value2 ... valueN`
italic	Italic type indicates a variable. Substitute a value for the variable: *`library_name`*
vertical line |	A vertical line indicates a choice within braces or brackets: `FILE filesize [K|M]`

Conventions

The following text conventions are used in this document:

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

1

Introduction to Oracle RAC

Provides an overview of Oracle Real Application Clusters (Oracle RAC) installation and administration, and various components and functions.

> **Note**
>
> A multitenant Container Database (CDB) is the only supported architecture in Oracle Database 21c and later releases.

- Overview of Oracle RAC
  Learn about Oracle Real Application Clusters (Oracle RAC), and the differences in functionality between Oracle RAC and a single-instance Oracle AI Database.
- Overview of Oracle Multitenant with Oracle RAC
  You can configure a multitenant Container Database (CDB) to use Oracle RAC.
- Overview of Installing Oracle RAC
  Install Oracle Grid Infrastructure and Oracle AI Database software using Oracle Universal Installer, and create your database with Oracle DBCA.
- Overview of Oracle Real Application Clusters One Node
  Oracle Real Application Clusters One Node (Oracle RAC One Node) is an option to Oracle AI Database Enterprise Edition.
- Overview of Oracle Clusterware for Oracle RAC
  Oracle Clusterware provides a complete, integrated clusterware management solution on all Oracle AI Database platforms.
- Overview of Oracle RAC Architecture and Processing
  Installing Oracle RAC requires software, a network, and a storage configuration.
- Overview of Automatic Workload Management with Dynamic Database Services
  Services represent groups of applications with common attributes, service level thresholds, and priorities.
- Overview of Blocker Resolver
  Blocker Resolver is an Oracle AI Database feature that automatically detects and resolves system delays.
- Overview of Database In-Memory and Oracle RAC
  Every Oracle RAC node has its own In-Memory (IM) column store. By default, populated objects are distributed across all IM column stores in the cluster.
- Overview of Managing Oracle RAC Environments
  When managing Oracle RAC, there are many considerations, such as the deployment type, the tools to use, how to monitor the system, and how to evaluate performance.

Overview of Oracle RAC

Learn about Oracle Real Application Clusters (Oracle RAC), and the differences in functionality between Oracle RAC and a single-instance Oracle AI Database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Non-cluster Oracle AI Database instances have a one-to-one relationship between Oracle AI Database and the instance. Oracle RAC environments, however, have a one-to-many relationship between the database and instances. An Oracle RAC database can have several instances, all of which access one Oracle AI Database. All database instances must use the same interconnect, which can also be used by Oracle Clusterware.

Oracle RAC databases differ architecturally from a non-cluster Oracle AI Database, in that each Oracle RAC database instance also has:

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

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

A **cluster** comprises multiple interconnected computers or servers that appear as if they are one server to end users and applications. The Oracle RAC option with Oracle AI Database enables you to cluster Oracle AI Database instances. Oracle RAC uses Oracle Clusterware for the infrastructure to bind multiple servers so they operate as a single system.

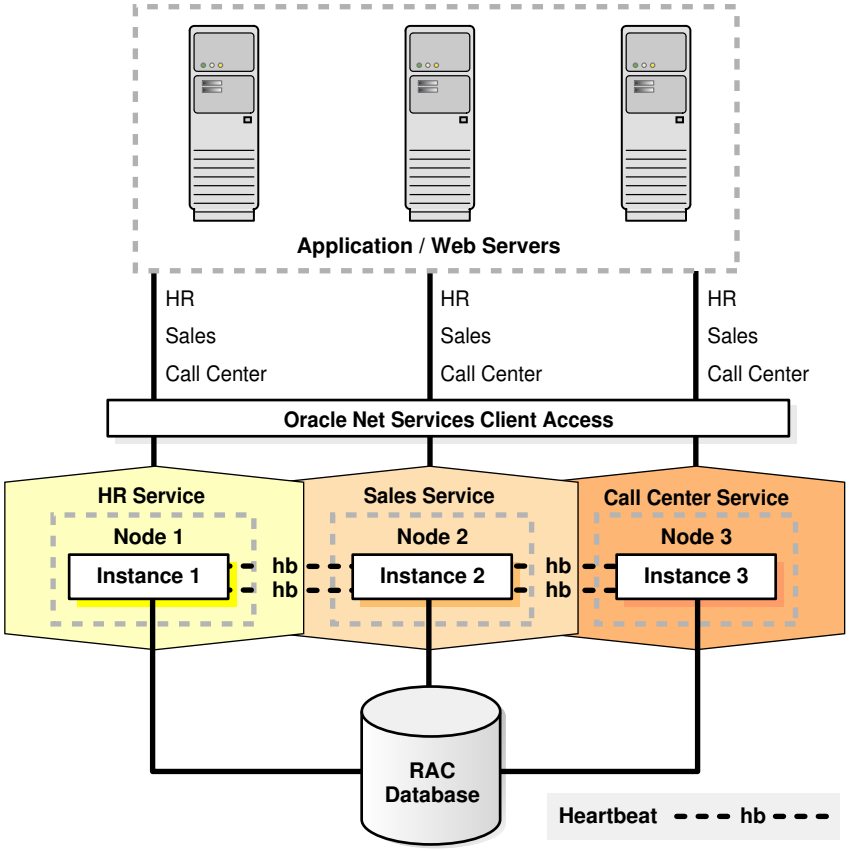
Oracle Clusterware is a portable cluster management solution that is integrated with Oracle AI Database. Oracle Clusterware is a required component for using Oracle RAC that provides the infrastructure necessary to run Oracle RAC. Oracle Clusterware also manages resources, such as **Virtual Internet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

Oracle Clusterware is the only clusterware that you need for most platforms on which Oracle RAC operates. Starting with Oracle Database 19c, the integration of vendor clusterware with Oracle Clusterware is deprecated, and is not supported in Oracle Database 21c and later releases.

The following figure shows how Oracle RAC is the Oracle AI Database option that provides a single system image for multiple servers to access one Oracle AI Database. In Oracle RAC, each Oracle instance must run on a separate server.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Figure 1-1 Oracle AI Database with Oracle RAC Architecture



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

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

Oracle RAC is a unique technology that provides high availability and scalability for all application types. The Oracle RAC infrastructure is also a key component for implementing the Oracle enterprise grid computing architecture. Having multiple instances access a single database prevents the server from being a single point of failure. Oracle RAC enables you to combine smaller commodity servers into a cluster to create scalable environments that support

Copyright © 1999, 2025, Oracle and/or its affiliates.

mission critical business applications. Applications that you deploy on Oracle RAC databases can operate without code changes.

Oracle RAC allows multiple instances running on different nodes to access the database. Oracle RAC Cache Fusion automatically ensures changes from multiple sessions running on different instances are coordinated. In Oracle Database 21c and later releases, these background processes that perform Oracle RAC Cache fusion functionality have been enhanced to handle irrecoverable errors. Depending on the cause, Oracle RAC can retry the operation or correct the error to prevent instance failure. This reduces the occurrence of instance failures and helps to prevent these workload impacts.

Related Topics

- Introduction to Oracle Clusterware
- Oracle Data Guard and Oracle Real Application Clusters
- Maximum Availability Architecture (MAA)

Overview of Oracle Multitenant with Oracle RAC

You can configure a multitenant Container Database (CDB) to use Oracle RAC.

You can make each PDB available on either every database instance of the Oracle RAC CDB or on a subset of instances. In either case, access to PDBs is regulated using dynamic database services. Applications uses these services to connect to a PDB, like they would connect to a single-instance non-CDB.

You can isolate PDBs to prevent certain operations from being performed on or within a particular PDB that may interfere with other PDBs sharing the same Oracle RAC database or instance. PDB isolation allows for greater consolidation.

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

Regardless of whether a PDB is available on multiple instances of an Oracle RAC CDB, the CDB is typically managed by the services running on the PDB. You can manually enable PDB access on each instance by starting the PDB manually on that instance.

Overview of Installing Oracle RAC

Install Oracle Grid Infrastructure and Oracle AI Database software using Oracle Universal Installer, and create your database with Oracle DBCA.

This ensures that your Oracle RAC environment has the optimal network configuration, database structure, and parameter settings for the environment that you selected.

Alternatively, you can install Oracle RAC using Fleet Patching and Provisioning, which offers all of the advantages of Oracle Universal Installer and Oracle DBCA previously specified. In addition, Fleet Patching and Provisioning allows for standardization and automation.

- Understanding Compatibility in Oracle RAC Environments
  As part of your deployment plan, review the release compatibility restrictions and guidelines for using different Oracle AI Database releases on Oracle Grid Infrastructure.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Oracle RAC Database Installation
  Learn how to install Oracle Real Application Clusters (Oracle RAC) and review the restrictions.
- Oracle RAC Database Creation
  Part of Oracle AI Database deployment is the creation of the database.
- Overview of Extending Oracle RAC Clusters
  To extend an Oracle RAC cluster, also known as cloning, and add nodes to your environment after your initial deployment, then you must to do this on multiple layers, considering the management style that you currently use in the cluster.

Related Topics

- *Oracle Real Application Clusters Installation Guide for Linux and UNIX*
- *Oracle Grid Infrastructure Installation and Upgrade Guide for Linux*

Understanding Compatibility in Oracle RAC Environments

As part of your deployment plan, review the release compatibility restrictions and guidelines for using different Oracle AI Database releases on Oracle Grid Infrastructure.

To run Oracle RAC in configurations with different releases of Oracle AI Database in the same cluster, you must first install Oracle Grid Infrastructure, which must be the same version, or higher, as the highest version of Oracle AI Database that you want to deploy in this cluster. For example, to run an Oracle RAC 19c database and an Oracle RAC 26ai database in the same cluster, you must install Oracle Grid Infrastructure 26ai. Contact My Oracle Support for more information about version compatibility in Oracle RAC environments.

Oracle RAC Database Installation

Learn how to install Oracle Real Application Clusters (Oracle RAC) and review the restrictions.

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

Before you install Oracle RAC, first install Oracle Grid Infrastructure. The release of Oracle Grid Infrastructure must be the same as or newer than the Oracle RAC release that you are installing. Oracle Universal Installer only enables you to deploy an Oracle AI Database home across the nodes in a cluster if you previously installed and configured Oracle Grid Infrastructure for the cluster. If Oracle Universal Installer does not give you an option to deploy the database home across all of the nodes in the cluster, then the server on which you are attempting to install Oracle RAC is not a cluster member node.

During installation, you must manually run Oracle DBCA to create an Oracle RAC or an Oracle RAC One Node database. In these cases, database creation is a two-step process. First, install the Oracle RAC software by running the Oracle AI Database installer. Then create and configure your Oracle RAC or Oracle RAC One Node database using Oracle DBCA.

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> Before you create a database, a default listener must be running in the Oracle Grid Infrastructure home. If there is no default listener in the Oracle Grid Infrastructure home, then Oracle DBCA returns an error instructing you to run NETCA from the Oracle Grid Infrastructure home to create a default listener.

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

The Oracle AI Database installation process creates a read-only Oracle home directory by default. You can use the read-only Oracle home as a software image to be shared across multiple database servers. This simplifies patching and mass rollout because only one Oracle home image needs to be updated to distribute a patch to multiple database servers. Read-only Oracle AI Database homes are more secure than traditional Oracle AI Database homes because there is a clear separation of configuration information from the actual software. Thus, there is no risk of new files being created inside the Oracle AI Database home from active processes.

Related Topics

- Oracle RAC Database Creation
  Part of Oracle AI Database deployment is the creation of the database.
- *Oracle AI Database Net Services Administrator's Guide*

Oracle RAC Database Creation

Part of Oracle AI Database deployment is the creation of the database.

You can create a database as part of the database deployment or you can only deploy the database software first, then create any database that is meant to run out of the newly created Oracle home by using Oracle DBCA.

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

In Oracle RAC environments, the values for `DB_UNIQUE_NAME.DB_DOMAIN` in its entirety must be unique for each database within your enterprise. The name of each Pluggable Database (PDB) should also be unique within the cluster.

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

Related Topics

- Oracle RAC Database Installation
  Learn how to install Oracle Real Application Clusters (Oracle RAC) and review the restrictions.

Overview of Extending Oracle RAC Clusters

To extend an Oracle RAC cluster, also known as cloning, and add nodes to your environment after your initial deployment, then you must to do this on multiple layers, considering the management style that you currently use in the cluster.

Oracle provides various means of extending Oracle RAC clusters. Choose from the following approaches to extend the current environment:

- Fleet Patching and Provisioning to provision new Oracle RAC databases and other software
- Cloning using cloning scripts
- Adding nodes using the `addnode.sh` (`addnode.bat` on Windows) script

Both approaches apply, regardless of how you initially deployed your environment. Both approaches copy the Oracle software on to the node that you plan to add to the cluster. Software that is copied to the node includes the Oracle Grid Infrastructure software and the Oracle AI Database homes.

For Oracle AI Database homes, ensure that the database software is deployed on all of the nodes on which database instances can potentially run. In either case, first deploy Oracle Grid Infrastructure on all of the nodes that are to be part of the cluster.

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

For new installations, or if you install only one Oracle RAC database, use the traditional automated and interactive installation methods, such as Oracle Universal Installer, Fleet Patching and Provisioning, or the Provisioning Pack feature of Oracle Enterprise Manager. To add or delete Oracle RAC from nodes in a cluster, use the procedures detailed in the *Adding and Deleting Oracle RAC from Nodes...* topics listed at the end of this topic.

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

Related Topics

- Fleet Patching and Provisioning
- Cloning Oracle RAC to Nodes in a New Cluster
  Learn how to clone Oracle Real Application Clusters (Oracle RAC) database homes on Linux and Unix systems to nodes in a new cluster.
- Adding and Deleting Oracle RAC from Nodes on Linux and Unix Systems
  Extend an existing Oracle Real Application Clusters (Oracle RAC) home to other nodes and instances in the cluster, and delete Oracle RAC from nodes and instances in the cluster.
- Adding and Deleting Oracle RAC from Nodes on Windows Systems
  Use these procedures to extend an existing Oracle Real Application Clusters (Oracle RAC) home on Microsoft Windows to other nodes and instances in the cluster, or delete Oracle RAC from nodes and instances in the cluster.

See Also

Oracle Enterprise Manager online help system for more information about the Provisioning Pack

Overview of Oracle Real Application Clusters One Node

Oracle Real Application Clusters One Node (Oracle RAC One Node) is an option to Oracle AI Database Enterprise Edition.

Oracle RAC One Node is a single instance of an Oracle RAC-enabled database running on one node in the cluster, only, under normal operations. This option adds to the flexibility that Oracle offers for database consolidation while reducing management overhead by providing a standard deployment for Oracle AI Databases in the enterprise. Oracle RAC One Node database requires Oracle Grid Infrastructure and, therefore, requires the same hardware setup as an Oracle RAC database.

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

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

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

Notification or Transparent Application Failover to minimize the impact of a relocation on the client.

Oracle RAC One Node databases are administered slightly differently from Oracle RAC or non-cluster databases. For Oracle RAC One Node databases, you must monitor the candidate node list and make sure a server is always available for failover, if possible.

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

Related Topics

- *Oracle Real Application Clusters Installation Guide for Linux and UNIX*
- Transaction Guard for Improving Client Failover
  Transaction Guard is a developer feature that your applications can use to determine `COMMIT_OUTCOME` following recoverable errors.

Overview of Oracle Clusterware for Oracle RAC

Oracle Clusterware provides a complete, integrated clusterware management solution on all Oracle AI Database platforms.

- Guidelines for Using Oracle Clusterware
  The functionality provided by Oracle Clusterware provides all of the features that are required to manage cluster databases, including node membership, group services, global resource management, and high availability functions.
- Overview of Temporary Tablespaces
  Oracle uses temporary tablespaces to write spill-overs to non-shared tablespaces that are created on local disks.

Guidelines for Using Oracle Clusterware

The functionality provided by Oracle Clusterware provides all of the features that are required to manage cluster databases, including node membership, group services, global resource management, and high availability functions.

You can install Oracle Grid Infrastructure independently or as a prerequisite to installing Oracle RAC. Oracle AI Database features, such as services, use the underlying Oracle Clusterware mechanisms to provide advanced capabilities. Starting with Oracle Database 21c, third-party clusterware products are no longer supported with Oracle RAC.

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

automatically start when the node starts and automatically restart if they fail. The Oracle Clusterware daemons run on each node.

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

Related Topics

- Managing Oracle Cluster Registry and Oracle Local Registry

Overview of Temporary Tablespaces

Oracle uses temporary tablespaces to write spill-overs to non-shared tablespaces that are created on local disks.

It is still possible for SQL operations, such as hash aggregations, sorts, hash joins, creations of cursor-duration temporary tables for the `WITH` clause, and star transformations to spill over to disk. The spill overs go to the tablespaces on shared disks. Management of temporary tablespaces is similar to that of existing tablespaces.

Temporary tablespaces improve tablespace management by:

- Storing temp files in private storage to take advantage of local storage I/O benefits.
- Avoiding expensive cross-instance temporary tablespace management.
- Increased addressability of temporary tablespace.
- Improving instance warm-up performance by eliminating on-disk space metadata management.

> **Note**
>
> You cannot use temporary tablespaces to store database objects, such as tables or indexes.

- Parallel Execution Support for Cursor-Duration Temporary Tablespaces
  The temporary tablespaces that are created using the `WITH` clause and star transformation exist in the temporary tablespace on shared disk.
- Temporary Tablespace Organization
  Review the following information when organizing the temporary tablespaces.
- Temporary Tablespace Hierarchy
  When you define temporary tablespaces, there is a hierarchy that determines how the tablespaces are used.
- Temporary Tablespace Features
  Review the following information when using temporary tablespaces.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Metadata Management of Temporary Files
  Instance-specific information, such as bitmap for allocation, current size for a temporary file, and the file status, is stored in the SGA and not in control files because such information can vary across instances.
- DDL Support for Temporary Tablespaces
  You can manage temporary tablespaces and temporary files with either the DDL command `ALTER TABLESPACE`, or `ALTER DATABASE`.
- Temporary Tablespace for Users
  When you create a user without explicitly specifying shared or temporary tablespace, the user inherits shared and temporary tablespace from the corresponding default database tablespaces.
- Atomicity Requirement for Commands
  All of the commands that you run from read-write instances are performed in an atomic manner.
- Temporary Tablespace and Dictionary Views
  Oracle extended dictionary views can display information about temporary tablespaces.

Related Topics

- *Oracle AI Database SQL Language Reference*

Parallel Execution Support for Cursor-Duration Temporary Tablespaces

The temporary tablespaces that are created using the `WITH` clause and star transformation exist in the temporary tablespace on shared disk.

A set of parallel query child processes loads the intermediate query results into these temporary tablespaces, which are then read by a different child processes. There is no restriction on how these child processes reading these results are allocated, because any parallel query child process on any instance can read the temporary tablespaces residing on the shared disk.

For read-write and read-only instance architecture, when the parallel query child processes load the intermediate results to the temporary tablespaces of these instances, the parallel query child processes of the instance where the intermediate results are stored share an affinity with the read operations for the intermediate results and can thus read them.

Temporary Tablespace Organization

Review the following information when organizing the temporary tablespaces.

For example:

```
CREATE TEMPORARY TABLESPACE temp_ts TEMPFILE\
   '/u01/app/oracle/database/23.4.0/dbs/temp_file'\
   EXTENT MANAGEMENT UNIFORM SIZE 1M AUTOEXTEND ON;
```

- Creation of a temporary tablespace results in the creation of temporary files on every instance and not a single file.
- You can create temporary tablespaces for both read-only and read-write instances.

```
CREATE TEMPORARY TABLESPACE FOR ALL temp_ts TEMPFILE\
  ‘/u01/app/oracle/database/23.4.0/dbs/temp_file’\
  EXTENT MANAGEMENT UNIFORM SIZE 1M AUTOEXTEND ON;
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Temporary Tablespace Hierarchy

When you define temporary tablespaces, there is a hierarchy that determines how the tablespaces are used.

There can be multiple temporary tablespaces in a database, such the default shared temporary tablespace for the database and multiple temporary tablespaces that are assigned to individual users. If a user has a shared temporary tablespace assigned, then that tablespace is used first, otherwise the database default temporary tablespace is used.

Once a tablespace has been selected for spilling during query processing, there is no switching to another tablespace. For example, if a user has a shared temporary tablespace assigned and during spilling it runs out of space, then there is no switching to an alternative tablespace. The spilling, in that case, results in an error. Additionally, remember that shared temporary tablespaces are shared among instances.

The allocation of temporary space for spilling to a temporary tablespace differs between read-only and read-write instances. For read-only instances, the following is the priority of selecting which temporary location to use for spills:

1. Allocate from a user's temporary tablespace.
2. Allocate from the database default temporary tablespace.

> **Note**
>
> If there is no temporary tablespace in the database, then read-only instances spill to shared temporary tablespace.

For read-write instances, the priority of allocation differs from the preceding allocation order, because shared temporary tablespaces are given priority, allocating as follows from:

1. A user’s shared temporary tablespace.
2. The database default shared temporary tablespace.

Temporary Tablespace Features

Review the following information when using temporary tablespaces.

Instances cannot share temporary tablespace. Therefore, one instance cannot take temporary tablespace from another. If an instance runs out of temporary tablespace during spilling, then the statement results in an error.

- Temporary tablespaces support only one `BIGFILE` per tablespace.
- To address contention issues arising from having only one `BIGFILE`-based temporary tablespace, you can assign multiple temporary tablespaces to different users as the default.
- A database administrator can specify the default temporary tablespace for a user using `ALTER USER` syntax. For example:

  ```
  ALTER USER MAYNARD TEMPORARY TABLESPACE temp_ts;
  ```

- You can configure a user with two default temporary tablespaces:

Copyright © 1999, 2025, Oracle and/or its affiliates.

- One temporary tablespace when the user is connected to the read-only instance.
- One shared temporary tablespace to be used when the same user is connected on the read-write instances that are running on a node.

Metadata Management of Temporary Files

Instance-specific information, such as bitmap for allocation, current size for a temporary file, and the file status, is stored in the SGA and not in control files because such information can vary across instances.

Currently, temporary file information (such as file name, creation size, creation SCN, temporary block size, and file status) is stored in the control file along with the initial and max files, as well as auto extent attributes. However, information about temporary files in the control file is common to all applicable instances.

When an instance starts, it reads the control file information and creates the temporary files that constitute the temporary tablespace for that instance. If there are two or more instances running on a node, then each instance has its own temporary files.

For temporary tablespaces, there is a separate file for each involved instance. The temporary file names follow a naming convention such that the instance numbers are appended to the temporary file names that are specified while creating the temporary tablespace.

For example, assume that a read-only node, N1, runs two Oracle read-only database instances with numbers 3 and 4. The following DDL command creates two files on node N1—`/temp/temp_file_3` and `/temp/temp_file_4`, for instance 3 and 4 respectively:

```
CREATE TEMPORARY TABLESPACE temp_ts TEMPFILE  '/temp/temp_file'\
   EXTENT MANAGEMENT UNIFORM SIZE 1M AUTOEXTEND ON;
```

Assuming that there are two read-write instances, instances 1 and 2, and two read-only instances, instances 3 and 4, the following DDL command creates four files—`/temp/temp_file_all_1` and `/temp/temp_file_all_2` for instances 1 and 2, respectively, and `/temp/temp_file_all_3` and `/temp/temp_file_all_4` for instances 3 and 4, respectively:

```
CREATE TEMPORARY TABLESPACE FOR ALL  temp_ts_for_all TEMPFILE  '/temp/
temp_file_all'\
   EXTENT MANAGEMENT UNIFORM SIZE 1M AUTOEXTEND ON;
```

DDL Support for Temporary Tablespaces

You can manage temporary tablespaces and temporary files with either the DDL command `ALTER TABLESPACE`, or `ALTER DATABASE`.

Run all temporary tablespace management and creation DDL commands from the read-write instances. Running all other DDL commands affects all instances in a homogeneous manner.

For example, the following command resizes the temporary files on all read-only instances:

```
ALTER TABLESPACE temp_ts RESIZE 1G;
```

For temporary tablespaces, Oracle supports the allocation options and their restrictions that are currently active for temporary files.

Copyright © 1999, 2025, Oracle and/or its affiliates.

To run a DDL command on a temporary tablespace on a read-only instance, there must be at least one read-only instance in the cluster. This restriction is not applicable when creating or altering temporary tablespaces `FOR ALL`. You can assign a default temporary tablespace to the database with the clause `DEFAULT TEMPORARY TABLESPACE` appended to the command `ALTER DATABASE`.

For example:

```
ALTER DATABASE DEFAULT TEMPORARY TABLESPACE temp_ts;
```

You can specify default temporary tablespaces when creating the database, as follows:

```
CREATE DATABASE .. DEFAULT TEMPORARY TABLESPACE temp_ts_for_dbtemp_ts
TEMPFILE\
   '/temp/temp_file_for_db' EXTENT MANAGEMENT UNIFORM SIZE 1M AUTOEXTEND ON;
```

You cannot specify default temporary tablespaces using the command `CREATE DATABASE`. When you create a database, its default temporary tablespace points to the default shared temporary tablespace. You must run the command `ALTER DATABASE` to assign an existing temporary tablespace as the default for the database.

Temporary Tablespace for Users

When you create a user without explicitly specifying shared or temporary tablespace, the user inherits shared and temporary tablespace from the corresponding default database tablespaces.

You can specify default temporary tablespace for a user, as follows:

```
CREATE USER new_user IDENTIFIED BY new_user TEMPORARY TABLESPACE
temp_ts_for_all;
```

You can change the temporary tablespace for a user using the `ALTER USER` command, as follows:

```
ALTER USER maynard TEMPORARY TABLESPACE temp_ts;
```

As previously mentioned, default user temporary tablespace can be shared temporary space. Consider the following items for the `ALTER USER...TEMPORARY TABLESPACE` command:

- You can change the user default temporary tablespace to any existing temporary tablespace.
- If you want to set the user default temporary tablespace to a shared temporary tablespace, `T`, then `T` must be the same as the default shared temporary tablespace.
- If a default user temporary tablespace points to a shared temporary tablespace, then, when you change the default shared temporary tablespace of the user, you also change the default temporary tablespace to that tablespace.

Following are some examples of temporary space management using the command `ALTER`:

- To take a temporary tablespace offline:

  ```
  ALTER DATABASE TEMPFILE ‘/temp/temp_file’ OFFLINE;
  ```


Copyright © 1999, 2025, Oracle and/or its affiliates.

- To decrease the size of a temporary tablespace:

```
ALTER TABLESPACE temp_ts SHRINK SPACE KEEP 20M
```

- To change the auto-extension attributes of a temporary file:

```
ALTER TABLESPACE temp_ts AUTOEXTEND ON NEXT 20G
```

- To resize a temporary file:

```
ALTER TABLESPACE temp_ts RESIZE 10G
```

> **Note**
>
> When you resize a temporary file, it applies to individual files.

Some read-only instances may be down when you run the preceding commands. This does not prevent the commands from succeeding because when a read-only instance starts up later, it creates the temporary files based on information in the control file. Creation is fast because Oracle AI Database reformats only the header block of the temporary file, recording information about the file size, among other things. If you cannot create the temporary files, then the read-only instance stays down. Commands that you submitted from read-write instances are replayed immediately on all open, read-only instances.

Atomicity Requirement for Commands

All of the commands that you run from read-write instances are performed in an atomic manner.

This means that the command succeeds only when it succeeds on all live instances.

Temporary Tablespace and Dictionary Views

Oracle extended dictionary views can display information about temporary tablespaces.

Note the following changes to data dictionary views:

- All of the diagnosibility information that is related to temporary tablespaces and temporary files that are exposed through AWR, SQL monitor, and other utilities, is also available for temporary tablespaces and temporary files. This information is available with the existing dictionary views for temporary tablespaces and temporary files: `DBA_TEMP_FILES`, `DBA_TEMP_FREE_SPACE`.
- The `USER_TABLESPACES` and `DBA_TABLESPACES` dictionary view are extended by a column called `SHARED`, that indicates if the temporary file is shared.
- The `DBA_TEMP_FILES` dictionary view is extended by two columns: `SHARED` and `INST_ID`. The `SHARED` column indicates if the temp file is shared. The `INST_ID` column contains the instance number. For shared temporary files, there is a single row per file and the `INST_ID` is null. For temporary files, this column contains information about temporary files for each instance, such as the size of the file in bytes (`BYTES` column).
- The `DBA_TEMP_FREE_SPACE` dictionary view is extended by two columns: `SHARED` and `INST_ID`. The `SHARED` column indicates of the temporary file is shared. The `INST_ID` column contains the instance number. For shared temporary files, there is a single row for each file and the `INST_ID` is null. For temporary files, this column contains information about

Copyright © 1999, 2025, Oracle and/or its affiliates.

temporary files for each instance, such as the total free space available (`FREE_SPACE` column).

- In the dictionary views, such as `DBA_TABLESPACES`, Oracle distinguishes the type of the tablespace using the `SHARED` column with the following values:
  - `SHARED`: for shared temporary tablespace
  - `LOCAL_ON_ALL`: for temporary tablespace on all of the instances

> **Note**
>
> Currently, spills onto temporary tablespace for queries, such as sort and hash join spills, are automatically encrypted. This is also true for spills to temporary tablespace.

Overview of Oracle RAC Architecture and Processing

Installing Oracle RAC requires software, a network, and a storage configuration.

Oracle RAC requires several components:

- Oracle Clusterware for concurrent access to the same storage and the same set of data files from all of the nodes in a cluster
- A communications protocol for enabling interprocess communication (IPC) across all of the nodes in a cluster
- Multiple database instances that process data as if the data resided on a logically combined, single cache
- A mechanism for monitoring and communicating the statuses of the nodes in a cluster
- Understanding Cluster-Aware Storage Solutions
  Learn about the Oracle Real Application Clusters shared everything architecture and what shared everything means for your storage options.
- Oracle RAC and Network Connectivity
  All nodes in Oracle RAC environments must connect to at least one Local Area Network (LAN). This network is commonly referred to as the *public network*, and it enables users and applications to access the database.
- Overview of Using Dynamic Database Services to Connect to Oracle AI Database
  Applications should use the Dynamic Database Services feature to connect to an Oracle AI Database over the public network.
- Overview of Virtual IP Addresses
  Node VIPs are virtual IP (VIP) addresses that clients use to connect to Oracle RAC databases.
- Restricted Service Registration in Oracle RAC
  The valid node checking feature provides the ability to configure and dynamically update a set of IP addresses or subnets from which registration requests are allowed by the listener.
- About Oracle RAC Software Components
  In an Oracle RAC environment, Oracle RAC Cache Fusion automatically synchronizes each Oracle instance's buffer cache.
- About Oracle RAC Background Processes
  The **global cache service** (GCS) and the **global enqueue service** (GES) processes, along with the **global resource directory** (GRD) collaborate to enable Cache Fusion.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Understanding Cluster-Aware Storage Solutions

Learn about the Oracle Real Application Clusters shared everything architecture and what shared everything means for your storage options.

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

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

- Oracle Automatic Storage Management (Oracle ASM)
  Oracle recommends this solution to manage your storage.
- A certified cluster file system
  - Oracle recommends Oracle Advanced Cluster File System (Oracle ACFS).
  - A third-party cluster file system on a cluster-aware volume manager that is certified for Oracle RAC. For example:
    * Oracle OCFS2 (Linux only)
    * IBM GPFS (IBM AIX only)
- Certified network file system (NFS) solution

Oracle RAC and Network Connectivity

All nodes in Oracle RAC environments must connect to at least one Local Area Network (LAN). This network is commonly referred to as the *public network*, and it enables users and applications to access the database.

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

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

> **Note**
>
> - Oracle supports interfaces with higher bandwidth but *does not* support using crossover cables with the interconnect.
> - Do not use the interconnect, the private network, for user communication, because **Cache Fusion** uses the interconnect for interinstance communication.

You can configure Oracle RAC to use either the User Datagram Protocol (UDP) or Reliable Data Socket (RDS) protocols for inter-instance communication on the interconnect. Oracle Clusterware uses the same interconnect using the UDP protocol, but cannot be configured to use RDS.

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

Overview of Using Dynamic Database Services to Connect to Oracle AI Database

Applications should use the Dynamic Database Services feature to connect to an Oracle AI Database over the public network.

Dynamic Database Services enable you to define rules and characteristics to control how users and applications connect to database instances. These characteristics include a unique name, workload balancing and failover options, and high availability characteristics.

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

Related Topics

- Overview of Automatic Workload Management with Dynamic Database Services
  Services represent groups of applications with common attributes, service level thresholds, and priorities.

Overview of Virtual IP Addresses

Node VIPs are virtual IP (VIP) addresses that clients use to connect to Oracle RAC databases.

Oracle Clusterware hosts the node VIP addresses on a public network. The process for a typical connection attempt from a database client to an Oracle RAC database instance can be summarized as follows:

1. The database client connects to SCAN (which includes a SCAN VIP on a public network), providing the SCAN listener with a valid service name.
2. The SCAN listener then determines which database instance hosts this service and routes the client to the local or node listener on the respective node.
3. The node listener, listening on a node VIP and a given port, retrieves the connection request and connects the client to an instance on the local node.

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

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

- The node on which a VIP address runs fails

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

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

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

Restricted Service Registration in Oracle RAC

The valid node checking feature provides the ability to configure and dynamically update a set of IP addresses or subnets from which registration requests are allowed by the listener.

Database instance registration with a listener succeeds only when the request originates from a valid node. The network administrator can specify a list of valid nodes, excluded nodes, or disable valid node checking altogether. The list of valid nodes explicitly lists the nodes and subnets that can register with the database. The list of excluded nodes explicitly lists the nodes that cannot register with the database. The control of dynamic registration results in increased manageability and security of Oracle RAC deployments.

By default, the SCAN listener agent sets `REMOTE_ADDRESS_REGISTRATION_`*`listener_name`* to a private IP endpoint. The SCAN listener accepts registration requests only from the private network. Remote nodes that are not accessible to the private network of the SCAN listener must be included in the list of valid nodes by using the `registration_invited_nodes_`*`alias`* parameter in the `listener.ora` file, or by modifying the SCAN listener using the command-line interface, SRVCTL.

> **Note**
>
> Starting with Oracle Grid Infrastructure 12c, for a SCAN listener, if the `VALID_NODE_CHECKING_REGISTRATION_`*`listener_name`* and `REGISTRATION_INVITED_NODES_`*`listener_name`* parameters are set in the `listener.ora` file, then the listener agent overwrites these parameters.

If you use the SRVCTL utility to set the `invitednodes` and `invitedsubnets` values, then the listener agent automatically sets `VALID_NODE_CHECKING_REGISTRATION_`*`listener_name`* to SUBNET and sets `REGISTRATION_INVITED_NODES_`*`listener_name`* to the specified list in the `listener.ora` file.

For other listeners managed by CRS, the listener agent sets `VALID_NODE_CHECKING_REGISTRATION_`*`listener_name`* to be SUBNET in the `listener.ora` file only if it is not already set in the `listener.ora` file. The SRVCTL utility does not support setting the `invitednodes` and `invitedsubnets` values for a non-SCAN listener. The listener agent does not update `REGISTRATION_INVITED_NODES_`*`listener_name`* in the `listener.ora` file for a non SCAN listener.

Copyright © 1999, 2025, Oracle and/or its affiliates.

About Oracle RAC Software Components

In an Oracle RAC environment, Oracle RAC Cache Fusion automatically synchronizes each Oracle instance's buffer cache.

Oracle RAC database automatically manages the block transfer between the instances without any manual intervention from the Database Administrators. The Global Cache Service (GCS) and the Global Enqueue Service (GES) perform the bulk of the work. The GCS and GES coordinate buffer cache blocks and enqueues in the Global Resource Directory (GRD). The GRD contents are distributed across all the active instances, resulting in a slight increase in the SGA size.

Note

- GRD and Oracle RAC Cache Fusion components use approximately 10 percent of the total SGA. When you move from a single-instance database to an Oracle RAC database, plan to increase your SGA accordingly.
- Starting with Oracle AI Database 26ai, Release Update 23.7:
  - The default value of the number of lock manager tickets has increased, resulting in improved performance in very busy concurrent workloads. This also results in a slight increase in the memory usage by the GCS process.
  - Improved monitoring capabilities by GCR* processes may result in a slight increase in memory usage. The V_GCR_METRICS view provides details on the new monitoring capabilities.

Once the cache is warm, the blocks can be transferred to any other instance needing the block via the private network. This is often faster than reading the block from a disk. Even in cases where a consistent block is requested from another instance, Cache Fusion transfers the block image directly.

In addition, Cache Fusion monitors the latency on the private networks and the service time on the disks, and automatically chooses the best path.

Related Topics

- *Oracle AI Database In-Memory Guide*

About Oracle RAC Background Processes

The **global cache service** (GCS) and the **global enqueue service** (GES) processes, along with the **global resource directory** (GRD) collaborate to enable Cache Fusion.

The Oracle RAC processes and their identifiers are as follows:

- `ACMS`: Atomic Controlfile to Memory Service (`ACMS`)

  In Oracle RAC environments, the `ACMS` process on each instance is an agent that contributes to ensuring a distributed SGA memory update is either globally committed on success or globally terminated if a failure occurs.
- `BG00`: Background Process

  The `ora_bg` is a standard background process in Oracle RAC environments that performs various maintenance and monitoring tasks. It is responsible for buffer cache management,

Copyright © 1999, 2025, Oracle and/or its affiliates.

log writer (LGWR) process support, checkpoint coordination, background process monitoring, and statistics gathering.

- GCW: Global Conflict Resolution (GCR) Monitor Process (LMHB)

  The ora_gcw is an infrastructure helper processes for LMHB. It monitors the LMHB process for stall and restarts, if necessary.

- GTX0-j: Global Transaction Process

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

- LMON: Global Enqueue Service Monitor

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

- LMD: Global Enqueue Service Daemon

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

- LMS: Global Cache Service Process

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

- LCK0: Instance Enqueue Process

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

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

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

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

- SCMN: Statistics Collection and Management

  SCMN is the main thread for a multithreaded process in a threaded Oracle RAC architecture. All other threads are spawned within the processes through SCMN based on the requests in the instance. It is an idle main thread, which waits for any requests, especially spawn threads, and takes care of them along with some periodic maintenance operations.

- VOSD: Virtual Operating System Daemon

  The VOSD process executes time bound Oracle AI Database service actions. VOSD is spawned at instance startup and it is responsible for executing system service actions critical for the database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

Related Topics

- *Oracle AI Database Reference*

Overview of Automatic Workload Management with Dynamic Database Services

Services represent groups of applications with common attributes, service level thresholds, and priorities.

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

Middle tier applications and clients select a service by specifying the service name as part of the connection in the TNS connect string. For example, data sources for Oracle WebLogic Server are set to route to a service. Using Net Easy*Connection, this connection comprises only the service name and network address, as follows: *`user_name/password`*`@SCAN/`*`service_name`*.

Server-side work, such as Oracle Scheduler, Parallel Query, and Oracle GoldenGate queues, set the service name as part of the workload definition. For Oracle Scheduler, jobs are assigned to job classes, and job classes run within services. For Parallel Query and Parallel DML, the query coordinator connects to a service, and the parallel query workers inherit the service for the duration of the parallel processing. For Oracle GoldenGate, streams queues are accessed using services. Work running under a service inherits the thresholds and attributes for the service and is measured as part of the service.

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

When users or applications connect to a database, Oracle recommends that you use a service that is specified in the `CONNECT_DATA` portion of the connect string. Oracle AI Database automatically creates one database service when the database is created but the behavior of this service is different from that of database services that you subsequently create. To enable more flexibility in the management of a workload that uses the database, Oracle AI Database enables you to create multiple services and specify on which instances the services start. If you are interested in greater workload management flexibility, then continue reading this chapter to understand the added features that you can use with services.

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

Dynamic Database Services

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

- **Services**: Oracle AI Database provides a powerful automatic workload management facility, called services, to enable the enterprise grid vision. Services are entities that you can define in Oracle Real Application Clusters (Oracle RAC) databases that enable you to group database workloads, route work to the optimal instances that are assigned to offer the service, and achieve high availability for planned and unplanned actions.
- **High Availability Framework**: An Oracle RAC component that enables Oracle AI Database to always maintain components in a running state.
- **Fast Application Notification (FAN)**: Provides information to Oracle RAC applications and clients about cluster state changes and Load Balancing Advisory events, such as `UP` and `DOWN` events for instances, services, or nodes. FAN has two methods for publishing events to clients, the Oracle Notification Service daemon, which is used by Java Database Connectivity (JDBC) clients including the Oracle Application Server, and Oracle GoldenGate Advanced Queueing, which is only used by previous releases of Oracle Call Interface (OCI) and Oracle Data Provider for .NET (ODP.NET) clients.

  > **Note**
  >
  > All Oracle Database clients use Oracle Notification Service.

- **Transaction Guard**: Provides a protocol and an API for at-most-once running of transactions in case of unplanned outages and duplicate submissions.
- **Application Continuity**: Provides a general purpose infrastructure that replays an in-flight request when a recoverable error is received, masking many system, communication, and storage outages, and hardware failures. Unlike existing recovery technologies, this feature attempts to recover the transactional and non-transactional session states beneath the application, so that the outage appears to the application as a delayed processing.
- **Connection Load Balancing**: An Oracle Net Services feature that balances incoming connections across all of the instances that provide the requested database service.
- **Load Balancing Advisory**: Provides information to applications about the current service levels that the database and its instances provide. The load balancing advisory makes recommendations to applications about where to direct application requests to obtain the best service based on the management policy that you have defined for that service. Load balancing advisory events are published through Oracle Notification Service.
- **Automatic Workload Repository (AWR)**: Tracks service-level statistics as metrics. You can create server-generated alerts for these metrics when the statistics exceed or fail to meet certain thresholds.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- **Fast Connection Failover (FCF)**: Enables Oracle Clients to provide rapid failover of connections by subscribing to FAN events.
- **Runtime Connection Load Balancing**: Enables Oracle Clients to provide intelligent allocations of connections in the connection pool, based on the current service level provided by the database instances when applications request a connection to complete some work.
- **Single Client Access Name (SCAN)**: Provides a single name to clients that connect to Oracle RAC that do not change throughout the life of a cluster, even if you add or remove nodes from the cluster. Clients connecting with SCAN can use a straightforward connection string, such as a thin JDBC URL or EZConnect, and still achieve load balancing and client connection failover goals.

You can deploy Oracle RAC and non-cluster Oracle AI Database environments to use dynamic database service features in many different ways. Depending on the number of nodes and your environment complexity and objectives, your choices for optimal automatic workload management and high-availability configuration depend on several considerations, which are explained in the Automatic Workload Management topics.

Related Topics

- *Oracle AI Database Administrator's Guide*
- Workload Management with Dynamic Database Services
  Workload management includes load balancing, enabling clients for Oracle Real Application Clusters (Oracle RAC), distributed transaction processing, and services.

Overview of Blocker Resolver

Blocker Resolver is an Oracle AI Database feature that automatically detects and resolves system delays.

> **Note**
>
> Starting with Oracle AI Database, Oracle Quality of Service Management (QoSM, or QoS Management) is desupported.

Blocker Resolver initially identified system delays and then dumped the relevant information about the delay into a trace file. Blocker Resolver can take action on and attempt to resolve the system delay. Blocker Resolver also runs in both single-instance and Oracle RAC database instances.

Blocker Resolver functions, as follows:

- First detects a system delay and then analyzes the delay and verifies the cause of the delay. It then applies heuristics to decide on a course of action to resolve the delay.
- Automates the tasks that used to require manual steps by a DBA to provide the trace files to My Oracle Support so that someone there could identify the source of the delay, minimizing or eliminating database and application downtime.
- Periodically scans all processes and analyzes a smaller subset of processes that are holding resources in successive scans. Blocker Resolver ignores processes if there is nothing waiting on the resource.
- Considers cross-instance delays, which are delays where the holder is a database process waiting on a response from an Oracle ASM instance.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Terminates the holder process so the next process waiting on that resource can move on and prevent a delay.
- Notifies a DBA with an ORA-32701 error message in the alert log.

Overview of Database In-Memory and Oracle RAC

Every Oracle RAC node has its own In-Memory (IM) column store. By default, populated objects are distributed across all IM column stores in the cluster.

Oracle recommends that you size the IM column stores equally on every Oracle RAC node. If an Oracle RAC node does not require an IM column store, then set the `INMEMORY_SIZE` parameter to `0`.

Starting with Oracle Database 19c, Database In-Memory has a new Base Level feature that allows you to use Database In-Memory with up to a 16 GB column store without requiring the Database In-Memory option. In an Oracle RAC database, the `INMEMORY_SIZE` setting in each database instance must not exceed 16 GB. Set the `INMEMORY_FORCE` parameter to `BASE_LEVEL` to enable this feature.

It is possible to have completely different objects populated on every node, or to have larger objects distributed across all of the IM column stores in the cluster. On Oracle Engineered Systems, it is also possible for the same objects to appear in the IM column store on every node. The distribution of objects across the IM column stores in a cluster is controlled by two subclauses to the `INMEMORY` attribute: `DISTRIBUTE` and `DUPLICATE`.

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

On an Oracle Engineered System, you can duplicate or mirror populated objects across the IM column stores in the cluster. This technique provides the highest level of redundancy. The `DUPLICATE` clause controls how an object is duplicated. If you specify only `DUPLICATE`, then one mirrored copy of the data is distributed across the IM column stores in the cluster. To duplicate the entire object in each IM column store, specify `DUPLICATE ALL`.

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

Overview of Managing Oracle RAC Environments

When managing Oracle RAC, there are many considerations, such as the deployment type, the tools to use, how to monitor the system, and how to evaluate performance.

- About Designing and Deploying Oracle RAC Environments
  Any enterprise that is designing and implementing a high availability strategy with Oracle RAC must begin by performing a thorough analysis of the business drivers that require high availability.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- About Administrative Tools for Oracle RAC Environments
  You administer a cluster database as a single-system image using the Server Control Utility (SRVCTL), Oracle Enterprise Manager, SQL*Plus, and other utilities.
- About Monitoring Oracle RAC Environments
  Web-based Oracle Enterprise Manager Cloud Control enables you to monitor an Oracle RAC database.
- About Evaluating Performance in Oracle RAC Environments
  You do not need to perform special tuning for Oracle RAC; Oracle RAC scales without special configuration changes.

About Designing and Deploying Oracle RAC Environments

Any enterprise that is designing and implementing a high availability strategy with Oracle RAC must begin by performing a thorough analysis of the business drivers that require high availability.

An analysis of business requirements for high availability combined with an understanding of the level of investment required to implement different high availability solutions enables the development of a high availability architecture that achieves both business and technical objectives.

Related Topics

- Design and Deployment Techniques
  Learn about methods to design and deploy Oracle RAC.

> **See Also**
>
> For help choosing and implementing the architecture that best fits your availability requirements:
>
> - "Design and Deployment Techniques" provides a high-level overview you can use to evaluate the high availability requirements of your business.
> - *Oracle AI Database High Availability Overview and Best Practices* describes how to select the most suitable architecture for your organization, describes several high availability architectures, and provides guidelines for choosing the one that best meets your requirements, and also provides information about the Oracle Maximum Availability Architecture

About Administrative Tools for Oracle RAC Environments

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

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

  You can use Oracle Enterprise Manager Cloud Control to also manage Oracle RAC One Node databases.
- **SQL*Plus**: SQL*Plus commands operate on the current instance. The current instance can be either the local default instance on which you initiated your SQL*Plus session, or it can be a remote instance to which you connect with Oracle Net Services.
- **Configuration Verification Utility (CVU)**: CVU is a command-line tool that you can use to verify a range of cluster and Oracle RAC components, such as shared storage devices, networking configurations, system requirements, and Oracle Clusterware, in addition to operating system groups and users. You can use CVU for preinstallation checks and for postinstallation checks of your cluster environment. CVU is especially useful during preinstallation and during installation of Oracle Grid Infrastructure and Oracle RAC components. Oracle Universal Installer runs CVU after installing Oracle Clusterware and Oracle AI Database to verify your environment.

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

  > **Note**
  >
  > Starting with Oracle Grid Infrastructure 26ai, Cluster Verification Utility (CVU) is renamed to Configuration Verification Utility (CVU).
- **Oracle DBCA**: The recommended utility for creating and initially configuring Oracle RAC, Oracle RAC One Node, and Oracle noncluster databases.
- **NETCA**: Configures the network for your Oracle RAC environment.

Related Topics

- *Oracle Clusterware Administration and Deployment Guide*
- Administering Database Instances and Cluster Databases
  This chapter describes how to administer Oracle Real Application Clusters (Oracle RAC) databases and database instances.
- Overview of Monitoring Oracle RAC and Oracle Clusterware
  Learn about the monitoring capabilities of Oracle Enterprise Manager, including the Cluster Database Homepage, the Interconnects page, and the Cluster Database Performance page.
- Server Control Utility Reference
  Use the Server Control Utility (SRVCTL) to manage Oracle Real Application Clusters (Oracle RAC) configuration information.
- *Oracle Clusterware Administration and Deployment Guide*
- *Oracle AI Database Net Services Administrator's Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

See Also

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

About Monitoring Oracle RAC Environments

Web-based Oracle Enterprise Manager Cloud Control enables you to monitor an Oracle RAC database.

Oracle Enterprise Manager Cloud Control is a central point of control for the Oracle environment that you access by way of a graphical user interface (GUI). See "Monitoring Oracle RAC and Oracle Clusterware" for more information about using Oracle Enterprise Manager to monitor Oracle RAC environments.

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

- Use Oracle Enterprise Manager Cloud Control to initiate cluster database management tasks.
- Use Oracle Enterprise Manager Cloud Control to administer multiple or individual Oracle RAC databases.
- Use the global views (`GV$` views), which are based on `V$` views. The `catclustdb.sql` script creates the `GV$` views. Run this script if you do not create your database with Oracle DBCA. Otherwise, Oracle DBCA runs this script for you.

  For almost every `V$` view, there is a corresponding global `GV$` view. In addition to the `V$` information, each `GV$` view contains an extra column named `INST_ID`, which displays the instance number from which the associated `V$` view information was obtained.
- Use the sophisticated management and monitoring features of the Oracle Database Diagnostic and Tuning packs within Oracle Enterprise Manager that include the Automatic Database Diagnostic Monitor (ADDM) and Automatic Workload Repository (AWR).

Note

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

Related Topics

- Overview of Monitoring Oracle RAC and Oracle Clusterware
  Learn about the monitoring capabilities of Oracle Enterprise Manager, including the Cluster Database Homepage, the Interconnects page, and the Cluster Database Performance page.
- *Oracle AI Database Performance Tuning Guide*

About Evaluating Performance in Oracle RAC Environments

You do not need to perform special tuning for Oracle RAC; Oracle RAC scales without special configuration changes.

If your application performs well on a noncluster Oracle AI Database, then it will perform well in an Oracle RAC environment. Many of the tuning tasks that you would perform on a noncluster Oracle AI Database can also improve Oracle RAC database performance. This is especially true if your environment requires scalability across a greater number of CPUs.

Some of the performance features specific to Oracle RAC include:

- Dynamic resource allocation
  - Oracle AI Database dynamically allocates Cache Fusion resources as needed
  - The dynamic acquiring of resources improves performance by keeping resources local to data blocks
- Cache Fusion enables a simplified tuning methodology
  - You do not have to tune any parameters for Cache Fusion
  - No application-level tuning is necessary
  - You can use a bottom-up tuning approach with virtually no effect on your existing applications
- More detailed performance statistics
  - More views for Oracle RAC performance monitoring
  - Oracle RAC-specific performance views in Oracle Enterprise Manager

Copyright © 1999, 2025, Oracle and/or its affiliates.

2

Administering Storage in Oracle RAC

Oracle recommends Oracle Automatic Storage Management (Oracle ASM) as a storage management solution that provides an alternative to conventional volume managers, file systems, and raw devices.

- About Oracle ASM
  Oracle Automatic Storage Management (Oracle ASM) is a volume manager and a file system for Oracle AI Database files that supports single-instance Oracle AI Database and Oracle Real Application Clusters (Oracle RAC) configurations.
- Overview of Storage Management for Oracle RAC
  All data files (including an undo tablespace for each instance) and redo log files (at least two for each instance) for an Oracle RAC database must reside on shared storage.
- Data File Access in Oracle RAC
- NFS Server for Storage
  An Oracle AI Database can serve as a network file system (NFS) server. The database responds to NFS requests from any NFS client and stores both the files and their metadata within the database.
- Redo Log File Storage in Oracle RAC
  Learn about redo log requirements for Oracle Real Application Clusters (Oracle RAC) databases.
- Automatic Undo Management in Oracle RAC
  Oracle AI Database automatically manages undo segments within a specific undo tablespace that is assigned to an instance.
- Oracle Automatic Storage Management with Oracle RAC
  Oracle Automatic Storage Management (Oracle ASM) automatically maximizes I/O performance by managing the storage configuration across the disks that Oracle ASM manages.

About Oracle ASM

Oracle Automatic Storage Management (Oracle ASM) is a volume manager and a file system for Oracle AI Database files that supports single-instance Oracle AI Database and Oracle Real Application Clusters (Oracle RAC) configurations.

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

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

The Oracle ASM volume manager functionality provides flexible server-based mirroring options. The Oracle ASM normal and high redundancy disk groups enable two-way and three-

Copyright © 1999, 2025, Oracle and/or its affiliates.

way mirroring respectively. You can use external redundancy to enable a Redundant Array of Independent Disks (RAID) storage subsystem to perform the mirroring protection function.

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

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

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

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

Related Topics

- Administering Oracle ASM Disk Groups
- Oracle ASM Instances and Disk Groups

Overview of Storage Management for Oracle RAC

All data files (including an undo tablespace for each instance) and redo log files (at least two for each instance) for an Oracle RAC database must reside on shared storage.

Oracle recommends that you use Oracle ASM to store these files in an Oracle ASM disk group.

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

> **Note**
>
> Oracle AI Database and related technologies, such as Oracle Clusterware, no longer support the use of raw (block) storage devices.

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

Related Topics

- Overview of Installing Oracle AI Database Software and Creating a Database
- Introducing Oracle Automatic Storage Management
- Oracle AI Database Structure and Storage

Copyright © 1999, 2025, Oracle and/or its affiliates.

Data File Access in Oracle RAC

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

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

NFS Server for Storage

An Oracle AI Database can serve as a network file system (NFS) server. The database responds to NFS requests from any NFS client and stores both the files and their metadata within the database.

Files associated with a primary database, such as SQL scripts, can be automatically replicated on a standby database. You can also store unstructured data, such as emails, on the database.

You can create or destroy an Oracle file system and access it though the NFS server using the procedures documented in *Oracle AI Database SecureFiles and Large Objects Developer's Guide*.

Related Topics

- *Oracle AI Database SecureFiles and Large Objects Developer's Guide*
- *Oracle AI Database PL/SQL Packages and Types Reference*
- *Oracle AI Database Reference*

Redo Log File Storage in Oracle RAC

Learn about redo log requirements for Oracle Real Application Clusters (Oracle RAC) databases.

In an Oracle RAC database, each instance must have at least two groups of redo log files. When you use Oracle DBCA to create the database, Oracle DBCA allocates redo log files to instances, as required, automatically. You can change the number of redo log groups and the size of the redo log files as required either during the initial database creation or as a post-creation step. If you add a node to the cluster, then the `addNode` script configures the redo logs on the new server.

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

Related Topics

- About Designing and Deploying Oracle RAC Environments
  Any enterprise that is designing and implementing a high availability strategy with Oracle RAC must begin by performing a thorough analysis of the business drivers that require high availability.
- Creating Redo Log Groups and Members
- ALTER DATABASE

Automatic Undo Management in Oracle RAC

Oracle AI Database automatically manages undo segments within a specific undo tablespace that is assigned to an instance.

Instances can always read all undo blocks throughout the cluster environment for consistent read purposes. Also, any instance can update any undo tablespace during transaction recovery, if that undo tablespace is not currently used by another instance for undo generation or transaction recovery.

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

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

Related Topics

- Setting SPFILE Parameter Values for Oracle RAC
  You can change SPFILE settings with Oracle Enterprise Manager or by using the `SET` clause of the `ALTER SYSTEM` statement.
- Guidelines for Managing Tablespaces

Oracle Automatic Storage Management with Oracle RAC

Oracle Automatic Storage Management (Oracle ASM) automatically maximizes I/O performance by managing the storage configuration across the disks that Oracle ASM manages.

Oracle ASM does this by evenly distributing the database files across all of the available storage assigned to the disk groups within Oracle ASM. Oracle ASM allocates your total disk space requirements into uniformly sized units across all disks in a disk group. Oracle ASM can also automatically mirror files to prevent data loss. Because of these features, Oracle ASM also significantly reduces your administrative overhead.

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

- Storage Management in Oracle RAC
  You can create Oracle ASM disk groups and configure mirroring for Oracle ASM disk groups using the Oracle ASM configuration assistant (ASMCA).
- Modifying Disk Group Configurations for Oracle ASM
  When you create a disk group for a cluster or add new disks to an existing clustered disk group, prepare the underlying physical storage on shared disks and give the Oracle user permission to read and write to the disk.
- Oracle ASM Disk Group Management
  To use Oracle ASM, you must first create disk groups with ASMCA before creating a database with Oracle DBCA.
- Configuring Preferred Mirror Read Disks in Extended Distance Clusters
  You can configure preferred read disks to improve performance.
- Converting Nonclustered Oracle ASM to Clustered Oracle ASM
- Administering Oracle ASM Instances with SRVCTL in Oracle RAC
  You can use the Server Control Utility (SRVCTL) to add or remove an Oracle ASM instance.

Related Topics

- Introducing Oracle Automatic Storage Management

Storage Management in Oracle RAC

You can create Oracle ASM disk groups and configure mirroring for Oracle ASM disk groups using the Oracle ASM configuration assistant (ASMCA).

Alternatively, you can use Oracle Enterprise Manager to administer Oracle ASM disk groups after you have discovered the respective servers with Oracle Enterprise Manager.

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

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

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

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

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

Related Topics

- *Oracle Clusterware Administration and Deployment Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

Modifying Disk Group Configurations for Oracle ASM

When you create a disk group for a cluster or add new disks to an existing clustered disk group, prepare the underlying physical storage on shared disks and give the Oracle user permission to read and write to the disk.

The shared disk requirement is the only substantial difference between using Oracle ASM with an Oracle RAC database compared to using it with a noncluster Oracle AI Database. Oracle ASM automatically redistributes the data files after you add or delete a disk or disk group.

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

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

Related Topics

- *Oracle Automatic Storage Management Administrator's Guide*

Oracle ASM Disk Group Management

To use Oracle ASM, you must first create disk groups with ASMCA before creating a database with Oracle DBCA.

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

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

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

Related Topics

- *Oracle Automatic Storage Management Administrator's Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

Configuring Preferred Mirror Read Disks in Extended Distance Clusters

You can configure preferred read disks to improve performance.

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

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

Related Topics

- Preferred Read Failure Groups
- ASM_PREFERRED_READ_FAILURE_GROUPS

Converting Nonclustered Oracle ASM to Clustered Oracle ASM

When installing Oracle Grid Infrastructure, any nonclustered Oracle Automatic Storage Management (Oracle ASM) instances are automatically converted to clustered Oracle ASM.

Related Topics

- Preferred Read Failure Groups

Administering Oracle ASM Instances with SRVCTL in Oracle RAC

You can use the Server Control Utility (SRVCTL) to add or remove an Oracle ASM instance.

To issue SRVCTL commands to manage Oracle ASM, log in as the operating system user who owns the Oracle Grid Infrastructure home and issue the SRVCTL commands from the bin directory of the Oracle Grid Infrastructure home.

Use the following syntax to add an Oracle ASM instance:

```
srvctl add asm
```

Use the following syntax to remove an Oracle ASM instance:

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

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

Use the following syntax to start an Oracle ASM instance:

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


Copyright © 1999, 2025, Oracle and/or its affiliates.

Use the following syntax to stop an Oracle ASM instance:

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

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

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

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

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

Related Topics

- Server Control Utility Reference
  Use the Server Control Utility (SRVCTL) to manage Oracle Real Application Clusters (Oracle RAC) configuration information.
- *Oracle Automatic Storage Management Administrator's Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

3

Administering Database Instances and Cluster Databases

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

> **Note**
>
> A multitenant Container Database (CDB) is the only supported architecture in Oracle Database 21c and later releases.

- Overview of Oracle RAC Database Administration
  Oracle RAC database administration requires certain privileges and administrative tasks can vary depending on the deployment model.
- Tools for Administering Oracle RAC
  The tools most commonly used to managed Oracle Real Application Clusters (Oracle RAC) databases and instances are the SRVCTL utility, Oracle Enterprise Manager, and SQL*Plus.
- Starting and Stopping Instances and Oracle RAC Databases
  You can start and stop instances with Oracle Enterprise Manager, SQL*Plus, or SRVCTL.
- Starting and Stopping PDBs in Oracle RAC
  You can use SRVCTL commands to manage PDBs.
- Local Rolling Database Maintenance
  Starting with Oracle AI Database 26ai, you can patch Oracle Real Application Clusters (Oracle RAC) and Oracle RAC One Node databases locally in out-of-place mode without affecting availability and workload of the database.
- Pluggable Database Rank
  The PDB `-rank` parameter defines relative importance of the PDBs, which are created specifying cardinality, in a database with the `RANK` management policy.
- Pluggable Database Placement
  Configure PDBs to either run explicitly in the specified CDB instances or run dynamically in any CDB or a subset of CDBs in the cluster.
- Example of Creating a Pluggable Database with Cardinality and Rank
  You can use these examples to see how to create an Oracle AI Database and create a Pluggable Database, specifying cardinality, in the Oracle AI Database.
- Reducing Downtime During Database and Instance Outages
  Outages can be either planned (maintenance) or unplanned. You can use features to help minimize both types of outages.
- Oracle RAC High Availability Best Practices
  Implement Oracle Real Application Clusters (Oracle RAC) best practices to achieve minimal downtime for patching and reconfiguration.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Verifying That Instances are Running
  To verify that a database instance is available, use Oracle Enterprise Manager, SRVCTL, or SQL*Plus.
- Terminating Sessions On a Specific Cluster Instance
  You can use the `ALTER SYSTEM KILL SESSION` statement to terminate a session on a specific instance.
- Overview of Initialization Parameter Files in Oracle RAC
  The initialization parameters for an Oracle RAC database are stored in a SPFILE.
- Initialization Parameter Use in Oracle RAC
  By default, most parameters are set to a default value and this value is the same across all instances.
- Quiescing Oracle RAC Databases
  The procedure for quiescing Oracle RAC databases is identical to quiescing a noncluster database.
- Administering Multiple Cluster Interconnects on Linux and UNIX Platforms
  In Oracle RAC environments that run on Linux and UNIX platforms, you can use the `CLUSTER_INTERCONNECTS` initialization parameter to specify an alternative interconnect to the one Oracle Clusterware is using for the private network.
- Customizing How Oracle Clusterware Manages Oracle RAC Databases
  Use these examples to minimize Oracle Clusterware control over Oracle RAC databases, which you may need to do during upgrades.
- Advanced Oracle Enterprise Manager Administration
  You can install, configure, and monitor an Oracle Real Application Clusters (Oracle RAC) database from a single location using Oracle Enterprise Manager Cloud Control.

> **See Also**
>
> The Oracle Enterprise Manager Cloud Control online help for more information about Oracle Enterprise Manager Cloud Control.

Overview of Oracle RAC Database Administration

Oracle RAC database administration requires certain privileges and administrative tasks can vary depending on the deployment model.

The policy-managed database deployment option is desupported in Oracle AI Database.

- Required Privileges for Oracle RAC Database Administration
  Use the SYSRAC administrative privilege to manage Oracle RAC databases.
- Oracle RAC Database Deployment Models
  Starting with Oracle Database 21c, there is a single, merged management style for Oracle RAC databases.

Required Privileges for Oracle RAC Database Administration

Use the SYSRAC administrative privilege to manage Oracle RAC databases.

To increase security and further separate administrative duties, Oracle RAC database administrators manage Oracle RAC databases with the SYSRAC administrative privilege.

Copyright © 1999, 2025, Oracle and/or its affiliates.

The SYSRAC administrative privilege is the default mode of connecting to the database by the Oracle Clusterware agent on behalf of Oracle RAC utilities, such as SRVCTL.

Related Topics

- *Oracle AI Database Security Guide*

Oracle RAC Database Deployment Models

Starting with Oracle Database 21c, there is a single, merged management style for Oracle RAC databases.

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

Prior to Oracle Database 21c, Oracle RAC databases support two different management styles and deployment models: administrator-managed deployment and policy-managed deployment.

Administrator-managed deployment requires that you statically configure each database instance to run on a specific node in the cluster. This deployment also requires that you configure database services to run on specific instances that belong to a particular database using the `preferred` and `available` designation.

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

Starting with Oracle Database 21c, the two management styles are merged into a single deployment model that combines the best features of each model. The administrator-managed database deployment style now has additional capabilities that were previously available only in policy-managed databases. These enhancements result in a new, converged deployment style. To use the merged database management style, you must have a Container Database (CDB) with at least one Pluggable Database (PDB).

You manage the merged database deployment model using the same commands and utilities (such as SRVCTL, Oracle DBCA, or Oracle Enterprise Manager) as before. All commands and utilities, except for the policy-management specific commands such as `srvpool` commands, maintain backward compatibility to support the management of databases prior to Oracle Database 21c.

The merged database management style simplifies management of dynamic systems. The clusters and databases can expand or shrink as requirements change. The Oracle home software must be installed on every node in the cluster.

Benefits of the merged management style

The PDBs in the cluster database are available on all nodes, or a subset of the nodes, based on the cardinality setting for the PDB. The *cardinality* of a PDB governs the number of nodes where a PDB can run at the same time. If you use a number for cardinality instead of `ALL`, then the instances in which the PDBs are opened depend on the resources available to each instance.

A database instance is started on every server in the cluster that hosts a PDB. If you are using Oracle Automatic Storage Management (Oracle ASM) with Oracle Managed Files for your

Copyright © 1999, 2025, Oracle and/or its affiliates.

database storage, then, when an instance starts and there is no redo thread available, Oracle RAC automatically enables one and creates the required redo log files and undo tablespace.

Clients can connect to a PDB using the same SCAN-based connect string irrespective of which servers the PDBs are running on at the time.

Tools for Administering Oracle RAC

The tools most commonly used to managed Oracle Real Application Clusters (Oracle RAC) databases and instances are the SRVCTL utility, Oracle Enterprise Manager, and SQL*Plus.

In many cases, you use these tools the same way to manage Oracle RAC environments as you would use them manage noncluster Oracle AI Database:

- Administering Oracle RAC with SRVCTL
  The Server Control Utility (SRVCTL) is a command-line interface that you can use to manage Oracle AI Databases in a centralized manner.
- Administering Oracle RAC with Oracle Enterprise Manager
  Oracle Enterprise Manager provides a central point of control for the Oracle RAC environment, allowing you to perform administrative tasks simultaneously on multiple cluster databases.
- Administering Oracle RAC with SQL*Plus
  Unlike SRVCTL or Oracle Enterprise Manager, SQL*Plus is an instance-oriented management tool.
- How SQL*Plus Commands Affect Instances
  You can use SQL*Plus to start and stop instances in the Oracle RAC database.

Administering Oracle RAC with SRVCTL

The Server Control Utility (SRVCTL) is a command-line interface that you can use to manage Oracle AI Databases in a centralized manner.

SRVCTL-based database management available for single-instance Oracle AI Database, using Oracle ASM in the Oracle Grid Infrastructure, for both a noncluster environment and Oracle RAC databases, based on Oracle Grid Infrastructure for a cluster. This enables homogeneous management of all Oracle AI Database types using SRVCTL. You can use SRVCTL to start and stop the database and instances, and to delete or move instances and services. You can also use SRVCTL to add services and manage configuration information, in addition to other resources in the cluster.

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

Related Topics

- Server Control Utility Reference
  Use the Server Control Utility (SRVCTL) to manage Oracle Real Application Clusters (Oracle RAC) configuration information.

Administering Oracle RAC with Oracle Enterprise Manager

Oracle Enterprise Manager provides a central point of control for the Oracle RAC environment, allowing you to perform administrative tasks simultaneously on multiple cluster databases.

Based on the Oracle Enterprise Manager Cloud Control (Grid Control in Oracle Enterprise Manager 11*g*) graphical user interface (GUI), you can manage both non-clustered and Oracle RAC environments.

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

Related Topics

- Advanced Oracle Enterprise Manager Administration
  You can install, configure, and monitor an Oracle Real Application Clusters (Oracle RAC) database from a single location using Oracle Enterprise Manager Cloud Control.

Administering Oracle RAC with SQL*Plus

Unlike SRVCTL or Oracle Enterprise Manager, SQL*Plus is an instance-oriented management tool.

SQL*Plus commands operate on the current instance. The current instance can be either the local default instance on which you initiated your SQL*Plus session, or it can be a remote instance to which you connect with Oracle Net Services. For an Oracle RAC environment that runs multiple instances on one database at the same time, this implies that you need to consider the extent to which SQL*Plus can operate on this instance.

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

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

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

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

Connecting as `SYSOPER` or `SYSRAC` enables you to perform privileged operations, such as instance startup and shutdown. Multiple SQL*Plus sessions can connect to the same instance at the same time. SQL*Plus automatically disconnects you from the first instance whenever you connect to another one.

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

Related Topics

- Authentication for Accessing Oracle ASM Instances
- Configuring Naming Methods
- Modifying a PDB with the ALTER PLUGGABLE DATABASE Statement

How SQL*Plus Commands Affect Instances

You can use SQL*Plus to start and stop instances in the Oracle RAC database.

Most SQL statements affect the current instance. You do not need to run SQL*Plus commands as `root` on Linux and UNIX systems or as `Administrator` on Windows systems. You need only the proper database account with the privileges that you normally use for a noncluster Oracle AI 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.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- `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.

The following table describes how SQL*Plus commands affect instances.

Table 3-1 How SQL*Plus Commands Affect Instances

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

Starting and Stopping Instances and Oracle RAC Databases

You can start and stop instances with Oracle Enterprise Manager, SQL*Plus, or SRVCTL.

Both Oracle Enterprise Manager and SRVCTL provide options to start and stop all of the instances in an Oracle Real Application Clusters (Oracle RAC) database with a single step.

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

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

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

- Starting One or More Instances and Oracle RAC Databases Using SRVCTL
  Use SRVCTL start Oracle RAC databases and instances.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Stopping One or More Instances and Oracle RAC Databases Using SRVCTL
  Use SRVCTL to stop instances and Oracle RAC databases.
- Stopping All Databases and Instances Using CRSCTL
  You can use the `crsctl stop crs` command on the node or the `crsctl stop cluster -all` command to stop all instances on a node or the entire cluster.
- Starting and Stopping Individual Instances Using SQL*Plus
  If you want to start or stop only one instance and you are connected to your local node, then you must first ensure that your current environment includes the SID for the local instance.

Related Topics

- Overview of Database Instance Startup and Shutdown

Starting One or More Instances and Oracle RAC Databases Using SRVCTL

Use SRVCTL start Oracle RAC databases and instances.

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

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

- To start your entire cluster database, that is, all of the instances and its dependencies, enter the following SRVCTL command:

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

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

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

- To start specific instances of a database, enter a comma-delimited list of instance names:

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

  You must enclose a comma-delimited list in double quotation marks ("").

-

- To start an instance of a database on a specific node, use the following command with a single node name:

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


Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.
> Policy-managed databases were deprecated in Oracle Database 21c, and admin-managed database deployment was enhanced with functions similar to policy managed databases. By converging the automation provided by policy-managed database with the consistency of an admin-managed database, Oracle seeks to simplify database management tasks for database administrators. This converged database deployment provides the best of both options, such as providing the options to rank and define the order of database startup, without requiring you to choose a specific style during deployment.

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

Related Topics

- Server Control Utility Reference
  Use the Server Control Utility (SRVCTL) to manage Oracle Real Application Clusters (Oracle RAC) configuration information.

Stopping One or More Instances and Oracle RAC Databases Using SRVCTL

Use SRVCTL to stop instances and Oracle RAC databases.

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

- In Oracle RAC, shutting down one instance does not interfere with the operation of other running instances.
- To shut down an Oracle RAC database completely, shut down every instance that has the database open or mounted.
- After a `NORMAL` or `IMMEDIATE` shutdown, instance recovery is not required. Recovery is required, however, after you issue the `SHUTDOWN ABORT` command or after an instance terminates unusually. An instance that is still running performs instance recovery for the instance that shut down. If no other instances are running, the next instance to open the database performs instance recovery for any instances needing it.
- Using the `SHUTDOWN TRANSACTIONAL` command with the `LOCAL` option is useful to shut down a particular Oracle RAC database instance. Transactions on other instances do not block this operation. If you omit the `LOCAL` option, then this operation waits until transactions on all other instances that started before you ran the `SHUTDOWN` command either commit or rollback, which is a valid approach, if you intend to shut down all instances of an Oracle RAC database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

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

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

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

  Use the `TRANSACTIONAL` stop option with the `srvctl stop database` command and the `TRANSACTIONAL LOCAL` stop option with the `srvctl stop instance` command.

- To stop all instances and their enabled services that are managed by Oracle Clusterware on one or more nodes, enter the following SRVCTL command:

  ```
  $ srvctl stop instance -node "node_list" [-stopoption stop_options]
  ```

  > **Note**
  >
  > If any of the instances has services running on it, then you must use the `-force` option to stop the services and the instance.

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

  ```
  $ srvctl stop instance -db db_unique_name {-node "node_list" | -instance
  "inst_name_list"}
     [-stopoption stop_options]
  ```

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

  The `srvctl stop instance` command also stops the services related to the terminated instances on the nodes where the instances were running. As an example, the following command shuts down the two instances, `orcl3` and `orcl4`, on the `orcl` database using the `failover` option for CRS to find another node to run the services from it and the `immediate` stop option:

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

Related Topics

- srvctl stop database
  Stops a database, its instances, and its services.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- srvctl stop instance
  The `srvctl stop instance` command stops instances and stops any services running on specified instances.
- Overview of Database and Instance Shutdown
- Shutting Down a Database

Stopping All Databases and Instances Using CRSCTL

You can use the `crsctl stop crs` command on the node or the `crsctl stop cluster -all` command to stop all instances on a node or the entire cluster.

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

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

Starting and Stopping Individual Instances Using SQL*Plus

If you want to start or stop only one instance and you are connected to your local node, then you must first ensure that your current environment includes the SID for the local instance.

Note that any subsequent commands in your session, whether inside or outside a SQL*Plus session, are associated with that same SID.

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

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

```
CONNECT / AS SYSRAC
STARTUP MOUNT
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

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

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

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

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

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

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

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

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

Related Topics

- *Oracle Automatic Storage Management Administrator's Guide*
- *SQL*Plus User's Guide and Reference*

Starting and Stopping PDBs in Oracle RAC

You can use SRVCTL commands to manage PDBs.

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> Starting with Oracle Database 21c, installation of non-CDB Oracle Database architecture is no longer supported. The policy-managed database deployment option is desupported in Oracle AI Database.

Starting with Oracle Database 21c, PDBs are a resource managed by Oracle Clusterware. Consider a admin-managed CDB called `raccont` that has a PDB called `spark`.

> **Note**
>
> If you attempt to create the service without first creating the PDB, then you will get an error message indicating you must create the PDB resource first.

If the `spark` PDB was created with cardinality set to 1, or 2, or ALL, then if you create a service named `plug` for the PDB, the service can use the `–cardinality` argument, too. If the `spark` PDB was created without specifying the `-cardinality` argument, then new services you create for the PDB use the `-preferred` or `-available` arguments, not the `–cardinality` argument.

Because PDBs are managed as an Oracle Clusterware resource, typical Oracle RAC-based management practices apply. For this reason, if the PDB `spark` has `AUTOMATIC` management policy, then the PDB is started when CDB starts. Similarly, if the PDB `spark` is in the online state when Oracle Clusterware is shut down on a server hosting this service and the management policy is set to `MANUAL`, then the PDB is restored to its original state after the restart of Oracle Clusterware on this server. The default PDB management policy is derived from the management policy of its CDB.

To start a Pluggable Database:

```
$ srvctl start pdb -db db_name -pdb pdb_name [-startoption start_options]
```

To start a Pluggable Database on specific nodes:

```
$ srvctl start pdb -db db_name -pdb pdb_name -node node_list
   [-startoption start_options]
```

To stop a PDB and all its services on all nodes within a database using the IMMEDIATE option:

```
$ srvctl stop pdb -db db_name -pdb pdb_name -stopoption IMMEDIATE -
drain_timeout 0
  -stopsvcoption IMMEDIATE
```

To stop a Pluggable Database on specific nodes:

```
$ srvctl stop pdb -db db_name -pdb pdb_name -node node_list
   [-stopoption stop_options] [-stopsvcoption stop_service_options
   [-drain_timeout timeout]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

If you do not want the `spark` PDB to restart when the Oracle RAC database is restarted on all, or on a specific node, use the following command:

```
srvctl disable pdb -db raccont -pdb spark [-node node_name]
```

To view the status of the PDB service `plug`, use the following command:

```
srvctl status service -db raccont -service plug -verbose
```

To view the status of the PDB `spark`, use the following command:

```
srvctl status pdb -db raccont -pdb spark -detail
```

To modify the configuration of the PDB, use the following command:

```
srvctl modify pdb -db db_unique_name -pdb pdb_name
     [-cardinality {num_of_instances | ALL}]
     [-maxcpu max_cpu_usage] [-mincpuunit min_cpu_usage]
     [-rank rank] [-startoption start_options]
     [-stopoption stop_options] [-policy policy]
```

> **Note**
>
> You can modify the `-cardinality` parameter only if you had set the `-cardinality` parameter when creating the PDB.

Related Topics

- Workload Management with Dynamic Database Services
  Workload management includes load balancing, enabling clients for Oracle Real Application Clusters (Oracle RAC), distributed transaction processing, and services.

Local Rolling Database Maintenance

Starting with Oracle AI Database 26ai, you can patch Oracle Real Application Clusters (Oracle RAC) and Oracle RAC One Node databases locally in out-of-place mode without affecting availability and workload of the database.

- About Local Rolling Database Maintenance
  Starting with Oracle AI Database 26ai, you can apply rolling patches and perform other maintenance operations locally for Oracle Real Application Clusters (Oracle RAC) and Oracle RAC One Node deployments.
- Requirements for Using Local Rolling Maintenance
  Learn what you need to do to configure and use local rolling database maintenance.
- Patching Oracle RAC Database in Local Rolling Mode
  You can patch Oracle Real Application Clusters (Oracle RAC) and Oracle RAC One Node databases in local rolling mode, reducing the time to migrate the connections.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- How to Recover from a Failed Transfer in Local Rolling Mode
  During local rolling database maintenance, some instances may fail to transfer to target Oracle home. Use any one of these procedures to recover from a failed transfer in local rolling mode.

About Local Rolling Database Maintenance

Starting with Oracle AI Database 26ai, you can apply rolling patches and perform other maintenance operations locally for Oracle Real Application Clusters (Oracle RAC) and Oracle RAC One Node deployments.

Local rolling creates and starts a second database instance from the second home on the same node, instead of stopping the database instance and relocating the workload to another instance on another node. Local rolling database maintenance reduces the time to migrate the connections. This feature reduces brownout time by keeping the workload on the local node. You must ensure that there are enough CPU, memory, and other computing resources to temporarily run two instances.

Local rolling ensures that at least the same number of Oracle RAC and Oracle RAC One Node instances are running in the cluster during rolling migration of the database between two Oracle homes. Using this feature, you can do out-of-place rolling patching within a single cluster node.

When you use local rolling out-of-place patching, Oracle RAC creates a second instance in the new Oracle home, starts the second instance, and then stops the first instance in the old Oracle home on the node you are patching. This feature also preserves the PDB placement and the services, and thus the node keeps doing the same work it was doing before patching.

Note

- The `srvctl modify database` command automatically creates a new instance with a unique instance name (`ORACL_SID`).
- The new instance name is the current instance name appended with an underscore (_) and a number. For example, if the current instance name is `sales_1`, then the new instance name could be `sales_2`. After the patching operation is complete, only the new instance runs on the patched node.
- When you perform rolling patching the second time, the original `ORACLE_SID` is restored. For example, the `ORACLE_SID` goes back to `sales_1` after the second rolling patching operation. The third rolling patching operation will use `sales_2`, and so on.

The local rolling out-of-place patching is optional in a multi-node environment. By default it is disabled. Use the `-localrolling` option with `srvctl modify database` to enable this feature.

Requirements for Using Local Rolling Maintenance

Learn what you need to do to configure and use local rolling database maintenance.

To take advantage of local rolling database maintenance capabilities in Oracle RAC, you must meet the following requirements:

- Configure Oracle Managed Files (OMF).
- Use the Server Parameters File (SPFILE).

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Reset the `THREAD` and `UNDO_TABLESPACE` initialization parameters after local rolling is complete.
- Do not run `srvctl add instance`, `srvctl remove instance`, or `srvctl modify instance` commands during local rolling patching.
- You must ensure that enough free storage space is available to create a new redo thread and a new undo tablespace for each instance. This feature creates a new redo thread and a new undo tablespace for each new instance when you use this feature for the first time. The second and subsequent time you use this feature, the previously used old redo thread and undo tablespace are used, and new redo and undo are not created.

Patching Oracle RAC Database in Local Rolling Mode

You can patch Oracle Real Application Clusters (Oracle RAC) and Oracle RAC One Node databases in local rolling mode, reducing the time to migrate the connections.

1. Download the Oracle AI Database installation image file (`db_home.zip`) and extract the image file into a new Oracle home directory.

   ```
   $ mkdir -p /u01/app/oracle/product/23.4.0/dbhome_1
   $ chgrp oinstall /u01/app/oracle/product/23.4.0/dbhome_1
   $ cd /u01/app/oracle/product/23.4.0/dbhome_1
   $ unzip -q /tmp/db_home.zip
   ```

2. From the new Oracle home directory, start the Oracle AI Database software installation and apply the required Release Updates (RUs).

   ```
   $ cd /u01/app/oracle/product/23.4.0/dbhome_1
   $ ./runInstaller -applyRU patch_directory_location
   ```

3. Prepare a new Oracle RAC or Oracle RAC One Node-enabled home.

   > **Note**
   >
   > You can also use Oracle FPP or any other method to install Oracle RAC software.

4. Enable local rolling for your database.

   ```
   $ srvctl modify database –db mydb -oraclehome new_Oracle_home -localrolling
   ```

   This command also generates a name for the new instance.

5. Transfer Oracle RAC and Oracle RAC One Node PDBs and services from the old Oracle home to the new Oracle home.

   ```
   $ srvctl transfer instance –d mydb [-node node_list]
   {[-stopoption stop_option] | -rollback} [-drain_timeout timeout] [-verbose]
   ```

   The new instance starts from the new Oracle home and waits for up to the time specified in the `-drain_timeout` parameter for the sessions to migrate to the new instance. The old instance stops when all the sessions are migrated to the new instance.

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> If `-drain_timeout` is not specified, then the maximum configured drain timeout of all the running services of the database on the specified nodes is used.

6. Verify the database configuration changes.

   ```
   $ srvctl config database -db mydb
   ```

7. Add or modify services and connect strings, as required.

Related Topics

- service Commands
  Use commands with the `service` keyword to add, modify, list the configuration of, enable, disable, start, stop, obtain the status of, relocate, and remove services.

How to Recover from a Failed Transfer in Local Rolling Mode

During local rolling database maintenance, some instances may fail to transfer to target Oracle home. Use any one of these procedures to recover from a failed transfer in local rolling mode.

1. Revert all instances from the target Oracle home to the old Oracle home.
   a. Revert the Oracle RAC local rolling configuration changes.

      ```
      srvctl modify database -db mydb -localrolling_revert
      ```

   b. Transfer all database instances to the old Oracle home.

      ```
      srvctl transfer instance -db mydb
      ```

      This procedure restores the database to the original state, before local rolling. All instances start running from the old Oracle home. You can delete the previous target Oracle home, create a new Oracle home and start local rolling procedure again.

2. Revert some instances from the target Oracle home to the old Oracle home.
   a. Revert the Oracle RAC local rolling configuration changes.

      ```
      srvctl modify database -db mydb -localrolling_revert
      ```

   b. Transfer some database instances to the old Oracle home.

      ```
      srvctl transfer instance -db mydb -node node_list
      ```

      Fix the local rolling target Oracle home on the transferred nodes. If any instances are still running from the local rolling target Oracle home, then the database goes in to the local rolling `REVERT` state.

   c. Modify the database to `TRANSFER` state using the same target Oracle home to transfer the instances.

      ```
      srvctl modify database -db mydb -localrolling -oraclehome
      target_Oracle_home
      ```


Copyright © 1999, 2025, Oracle and/or its affiliates.

3. Revert some instances from the target Oracle home to the old Oracle home.
   a. Rollback some instances from the target Oracle home to the old Oracle home.

      ```
      srvctl transfer instance -db mydb -node node_list -rollback
      ```

      The database goes into the `TRANSFER` state.
   b. Fix the target Oracle home.
   c. Transfer the database instance back to the target Oracle home.

      ```
      srvctl transfer instance -db mydb
      ```

Related Topics

- service Commands
  Use commands with the `service` keyword to add, modify, list the configuration of, enable, disable, start, stop, obtain the status of, relocate, and remove services.

Pluggable Database Rank

The PDB `-rank` parameter defines relative importance of the PDBs, which are created specifying cardinality, in a database with the `RANK` management policy.

The Pluggable Database (PDB) rank is a predefined value that you can assign to a PDB to specify workload importance of the PDB. Oracle Clusterware makes several decisions based on the rank of the PDB. By default, Oracle Clusterware has the same workload importance for each PDB. However, with the PDB `-rank` parameter, you can choose from a set of predefined values to distinguish workload importance of a PDB. The higher rank means the higher PDB workload importance, for example, the PDB rank 5 is the highest rank and 1 is the lowest rank.

The PDB `-rank` parameter is optional, and it is not set by default. You can configure it using the `srvctl modify pdb` command. When the `-rank` parameter is set, Oracle Clusterware gives precedence to the PDBs with ranks to perform the following operations:

- Determines the startup order of the PDBs in the cluster. Oracle Clusterware attempts to start PDBs with the highest rank before the other PDBs with the lower ranks.
- Shuts down a cluster database instance, with the `RANK` management policy, if there are no running PDBs that require that database instance.
- Decides whether to refuse starting the PDBs or stop running the PDBs when the cluster does not have sufficient resources to meet resource requirements of the PDBs with higher ranks when PDB's resource requirements are set to non-default values. The PDBs with non-default rank and resource requirement values have higher priority than the PDBs with default rank and resource requirement values.

  If a PDB in CDB1 has RANK 3 and a PDB in CDB2 has RANK 2, and if there are only enough resources to start only one CDB, then Oracle Clusterware starts CDB1 by dependency because the PDB in CDB1 has a higher rank. Oracle Clusterware does not start CDB2 because the PDB in CDB2 has a lower rank.

How PDB Rank Works?

If the PDB `-rank` parameter is defined, then Oracle Clusterware first considers PDB with the highest rank and then considers number of required CPUs while failing over the PDBs. For example, in a three node cluster with four CPUs in each node, and three PDBs named PDB1

Copyright © 1999, 2025, Oracle and/or its affiliates.

with RANK 1 and CPU count 4, PDB2 with RANK 2 and CPU count 4, and PDB3 with RANK 3 and CPU count 4, Oracle Clusterware handles failover in the following order:

1. The cluster has enough resources to start all PDBs. When the first node fails, PDB2 and PDB3 keep running, but Oracle Clusterware stops PDB1 because it has the lowest rank. If PDB2 and PDB3 were hosted on the failed node, then Oracle Clusterware stops PDB1 on the running nodes and starts PDB2 and PDB3 on these nodes.
2. When the second node fails, PDB3 keeps running, but Oracle Clusterware stops PDB2 because it has the lowest rank. If PDB3 was hosted on the failed node, then Oracle Clusterware stops PDB2 on the running node and starts PDB3 on that node.
3. PDBs can run from different CDBs. If you have multiple CDBs and any of these CDBs does not have a running PDB, then Oracle Clusterware shuts down the CDBs that do not have any running PDBs.

The rank of a PDB is set for the entire cluster, not only for the CDB in which you create a PDB.

Pluggable Database Placement

Configure PDBs to either run explicitly in the specified CDB instances or run dynamically in any CDB or a subset of CDBs in the cluster.

You can choose from the following two placement options for the PDBs:

- **Preferred and Available PDBs:** These PDBs can run only in the explicitly specified CDB instances that are running on the list of specified cluster nodes. While configuring such PDBs, you need to provide a CDB name and the list of instances or nodes where the CDB can run. You can modify the list of instances or nodes where the CDB can run.
- **Floating PDBs:** These PDBs are created with specified cardinality and they can run on any instance of the CDB in which they are created. The cardinality of a PDB governs the number of nodes where a PDB can run at the same time. If you use a number for cardinality instead of `ALL`, then which instances the PDBs are opened in depends on the available resources of each instance.

  Oracle Clusterware evaluates resources for each cluster database instance based on the values of the `-maxcpu` and `-mincpuunit` parameters for the PDB. You must be logged in as either the `grid` or the `root` user to modify the `-maxcpu`, `-mincpuunit`, and `-rank` parameters.

You can configure the PDB placement option either while creating a new PDB or by modifying an existing PDB.

Example of Creating a Pluggable Database with Cardinality and Rank

You can use these examples to see how to create an Oracle AI Database and create a Pluggable Database, specifying cardinality, in the Oracle AI Database.

You can define relative importance of Pluggable Databases in an Oracle AI Database using the `-rank` parameter. This option works for the Pluggable Databases that are created specifying cardinality in a database with the RANK management policy.

The following examples show you how to create an Oracle AI Database with the `RANK` management policy, add a Pluggable Database in the Oracle AI Database specifying cardinality, and modify the PDB cardinality.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example 3-1 Creating an Oracle AI Database Using SRVCTL

In this example, you create an Oracle AI Database `DATA` with the `RANK` management policy using SRVCTL:

```
$ srvctl add database -db db_unique_name -policy RANK -
oraclehome $ORACLE_HOME -dbname DATA
```

Example 3-2 Creating a Pluggable Database Using SRVCTL

In this example, you create a Pluggable Database `MYPDB`, specifying cardinality, in Oracle AI Database `DATA` using SRVCTL:

```
$ srvctl add pdb -db DATA -pdb MYPDB -cardinality 2
```

> **Note**
>
> If you set the management policy of your CDB to `RANK`, then the default policy of your PDBs that you create in that CDB will be set to `RESTART`.

Example 3-3 Checking PDB Configuration Using SRVCTL

In this example, you check configuration of Pluggable Database `MYPDB` using SRVCTL:

```
$ srvctl config pdb -db DATA -pdb MYPDB
Pluggable database name: MYPDB
Application Root PDB:
Cardinality: 2
Maximum CPU count (whole CPUs): 0
Minimum CPU count unit (1/100 CPU count): 0
Start Option: open
Stop Option: immediate
```

Example 3-4 Modifying Pluggable Database Cardinality and Rank Using SRVCTL

In this example, as the `grid` user, you modify cardinality of Pluggable Database `MYPDB`, and set maximum CPU usage, minimum CPU usage, and rank using SRVCTL:

```
$ srvctl modify pdb -db DATA -pdb MYPDB -cardinality 1 -maxcpu 3 -mincpuunit
20 -rank 2
```

Reducing Downtime During Database and Instance Outages

Outages can be either planned (maintenance) or unplanned. You can use features to help minimize both types of outages.

In an Oracle RAC database, the outage of a single instance does not affect database availability. If a server or instance fails, restart and recovery are automatic, including the restarting of the subsystems, such as the listener and the Oracle Automatic Storage Management (Oracle ASM) processes, not only the database. User sessions that connect using a service can be transitioned to a remaining instance automatically. This happens transparently, with little impact to the users.

Copyright © 1999, 2025, Oracle and/or its affiliates.

If the entire database needs to be stopped, then this can be done in a rolling fashion. You can stop each instance individually with stopping the entire database. While an instance is stopped, you perform the task that required the database to be stopped, and then restart the instance. This process is repeated until each instance in the Oracle RAC database has been stopped and restarted.

There are additional features you can use to minimize outages, both planned and unplanned:

- The Oracle RAC high-availability framework maintains service availability by using Oracle Clusterware and resource profiles. **Oracle Clusterware** recovers and balances services according to business rules and the service attributes. If these services are used for client connections to the database, then they are automatically redirected to a remaining instance instead of getting an outage error.
- For repairs, upgrades, and changes that require you to isolate one or more instances or nodes, Oracle RAC provides interfaces that relocate, disable, and enable services to minimize service disruption to application users.
- **Fast Application Notification (FAN)** provides immediate interrupt of clients following outages related to the database, nodes, and networks. FAN notifies clients immediately when resources become available and initiates draining of database sessions so clients experience no outages during planned maintenance. Oracle connection pools, for example, use FAN to receive very fast notification of failures, to balance connections following failures, and to balance connections again after the failed components are repaired.
- **Application Continuity** is a feature that enables the replay, in a non-disruptive and rapid manner, of a request against the database after a recoverable error that makes the database session unavailable so an outage appears to the user as no more than a delayed processing of the request.

Oracle RAC High Availability Best Practices

Implement Oracle Real Application Clusters (Oracle RAC) best practices to achieve minimal downtime for patching and reconfiguration.

- Oracle RAC Two-Stage Rolling Updates
  Starting with Oracle AI Database 26ai, the Oracle RAC two-stage rolling patches feature enables you to apply previously non-rolling patches in a rolling fashion.
- Smooth Reconfiguration of Oracle RAC Instances
  Smooth reconfiguration of Oracle Real Application Clusters (Oracle RAC) instances reduces brownout time during cluster reconfiguration.
- Ordered Sequence Optimizations in Oracle RAC
  Sequence is a database object from which multiple users may generate unique integers. You can use sequences to automatically generate primary key values.

Oracle RAC Two-Stage Rolling Updates

Starting with Oracle AI Database 26ai, the Oracle RAC two-stage rolling patches feature enables you to apply previously non-rolling patches in a rolling fashion.

Oracle RAC two-stage rolling patches are new types of patches, which you can apply in a rolling fashion in stages. Once the patch is applied on the first node, the second node is patched, and so on. When all the nodes are patched, you can enable the patches. Fixes that you apply using this feature are disabled by default.

Copyright © 1999, 2025, Oracle and/or its affiliates.

You can enable these fixes using the `alter system enable RAC two_stage rolling updates all;` command, after all the nodes are patched successfully. You have the option to enable two-stage rolling patches either immediately after applying the patches or later. However, these patches are automatically enabled, if another patch or Release Update (RU) is applied using two-stage rolling patching. Use the `V$RAC_TWO_STAGE_ROLLING_UPDATES` view to list the patches applied using the Oracle RAC two-stage rolling updates.

> **Note**
>
> Oracle RAC two-stage rolling updates apply to non-rolling RUs, but it does not apply to major Oracle RAC database upgrades.

Oracle RAC two-stage rolling updates reduce the need for downtime to apply non-rolling patches. However, not all non-rolling fixes can be applied in RAC rolling fashion. With this feature, the number of Oracle RAC non-rolling patches is significantly reduced.

> **Note**
>
> Review the patch README file for instruction on how to apply the patch, and complete all the required steps before starting the patch upgrade.

This feature enables you to include all Oracle RAC bug fixes in RUs and new features. You are recommended to patch all the instances in a single maintenance window to avoid release version mismatch.

Smooth Reconfiguration of Oracle RAC Instances

Smooth reconfiguration of Oracle Real Application Clusters (Oracle RAC) instances reduces brownout time during cluster reconfiguration.

The smooth reconfiguration feature reduces the brownout time caused by certain Oracle RAC operations such as nodes joining or leaving an Oracle RAC cluster, or when a node is undergoing maintenance or suffers a failure. This feature ensures continuous availability of Oracle RAC services and client applications.

The brownout time depends on the SGA size of the database. The bigger SGA size causes longer brownout. The brownout is related to redistribution of the resources for Global Enqueue Service (GES) and Cache Fusion to new instances when a new node joins or to redistribute resources to existing instances when a node fails or leaves the cluster. The Cache Fusion requests can trigger reconfiguration on an individual resource basis.

The smooth reconfiguration feature allows on-demand Cache Fusion reconfiguration for the Cache Fusion resources that client requests during cluster reconfiguration. The Cache Fusion request reconfigures immediately when the client requests so that the client can complete the request after the on-demand reconfiguration. The on-demand Cache Fusion reconfiguration can work in parallel with the ongoing cluster reconfiguration.

Ordered Sequence Optimizations in Oracle RAC

Sequence is a database object from which multiple users may generate unique integers. You can use sequences to automatically generate primary key values.

Oracle Real Application Clusters (Oracle RAC) supports the following sequence configurations:

Copyright © 1999, 2025, Oracle and/or its affiliates.

- **CACHE and ORDER**: If your application requires sequence number ordering but can tolerate gaps, then use `CACHE` and `ORDER` to cache and order sequence numbers in Oracle RAC. Every instance caches the same set of numbers.
- **CACHE and NOORDER**: If you use sequence numbers, then always use `CACHE` with the `NOORDER` option for optimal performance in sequence number generation. With the `CACHE` option, however, you may have gaps in the sequence numbers. This configuration has the least performance impact and it is the default configuration when creating a new sequence without options.
- **NOCACHE and ORDER**: If your application requires ordered sequence numbers without gaps, then use `NOCACHE` and `ORDER`. The `NOCACHE` and `ORDER` combination has the most negative effect on performance compared to other caching and ordering combinations.
- **NOCACHE and NOORDER**: If government regulations or laws legally require sequence numbers with limited gaps, then use `NOCACHE` and `NOORDER`. Ordering is not guaranteed in this configuration, however, it provides better performance than `NOCACHE` and `ORDER`.

> **Note**
>
> You can use scalable sequences to provide better data load scalability instead of configuring a very large sequence cache. Scalable sequences improve the performance of concurrent data load operations, especially when the sequence values are used for populating primary key columns of tables.

In Oracle AI Database 26ai, ordered sequences in Oracle RAC environments are optimized for performance by reducing the number of lock acquisitions of the ordering enqueue. These improvements do not require any manual intervention or modification of the sequence. Ordered sequence optimization, along with the automatic sizing of the sequence cache feature that was introduced in Oracle Database 19c, works to improve performance of workloads that use ordered sequences.

Related Topics

- *Oracle AI Database Administrator's Guide*

Verifying That Instances are Running

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

- Using SRVCTL to Verify That Instances are Running
  You can use SRVCTL to verify that instances are running on a particular database.
- Using SQL*Plus to Verify That Instances are Running
  You can use SQL*Plus to verify that database instances are running.

Using SRVCTL to Verify That Instances are Running

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

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

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


Copyright © 1999, 2025, Oracle and/or its affiliates.

This command returns output similar to the following:

```
Instance mail1 is running on node node11
Instance mail2 is running on node node10
```

Additionally, you can check whether a PDB is running in the cluster, as follows:

```
$ srvctl status pdb -db db_unique_name -pdb pdb_name
```

Using SQL*Plus to Verify That Instances are Running

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

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

   ```
   CONNECT /@db1 as SYSRAC
   ```

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

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

   This query returns output similar to the following:

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

The output columns for this example are shown in the following table.

Table 3-2 Descriptions of V$ACTIVE_INSTANCES Columns

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

Related Topics

- Viewing the Open Mode of Each PDB

Terminating Sessions On a Specific Cluster Instance

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

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

> **Note**
>
> You can use Application Continuity to hide the cancellation of a transaction from the user, if the application initially used an Application Continuity-enabled dynamic database service to connect to the database instance.

To terminate sessions, follow these steps:

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

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

   - For *`integer1`*, specify the value of the SID column.
   - For *`integer2`*, specify the value of the `SERIAL#` column.
   - For the optional *`integer3`*, specify the ID of the instance where the session to be terminated exists. You can find the instance ID by querying the `GV$` tables.

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

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

Examples of Identifying and Terminating Sessions

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example 3-5 Identify and terminate the session on an busy instance

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

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

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

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

Example 3-6 Identify and terminate the session on an idle instance

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

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

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

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

System altered.

SQL>
```

Example 3-7 Using the IMMEDIATE parameter

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

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

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

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

System altered.

SQL>
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Related Topics

- *Oracle AI Database Administrator's Guide*
- *Oracle AI Database Get Started with Performance Tuning*
- About Application Continuity
  Application Continuity provides high availability for your application workloads.

Overview of Initialization Parameter Files in Oracle RAC

The initialization parameters for an Oracle RAC database are stored in a SPFILE.

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

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

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

- About Creating an SPFILE for Oracle RAC
  All instances in an Oracle Real Application Clusters environment must use the same server parameter file.
- Setting SPFILE Parameter Values for Oracle RAC
  You can change SPFILE settings with Oracle Enterprise Manager or by using the `SET` clause of the `ALTER SYSTEM` statement.
- Parameter File Search Order in Oracle RAC
  Oracle AI Database searches for your parameter file in a particular order depending on your platform. For Oracle RAC databases, you can easily determine the location of the parameter file by using the `srvctl config database` command.
- Backing Up the Server Parameter File
  Oracle recommends that you regularly back up the server parameter file for recovery purposes.

About Creating an SPFILE for Oracle RAC

All instances in an Oracle Real Application Clusters environment must use the same server parameter file.

However, when otherwise permitted, individual instances can have different settings of the same parameter within this one file. Instance-specific parameter definitions are specified as *`SID.parameter = value`*, where *`SID`* is the instance identifier.

Copyright © 1999, 2025, Oracle and/or its affiliates.

For Oracle RAC, the location of the SPFILE is an attribute of the database resource managed by Oracle Clusterware. When creating a new SPFILE, if the instance from which you issued the command is running, then the following command creates a new SPFILE and automatically updates the database resource with the new SPFILE location:

```
CREATE SPFILE='location' FROM PFILE;
```

In this case, you can start up the database without referring to the server parameter file by name.

If the instance from which you issued the command is not running, then the SPFILE in the database resource must be updated manually using `srvctl modify database -db dbname -spfile spfile_path`. Also, if you use the following commands, then the SPFILE location is not automatically updated in the database resource:

```
CREATE SPFILE FROM PFILE [AS COPY];


CREATE SPFILE='location' FROM PFILE AS COPY;


CREATE SPFILE FROM MEMORY;
```

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

Setting SPFILE Parameter Values for Oracle RAC

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

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

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

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

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

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 AI Database updates the setting on all instances except the instance with SID `prod1`:

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

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

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

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

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

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

```
prod1.OPEN_CURSORS=1000
```

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

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

Parameter File Search Order in Oracle RAC

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

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

1. The location specified by the `-spfile` attribute for the database resource managed by Oracle Clusterware.
2. The `spfile`*`sid`*`.ora` file in the location returned by the `$ORACLE_HOME/bin/orabaseconfig` utility, in the subdirectory `/dbs`.
3. The `spfile.ora` file in the location returned by the `$ORACLE_HOME/bin/orabaseconfig` utility, in the subdirectory `/dbs`.
4. The `init`*`sid`*`.ora` file in the location returned by the `$ORACLE_HOME/bin/orabaseconfig` utility, in the subdirectory `/dbs`.

On Windows platforms, the search order is as follows:

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

3. %ORACLE_HOME%\database\init*sid*.ora

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

Related Topics

- srvctl config database
  Displays the configuration for an Oracle RAC database or lists all configured databases that are registered with Oracle Clusterware.

Backing Up the Server Parameter File

Oracle recommends that you regularly back up the server parameter file for recovery purposes.

Do this using Oracle Enterprise Manager or use the CREATE PFILE statement. For example:

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

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

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

Related Topics

- CREATE SPFILE
- *Oracle AI Database Backup and Recovery Reference*

Initialization Parameter Use in Oracle RAC

By default, most parameters are set to a default value and this value is the same across all instances.

However, many initialization parameters can also have different values on different instances as described in Initialization Parameters Specific to Oracle RAC. Other parameters *must* either be unique or identical as described in the following sections:

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Initialization Parameters Specific to Oracle RAC
  The following table summarizes the initialization parameters used specifically for Oracle RAC databases.
- Parameters That Must Have Identical Settings on All Instances
  Certain parameters that are critical at database creation or that affect certain database operations must have the same value for every instance in an Oracle RAC database.
- Parameters That Should Have Identical Settings on All Instances
  Oracle recommends that the parameters listed here have identical settings on all instances.

Related Topics

- Initialization Parameters

Initialization Parameters Specific to Oracle RAC

The following table summarizes the initialization parameters used specifically for Oracle RAC databases.

Parameter	Description
ACTIVE_INSTANCE_COUNT	This initialization parameter was deprecated in Oracle RAC 11g release 2 (11.2). Instead, use a service with one preferred and one available instance.
ASM_PREFERRED_READ_FAILURE_GROUPS	Specifies a set of disks to be the preferred disks from which to read mirror data copies. The values you set for this parameter are instance specific and need not be the same on all instances.
CLUSTER_DATABASE	Enables a database to be started in cluster mode. Set this parameter to TRUE.
CLUSTER_DATABASE_INSTANCES	Oracle RAC uses this parameter to allocate adequate memory resources. It must be set to the same value on all instances. **Note:**The policy-managed database deployment option is desupported in Oracle AI Database. You can set this parameter to a value that is *greater than* the current number of instances, if you are planning to add instances.
CLUSTER_INTERCONNECTS	Specifies an alternative cluster interconnect for the private network when there are multiple interconnects. **Notes:** • Oracle recommends that all Oracle AI Databases and Oracle Clusterware use the same interconnect network. • Oracle does not recommend setting the CLUSTER_INTERCONNECTS parameter except in certain situations. • This parameter is stored in the Grid Plug and Play profile in a Grid Plug and Play environment.
DB_NAME	If you set a value for DB_NAME in instance-specific parameter files, the setting must be identical for all instances.

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

Related Topics

- Administering Multiple Cluster Interconnects on Linux and UNIX Platforms
  In Oracle RAC environments that run on Linux and UNIX platforms, you can use the CLUSTER_INTERCONNECTS initialization parameter to specify an alternative interconnect to the one Oracle Clusterware is using for the private network.
- About Dispatchers

Parameters That Must Have Identical Settings on All Instances

Certain parameters that are critical at database creation or that affect certain database operations must have the same value for every instance in an Oracle RAC database.

Specify these initialization parameter values in the SPFILE or in the individual PFILEs for each instance. The following list contains the parameters that must be identical on every instance:

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

```
DML_LOCKS
RESULT_CACHE_MAX_SIZE
```

Parameters That Should Have Identical Settings on All Instances

Oracle recommends that the parameters listed here have identical settings on all instances.

Oracle recommends that you set the values for the parameters in Table 3-3 to the same value on all instances. Although you can have different settings for these parameters on different instances, setting each parameter to the same value on all instances simplifies administration.

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

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

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

Quiescing Oracle RAC Databases

The procedure for quiescing Oracle RAC databases is identical to quiescing a noncluster database.

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

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

These conditions apply to Oracle RAC:

- If you issued the ALTER SYSTEM QUIESCE RESTRICTED statement but Oracle AI Database has not finished processing it, you cannot open the database.
- You cannot open the database if it is in a quiesced state.
- The ALTER SYSTEM QUIESCE RESTRICTED and ALTER SYSTEM UNQUIESCE statements affect all instances in an Oracle RAC environment, not only the instance that issues the command.

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

Related Topics

- *Oracle AI Database Administrator's Guide*
- ALTER SYSTEM

Copyright © 1999, 2025, Oracle and/or its affiliates.

Administering Multiple Cluster Interconnects on Linux and UNIX Platforms

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

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

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

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

Instead, the best practice is to use Redundant Interconnect Usage, available with Oracle Grid Infrastructure for Oracle RAC and Oracle Real Application Clusters One Node databases, and later. Oracle AI Database uses operating system-based network bonding technologies to enable high availability (and load balancing) for network interface cards meant to be used as the cluster interconnect. If you want to use multiple database versions in one cluster, you can combine both techniques. Redundant Interconnect Usage will use the interfaces as presented on the operating system level, regardless of bonding. For more information regarding bonding technologies contact your operating system vendor.

- Use Cases for Setting the CLUSTER_INTERCONNECTS Parameter

Related Topics

- *Oracle Clusterware Administration and Deployment Guide*

Use Cases for Setting the CLUSTER_INTERCONNECTS Parameter

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> Oracle recommends that all databases and Oracle Clusterware use the same interconnect network.

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

- The cluster is running multiple databases and you need the interconnect traffic to be separated and you do not use Redundant Interconnect Usage.
- You have a single IP address that is made highly available by the operating system, and it does not have a stable interface name (for example, the name can change when you restart).

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

- If you want to use Redundant Interconnect Usage.
- If you have only one cluster interconnect.
- If the default cluster interconnect meets the bandwidth requirements of your Oracle RAC database, which is typically the case.

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

- The `CLUSTER_INTERCONNECTS` initialization parameter is useful only in Linux and UNIX environments where UDP IPC is enabled.
- Specify a different value for each instance of the Oracle RAC database when setting the `CLUSTER_INTERCONNECTS` initialization parameter in the parameter file.
- The IP addresses you specify for the different instances of the same database on different nodes must belong to network adapters that connect to the same interconnect network.
- If you specify multiple IP addresses for this parameter, then list them in the same order for all instances of the same database. For example, if the parameter for the first instance on `node1` lists the IP addresses of the `alt0:`, `fta0:`, and `ics0:` devices in that order, then the parameter for the second instance on `node2` must list the IP addresses of the equivalent network adapters in the same order.
- If an operating system error occurs while Oracle AI Database is writing to the interconnect that you specify with the `CLUSTER_INTERCONNECTS` parameter, then Oracle AI Database returns an error even if some other interfaces are available. This is because the communication protocols between Oracle AI Database and the interconnect can vary greatly depending on your platform. See your Oracle AI Database platform-specific documentation for more information.

Example

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

For example, if you have two databases with high interconnect bandwidth requirements, then you can override the default interconnect provided by your operating system and nominate a

Copyright © 1999, 2025, Oracle and/or its affiliates.

different interconnect for each database using the following syntax in each server parameter file where ip*n* is an IP address in standard dot-decimal format, for example: `144.25.16.214`:

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

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

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

Related Topics

- *Oracle AI Database Reference*

Customizing How Oracle Clusterware Manages Oracle RAC Databases

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

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

To prevent Oracle Clusterware from restarting your Oracle RAC database when you restart your system, or to avoid restarting failed instances more than once, configure a management policy to define the degree of control. There are three management policies:

- AUTOMATIC: This is the default management policy. The database is automatically restored to its previous running condition (started or stopped) upon restart of the database host computer.
- MANUAL: The database is never automatically restarted upon restart of the database host computer. A MANUAL setting does not prevent Oracle Restart from monitoring the database while it is running and restarting it if a failure occurs.
- NORESTART: Similar to the MANUAL setting, the database is never automatically restarted upon restart of the database host computer. A NORESTART setting, however, never restarts the database even if a failure occurs.

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

Example 1: Display the Current Management Policy

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

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


Copyright © 1999, 2025, Oracle and/or its affiliates.

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

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

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

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

Example 3: Specify a Management Policy for a New Database

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

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

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

Related Topics

- srvctl config database
  Displays the configuration for an Oracle RAC database or lists all configured databases that are registered with Oracle Clusterware.
- srvctl modify database
  Modifies the configuration for a database.
- srvctl add database
  Adds a database configuration to Oracle Clusterware.

Advanced Oracle Enterprise Manager Administration

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

This section provides advanced administration tasks that are not covered in Monitoring and Tuning Oracle RAC Databases.

- Using Oracle Enterprise Manager Cloud Control to Discover Nodes and Instances
  Discovering Oracle RAC database and instance targets in Oracle Enterprise Manager enables monitoring and administration.
- Other Oracle Enterprise Manager Capabilities
  Oracle Enterprise Manager provides a variety of administrative capabilities.
- Administering Jobs and Alerts in Oracle RAC
  You can use the **Administration** tab in Oracle Enterprise Manager for an Oracle RAC database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Using Oracle Enterprise Manager Cloud Control to Discover Nodes and Instances

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

Oracle Enterprise Manager Cloud Control enables you to use the Oracle Enterprise Manager console interface to discover Oracle Real Application Clusters (Oracle RAC) database and instance targets.

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

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

1. Log in to Oracle Enterprise Manager and click the **Targets** tab.
2. Click the **Database** tab to view all of the available targets. The column labeled **Types** shows the Oracle RAC databases using the entry *Cluster Database*.
3. Add the database target by selecting the target name, then clicking **Add**. The Add Database Target: Specify Host page appears, which enables you to add databases, listeners, and Oracle Automatic Storage Management (Oracle ASM) as monitored targets.
4. Click the flashlight icon to display the available host names, select a host, then click **Continue**. The Add Database: Specify Source page appears.
5. Either request Oracle Enterprise Managerr to discover only noncluster databases and listeners, or to discover all cluster databases, noncluster databases, and listeners on the cluster, then click **Continue**.
6. If this procedure did not discover your reconfigured cluster database and all of its instances, you can use the Targets Discovered on Cluster page to manually configure your cluster database and noncluster databases.

Other Oracle Enterprise Manager Capabilities

Oracle Enterprise Manager provides a variety of administrative capabilities.

- The Oracle Grid Infrastructure/Oracle RAC Provisioning deployment procedure provisions Oracle RAC and Oracle Grid Infrastructure. This procedure also has a feature called *Profiles*, which enables you to record the inputs and subsequently use them for repeated deployments.
- Dynamic prerequisites for the new procedures enable Oracle Enterprise Manager, when connected to My Oracle Support, to download the latest prerequisites and tools for Oracle RAC provisioning.
- The existing *One-Click Extend Cluster Database* capability now supports Oracle RAC stack.
- The existing *Delete/Scale down Oracle Real Application Clusters* capability is certified with Oracle RAC clusters.
- The existing *Oracle AI Database Provisioning* procedure now supports provisioning of single instances of Oracle AI Database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

Administering Jobs and Alerts in Oracle RAC

You can use the **Administration** tab in Oracle Enterprise Manager for an Oracle RAC database.

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

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

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

- Administering Jobs in Oracle RAC
  You can administer Oracle Enterprise Manager jobs at both the database and instance levels.
- Administering Alerts in Oracle RAC with Oracle Enterprise Manager
  You can use Oracle Enterprise Manager to configure Oracle RAC environment alerts.
- Using Defined Suspensions in Oracle Enterprise Manager
  You can define suspensions (which are time periods in which database monitoring is suspended so that maintenance operations do not skew monitoring data or generate needless alerts) for all managed targets of an Oracle Real Application Clusters (Oracle RAC) database.

Administering Jobs in Oracle RAC

You can administer Oracle Enterprise Manager jobs at both the database and instance levels.

For example, you can create a job at the cluster database level to run on any active instance of the target Oracle Real Application Clusters (Oracle RAC) database. Or you can create a job at the instance level to run on the specific instance for which you created it. If there is a failure, then recurring jobs can run on a remaining instance.

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

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

1. Click **Targets** and click the database for which you want to create the job.
2. Log in to the target database.
3. When Oracle Enterprise Manager displays the Database Home page, click **Maintenance**.
4. Complete the Enterprise Manage Job Wizard pages to create the job.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Administering Alerts in Oracle RAC with Oracle Enterprise Manager

You can use Oracle Enterprise Manager to configure Oracle RAC environment alerts.

You can also configure special Oracle RAC database tests, such as global cache converts, consistent read requests, and so on.

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

Related Topics

- *Oracle AI Database PL/SQL Packages and Types Reference*

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

Using Defined Suspensions in Oracle Enterprise Manager

You can define suspensions (which are time periods in which database monitoring is suspended so that maintenance operations do not skew monitoring data or generate needless alerts) for all managed targets of an Oracle Real Application Clusters (Oracle RAC) database.

Defining suspensions prevents alerts from occurring while performing maintenance. You can define suspensions for an entire cluster database or for specific cluster database instances.

Copyright © 1999, 2025, Oracle and/or its affiliates.

4

Administering Oracle RAC One Node

Learn how to administer Oracle Real Application Clusters One Node (Oracle RAC One Node).

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

- Creating an Oracle RAC One Node Database
  You can create Oracle RAC One Node databases by using Fleet Patching and Provisioning or the Oracle DBCA, as with any other Oracle AI Database (manually created scripts are also a valid alternative).
- Converting Databases
  Using `SRVCTL`, you can convert an Oracle Real Application Clusters (Oracle RAC) database with one instance to an Oracle RAC One Node database, or back to an Oracle RAC database instance.
- Online Database Relocation
  You can relocate an Oracle RAC One Node database to another node while still maintaining service availability using the online database relocation feature.

Creating an Oracle RAC One Node Database

You can create Oracle RAC One Node databases by using Fleet Patching and Provisioning or the Oracle DBCA, as with any other Oracle AI Database (manually created scripts are also a valid alternative).

You can create an Oracle RAC One Node database using Fleet Patching and Provisioning and the `rhpctl add database` command with the `-dbtype RACONENODE` parameter. You can also include an Oracle RAC One Node database using the `rhpctl add workingcopy` command.

Oracle RAC One Node databases may also be the result of a conversion from either a single-instance Oracle AI Database or an Oracle RAC database. Typically, Oracle-provided tools register the Oracle RAC One Node database with Oracle Clusterware. Depending on your configuration, automatic registration of an Oracle RAC One Node database with Oracle Clusterware may not have happened. If this is the case, then follow the steps in this section to register the Oracle RAC One Node database with Oracle Clusterware.

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

```
$ srvctl add database -dbtype  RACONENODE [-server server_list]
  [-instance instance_name] [-timeout timeout]
```

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

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

For Oracle RAC One Node databases, you must configure at least one dynamic database service (in addition to and opposed to the default database service). When using an Oracle RAC One Node database, service registration is performed as with any other Oracle RAC database.

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

Related Topics

- Fleet Patching and Provisioning
- srvctl add database
  Adds a database configuration to Oracle Clusterware.
- Using Oracle DBCA in Interactive Mode to Add Database Instances to Target Nodes
  To add a database instance to a target node with Oracle DBCA in interactive mode, perform the steps described here.

Converting Databases

Using `SRVCTL`, you can convert an Oracle Real Application Clusters (Oracle RAC) database with one instance to an Oracle RAC One Node database, or back to an Oracle RAC database instance.

- Converting a Database from Oracle RAC to Oracle RAC One Node
  Use SRVCTL to convert an Oracle RAC database to an Oracle RAC One Node database.
- Converting a Database from Oracle RAC One Node to Oracle RAC
  You can convert an Oracle RAC One Node database to an Oracle RAC database using SRVCTL.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Converting a Database from Oracle RAC to Oracle RAC One Node

Use SRVCTL to convert an Oracle RAC database to an Oracle RAC One Node database.

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

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

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

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

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

Related Topics

- srvctl remove instance
- srvctl stop instance
- srvctl convert database

Converting a Database from Oracle RAC One Node to Oracle RAC

You can convert an Oracle RAC One Node database to an Oracle RAC database using SRVCTL.

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

Log in as the Oracle RAC One Node database owner and enter the following SRVCTL command:

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


Copyright © 1999, 2025, Oracle and/or its affiliates.

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

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

Related Topics

- srvctl convert database
  Converts a database either to or from an Oracle RAC One Node database.

Online Database Relocation

You can relocate an Oracle RAC One Node database to another node while still maintaining service availability using the online database relocation feature.

Only during a planned online database relocation is a second instance of an Oracle RAC One Node database created, so that any database sessions can continue while the database is relocated to a new node. You can only use online database relocation with Oracle RAC One Node databases but you cannot use online database relocation with Oracle RAC databases.

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

Online database relocation occurs, as follows:

1. Start a new database instance in a different location.
2. Move all the services to the relocated instance.
3. Wait for all the connections to migrate to the relocated instance.
4. Shut down the former database instance, forcing any remaining connections to move to the relocated instance.

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

Before you initiate the online relocation of a database instance, perform the following tasks:

- When you relocate a database instance to a target node that is not currently in the candidate server list for the database, you must copy the password file, if configured, to the target node, unless you use shared password files stored in Oracle ASM.
- When you use password file-based authentication for remote management of Oracle RAC One Node databases without any shared password file, you must have two password files on each node where the database can run: one named *`SID_prefix`*`_1` and the other named *`SID_prefix`*`_2`. You must recopy both of these files to all candidate nodes every time you update the password file.

  Oracle recommends using Oracle Clusterware to start and stop the database, and defining users in the data dictionary for other management.
- If your operating system is Microsoft Windows, then before you relocate a database instance, you must ensure that the database service user is added to the wallet. Run

Copyright © 1999, 2025, Oracle and/or its affiliates.

`crsctl query wallet -type OSUSER -all` to check whether the database service user is in the wallet. If not, then run `crsctl add wallet -type OSUSER -user` *`user_name`* `-passwd` to add the database service user to the wallet.

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

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

Related Topics

- Creating and Maintaining a Database Password File
- srvctl relocate database
  Initiates the relocation of an Oracle RAC One Node database from one node to another node.

Copyright © 1999, 2025, Oracle and/or its affiliates.

5

Workload Management with Dynamic Database Services

Workload management includes load balancing, enabling clients for Oracle Real Application Clusters (Oracle RAC), distributed transaction processing, and services.

- Connection Load-Balancing
  Learn how Oracle Net Services provide the ability to distribute client connections across the instances in an Oracle RAC configuration.
- Load Balancing Advisory
  Learn about how to configure and use the load balancing advisory, and how to manage FAN events.
- Enabling Clients for Oracle RAC
  Learn how FAN is integrated with Oracle Clients, and how to enable FAN events for the several specific client development environments.
- Distributed Transaction Processing in Oracle RAC
  Learn how Oracle Real Application Clusters (Oracle RAC) supports global (XA) transactions and DTP processing
- Oracle RAC Sharding
  Oracle RAC Sharding creates an affinity between table partitions and Oracle RAC instances, and routes database requests that specify a partitioning key to the instance that logically holds the corresponding partition.
- Automatic Workload Repository
  The Automatic Workload Repository (AWR) collects, processes, and maintains performance statistics for the database.
- Measuring Performance by Service Using the Automatic Workload Repository
  Services add a new dimension for performance tuning because workloads are visible and measurable, and therefore resource consumption and wait times are attributable by application.
- Automatic Workload Repository Service Thresholds and Alerts
  To provide accountability for your required levels of service, you can use the Oracle Real Application Clusters (Oracle RAC) Automatic Workload Repository (AWR) service.
- Using Oracle Services
  To manage workloads or a group of applications, you can define services that you assign to a particular application, or to a subset of an application's operations, or you can group work by type under services.
- Service Deployment Options
  Learn about services in Oracle Real Application Clusters (Oracle RAC) databases, and how to define and deploy services.
- Administering Services
  Learn how to create and administer services, and perform other service-related tasks using Oracle Enterprise Manager and the SRVCTL utility.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Global Services
  Oracle RAC supports database services and enables service-level workload management across instances in a single cluster.
- Service-Oriented Buffer Cache Access
  Service-oriented buffer cache access improves performance by managing data with the service to which the data belongs.
- Connecting to a Service: An Example
  You can use this example to see how to create a service, and see several examples of connecting to that service using different client methods.

Connection Load-Balancing

Learn how Oracle Net Services provide the ability to distribute client connections across the instances in an Oracle RAC configuration.

- About Connection Load-Balancing
  There are two types of load balancing that you can implement: client-side and server-side load balancing.
- Server-Side Load Balancing
  Using Oracle DBCA to create an Oracle Real Application Clusters (Oracle RAC) database enables you to obtain server-side load-balancing configuration automatically.
- Generic Database Clients
  Oracle Net Services enables you to add the `CONNECT_TIMEOUT`, `RETRY_COUNT`, and `TRANSPORT_CONNECT_TIMEOUT` parameters to the `tnsnames.ora` connection string.
- Client-Side Connection Configuration for Older Clients
  Learn about the ways you can set up connection failovers and timeouts with Java Database Connectivity (JDBC) Thin Clients, and Oracle Call Interface (OCI) clients.
- Client-Side Load Balancing
  Learn about client-side load balancing, and how a Single Client Access Name (SCAN) can assist with connection loads.
- Smart Connection Rebalance
  Smart Connection Rebalance automatically routes sessions to an instance with the intent to optimize performance by monitoring the access patterns of the underlying objects of the workload.

About Connection Load-Balancing

There are two types of load balancing that you can implement: client-side and server-side load balancing.

With client-side load balancing, connection requests are distributed across the listeners, independently at each client. With server-side load balancing, the SCAN listener directs a connection request to the best instance currently providing the service, based on the `-clbgoal` and `-rlbgoal` settings for the service.

The SCAN listener is aware of the HTTP protocol. With this awareness, the SCAN can redirect HTTP clients to the appropriate handler, which can reside on different nodes in the cluster, not only the node on which the SCAN listener resides.

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

Related Topics

- *Oracle AI Database Net Services Administrator's Guide*

Server-Side Load Balancing

Using Oracle DBCA to create an Oracle Real Application Clusters (Oracle RAC) database enables you to obtain server-side load-balancing configuration automatically.

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

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

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

> **Note**
>
> Note: If you set the `REMOTE_LISTENER` parameter manually, then set this parameter to *`scan_name:scan_port`*.

FAN, Fast Connection Failover, and the load balancing advisory depend on an accurate connection load-balancing configuration that includes setting the connection load-balancing goal for the service. You can use a goal of either `LONG` or `SHORT` for connection load-balancing. These goals have the following characteristics:

- **SHORT**: Use the `SHORT` connection load-balancing method for applications that benefit from using run-time load-balancing. The following is an example of modifying a service using SRVCTL to set the connection load balancing goal to `SHORT`:

  ```
  $ srvctl modify service -db db_unique_name -service service_name -clbgoal
  SHORT
  ```

- **LONG**: Use the `LONG` connection load-balancing method for applications that benefit from server-side load-balancing and do not require run-time load balancing. `LONG` is the default connection load balancing goal. The following is an example of modifying a service using SRVCTL to set the connection load balancing goal to `LONG`:

  ```
  $ srvctl modify service -db db_unique_name -service service_name -clbgoal
  LONG
  ```

Generic Database Clients

Oracle Net Services enables you to add the `CONNECT_TIMEOUT`, `RETRY_COUNT`, and `TRANSPORT_CONNECT_TIMEOUT` parameters to the `tnsnames.ora` connection string.

For example, when using SCAN addresses for the remote listeners at the database:

```
jdbc:oracle:thin:@(DESCRIPTION =
 (TRANSPORT_CONNECT_TIMEOUT=3)(CONNECT_TIMEOUT=60)
 (RETRY_COUNT=3)(FAILOVER=ON)
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

```
 (ADDRESS_LIST =(ADDRESS=(PROTOCOL=tcp)
  (HOST=CLOUD-SCANVIP.example.com)(PORT=5221))
 (CONNECT_DATA=(SERVICE_NAME=orcl)))
Remote_listeners=CLOUD-SCANVIP.example.com:5221
```

For example, when using remote listeners pointing to VIPs at the database:

```
jdbc:oracle:thin:@(DESCRIPTION =
 (TRANSPORT_CONNECT_TIMEOUT=3)
 (CONNECT_TIMEOUT=60)(RETRY_COUNT=20)
 (RETRY_DELAY=3)(FAILOVER=ON)
 (ADDRESS_LIST=
 (ADDRESS=(PROTOCOL=tcp)(HOST=CLOUD-VIP1)(PORT=1521) )
 (ADDRESS=(PROTOCOL=tcp)(HOST=CLOUD-VIP2)(PORT=1521) )
 (ADDRESS=(PROTOCOL=tcp)(HOST=CLOUD-VIP3)(PORT=1521) ))
 (CONNECT_DATA=(SERVICE_NAME=GOLD)))
```

The value of these parameters is expressed in seconds. In the preceding examples, Oracle Net waits for 60 seconds for each full connection to receive a response, after which it assumes that a failure occurred and retries the next address in the `ADDRESS_LIST`. Oracle Net retries the address list 3 times before it returns a failure message to the client. The `TRANSPORT_CONNECT_TIMEOUT` parameter establishes the time to wait to establish a TCP connection to the database server.

For SCAN, Oracle Net Services tries all three addresses (returned by the SCAN address) before returning a failure to the client. EZConnect with SCAN includes this connection failover feature.

This behavior is called Oracle Net *connection failover*. If an error is returned from a chosen address in the list, then Oracle Net Services tries the next address in the list until it is either successful or it has exhausted all addresses in its list.

Client-Side Connection Configuration for Older Clients

Learn about the ways you can set up connection failovers and timeouts with Java Database Connectivity (JDBC) Thin Clients, and Oracle Call Interface (OCI) clients.

- About Client-Side Connection Configuration for Older Clients
  Oracle Net Services provide connection failover and availability features for service requests from older clients.
- JDBC Thin Clients
  You can avoid delays by setting the `oracle.net.ns.SQLnetDef.TCP_CONNTIMEOUT_STR` property
- OCI Clients
  For Oracle Call Interface (OCI) clients, create a local `sqlnet.ora` file on the client side.

About Client-Side Connection Configuration for Older Clients

Oracle Net Services provide connection failover and availability features for service requests from older clients.

In addition to client-side load balancing, Oracle Net Services include **connection failover**. If an error is returned from the chosen address in the list, Oracle Net Services tries the next address in the list until it is either successful or it has exhausted all addresses in its list. For

Copyright © 1999, 2025, Oracle and/or its affiliates.

SCAN, Oracle Net Services tries all three addresses before returning a failure to the client. EZConnect with SCAN includes this connection failover feature.

To increase availability, you can specify a timeout that specifies how long Oracle Net waits for a response from the listener before returning an error. The method of setting this timeout parameter depends on the type of client access. Oracle Net maintains these parameters for backward compatibility.

JDBC Thin Clients

You can avoid delays by setting the `oracle.net.ns.SQLnetDef.TCP_CONNTIMEOUT_STR` property

Use the following example to see how to set the property for Java Database Connectivity (JDBC) thin clients:

```
Properties prop = new Properties ();
prop.put (oracle.net.ns.SQLnetDef.TCP_CONNTIMEOUT_STR,
"" + (1 * 1000)); // 1 second
dbPools[ poolIndex ].setConnectionProperties ( prop );
```

The parameter value is specified in milliseconds, so you can configure a timeout less than one second. For example, if the application retries connecting, it is possible to reduce the timeout to 500Ms.

OCI Clients

For Oracle Call Interface (OCI) clients, create a local `sqlnet.ora` file on the client side.

To configure the connection timeout in the `sqlnet.ora` file, add the following line:

```
sqlnet.outbound_connect_timeout = number_of_seconds
```

The granularity of the timeout value for the OCI client is in seconds. The `sqlnet.ora` file affects all connections using this client.

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

Related Topics

- *Oracle Call Interface Developer's Guide*

Client-Side Load Balancing

Learn about client-side load balancing, and how a Single Client Access Name (SCAN) can assist with connection loads.

Client-side load balancing is defined in your client connection definition (`tnsnames.ora` file, for example) by setting the parameter `LOAD_BALANCE=ON`. When you set this parameter to `ON`, Oracle AI Database randomly selects an address in the address list, and connects to that node's listener. This balances client connections across the available SCAN listeners in the cluster.

Copyright © 1999, 2025, Oracle and/or its affiliates.

If you configured SCAN for connection requests, then client-side load balancing is not relevant for those clients that support SCAN access. When clients connect using SCAN, Oracle Net automatically balances the load of client connection requests across all available IP addresses you defined for the SCAN, unless you are using Easy Connect string.

The SCAN listener redirects the connection request to the local listener of the instance that is least loaded (if `-clbgoal` is set to `SHORT`) and provides the requested service. When the listener receives the connection request, the listener connects the user to an instance that the listener knows provides the requested service. To see what services a listener supports, run the `lsnrctl services` command.

> **Note**
>
> Oracle recommends `-clbgoal LONG` for client-side load balancing.

If you are using clients that do not support SCAN, then, to use SCAN you must change the client `tnsnames.ora` to include the SCAN VIPs, and set `LOAD_BALANCE=ON` to balance requests across the VIPs. For example:

```
Sales.example.com=(DESCRIPTION=
  (ADDRESS_LIST=(LOAD_BALANCE=ON)(FAILOVER=ON)
    (ADDRESS=(PROTOCOL=TCP)(HOST=172.22.67.192)(PORT=1521))
    (ADDRESS=(PROTOCOL=TCP)(HOST=172.22.67.193)(PORT=1521))
    (ADDRESS=(PROTOCOL=TCP)(HOST=172.22.67.194)(PORT=1521)))
    (CONNECT_DATA=(SERVICE_NAME=salesservice.example.com))
  )
```

Smart Connection Rebalance

Smart Connection Rebalance automatically routes sessions to an instance with the intent to optimize performance by monitoring the access patterns of the underlying objects of the workload.

Oracle Real Application Clusters (Oracle RAC) offers two options for load balancing: client-side load balancing and server- side load balancing. Sessions connect to an Oracle RAC instance using Single Client Access Network (SCAN) and a user-defined service name. You can configure a service to run on all or a subset of Oracle RAC instances. By default, SCAN redirects the sessions to the local listener and the SCAN listener directs a connection request to the best instance currently hosting the service, based on the `-clbgoal` and `-rlbgoal` settings for the service.

Smart Connection Rebalance avoids resource conflict and ensures that workloads accessing similar objects end up in one instance and benefit from the reduced inter-instance network messages and data block transfers over the private network. This feature ensures optimum load balancing and performance. Oracle RAC features, such as partitioning, local indexes, Right Growing Index (RGI) optimizations, and Exafusion help reduce resource conflict.

You can enable Smart Connection Rebalance by setting the `-rlbgoal` attribute to `SMART_CONN`:

```
$ srvctl modify service -db db_unique_name -service service_name -rlbgoal
SMART_CONN
```

To disable Smart connection load balancing, set the `-rlbgoal` of that service to `Service_TIME`.

Copyright © 1999, 2025, Oracle and/or its affiliates.

This feature performs real-time monitoring of different workloads and attempt to transparently relocate service-based connections across Oracle RAC instances to significantly improve database performance.

> **Note**
>
> The connection relocation is automatic and does not need database administrators to manually distribute the sessions.

Related Topics

- Load Balancing Advisory

Load Balancing Advisory

Learn about how to configure and use the load balancing advisory, and how to manage FAN events.

- Overview of the Load Balancing Advisory
  Learn about load balancing, and about guidelines that Oracle recommends for load balancing on Oracle Real Application Clusters (Oracle RAC).
- Configuring Your Environment to Use the Load Balancing Advisory
  You can configure your environment to use the load balancing advisory by defining service-level goals for each service for which you want to enable load balancing.
- Load Balancing Advisory FAN Events
  The load balancing advisory FAN events provide metrics for load balancing algorithms.
- Monitoring Load Balancing Advisory FAN Events
  To monitor load balancing advisory events for an instance, use this query.

Overview of the Load Balancing Advisory

Learn about load balancing, and about guidelines that Oracle recommends for load balancing on Oracle Real Application Clusters (Oracle RAC).

Load balancing distributes work across all of the available Oracle RAC database instances. Oracle recommends that applications use connection pools with persistent connections that span the instances that offer a particular service. When using persistent connections, connections are created infrequently and exist for a long duration. Work comes into the system with high frequency, borrows these connections, and exists for a relatively short duration. The load balancing advisory provides advice about how to direct incoming work to the instances that provide the optimal service quality for that work. This minimizes the need to relocate the work later.

By using the Load Balancing Advisory and run-time connection load balancing goals, feedback is built in to the system. Work is routed to provide the best service times globally, and routing responds gracefully to changing system conditions. In a steady state, the system approaches equilibrium with improved throughput across all of the Oracle RAC instances.

Standard architectures that can use the load balancing advisory include connection load balancing, transaction processing monitors, application servers, connection concentrators, hardware and software load balancers, job schedulers, batch schedulers, and message queuing systems. All of these applications can allocate work.

Copyright © 1999, 2025, Oracle and/or its affiliates.

The load balancing advisory is deployed with key Oracle clients, such as a listener, the JDBC universal connection pool, OCI session pool, Oracle WebLogic Server Active GridLink for Oracle RAC, and the ODP.NET Connection Pools. Third-party applications can also subscribe to load balancing advisory events by using JDBC and Oracle RAC FAN API or by using callbacks with OCI.

Configuring Your Environment to Use the Load Balancing Advisory

You can configure your environment to use the load balancing advisory by defining service-level goals for each service for which you want to enable load balancing.

Configuring a service-level goal enables the load balancing advisory and the publishing of FAN load balancing events for that service. There are three types of service-level goals for run-time connection load balancing:

- `SMART_CONN`: Enables Smart Connection Rebalance and tries to ensure that workloads accessing similar objects end up in one instance and benefit from the reduced inter-instance network messages and data block transfers over the private network. The following example shows how to set the goal to `SMART_CONN` for connections using the `odpapp` service:

  ```
  $ srvctl modify service -db db_unique_name -service odpapp
    -rlbgoal SMART_CONN
  ```

- `SERVICE_TIME`: Attempts to direct work requests to instances according to response time. Load balancing advisory data is based on elapsed time for work done in the service plus available bandwidth to the service. An example for the use of `SERVICE_TIME` is for workloads such as internet shopping where the rate of demand changes. The following example shows how to set the goal to `SERVICE_TIME` for connections using the `online` service:

  ```
  $ srvctl modify service -db db_unique_name -service online
    -rlbgoal SERVICE_TIME -clbgoal SHORT
  ```

- `THROUGHPUT`: Attempts to direct work requests according to throughput. The load balancing advisory is based on the rate that work is completed in the service plus available bandwidth to the service. An example for the use of `THROUGHPUT` is for workloads such as batch processes, where the next job starts when the last job completes. The following example shows how to set the goal to `THROUGHPUT` for connections using the `sjob` service:

  ```
  $ srvctl modify service -db db_unique_name -service sjob
    -rlbgoal THROUGHPUT -clbgoal LONG
  ```

Setting the run-time connection load balancing goal to `NONE` disables load balancing for the service. You can see the goal settings for a service in the data dictionary by querying the `DBA_SERVICES`, `V$SERVICES`, and `V$ACTIVE_SERVICES` views. You can also review the load balancing settings for a service using Oracle Enterprise Manager.

Related Topics

- Administering Services
  Learn how to create and administer services, and perform other service-related tasks using Oracle Enterprise Manager and the SRVCTL utility.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Load Balancing Advisory FAN Events

The load balancing advisory FAN events provide metrics for load balancing algorithms.

The easiest way to take advantage of these events is to use the run-time connection load balancing feature of an Oracle integrated client such as JDBC, Universal Connection Pool (or the deprecated Implicit Connection Cache), ODP.NET Connection Pools, OCI session pools, or Oracle WebLogic Server Active GridLink for Oracle RAC. Other client applications can take advantage of FAN programatically by using the Oracle RAC FAN API to subscribe to FAN events and perform event-handling actions upon receipt. Table 5-1 describes the load balancing advisory FAN event parameters.

> **See Also**
>
> *Oracle AI Database JDBC Developer's Guide* for more information about the Oracle RAC FAN API

Table 5-1 Load Balancing Advisory FAN Events

Parameter	Description
`VERSION`	Version of the event record. Used to identify release changes.
`EVENT_TYPE`	A load balancing advisory event is always of the `SERVICEMETRICS` event type.
`SERVICE`	The service name; matches the value of `NAME` in `DBA_SERVICES`.
`DATABASE`	The unique database supporting the service; matches the initialization parameter value for `DB_UNIQUE_NAME`, which defaults to the value of the initialization parameter `DB_NAME`.
`INSTANCE`	The name of the instance that supports the service; matches the `ORACLE_SID` value.
`PERCENT`	The percentage of work requests to send to this database instance.
`FLAG`	Indication of the service quality relative to the service goal. Valid values are `GOOD`, `VIOLATING`, `NO DATA`, and `BLOCKED`.
`TIMESTAMP`	The local time zone to use when ordering notification events.

> **Note**
>
> The `INSTANCE`, `PERCENT`, and `FLAG` event parameters are generated for each instance offering the service. Each set of instance data is enclosed within braces ({}).

Related Topics

- *Oracle AI Database JDBC Developer's Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

Monitoring Load Balancing Advisory FAN Events

To monitor load balancing advisory events for an instance, use this query.

You can use the following query against the internal queue table for load balancing advisory FAN events to monitor load balancing advisory events generated for an instance:

```
SET PAGES 60 COLSEP '|' LINES 132 NUM 8 VERIFY OFF FEEDBACK OFF
COLUMN user_data HEADING "AQ Service Metrics" FORMAT A60 WRAP
BREAK ON service_name SKIP 1
SELECT
 TO_CHAR(enq_time, 'HH:MI:SS') Enq_time, user_data
 FROM sys.sys$service_metrics_tab
 ORDER BY 1 ;
```

The results of this query contain rows similar to the following:

```
02:56:05|SYS$RLBTYP('hr', 'VERSION=1.0 database=sales service=hr
   { {instance=sales_4 percent=38 flag=GOOD aff=TRUE}{instance=sales_1
   percent=62 flag=GOOD aff=TRUE} } timestamp=2012-07-16 07:56:05')
```

Following is an example of a load balancing advisory event for the `lba_serv` service offered on two instances (`orcl1` and `orcl2`), as captured from Oracle Notification Service using the Oracle RAC FAN API:

```
Notification Type: database/event/servicemetrics/lba_serv.example.com
   VERSION=1.0 database=orcl service=lba_serv.example.com { {instance=orcl2
   percent=50 flag=UNKNOWN aff=FALSE}{instance=orcl1 percent=50 flag=UNKNOWN
   aff=FALSE} } timestamp=2012-07-06 13:19:12
```

> **Note**
>
> The `SERVICMETRICS` events are not visible through the FAN callout mechanism.

Enabling Clients for Oracle RAC

Learn how FAN is integrated with Oracle Clients, and how to enable FAN events for the several specific client development environments.

Oracle has integrated FAN with many of the common client application environments used to connect to Oracle Real Application Clusters (Oracle RAC). The easiest way to use FAN is to use an integrated Oracle Client.

- Overview of Oracle Integrated Clients and FAN
  The overall goals of FAN are to enable end-to-end, lights-out recovery of applications and load balancing based on real transaction performance.
- Enabling JDBC-Thin Clients for Fast Connection Failover
  Enabling Fast Connection Failover (FCF) for Universal Connection Pool and Oracle WebLogic Server Active GridLink for Oracle RAC enables the use of FAN HA and load balancing advisory events.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Enabling JDBC Clients for Run-time Connection Load Balancing
  Run-time connection load balancing requires the use of an Oracle JDBC driver and an Oracle RAC database.
- Configuring JDBC-Thin Clients for Application Continuity for Java
  This `oracle.jdbc.datasource.impl.OracleDataSource` data source is a JDBC-thin data source that Application Continuity requires for Java.
- Configuring JDBC-Thin Clients for Transaction Guard
  Transaction Guard provides a protocol and a generic tool for applications to use for at-most-once processing in case of planned and unplanned outages.
- Enabling OCI Clients for Fast Connection Failover
  Complete the procedure that enables Fast Connection Failover (FCF) by registering to receive notifications for Oracle RAC high availability Fast Application Notification (FAN) events.
- Enabling OCI Clients for Run-time Connection Load Balancing
  Oracle Call Interface (OCI) session pooling enables multiple threads of an application to use a dynamically managed set of pre-created database sessions.
- Configuring OCI Clients to use Transaction Guard
  OCI supports FAN messages and Transaction Guard. FAN is designed to quickly notify an OCI-based application of outages at the node, database, instance, service, and public network levels.
- Enabling ODP.NET Clients to Receive FAN High Availability Events
  ODP.NET connection pools can subscribe to FAN HA notifications that indicate when nodes, services, and service members are down.
- Enabling ODP.NET Clients to Receive FAN Load Balancing Advisory Events
  When connecting to Oracle AI Database, ODP.NET uses Oracle Notification Service, rather than Advanced Queuing.
- Configuring ODP.NET Clients to use Transaction Guard
  ODP.NET supports FAN messages and Transaction Guard. FAN is designed to quickly notify an ODP.NET-based application of outages at the node, database, instance, service, and public network levels.

Overview of Oracle Integrated Clients and FAN

The overall goals of FAN are to enable end-to-end, lights-out recovery of applications and load balancing based on real transaction performance.

Applications use the FAN high availability (HA) events to achieve very fast detection of failures, balancing of connection pools following failures, and distribution of connections again when the failed components are repaired.

The FAN events carrying load balancing advice help connection pools consistently deliver connections to available instances that provide the best service. FAN HA is integrated with the JDBC-thin and OCI drivers. FAN HA and FAN load balancing are both integrated with the JDBC Universal Connection Pool (and the deprecated Implicit Connection Cache), the OCI session pools, the ODP.NET connection pool, and Oracle WebLogic Server Active GridLink for Oracle RAC.

Due to the integration with FAN, Oracle integrated clients are more aware of the current status of an Oracle RAC cluster. This prevents client connections from waiting or trying to connect to instances or services that are no longer available. When instances start, Oracle RAC uses FAN to notify the connection pool so that the connection pool can create connections to the recently started instance and take advantage of the additional resources that this instance provides.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Oracle client drivers that are integrated with FAN can:

- Remove terminated connections immediately when a service is declared `DOWN` at an instance, and immediately when nodes are declared `DOWN`
- Report errors to clients immediately when Oracle AI Database detects the `NOT RESTARTING` state, instead of making the client wait while the service repeatedly attempts to restart

Oracle connection pools that are integrated with FAN can:

- Balance connections across all of the Oracle RAC instances when a service starts; this is preferable to directing the sessions that are defined for the connection pool to the first Oracle RAC instance that supports the service
- Balance work requests at run time using load balancing advisory events

The use of client drivers or connection pools and FAN requires that you properly configure the Oracle Notification Service to deliver the FAN events to the clients. In addition, for load balancing, configure database connection load balancing across all of the instances that provide the services used by the connection pool. Oracle recommends that you configure both client-side and server-side load balancing with Oracle Net Services. If you use Oracle DBCA to create your database, then both client-side and server-side load balancing are configured by default.

Related Topics

- Connection Load-Balancing
  Learn how Oracle Net Services provide the ability to distribute client connections across the instances in an Oracle RAC configuration.
- Fast Application Notification (FAN)
  The Oracle RAC high availability framework monitors a database and its services, and sends Fast Application Notification (FAN) event notifications.

Enabling JDBC-Thin Clients for Fast Connection Failover

Enabling Fast Connection Failover (FCF) for Universal Connection Pool and Oracle WebLogic Server Active GridLink for Oracle RAC enables the use of FAN HA and load balancing advisory events.

- About Fast Connection Failover and JDBC-Thin Clients
  Learn how Oracle Real Application Clusters (Oracle RAC) FAN APIs enable application code to receive and respond to FAN event notifications.
- Oracle Notification Service for JDBC-Thin Clients
  Learn about the benefits of using Remote Oracle Notification Service subscription with your Oracle Real Application Clusters (Oracle RAC) database.
- Configuring FCF for JDBC/OCI and JDBC-Thin Driver Clients
  You can enable FCF for Universal Connection Pool or Implicit Connection Cache.

About Fast Connection Failover and JDBC-Thin Clients

Learn how Oracle Real Application Clusters (Oracle RAC) FAN APIs enable application code to receive and respond to FAN event notifications.

For Universal Connection Pool to use Oracle RAC Fast Application Notification (FAN), your application can use the Java Database Connectivity (JDBC) development environment for either JDBC OCI or JDBC Thin clients. The Java Database Connectivity Oracle Call Interface (JDBC/OCI) driver connection pooling functionality is part of the JDBC-thin client. This functionality is provided by the `OracleOCIConnectionPool` class.

Copyright © 1999, 2025, Oracle and/or its affiliates.

To enable FCF for the JDBC-thin client, call the method `setFastConnectionFailoverEnabled(true)` of the `OracleDataSource` class in the `oracle.jdbc.pool` package before making the first `getConnection()` request. When you enable FCF for the JDBC-thin client, the failover property applies to every connection in the connection pool. Enabling FCF with JDBC-thin driver or JDBC/OCI clients enables the connection pool to receive and react to all FAN events.

JDBC application developers can programmatically integrate with FAN by using a set of APIs. The Oracle RAC FAN APIs enable application code to receive and respond to FAN event notifications sent by Oracle RAC in the following ways:

- Listening for Oracle RAC service down, service up, and node down events
- Listening for load balancing advisory events and responding to them

Related Topics

- *Oracle AI Database JDBC Developer's Guide*

Oracle Notification Service for JDBC-Thin Clients

Learn about the benefits of using Remote Oracle Notification Service subscription with your Oracle Real Application Clusters (Oracle RAC) database.

FCF relies on Oracle Notification Service to propagate database events between the connection pool and the Oracle RAC database. At run time, the connection pool must be able to setup an Oracle Notification Service environment. Oracle Notification Service (`ons.jar`) is included as part of the Oracle Client software. Oracle Notification Service can be configured using either remote configuration or client-side Oracle Notification Service daemon configuration. Remote Oracle Notification Service subscription offers the following advantages:

- Support for an All Java mid-tier software
- An Oracle Notification Service daemon is not necessary on the client system, so you do not have to manage this process
- Configuration by way of a `DataSource` property

Configuring FCF for JDBC/OCI and JDBC-Thin Driver Clients

You can enable FCF for Universal Connection Pool or Implicit Connection Cache.

Oracle recommends using the Universal Connection Pool for Java because the Implicit Connection Cache is deprecated. You can also use Oracle WebLogic Server Active GridLink for Oracle RAC.

This procedure explains how to enable FCF for JDBC. You can also configure Application Continuity and Transaction Guard.

To enable FCF, you must first enable the Universal Connection Pool, as described in the following procedure:

1. Create the connection pool and set `setFastConnectionFailoverEnabled(true)`.

   The following example creates a connection pool and enables FCF. The `ucp.jar` library must be included in the classpath of an application to use this example.

   ```
   PoolDataSource pds = PoolDataSourceFactory.getPoolDataSource();
   pds.setFastConnectionFailoverEnabled(true);
   ```

2. Determine the ports to use for Oracle Notification Service remote subscriptions.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Use the following command to view the Oracle Notification Service configuration on each node that is running Oracle Clusterware as in the following example:

```
srvctl config nodeapps -onsonly
```

The output of this command lists the local and remote ports configured for Oracle Notification Service.

> **Note**
>
> Oracle Notification Service configuration should have been automatically completed during the Oracle Grid Infrastructure installation.

3. Configure the remote Oracle Notification Service subscription.

   When using the Universal Connection Pool, an application calls `setONSConfiguration` for an `OracleDataSource` instance and specifies the nodes and port numbers to use. The port numbers used for each node are the same as the remote port displayed for each node in Step 2, as shown in the following example. The `ons.jar` library must be included in the classpath of an application to use this example.

   ```
   pds.setONSConfiguration("nodes=racnode1:6200,racnode2:6200");
   ```

   Applications that use remote Oracle Notification Service configuration must set the `oracle.ons.oraclehome` system property to the location of `ORACLE_HOME` before starting the application, for example:

   ```
   java -Doracle.ons.oraclehome=$ORACLE_HOME ...
   ```

4. Configure the connection URL.

   A connection factory's connection URL must use the service name syntax when using FCF. The service name is used to map the connection pool to the service. The following example demonstrates configuring the connection URL:

   ```
   pds.setConnectionFactoryClassName("oracle.jdbc.pool.OracleDataSource");
   pds.setURL("jdbc:oracle:thin@//SCAN_name:service_name");...
   ```

Related Topics

- *Oracle AI Database JDBC Developer's Guide*
- *Oracle Universal Connection Pool Developer's Guide*

Enabling JDBC Clients for Run-time Connection Load Balancing

Run-time connection load balancing requires the use of an Oracle JDBC driver and an Oracle RAC database.

Oracle JDBC Universal Connection Pool and Oracle WebLogic Server Active GridLink for Oracle RAC leverage the load balancing functionality provided by an Oracle RAC database.

The Universal Connection Pool and Oracle WebLogic Server Active GridLink for Oracle RAC are integrated to take advantage of Load Balancing Advisory information.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Run-time connection load balancing requires that FCF is enabled and configured properly. In addition, the Oracle RAC load balancing advisory must be configured with service-level goals for each service used by the connection pool.

```
srvctl modify service -db db_unique_name -service service_name
   -rlbgoal SERVICE_TIME -clbgoal LONG
```

Related Topics

- Configuring FCF for JDBC/OCI and JDBC-Thin Driver Clients
  You can enable FCF for Universal Connection Pool or Implicit Connection Cache.
- *Oracle Universal Connection Pool Developer's Guide*

Configuring JDBC-Thin Clients for Application Continuity for Java

This `oracle.jdbc.datasource.impl.OracleDataSource` data source is a JDBC-thin data source that Application Continuity requires for Java.

This data source serves as the connection factory that produces new physical JDBC connections, for both Universal Connection Pool and Oracle WebLogic Server Active GridLink for Oracle RAC data sources. The JDBC replay driver maintains a history of calls during a client conversation with Oracle AI Database. Following any outage of the session caused by a loss of database service, planned or unplanned, under the direction of the database, the JDBC replay driver attempts to rebuild the non-transactional and transactional database session states, so that the outage appears as a delayed processing.

Application Continuity for Java is supported in the following configurations:

- JDBC applications using Oracle JDBC data source and using neither Universal Connection Pool or Oracle WebLogic Server Active GridLink—typical third-party, JDBC-based connection pools
- JDBC applications using Universal Connection Pool data sources—standalone or third-party application servers configured to use a Universal Connection Pool data source
- JDBC applications using only Oracle WebLogic Server Active GridLink but not Universal Connection Pool data sources—typical Oracle WebLogic Server J2EE cases

To configure JDBC-thin clients to use the JDBC Replay Driver:

1. Ensure that you are using an application that is certified for replay.
2. Use SRVCTL to create a service for use by the application, if one does not already exist. Set the `-failovertype` parameter to `TRANSACTION` and the `-commit_outcome` parameter to `TRUE` for this service.
3. Configure the connection element using the `PoolDataSource` object, as shown in the following example:

```
PoolDataSource rds = PoolDataSourceFactory.getPoolDataSource();
rds.setConnnectionPoolName("replayExample");
rds.setONSConfiguration("nodes=racnode1:4200,racnode2:4200");
rds.setFastConnectionFailoverEnabled(true);
rds.setConnectionFactoryClassName("oracle.jdbc.datasource.impl.OracleDataSo
urce");

Connection conn = rds.getConnection();
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

4. When connecting to the database, use a URL that can access all instances offering the service.

Related Topics

- About Application Continuity
  Application Continuity provides high availability for your application workloads.
- Creating Services for Application Continuity and Transaction Guard
  To configure services for Application Continuity, when you create a service using SRVCTL, set the `-failovertype` parameter to `TRANSACTION` and `-commit_outcome` to `TRUE`.
- Configuring FCF for JDBC/OCI and JDBC-Thin Driver Clients
  You can enable FCF for Universal Connection Pool or Implicit Connection Cache.
- *Oracle AI Database JDBC Developer's Guide*
- *Oracle Universal Connection Pool Developer's Guide*

> **See Also**
>
> *Oracle AI Database JDBC Developer's Guide* for information about configuring Transaction Guard without enabling Application Continuity

Configuring JDBC-Thin Clients for Transaction Guard

Transaction Guard provides a protocol and a generic tool for applications to use for at-most-once processing in case of planned and unplanned outages.

Applications use the logical transaction ID to determine the outcome of the last transaction open in a database session following an outage. Without Transaction Guard, end users or applications that attempt to retry operations following outages can cause logical corruption by committing duplicate transactions or committing transactions out of order.

Related Topics

- *Oracle AI Database JDBC Developer's Guide*
- *Oracle AI Database Development Guide*
- *Oracle Call Interface Developer's Guide*

Enabling OCI Clients for Fast Connection Failover

Complete the procedure that enables Fast Connection Failover (FCF) by registering to receive notifications for Oracle RAC high availability Fast Application Notification (FAN) events.

Oracle Call Interface (OCI) clients can enable FCF by registering to receive notifications for Oracle Real Application Clusters (Oracle RAC) high availability FAN events and responding when events occur. Using FCF improves the session failover response time in OCI applications and also removes connections to nonfunctioning instances from connection and session pools. FCF can be used in OCI applications that also use TAF, OCI drivers (including your own connection pools), OCI connection pool, and OCI session pools. FAN is posted over the Oracle Notification Service for both high availability and load balancing events.

To use FCF, you must use a service with FAN enabled. FAN is published over Oracle Notification Service. Client applications can also register callbacks that are used whenever an event occurs. This reduces the time that it takes to detect a connection failure.

Copyright © 1999, 2025, Oracle and/or its affiliates.

During DOWN event processing, OCI:

- Terminates affected connections at the client and returns an error
- Removes connections from the OCI connection pool and the OCI session pool—the session pool maps each session to a physical connection in the connection pool, and there can be multiple sessions for each connection
- Fails over the connection if you have configured TAF. If TAF is not configured, then the client only receives an error if the instance it is connected to fails.

If your application is using TAF, then you must enable the TAF properties for the service using SRVCTL or Oracle Enterprise Manager. Configure your OCI client applications to connect to an Oracle RAC database using the configured service.

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

Example 5-1 Configuring FCF for OCI Clients

OCI applications must connect to an Oracle RAC instance to enable HA event notification. Furthermore, these applications must perform the following steps to configure FCF for an OCI client:

1. Configure the service for your OCI connection pool to enable FAN, connection load balancing, and run-time connection load balancing, as shown in the following example:

   ```
   $ srvctl modify service -db crm -service ociapp.example.com -notification
   TRUE
   ```

2. Link the application with a thread library.
3. After linking with the thread library, the applications can register a callback that is invoked whenever a FAN event occurs.

Related Topics

- *Oracle AI Database Net Services Administrator's Guide*
- *Oracle Call Interface Developer's Guide*

Enabling OCI Clients for Run-time Connection Load Balancing

Oracle Call Interface (OCI) session pooling enables multiple threads of an application to use a dynamically managed set of pre-created database sessions.

In connection pooling, the pool element is a connection, but in session pooling, the pool element is a session. Oracle AI Database continually reuses the sessions in the session pool to form nearly permanent channels to the instances, thus saving the overhead of creating and closing sessions every time applications need them.

Run-time connection load balancing is enabled by default in Oracle AI Database. For Oracle RAC environments, session pools use service metrics received from the Oracle RAC load balancing advisory through Fast Application Notification (FAN) events to balance application session requests. The work requests coming into the session pool can be distributed across the instances of Oracle RAC offering a service, using the current service performance.

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> Run-time connection load balancing is basically routing work requests to sessions in a session pool that can best serve the work. It comes into effect when selecting a session from an existing session pool. Thus, run-time connection load balancing is a very frequent activity.

Example 5-2 Configuring OCI Clients to Receive Load Balancing Advisory FAN Events

For Oracle RAC environments, session pools use service metrics received from the Oracle RAC load balancing advisory through Fast Application Notification (FAN) events to balance application session requests. To enable your application to receive the service metrics based on the service time, ensure that you configure FAN, the load balancing advisory goal (`-rlbgoal` parameter), and the connection load balancing goal (`-clbgoal` parameter) for a service that is used by the session pool, as shown in the following example:

```
$ srvctl modify service -db crm -service ociapp.example.com -rlbgoal
SERVICE_TIME
  -clbgoal LONG -notification TRUE
```

Related Topics

- *Oracle Call Interface Developer's Guide*

Configuring OCI Clients to use Transaction Guard

OCI supports FAN messages and Transaction Guard. FAN is designed to quickly notify an OCI-based application of outages at the node, database, instance, service, and public network levels.

Once notified of the failure, an application can leverage Transaction Guard to reliably determine the outcome of the last in-flight transaction.

Transaction Guard avoids the costs of ambiguous errors that lead to user frustration, customer support calls, and lost opportunities. Transaction Guard is safer and performs better, with lower overheads, than home grown solutions for a known outcome.

Related Topics

- Fast Application Notification (FAN)
  The Oracle RAC high availability framework monitors a database and its services, and sends Fast Application Notification (FAN) event notifications.
- Enabling Clients for Oracle RAC
  Learn how FAN is integrated with Oracle Clients, and how to enable FAN events for the several specific client development environments.
- *Oracle Call Interface Developer's Guide*

Enabling ODP.NET Clients to Receive FAN High Availability Events

ODP.NET connection pools can subscribe to FAN HA notifications that indicate when nodes, services, and service members are down.

After a `DOWN` event, Oracle AI Database cleans up sessions in the connection pool that go to the instance and ODP.NET proactively removes connections that are no longer valid. ODP.NET establishes additional connections to existing Oracle RAC instances if the removal of invalid

Copyright © 1999, 2025, Oracle and/or its affiliates.

connections reduces the total number of connections to below the value for the `Min Pool Size` parameter.

When connecting to Oracle AI Database, ODP.NET uses Oracle Notification Service, rather than Advanced Queuing.

Enable Fast Connection Failover for ODP.NET connection pools by subscribing to FAN high availability events. To enable Fast Connection Failover, include "`HA Events=true`" and "`pooling=true`" (the default value) in the connection string, as shown in the following example where *`user_name`* is the name of the database user and *`password`* is the password for that user:

```
con.ConnectionString =
   "User Id=user_name;Password=password;Data Source=odpnet;" +
   "Min Pool Size=10;Connection Lifetime=120;Connection Timeout=60;" +
   "HA Events=true;Incr Pool Size=5;Decr Pool Size=2";
```

Related Topics

- *Oracle Data Provider for .NET Developer's Guide*
- Fast Application Notification (FAN)
  The Oracle RAC high availability framework monitors a database and its services, and sends Fast Application Notification (FAN) event notifications.

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

When connecting to Oracle AI Database, ODP.NET uses Oracle Notification Service, rather than Advanced Queuing.

Use the following procedure to enable ODP.NET clients or applications to receive FAN load balancing advisory events:

1. Enable Oracle Notification Service notifications by using SRVCTL, and set the run-time load balancing goal, as shown in the following example:

   ```
   $ srvctl modify service -db crm -service odpapp.example.com
     -notification TRUE -clbgoal LONG -rlbgoal SERVICE_TIME
   ```

2. Ensure Oracle Notification Service (ONS) is configured for FAN events including run-time load balancing advice.
3. To take advantage of load balancing events with ODP.NET connection pools, set the load balancing attribute in the ConnectionString to `TRUE` (the default is `FALSE`). You can do this at connect time. This only works if you are using connection pools, or when the pooling attribute is set to `TRUE` which is the default.

   The following example demonstrates how to configure the ConnectionString to enable load balancing, where *`user_name`* is the name of the user and *`password`* is the password:

   ```
   con.ConnectionString =
     "User Id=user_name;Password=password;Data Source=odpapp;" +
     "Min Pool Size=10;Connection Lifetime=120;Connection Timeout=60;" +
     "Load Balancing=true;Incr Pool Size=5;Decr Pool Size=2";
   ```


Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> ODP.NET does not support connection redistribution when a node starts (UP events). However, if you have enabled failover on the server-side, then ODP.NET can migrate connections to newly available instances.

Related Topics

- srvctl modify service
  Modifies a service configuration.
- *Oracle Data Provider for .NET Developer's Guide*
- Fast Application Notification (FAN)
  The Oracle RAC high availability framework monitors a database and its services, and sends Fast Application Notification (FAN) event notifications.

Configuring ODP.NET Clients to use Transaction Guard

ODP.NET supports FAN messages and Transaction Guard. FAN is designed to quickly notify an ODP.NET-based application of outages at the node, database, instance, service, and public network levels.

Once notified of the failure, an application can leverage Transaction Guard to reliably determine the outcome of the last in-flight transaction.

Transaction Guard avoids the costs of ambiguous errors that lead to user frustration, customer support calls, and lost opportunities. Transaction Guard is safer and performs better, with lower overheads, than home grown solutions for a known outcome.

Related Topics

- Fast Application Notification (FAN)
  The Oracle RAC high availability framework monitors a database and its services, and sends Fast Application Notification (FAN) event notifications.
- Administering Services
  Learn how to create and administer services, and perform other service-related tasks using Oracle Enterprise Manager and the SRVCTL utility.
- Creating Services for Application Continuity and Transaction Guard
  To configure services for Application Continuity, when you create a service using SRVCTL, set the `-failovertype` parameter to `TRANSACTION` and `-commit_outcome` to `TRUE`.
- *Oracle Data Provider for .NET Developer's Guide*

Distributed Transaction Processing in Oracle RAC

Learn how Oracle Real Application Clusters (Oracle RAC) supports global (XA) transactions and DTP processing

The X/Open Distributed Transaction Processing (DTP) architecture defines a standard architecture or interface that enables multiple application programs (APs) to share resources provided by multiple, and possibly different, resource managers (RMs). It coordinates the work between APs and RMs into global transactions.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Overview of XA Transactions and Oracle RAC
  A global (XA) transaction can span Oracle RAC instances by default, allowing any application that uses the Oracle XA library to take full advantage of the Oracle RAC environment to enhance the availability and scalability of the application.
- Using Global Transactions and XA Affinity for XA Transactions
  To provide improved application performance with distributed transaction processing (DTP) in Oracle RAC, you can take advantage of XA affinity.
- Using Services with XA Transactions on Oracle RAC
  Most applications using XA on Oracle RAC can use uniform or (all preferred) services with XA affinity provided by the connection pool or transaction processing monitor.
- Configuring Services for XA Applications
  To create distributed transaction processing (DTP) services for distributed transaction processing, use this procedure.
- Relocating Services in Administrator-Managed Databases
  Beginning with Oracle Real Application Clusters 11*g* release 1 (11.1), global transactions and XA affinity replace the need for distributed transaction processing (DTP) services.

Overview of XA Transactions and Oracle RAC

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

GTX*n* background processes support XA transactions in an Oracle RAC environment. The `GLOBAL_TXN_PROCESSES` initialization parameter, which is set to `1` by default, specifies the initial number of GTX*n* background processes for each Oracle RAC instance. Use the default value for this parameter clusterwide to allow distributed transactions to span multiple Oracle RAC instances. Using the default value allows the units of work performed across these Oracle RAC instances to share resources and act as a single transaction (that is, the units of work are *tightly coupled*). It also allows 2PC requests to be sent to any node in the cluster.

Before Oracle RAC 11g release 1 (11.1), the way to achieve tight coupling in Oracle RAC was to use distributed transaction processing (DTP) services, that is, services whose cardinality (one) ensured that all tightly-coupled branches landed on the same instance—regardless of whether load balancing was enabled. If the XA application does not use suspend and resume on the same transaction branch, and does not issue savepoints that span branches, then tightly coupled XA transactions no longer require the special type of singleton services to be deployed on Oracle RAC databases. If your application cannot determine whether a transaction branch has been suspended and resumed, then the application must continue to use DTP services or preferably use XA affinity.

XA affinity (placing all branches of the same XA transaction at the same Oracle RAC instance) is a requirement when suspending and resuming the same XA branch or if using savepoints across branches. It also provides much better performance because different transactions can be balanced. XA affinity is available with Oracle WebLogic Server Active GridLink for Oracle RAC, JDBC Universal Connection Pool, and Oracle Tuxedo. XA affinity is also standard for RedHat JBoss, IBM WebSphere, and IBM Liberty.

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> Transaction Processing monitors with XA work best when using services with one preferred instance and many available instances. Oracle does not recommend using the DTP setting for Oracle AI Database.

Related Topics

- Using Global Transactions and XA Affinity for XA Transactions
  To provide improved application performance with distributed transaction processing (DTP) in Oracle RAC, you can take advantage of XA affinity.
- *Oracle AI Database Reference*

Using Global Transactions and XA Affinity for XA Transactions

To provide improved application performance with distributed transaction processing (DTP) in Oracle RAC, you can take advantage of XA affinity.

Using XA affinity, you can direct all branches of a distributed transaction to a single instance in the cluster. To implement XA affinity you can use an application server that provides XA affinity, such as WebLogic Server and Universal Connection Pool. If your application server does not have XA affinity, then you can also use singleton services across Oracle RAC.

Connection pools at the application server tier that load balance across multiple connections to an Oracle RAC database use XA affinity to ensure that all tightly-coupled branches of a global distributed transaction run on only one Oracle RAC instance. When using a connection pool with XA affinity, your services using XA can span Oracle RAC. This is also true in distributed transaction environments using protocols such as X/Open Distributed Transaction Processing or the Microsoft Distributed Transaction Coordinator.

To enhance the performance of distributed transactions, use services with one preferred instance. A singleton service runs on one Oracle RAC instance at time in an Oracle RAC database. This service still allows draining for maintenance, so has better high-availability characteristics than an older DTP service. To load balance across the cluster, it is better to have several groups of smaller application servers with each group directing its transactions to a single service, or set of services, than it is to have one or two larger application servers. Using singleton services, global distributed transactions performed through the services have their tightly-coupled branches running on a single Oracle RAC instance. This has the following benefits:

- The changes are available locally within one Oracle RAC instance when tightly coupled branches need information about changes made by each other
- Relocation and failover of services are fully supported using global transactions
- By using more singleton services than there are Oracle RAC instances, Oracle AI Database can balance the load by services across all of the Oracle RAC database instances

> **Note**
>
> Oracle does not recommend using the DTP setting for Oracle AI Database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Using Services with XA Transactions on Oracle RAC

Most applications using XA on Oracle RAC can use uniform or (all preferred) services with XA affinity provided by the connection pool or transaction processing monitor.

The application may also use singleton services to provide XA affinity.

When using singleton services, to leverage all of the instances in a cluster, create one or more singleton services for each Oracle RAC instance that hosts distributed transactions. Choose different services for application servers to balance the workload among the Oracle RAC database instances. Because all of the branches of a distributed transaction are on one instance, you can leverage all of the instances to balance the load of many transactions through multiple singleton services, thereby maximizing application throughput.

If you add or delete nodes from your cluster database, then you may have to identify and relocate services to ensure that you maintain optimum performance levels. Using singleton services, current work can complete.

Configuring Services for XA Applications

To create distributed transaction processing (DTP) services for distributed transaction processing, use this procedure.

1. Create a singleton service using Oracle Enterprise Manager or SRVCTL.

   For an administrator-managed database, define only one instance as the preferred instance. You can have as many available instances as you want, for example:

   ```
   $ srvctl add service -db crm -service xa_01.example.com -preferred RAC01
     -available RAC02,RAC03
   ```

2. Set the DTP parameter (`-dtp`) for the service to `TRUE` (the default value is `FALSE`). You can use Oracle Enterprise Manager or SRVCTL to modify the DTP property of the singleton service. The following example shows how to modify the `xa_01.example.com` service using SRVCTL:

   ```
   $ srvctl modify service -db crm -service xa_01.example.com -dtp TRUE
   ```

   > **Note**
   >
   > If the application does require DTP services, then use the `-dtp` parameter. If not, then use the preceding example with no `-dtp` parameter.

Related Topics

- srvctl add service
  Adds services to a database and assigns them to instances.
- srvctl modify service
  Modifies a service configuration.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Relocating Services in Administrator-Managed Databases

Beginning with Oracle Real Application Clusters 11*g* release 1 (11.1), global transactions and XA affinity replace the need for distributed transaction processing (DTP) services.

Most XA deployments should be using global transactions with XA affinity for improved load balancing and flexibility rather than the DTP attribute.

If services migrate to other instances, then you might have to force the relocation of the service back to the preferred instance after it is restarted to evenly re-balance the load on all of the available hardware. You can use data from the `GV$ACTIVE_SERVICES` view to determine whether you need to relocate the DTP service.

Oracle RAC Sharding

Oracle RAC Sharding creates an affinity between table partitions and Oracle RAC instances, and routes database requests that specify a partitioning key to the instance that logically holds the corresponding partition.

Oracle routes database requests to Oracle RAC instances in such a way that each instance always gets requests for a disjoint subset of rows in the database, which creates affinity of rows with instances. The affinity leads to higher Oracle RAC performance and scalability because of improved cache locality and reduced inter-node synchronization and block pings.

Sharding for Oracle RAC affinity uses client and server-side support for key-based routing, which is part of the Oracle AI Database sharding. An application that supplies a sharding key in the database using the same API implemented for sharding support in Oracle connection pools (such as Universal Connection Pool, OCI), in the same way it is done for sharding, utilizes key-based routing and, by doing so, enables Oracle RAC affinity.

Application changes that are required to supply the sharding key, do not have to affect all modules of the application. Changes can only be applied to a few frequently processed database requests. Requests that do not provide the sharding key in the connect string are routed based on the load-balancing policy. Keyless requests do not have any negative impact on data affinity because of the explicit ownership assignment of data objects to instances.

> **Note**
>
> Oracle only supports Oracle RAC affinity for partitioned tables. You can partition a table using any supported method without making changes to the database schema to enable this feature and then run the `ALTER SYSTEM ENABLE AFFINITY` command.

If you want to make changes to your applications to take advantage of affinity-enabling routing, then you may also take advantage of sharding when data is distributed across multiple independent databases. You can later move to distributed sharding if you require extreme scalability or fault isolation.

Related Topics

- *Oracle AI Database SQL Language Reference*
- *Oracle AI Database Net Services Administrator's Guide*
- *Oracle Globally Distributed Database Guide*
- *Oracle AI Database JDBC Developer's Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

- *Oracle Call Interface Developer's Guide*

Automatic Workload Repository

The Automatic Workload Repository (AWR) collects, processes, and maintains performance statistics for the database.

The gathered data can be displayed in both reports and views. If you use services with your database, then AWR tracks metrics at the service level.

Metrics can be measured against a variety of units, including time, transactions, or database calls. For example, the number of database calls per second is a metric. Server generated alerts can be placed on these metrics when they exceed or fail to meet user-specified thresholds. The database or system administrator can then respond, for example, by:

- Using the Oracle AI Database Resource Manager to configure the service level for one service to have priorities relative to other services
- Stopping overloaded processes
- Modifying a service level requirement
- Implementing recovery scenarios in response to service quality changes

Using AWR metrics and performance alerts enables you to maintain continued service availability despite service level changes. It also enables you to measure the service quality provided by the database services.

The AWR ensures that the Oracle Clusterware workload management framework and the database resource manager have persistent and global representations of performance data. This information helps Oracle AI Database schedule job classes by service and to assign priorities to consumer groups. If necessary, you can rebalance workloads manually with either Oracle Enterprise Manager or SRVCTL. You can also disconnect a series of sessions, but leave the service running.

> **Note**
>
> Oracle *does not* recommend using the DBMS_SERVICE package for use with services used by an Oracle RAC database. Use SRVCTL or Oracle Enterprise Manager to create database services for Oracle RAC.

Related Topics

- *Oracle AI Database Get Started with Performance Tuning*
- *Oracle AI Database Performance Tuning Guide*
- *Oracle AI Database PL/SQL Packages and Types Reference*

Measuring Performance by Service Using the Automatic Workload Repository

Services add a new dimension for performance tuning because workloads are visible and measurable, and therefore resource consumption and wait times are attributable by application.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Tuning by using "service and SQL" replaces tuning by "session and SQL" in the majority of systems where all sessions are anonymous and shared.

The AWR maintains performance statistics that include information about response time, throughput, resource consumption, and wait events for all services and work that a database performs. Oracle AI Database also maintains metrics, statistics, wait events, wait classes, and SQL-level traces for services. You can optionally augment these statistics by defining modules within your application to monitor certain statistics. You can also define the actions within those modules that business critical transactions should run in response to particular statistical values.

Enable module and action monitoring using the `DBMS_MONITOR` PL/SQL package. For example, for connections that use the `erp` service, the following command enables monitoring for the `exceptions pay` action in the `payroll` module:

```
EXECUTE DBMS_MONITOR.SERV_MOD_ACT_STAT_ENABLE(SERVICE_NAME => 'ERP',
   MODULE_NAME=> 'PAYROLL', ACTION_NAME => 'EXCEPTIONS PAY');
```

For connections that use the `erp` service, the following command enables monitoring for all actions in the `payroll` module:

```
EXECUTE DBMS_MONITOR.SERV_MOD_ACT_STAT_ENABLE(SERVICE_NAME => 'ERP',
   MODULE_NAME=> 'PAYROLL', ACTION_NAME => NULL);
```

Use the `DBA_ENABLED_AGGREGATIONS` view to verify that you have enabled monitoring for application modules and actions.

Statistics aggregation and tracing by service are global in scope for Oracle RAC databases. In addition, these statistic aggregations are persistent across instance restarts and service relocations for both Oracle RAC and noncluster Oracle AI Database.

The service, module, and action names are visible in `V$SESSION`, `V$ACTIVE_SESSION_HISTORY`, and `V$SQL` views. The call times and performance statistics are visible in `V$SERVICE_STATS`, `V$SERVICE_EVENT`, `V$SERVICE_WAIT_CLASS`, `V$SERVICEMETRIC`, and `V$SERVICEMETRIC_HISTORY`. When you enable statistics collection for an important transaction, you can see the call speed for each service, module, and action name at each database instance using the `V$SERV_MOD_ACT_STATS` view.

The following sample SQL*Plus script provides service quality statistics for a five second interval. You can use these service quality statistics to monitor the service quality, to direct work, and to balance services across Oracle RAC instances:

```
SET PAGESIZE 60 COLSEP '|' NUMWIDTH 8 LINESIZE 132 VERIFY OFF FEEDBACK OFF
COLUMN service_name FORMAT A20 TRUNCATED HEADING 'Service'
COLUMN begin_time HEADING 'Begin Time' FORMAT A10
COLUMN end_time HEADING 'End Time' FORMAT A10
COLUMN instance_name HEADING 'Instance' FORMAT A10
COLUMN service_time HEADING 'Service Time|mSec/Call' FORMAT 999999999
COLUMN throughput HEADING 'Calls/sec'FORMAT 99.99
BREAK ON service_name SKIP 1
SELECT
    service_name
  , TO_CHAR(begin_time, 'HH:MI:SS') begin_time
  , TO_CHAR(end_time, 'HH:MI:SS') end_time
  , instance_name
  , elapsedpercall  service_time
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

```
  ,  callspersec  throughput
FROM
    gv$instance i
  , gv$active_services s
  , gv$servicemetric m
WHERE s.inst_id = m.inst_id
  AND s.name_hash = m.service_name_hash
  AND i.inst_id = m.inst_id
  AND m.group_id = 10
ORDER BY
   service_name
 , i.inst_id
 , begin_time ;
```

Automatic Workload Repository Service Thresholds and Alerts

To provide accountability for your required levels of service, you can use the Oracle Real Application Clusters (Oracle RAC) Automatic Workload Repository (AWR) service.

- About Automatic Workload Repository Service Thresholds and Alerts
  Learn how you can maintain the service quality delivery by using the Automatic Workload Repository (AWR) service.
- Example of Services and Thresholds Alerts
  In this task scenario, you need to check the thresholds for the payroll service. To obtain the threshold information, you can use the AWR report.
- Enable Service, Module, and Action Monitoring
  To enable performance data tracing for important modules and actions within each service, you can use the `V$SERV_MOD_ACT_STATS` view.

About Automatic Workload Repository Service Thresholds and Alerts

Learn how you can maintain the service quality delivery by using the Automatic Workload Repository (AWR) service.

Service level thresholds enable you to compare actual service levels against required levels of service. This provides accountability for the delivery or the failure to deliver an agreed service level. The end goal is a predictable system that achieves service levels. There is no requirement to perform as fast as possible with minimum resource consumption; the requirement is to meet the service quality.

AWR enables you to explicitly specify two performance thresholds for each service: the response time for calls (`ELAPSED_TIME_PER_CALL`), and the CPU time for calls (`CPU_TIME_PER_CALL`). The response time threshold indicates that the elapsed time for each user call for each service should not exceed a certain value, and the CPU time for calls threshold indicates that the time spent using the CPU for each call for each service should not exceed a certain value. Response time is a fundamental measure that reflects all delays and faults that might be blocking the call from running on behalf of the user. Response time can also indicate differences in node power across the nodes of an Oracle RAC database.

You must set these thresholds on each instance of an Oracle RAC database. The elapsed time and CPU time are calculated as the moving average of the elapsed, server-side call time. The AWR monitors the elapsed time and CPU time and publishes AWR alerts when the performance exceeds the thresholds. You can schedule actions using Oracle Enterprise Manager jobs for these alerts, or you can schedule actions to occur programmatically when the alert is received. You can respond to these alerts by changing the priority of a job, stopping

Copyright © 1999, 2025, Oracle and/or its affiliates.

overloaded processes, or by relocating, starting or stopping a service. This permits you to maintain service availability despite changes in demand.

Example of Services and Thresholds Alerts

In this task scenario, you need to check the thresholds for the payroll service. To obtain the threshold information, you can use the AWR report.

To prepare for checking the payroll service thresholds, you should compare the results from reports run over several successive intervals during which time the system is running optimally. For example, assume that for servers accessed by a payroll application, the AWR report runs each Thursday during the peak usage times of 1:00 p.m. to 5:00 p.m. The AWR report contains the response time, or elapsed database time, and the CPU consumption time, or CPU time, for calls for each server, including the `payroll` service. The AWR report also provides a breakdown of the work done and the wait times that are contributing to the response times.

Using `DBMS_MONITOR`, you set a warning threshold for the elapsed time per call for the `payroll` service at 0.5 seconds (500000 microseconds). You also set a critical threshold for the elapsed time per call for the `payroll` service at 0.75 seconds (750000 microseconds).

Example 5-3 Adding Thresholds to Check for Service Quality

In this example, The commands add thresholds for the `payroll` service:

```
EXECUTE DBMS_SERVER_ALERT.SET_THRESHOLD(
METRICS_ID => DBMS_SERVER_ALERT.ELAPSED_TIME_PER_CALL
, warning_operator => DBMS_SERVER_ALERT.OPERATOR_GE
, warning_value => '500000'
, critical_operator => DBMS_SERVER_ALERT.OPERATOR_GE
, critical_value => '750000'
, observation_period => 30
, consecutive_occurrences => 5
, instance_name => NULL
, object_type => DBMS_SERVER_ALERT.OBJECT_TYPE_SERVICE
, object_name => 'payroll');
```

To verify that the threshold configuration is set on all the instances, you can use the following `SELECT` statement:

```
SELECT METRICS_NAME, INSTANCE_NAME, WARNING_VALUE, CRITICAL_VALUE,
OBSERVATION_PERIOD FROM dba_thresholds ;
```

Enable Service, Module, and Action Monitoring

To enable performance data tracing for important modules and actions within each service, you can use the `V$SERV_MOD_ACT_STATS` view.

To see how you can trace performance data for modules and actions within services using the `V$SERV_MOD_ACT_STATS` view, suppose you want to set the following performance checks:

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

The following commands show how to enable the module and action monitoring for the services:

```
EXECUTE DBMS_MONITOR.SERV_MOD_ACT_STAT_ENABLE(service_name => 'erp',
module_name=>
 'payroll', action_name => 'exceptions pay');
EXECUTE DBMS_MONITOR.SERV_MOD_ACT_STAT_ENABLE(service_name => 'erp',
module_name=>
 'payroll');
EXECUTE DBMS_MONITOR.SERV_MOD_ACT_STAT_ENABLE(service_name => 'hot_batch',
module_name =>'posting');
```

To verify monitoring is enabled for the service, module, and actions, use the following `SELECT` statement:

```
COLUMN AGGREGATION_TYPE FORMAT A21 TRUNCATED HEADING 'AGGREGATION'
COLUMN PRIMARY_ID FORMAT A20 TRUNCATED HEADING 'SERVICE'
COLUMN QUALIFIER_ID1 FORMAT A20 TRUNCATED HEADING 'MODULE'
COLUMN QUALIFIER_ID2 FORMAT A20 TRUNCATED HEADING 'ACTION'
SELECT * FROM DBA_ENABLED_AGGREGATIONS ;
```

The output is similar to the following:

```
AGGREGATION            SERVICE               MODULE        ACTION
------------           --------------------  ----------    -------------
SERVICE_MODULE_ACTION  erp                   payroll       exceptions pay
SERVICE_MODULE         erp                   payroll
SERVICE_MODULE         hot_batch             posting
```

Using Oracle Services

To manage workloads or a group of applications, you can define services that you assign to a particular application, or to a subset of an application's operations, or you can group work by type under services.

To understand how you can manage workloads by defining services or groups, consider this example: To connect to the database, you configure one service for online users, while batch processing uses another service, and reporting uses yet another service. You can thus track workloads by services.

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

Service Deployment Options

Learn about services in Oracle Real Application Clusters (Oracle RAC) databases, and how to define and deploy services.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Service Usage in an Oracle RAC Database
  Learn how several database features use services for an Oracle Real Application Clusters (Oracle RAC) database.
- Service Characteristics
  When you create new services for your Oracle Real Application Clusters (Oracle RAC) database, you should define the automatic workload management characteristics for each service.
- Default Service Connections
  Oracle Real Application Clusters (Oracle RAC) includes default service connections, which you should not attempt to manage.
- Restricted Service Registration
  This feature allows listener registration only from local IP addresses, by default, and provides the ability to configure and dynamically update a set of IP addresses or subnets from which registration requests are allowed by the listener.

Service Usage in an Oracle RAC Database

Learn how several database features use services for an Oracle Real Application Clusters (Oracle RAC) database.

Services provide location transparency. A service name can identify multiple database instances, and an instance can belong to multiple services.

- Oracle Clusterware Resources for a Service
- Database Resource Manager Consumer Group Mappings for Services
- Performance Monitoring by Service with AWR
- Parallel Operations and Services
  By default, in an Oracle RAC environment, a SQL statement processed in parallel can run across all of the nodes in the cluster.
- Oracle GoldenGate and Oracle RAC
  Oracle GoldenGate takes advantage of Oracle RAC features.

Oracle Clusterware Resources for a Service

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

Database Resource Manager Consumer Group Mappings for Services

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

Performance Monitoring by Service with AWR

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

Related Topics

- *Oracle AI Database Performance Tuning Guide*

Parallel Operations and Services

By default, in an Oracle RAC environment, a SQL statement processed in parallel can run across all of the nodes in the cluster.

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

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

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

Oracle GoldenGate and Oracle RAC

Oracle GoldenGate takes advantage of Oracle RAC features.

When Oracle GoldenGate is configured in an Oracle RAC environment, each queue table has an owning instance. If the instance that hosts a queue table fails, another instance in the Oracle RAC database becomes the owning instance for the queue table, allowing Oracle GoldenGate to continue operating.

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

Service Characteristics

When you create new services for your Oracle Real Application Clusters (Oracle RAC) database, you should define the automatic workload management characteristics for each service.

- Service Name
  The service name is used by clients to connect to one or more instances.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Service Edition
  To upgrade an application's objects while these objects are in use, you can use edition-based redefinition of database objects.
- Service Management Policy
  When you use Oracle Clusterware to manage your database, you can configure startup options for each individual database service when you add the service using the `srvctl add service` command with the `-policy` parameter.
- Database Role for a Service
  If you configured Oracle Data Guard in your environment, then you can define a role for services when you add or modify a service using SRVCTL and the `-role` parameter with the appropriate command.
- Instance Preference
  When you define a service for an administrator-managed database, you define which instances normally support that service using SRVCTL with the `-preferred` parameter.
- Service Co-location
  Oracle RAC routes clients with the same `COLOCATION_TAG` to the same database instance, when possible.
- Load Balancing Advisory Goal for Run-time Connection Load Balancing
  To provide better service to users, take advantage of load balancing advisory events to manage workloads.
- Connection Load Balancing Goal
  Oracle Net Services provide connection load balancing to enable you to spread user connections across all of the instances that are supporting a service.
- Distributed Transaction Processing
  Learn about Oracle XA applications in Oracle Real Application Clusters (Oracle RAC).

Service Name

The service name is used by clients to connect to one or more instances.

Each service has a service name. The service name must be unique throughout your system.

The service name must meet the following qualifications:

- The name must consist of alphanumeric characters (a-z, A-Z, 0-9), underscore (_), and hyphen (-).
- The service domain portion of the name must consist of alphanumeric characters (a-z, A-Z, 0-9), underscore (_), dollar sign ($), number sign (#), period (.), and hyphen (-)
- A domain qualified service name is of the form *service_name*.*service_domain*.
- You cannot create a service with the same name as the database default service, which is *db_unique_name*.*db_domain*.

Service Edition

To upgrade an application's objects while these objects are in use, you can use edition-based redefinition of database objects.

You can set the **edition** attribute of a database service when you create it, or modify an existing service to set the edition. When you set the service edition, connections that use this service use this edition as the initial session edition. If the service does not specify the edition name, then the initial session edition is the database default edition.

Copyright © 1999, 2025, Oracle and/or its affiliates.

In this example, you use SRVCTL to set the service edition:

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

Service Management Policy

When you use Oracle Clusterware to manage your database, you can configure startup options for each individual database service when you add the service using the `srvctl add service` command with the `-policy` parameter.

If you set the management policy for a service to `AUTOMATIC` (the default), then the service starts automatically when you start the database with SRVCTL. If you set the management policy to `MANUAL`, then the service does not automatically start, and you must manually start it with SRVCTL. A `MANUAL` setting does not prevent Oracle Clusterware from monitoring the service when it is running and restarting it if a failure occurs. Before Oracle RAC 11*g* release 2 (11.2), all services worked as though they were defined with a `MANUAL` management policy.

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

> **Note**
>
> When you use automatic services in an administrator-managed database, during planned database startup, services may start on the first instances to start rather than their preferred instances, provided that the started instances are in the (combined) preferred and available services list.

Related Topics

- srvctl add service
  Adds services to a database and assigns them to instances.

Database Role for a Service

If you configured Oracle Data Guard in your environment, then you can define a role for services when you add or modify a service using SRVCTL and the `-role` parameter with the appropriate command.

When you specify a role for a service, Oracle Clusterware automatically starts the service only when the database role matches the role you specified for the service. Valid roles are `PRIMARY`, `PHYSICAL_STANDBY`, `LOGICAL_STANDBY`, and `SNAPSHOT_STANDBY` and you can specify more than one role for a service.

> **Note**
>
> - The service role only controls automatic startup of services. Using SRVCTL to manually start a service will succeed even if the roles do not match.
> - If the service role specifies only read-only database roles (`*STANDBY`), then changes to the service resource attributes with SRVCTL cannot take effect in the database until you start the service on a read-write database at least once. This is because `DBMS_SERVICE.MODIFY_SERVICE` cannot work in a read-only database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Redo Apply (physical standby database) can run on all or some standby instances that you can configure. This enables Redo Apply performance to scale, if necessary, by adding additional standby instances.

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

Related Topics

- *Oracle Data Guard Concepts and Administration*

Instance Preference

When you define a service for an administrator-managed database, you define which instances normally support that service using SRVCTL with the `-preferred` parameter.

These are known as the *preferred* instances. You can also define other instances to support a service if the service's preferred instance fails using SRVCTL with the `-available` parameter. These are known as *available* instances.

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

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

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

- The service is running on the desired number of instances.
- Maintaining the service on the current instance provides a higher level of service availability.
- Not moving the service back to the initial preferred instance prevents a second outage.

Starting with Oracle Database release 19.3, if you specify `yes` for the `-failback` attribute of a service, then, after failing over to an available instance when the last preferred instance went down, the service transfers back to a preferred instance when one becomes available. For earlier releases, you can automate fail back to the preferred instance by using FAN callouts.

Related Topics

- Tools for Administering Oracle RAC
  The tools most commonly used to managed Oracle Real Application Clusters (Oracle RAC) databases and instances are the SRVCTL utility, Oracle Enterprise Manager, and SQL*Plus.

Service Co-location

Oracle RAC routes clients with the same `COLOCATION_TAG` to the same database instance, when possible.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Co-location of sessions on the same instance can help decrease inter-instance communication and increase performance for workloads that benefit from being run in the same instance. You configure the `COLOCATION_TAG` in the `CONNECT_DATA` parameter of the TNS connect string used by the service as described in *Oracle AI Database Net Services Reference*.

Related Topics

- COLOCATION_TAG

Load Balancing Advisory Goal for Run-time Connection Load Balancing

To provide better service to users, take advantage of load balancing advisory events to manage workloads.

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

The load balancing advisory also recommends how much of the workload should be sent to that instance. The goal determines whether connections are made to the service based on best service quality (how efficiently a single transaction completes) or best throughput (how efficiently a complete job or long-running query completes).

Connection Load Balancing Goal

Oracle Net Services provide connection load balancing to enable you to spread user connections across all of the instances that are supporting a service.

For each service, you can use SRVCTL to define the method you want the listener to use for load balancing by setting the connection load balancing goal, specified with the `-clbgoal` parameter. Connections are classified as `LONG`, which tells the listener to use session count, or `SHORT`, which tells the listener to use response-time or throughput statistics.

If load balancing advisory is enabled (the `-rlbgoal` parameter *does not* equal `NONE`), then connection load balancing attempts to use load balancing advisory (whether load balancing goal is set to `SHORT` or `LONG`). If load balancing is set to `SHORT`, then it uses the `GOODNESS` value of a service to try to prevent all connection requests from going to one instance. If load balancing is set to `LONG`, then it uses `run queue length` if the service is singleton, or `session count` if the service is uniform.

> **Note**
>
> Oracle recommends `-clbgoal LONG` for both server-side and client-side load balancing.

Distributed Transaction Processing

Learn about Oracle XA applications in Oracle Real Application Clusters (Oracle RAC).

Oracle XA is the Oracle implementation of the X/Open Distributed Transaction Processing (DTP) XA interface. Oracle XA applications have unique requirements. Oracle provides global transactions across Oracle RAC. For best performance, Oracle recommends that you use XA

Copyright © 1999, 2025, Oracle and/or its affiliates.

affinity (all branches at the same instance) for most transactions, and global transactions when needed. You can use XA affinity with connection pools, such as Universal Connection Pools and WebLogic Server. You can also use singleton services that you create using `SRVCTL`. If you want to suspend and resume the same Oracle XA branch, then you also use SRVCTL to set the distributed transaction processing parameter (`-dtp`) to `TRUE`. However, Oracle recommends that you do not do this in general, because managing branches this way does not offer rolling planned maintenance.

Related Topics

- Distributed Transaction Processing in Oracle RAC
  Learn how Oracle Real Application Clusters (Oracle RAC) supports global (XA) transactions and DTP processing
- srvctl add service
  Adds services to a database and assigns them to instances.

Default Service Connections

Oracle Real Application Clusters (Oracle RAC) includes default service connections, which you should not attempt to manage.

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

- `SYS$BACKGROUND` is used by the background processes only
- `SYS$USERS` is the default service for user sessions that are not associated with any application service

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

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

Restricted Service Registration

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

Security is a high priority to all enterprises, and network security and controlling access to the database is a critical component of overall security endeavours. Database Instance registration with a listener succeeds only when the request originates from a valid node. The network administrator can specify a list of valid nodes, excluded nodes, or disable valid node checking. The list of valid nodes explicitly lists the nodes and subnets that can register with the database. The list of excluded nodes explicitly lists the nodes that cannot register with the database. The

Copyright © 1999, 2025, Oracle and/or its affiliates.

control of dynamic registration results in increased manageability and security of Oracle RAC deployments.

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

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

> **Note**
>
> Starting with Oracle Grid Infrastructure 12c, for a SCAN listener, if the `VALID_NODE_CHECKING_REGISTRATION_listener_name` and `REGISTRATION_INVITED_NODES_listener_name` parameters are set in the `listener.ora` file, then the listener agent overwrites these parameters.

If you use the SRVCTL utility to set the `invitednodes` and `invitedsubnets` values, then the listener agent automatically sets `VALID_NODE_CHECKING_REGISTRATION_listener_name` to SUBNET and sets `REGISTRATION_INVITED_NODES_listener_name` to the specified list in the `listener.ora` file.

For other listeners managed by CRS, the listener agent sets `VALID_NODE_CHECKING_REGISTRATION_listener_name` in the `listener.ora` file only if it is not already set in the `listener.ora` file. The SRVCTL utility does not support setting the `invitednodes` and `invitedsubnets` values for a non-SCAN listener. The listener agent does not update `REGISTRATION_INVITED_NODES_listener_name` in the `listener.ora` file for a non SCAN listener.

Related Topics

- *Oracle AI Database Net Services Administrator's Guide*

Administering Services

Learn how to create and administer services, and perform other service-related tasks using Oracle Enterprise Manager and the SRVCTL utility.

> **Note**
>
> You can also use the `DBMS_SERVICE` package to create or modify services and service attributes, but `SRVCTL` will override any settings made using this package. Oracle does not recommend using the `DBMS_SERVICE` package with services used either by an Oracle Real Application Clusters (Oracle RAC) database, or with Oracle Restart, or when Oracle Clusterware is managing a single-instance database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Overview of Service Administration
  When you create and administer services, you are dividing the work that your database performs into manageable units.
- Administering Services with Oracle Enterprise Manager
  The Cluster Managed Database Services page is the main page for beginning all tasks related to services.
- Administering Services with SRVCTL
  Learn how to use SRVCTL to perform service administration on an Oracle Real Application Clusters (Oracle RAC) database.

Overview of Service Administration

When you create and administer services, you are dividing the work that your database performs into manageable units.

The goal of using services is to achieve optimal utilization of your database infrastructure. You can create and deploy services based on business requirements. Oracle AI Database can measure the performance for each service. Using the `DBMS_MONITOR` package, you can define both the application modules within a service and the individual actions for a module and monitor thresholds for these actions, enabling you to manage workloads to deliver capacity on demand.

When you create new services for your database, you should define the automatic workload management characteristics for each service, as described in "Service Characteristics".

In addition to creating services, you can:

- **Delete a service**. You can delete services that you created. However, you cannot delete or modify the properties of the default database service that Oracle AI Database created.
- **Check the status of a service**. A service can be assigned different roles among the available instances. In a complex database with many services, you may not remember the details of every service. Therefore, you may have to check the status on an instance or service basis. For example, you may have to know the status of a service for a particular instance before you make modifications to that instance or to the Oracle home from which it runs.
- **Start or stop a service for a database or an instance**. A service must be started before it can be used for client connections to that instance. If you shut down your database, for example, by running the SRVCTL command `srvctl stop database -db` *`db_unique_name`* where *`db_unique_name`* is the name of the database you want to stop, then Oracle AI Database stops all services for that database. Depending on the service management policy, you may have to manually restart the services when you start the database. Both the `srvctl stop database` and `srvctl stop service` commands accept the `-force` option to forcibly disconnect connections. To drain sessions for planned outages do not use the `-force` option.
- **Map a service to a consumer group**. You can map services to Resource Manager Consumer groups to limit the amount of resources that services can use in an instance. You must create the consumer group and then map the service to the consumer group.
- **Enable or disable a service for a database or an instance**. By default, Oracle Clusterware attempts to restart a service automatically after failures. You can prevent this behavior by disabling a service. Disabling a service is useful when you must perform database or instance maintenance, such as when you are performing an upgrade and you want to prevent connection requests from succeeding.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- **Relocate a service to a different instance**. You can move a service from one instance to another instance to re-balance workloads, for example, after adding or deleting cluster nodes.

> **Note**
>
> - When you use services, do not set a value for the `SERVICE_NAMES` parameter; Oracle AI Database controls the setting for this parameter for the services that you create and for the default database service. The service features that this chapter describes are not directly related to the features that Oracle AI Database provides when you set `SERVICE_NAMES`. In addition, setting a value for this parameter may override some benefits of using services.
> - Service status information must be obtained from SRVCTL or from the service-related database views, such as `dba_services`.
> - If you specify a service using the `DISPATCHERS` initialization parameter, it overrides any service in the `SERVICE_NAMES` parameter, and cannot be managed. (For example, stopping the service with a SRVCTL command does not stop users connecting with the service.)

Related Topics

- Enabling Clients for Oracle RAC
  Learn how FAN is integrated with Oracle Clients, and how to enable FAN events for the several specific client development environments.
- *Oracle AI Database PL/SQL Packages and Types Reference*

Administering Services with Oracle Enterprise Manager

The Cluster Managed Database Services page is the main page for beginning all tasks related to services.

Access this page, as follows:

1. In Oracle Enterprise Manager, go to the Cluster Database Home page.
2. From the **Availability** menu, select **Cluster Managed Database Services** to display the Cluster Managed Database Services page.
3. Enter or confirm the credentials for the Oracle RAC database and host operating system and click **Continue** to display the Cluster Managed Database Services page.

From the Cluster Managed Database Services page you can drill down to perform the following tasks:

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Delete a service

> **Note**
>
> You must have `SYSDBA` credentials to access a cluster database. Cluster Managed Database Services does not permit you to connect as anything other than `SYSDBA`.

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

Administering Services with SRVCTL

Learn how to use SRVCTL to perform service administration on an Oracle Real Application Clusters (Oracle RAC) database.

> **Note**
>
> When you create a service using SRVCTL, you must start the service with a separate SRVCTL command. However, you may later have to manually stop or restart the service. You may also have to disable the service to prevent automatic restarts, to manually relocate the service, or to obtain status information about the service.

- Creating Services with SRVCTL
  To create a service with SRVCTL, use the `srvctl add service` command on the command line.
- Creating Services for Application Continuity and Transaction Guard
  To configure services for Application Continuity, when you create a service using SRVCTL, set the `-failovertype` parameter to `TRANSACTION` and `-commit_outcome` to `TRUE`.
- Starting and Stopping Services with SRVCTL
  To start or stop a service on Oracle Real Application Clusters (Oracle RAC), use the SRVCTL syntax described here.
- Enabling and Disabling Services with SRVCTL
  To enable or disable a service on Oracle Real Application Clusters (Oracle RAC), use the SRVCTL syntax described here.
- Relocating Services with SRVCTL
  To relocate a service on Oracle Real Application Clusters (Oracle RAC), use the SRVCTL syntax described here.
- Obtaining the Status of Services with SRVCTL
  to obtain the status of a service on Oracle Real Application Clusters (Oracle RAC), run the `srvctl status service` command from the command line.
- Obtaining the Configuration of Services with SRVCTL
  To obtain the high availability configuration of a service on Oracle Real Application Clusters (Oracle RAC), run the `srvctl config service` command from the command line.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Creating Services with SRVCTL

To create a service with SRVCTL, use the `srvctl add service` command on the command line.

Related Topics

- srvctl add service
  Adds services to a database and assigns them to instances.

Creating Services for Application Continuity and Transaction Guard

To configure services for Application Continuity, when you create a service using SRVCTL, set the `-failovertype` parameter to `TRANSACTION` and `-commit_outcome` to `TRUE`.

When using Application Continuity and Transaction Guard with your applications, you must configure a service. This section describes how to configure these application services depending on the functionality you plan to implement.

Creating Services for Application Continuity

Additionally, you can set values for these other service parameters for Application Continuity and load balancing:

- `-replay_init_time`: Specifies how long, in seconds, you allow replay to start. Oracle recommends that you choose a value based on how long you will allow replay to be initiated. The default value is 300 seconds.
- `-retention`: Specifies the time (in seconds) that the commit outcome information is stored in the database. The default value is 86400 (1 day).
- `-session_state`: After a COMMIT has run, if the state was changed in that transaction, then it is not possible to replay the transaction to reestablish that state if the session is lost. When configuring Application Continuity, the applications are categorized depending on whether it is correct to continue past a COMMIT operation within a request.
  - **Dynamic**: (default) A session has a dynamic state if the session state changes are not fully encapsulated by the initialization, and cannot be fully captured in a callback at failover. Once the first transaction in a request commits, failover is internally disabled until the next request begins. This is the default mode that almost *all* applications should use for requests.
  - **AUTO**: Use session state `AUTO` so that the session state is managed as the application evolves. Session state `AUTO` is available with TAC. When using TAC, both the client visible and server visible session states must match for a successful failover or session migration. When there is unrestorable session state at the beginning of a request, TAC does not enable as the state cannot be proven at the beginning of the request. Use ACCHK and the protection statistics to detect unrestorable session state at the beginning of request. Use `RESET_STATE` to clean session state automatically by Oracle AI Database between requests.
  –
- `-failoverretry`: Number of connection retries for each connection attempt; recommended value is 30.
- `-failoverdelay`: Delay in seconds between each connection attempt; recommended value is 10.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- `-notification`: FAN is highly recommended—set this value to `TRUE` to enable FAN for OCI and ODP.Net clients.
- `-clbgoal`: For connection load balancing, use `LONG` when using run-time load balancing.
- `-rlbgoal`: For run-time load balancing, set to `SERVICE_TIME`.

You can use SRVCTL to modify an existing service for Application Continuity, similar to the following command, where `racdb` is the name of your Oracle RAC database, and `app1` is the name of the service you are modifying:

```
$ srvctl modify service -db racdb -service app1 -clbgoal LONG
  -rlbgoal SERVICE_TIME -failoverretry 30 -failoverdelay 10
  -failovertype TRANSACTION -commit_outcome TRUE -replay_init_time 1800
  -retention 86400 -notification TRUE
```

Creating Services for Transaction Guard

To enable Transaction Guard, but not Application Continuity, create the service using SRVCTL and set only `-commit_outcome TRUE`.

You can use SRVCTL to modify an existing service to enable Transaction Guard, similar to the following command, where `racdb` is the name of your Oracle RAC database, and `app2` is the name of the service you are modifying:

```
$ srvctl modify service -db racdb -service app2 -commit_outcome TRUE
  -retention 86400 -notification TRUE
```

In the preceding example, the `-retention` parameter specifies how long, in seconds, to maintain the history. Additionally the `–notification` parameter is set to `TRUE`, enabling FAN events.

To use Transaction Guard, a DBA must grant permission, as follows:

```
GRANT EXECUTE ON DBMS_APP_CONT;
```

Related Topics

- *Oracle AI Database Development Guide*

Starting and Stopping Services with SRVCTL

To start or stop a service on Oracle Real Application Clusters (Oracle RAC), use the SRVCTL syntax described here.

For applications to connect using a server, the service must be started. If you stop a service, then it is temporarily unavailable, but is still subject to automatic restart and failover.

To start a service, enter the following SRVCTL syntax at the command line:

```
$ srvctl start service -db db_unique_name [-service service_name_list]
    [-instance inst_name] [-startoption start_options]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

To stop a service, enter the following SRVCTL syntax at the command line:

```
$ srvctl stop service -db db_unique_name -service service_name_list
    [-instance inst_name] [-stopoption stop_options]
```

Enabling and Disabling Services with SRVCTL

To enable or disable a service on Oracle Real Application Clusters (Oracle RAC), use the SRVCTL syntax described here.

If you disable a service, then Oracle Clusterware does not consider the service for automatic startup, failover, or restart. You might disable a service when performing application maintenance, to ensure the service is not accidentally restarted by Oracle Clusterware until your maintenance operations are complete. To make the service available for normal operation again, you enable the service.

To enable services, use the following SRVCTL syntax from the command line:

```
$ srvctl enable service -db db_unique_name -service service_name_list
    [-instance inst_name]
```

To disable services, use the following SRVCTL syntax from the command line:

```
$ srvctl disable service -db db_unique_name -service service_name_list
    [-instance inst_name]
```

Relocating Services with SRVCTL

To relocate a service on Oracle Real Application Clusters (Oracle RAC), use the SRVCTL syntax described here.

to relocate a service, you run the `srvctl relocate service` command from the command line. For example, you can use this command when a service has failed over to an available instance, but you want to move it back to the preferred instance after that instance is restarted.

In the following example, the `srvctl relocate service` command relocates the `crm` service from instance `apps1` to instance `apps3`:

```
$ srvctl relocate service -db apps -service crm -oldinst apps1 -newinst apps3
```

In the following example, the `srvctl relocate service` command relocates the `crm` service from `node1` to `node3` using node syntax:

```
$ srvctl relocate service -db apps -service crm -currentnode node1
    -targetnode node3
```

Obtaining the Status of Services with SRVCTL

to obtain the status of a service on Oracle Real Application Clusters (Oracle RAC), run the `srvctl status service` command from the command line.

Copyright © 1999, 2025, Oracle and/or its affiliates.

In the following example, the `srvctl status service` command returns the status of the services that are running on the `apps` database:

```
$ srvctl status service -db apps

Service erp is running on nodes: apps02,apps03
Service hr is running on nodes: apps02,apps03
Service sales is running on nodes: apps01,apps04
```

Obtaining the Configuration of Services with SRVCTL

To obtain the high availability configuration of a service on Oracle Real Application Clusters (Oracle RAC), run the `srvctl config service` command from the command line.

In the following example, the `srvctl config service` command returns the configuration of the `erp` service that is running on the `apps` database:

```
$ srvctl config service -db apps -service erp

Service name: erp
Service is enabled
Cardinality: 1
Disconnect: false
Service role: PRIMARY
Management policy: AUTOMATIC
DTP transaction: false
AQ HA notifications: true
Global: false
Commit Outcome: true
Failover type: TRANSACTION
Failover method: NONE
TAF failover retries: 30
TAF failover delay: 10
Connection Load Balancing Goal: LONG
Runtime Load Balancing Goal: SERVICE_TIME
TAF policy specification: NONE
Edition:
Pluggable database name:
Maximum lag time: ANY
SQL Translation Profile:
Retention: 86400 seconds
Replay Initiation Time: 1800 seconds
Session State Consistency: AUTO
Preferred instances: apps
Available instances:
```

Global Services

Oracle RAC supports database services and enables service-level workload management across instances in a single cluster.

Global services provide dynamic load balancing, failover, and centralized service management for a set of replicated databases that offer common services. The set of databases may include

Copyright © 1999, 2025, Oracle and/or its affiliates.

Oracle RAC and non-clustered Oracle AI Databases interconnected by Oracle Data Guard, Oracle GoldenGate, or any other replication technology.

When you create and use global services, the following workload management features are available:

- Ability to specify preferred and available databases for a global service
- Handling of replication lag
- Geographical affinity between clients and servers
- Connection load balancing
- Run-time load balancing
- Inter-database service failover
- Fast connection failover
- Connect-time failover
- Application Continuity
- Transaction Guard
- Backward compatibility with existing clients

> **Note**
>
> You can manage instance placement of a global service within an Oracle RAC database with SRVCTL but you can only manage other global service attributes with GDSCTL.

Related Topics

- *Oracle AI Database Global Data Services Concepts and Administration Guide*

Service-Oriented Buffer Cache Access

Service-oriented buffer cache access improves performance by managing data with the service to which the data belongs.

Access of an object, over time, through a service is mapped and persisted to the database, and this information can be used to improve performance. Blocks that are accessed through the service are cached in the instances where the services are running and, more importantly, the information is not cached where the services are not running.

This information can also be used to pre-warm the cache prior to a service starting. The service start-up can be triggered either by instance start-up or by service relocation. Service-oriented buffer cache access provides consistent performance to any user of that service because the blocks that the service user accesses are cached in the new relocated instance.

Connecting to a Service: An Example

You can use this example to see how to create a service, and see several examples of connecting to that service using different client methods.

The service is enabled for run-time load balancing using the following scenario:

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Service Name: `HR.example.com`
  - Running on database named CRM
  - The system consists of 4 nodes
- `SERVICE_TIME` is specified as the value for the `-rlbgoal` parameter
- The SCAN address of the listener is `rws3010104-scan.example.com`
- The Listener port is 1585

The service has a cardinality of two, but if needed, can be offered by any of the CRM database instances. The service configuration is as follows:

- Preferred Instances: CRM1, CRM2
- Available Instances: CRM3, CRM4
- `LONG` is specified as the value for the `-clbgoal` parameter

The application using this service takes advantage of Application Continuity, so you must set `-failovertype` and `-commit_outcome`. Use the default retention parameters, but set a 10 second delay between connection attempts, and set a limit of up to 40 retries before failing to get a connection.

Example 5-4 Creating the HR Service Using SRVCTL

In this example, you create the HR service using SRVCTL:

```
$ srvctl add service –db CRM –service HR.example.com –preferred CRM1,CRM2
  –available CRM3,CRM4 –clbgoal LONG –failovertype TRANSACTION
  –commit_outcome TRUE –failoverdelay 10 –failoverretry 40
```

Next, start the `HR.example.com` service:

```
$ srvctl start service –db CRM –service HR.example.com
```

The service is now available on up to two instances, and CRM1 and CRM2 are the preferred instances.

Example 5-5 Connecting to the HR Service from a JDBC Application

In this example, the application that connects to the HR service is a Java Database Connectivity (JDBC) application that is using the JDBC Universal Connection Pool with the JDBC thin driver.

A URL is constructed specifying the thin-style service name format for the database specifier. Fast Connection Failover is enabled, and remote Oracle Notification Service is configured, where the Oracle Notification Service daemon on the cluster listens on port 6200.

```
//import packages and register the driver
import java.sql.Connection;
import java.sql.SQLException;
import java.sql.Statement;
import oracle.ucp.jdbc.PoolDataSourceFactory;
import oracle.ucp.jdbc.PoolDataSource;

PoolDataSource  pds = PoolDataSourceFactory.getPoolDataSource();

//set the connection properties on the data source.
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

```
pds.setConnectionPoolName("FCFPool");
pds.setFastConnectionFailoverEnabled(true);
pds.setONSConfiguration("nodes=rws3010104-scan.example.com:6200");
pds.setConnectionFactoryClassName("oracle.jdbc.pool.OracleDataSource");
pds.setURL("jdbc:oracle:thin:@//rws3010104-scan.example.com:1585/
HR.example.com");
pds.setUser("HR");
pds.setPassword("hr");


//Override any pool properties.
pds.setInitialPoolSize(5);

//Get a database connection from the datasource.

Connection conn = pds.getConnection();

// do some work

//return connection to pool
conn.close();
conn=null
```

Related Topics

- *Oracle Universal Connection Pool Developer's Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

6

Ensuring Application Continuity

Application Continuity is a feature that enables the replay, in a non-disruptive and rapid manner, of a request against the database after a recoverable error that makes the database session unavailable so an outage appears to the user as no more than a delayed processing of the request.

AC needs request boundaries, which means AC depends on Oracle or third-party pools that embed request boundaries when connections are borrowed and returned to the pool. Whereas, TAC uses provided request boundaries and creates its own. With TAC request boundaries advance automatically when there is no transaction, the session state is restorable, and cursors are closed.

To use this chapter, you should be familiar with the major relevant concepts and techniques of the technology or product environment in which you are using Application Continuity, such as Oracle RAC or Oracle Active Data Guard (Oracle ADG).

- Understanding Application Continuity
  Application Continuity can be used to mask outages from clients and recover in-flight transactions that would otherwise be lost.
- Transparent Application Continuity
  A version of Application Continuity for simple applications that uses discovery to detect request boundaries.
- Configuring Application Continuity
  Application Continuity (AC) requires using an Oracle Pool or an application that allows Transparent Application Continuity (TAC) to discover request boundaries.
- Administering Application Continuity Operation and Usage
  Learn how to manage the use of Application Continuity, and how you can use it in applications.
- Fast Application Notification (FAN)
  The Oracle RAC high availability framework monitors a database and its services, and sends Fast Application Notification (FAN) event notifications.
- Configure for Unplanned Outages
  You can assign services to one or more instances in an administrator-managed Oracle RAC database to hide outages.
- Managing Planned Maintenance
  To minimize service disruption to application users, Oracle Real Application Clusters (Oracle RAC) provides interfaces that relocate, disable, and enable services.
- Transaction Guard for Improving Client Failover
  Transaction Guard is a developer feature that your applications can use to determine `COMMIT_OUTCOME` following recoverable errors.
- Application Continuity During Major Database Version Upgrades with DBMS_ROLLING
  Starting with Oracle AI Database 26ai, Application Continuity and draining of database sessions are supported during major database version upgrades and database restructuring when using Application Continuity with `DBMS_ROLLING`.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Reset Database Session State
  RESET_STATE is an Oracle-exclusive feature that automatically prevents session states from leaking between application usage.

> **See Also**
>
> *Oracle AI Database High Availability Overview and Best Practices* describes how you can choose and implement the level of Application Continuity protection that is the best for your applications.

Understanding Application Continuity

Application Continuity can be used to mask outages from clients and recover in-flight transactions that would otherwise be lost.

- About Application Continuity
  Application Continuity provides high availability for your application workloads.
- Key Concepts for Application Continuity
  This section describes several terms and concepts that you must understand to use Application Continuity.
- How Application Continuity Works for Applications
  If a recoverable error occurs and if you enabled replay, then Application Continuity attempts recovery of the database session.
- Support for Oracle Application Continuity and Transparent Application Continuity
  Support for Application Continuity (AC) is integrated into many Oracle applications.
- Restrictions and Other Considerations for Application Continuity
  Be aware of these restrictions and considerations when using Application Continuity.
- Application Continuity for Various Applications
  Transparent Application Continuity covers applications that belong to three different groups, automatically tracked by the state tracking system.

About Application Continuity

Application Continuity provides high availability for your application workloads.

Application Continuity is a high availability feature that masks many recoverable Oracle AI Database outages (when replay is successful) from applications and users by restoring the database session: the full session, including all states, cursors, variables, and the last transaction if there is one. Application Continuity addresses the problem that arises when an application is trying to access the database and the database instance becomes unavailable due to an unplanned outage or planned maintenance (such as timeout, network outage, instance failure, repair, configuration change, and patch apply). Without Application Continuity in place, database recovery does not mask outages to applications and end users. In such scenarios, developers and users must handle the exception conditions, and users can be left not knowing what happened to their funds transfers, time sheets, orders, and bill payments. Users might lose screens of uncommitted data, and must log in again and reenter that data. In the worst cases, the administrator might be forced to restart the middle tier to recover from an overwhelming number of logins.

With Application Continuity, if the database instance becomes unavailable, then Application Continuity attempts to rebuild the session and any open transactions using the correct states; and if the transaction committed and need not be resubmitted, then the successful return

Copyright © 1999, 2025, Oracle and/or its affiliates.

status is returned to the application. If replay is successful, then the request can continue safely without risk of duplication. If replay cannot restore data that the application has already processed and potentially made decisions on, then the database rejects the replay and the application receives the original error.

Client requests can contain transactional and non-transactional work. After a successful replay on Oracle AI Database, the application can continue where that database session left off, instead of having users left in doubt, not knowing what happened to their transactions. Recovering these client requests also helps to avoid the need to reboot mid-tier servers to recover from an overload of logins when the application comes back online. With Application Continuity, the end-user experience is improved by masking many outages, planned and unplanned, without the application developer needing to attempt to recover the request.

Application Continuity performs the recovery of in-flight transactions and database session state, while ensuring the transaction idempotence provided by Transaction Guard. Each database session is tagged with a logical transaction ID (LTXID), so the database recognizes whether each replay committed any transactions, and if it did commit any transactions, whether the work also ran to completion. While Application Continuity attempts to replay, the replay appears to the application as a delayed processing, or the application receives the commit response for the original transaction (if the last transaction had completed before the outage).

Application Continuity is supported for Oracle RAC, Oracle Active Data Guard, and `DBMS_ROLLING`. It is supported for Oracle AI Database using the multitenant architecture (with failover at the Pluggable Database level). It is not currently supported for Oracle GoldenGate, Logical Standby, third-party replication solutions, or DML redirection if using Oracle Active Data Guard.

Related Topics

- Administering Application Continuity Operation and Usage
  Learn how to manage the use of Application Continuity, and how you can use it in applications.
- Restrictions and Other Considerations for Application Continuity
  Be aware of these restrictions and considerations when using Application Continuity.

Key Concepts for Application Continuity

This section describes several terms and concepts that you must understand to use Application Continuity.

The following terms are used throughout this chapter:

Database request

A database request is a unit of work submitted to the database from the application, such as a transaction. A request typically corresponds to the SQL and PL/SQL, and other database calls, of a single web request on a single database connection. A request is generally demarcated by the calls made to check-out and check-in the database connection from a connection pool.

Request Boundaries

Request Boundaries demarcate where applications and application servers borrow and return connections from their connection pools. Request Boundaries indicate when a session is not in use. When request boundaries are visible to the database, it enables functionality such as draining for planned maintenance, load balancing, and multiplexing to be isolated at the database layer. Sessions can be re-established with no visible disruption to the application layers above.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Recoverable error

A recoverable error is an error that arises due to an external system failure, independent of the application session logic that is running, such as a lost or invalid connection. Recoverable errors occur following planned and unplanned outages of foregrounds, networks, nodes, storage, and databases. The application receives an error code that can leave the application not knowing the status of the last operation submitted. Application Continuity reestablishes database sessions and resubmits the pending work for the class of recoverable errors.

Application Continuity does not resubmit work following call failures due to nonrecoverable errors. An example of a nonrecoverable error that would *not* be replayed is submission of invalid data values.

Commit outcome

A transaction is committed by updating its entry in the transaction table. Oracle AI Database generates a redo-log record corresponding to this update and writes out this redo-log record. Once this redo-log record is written out to the redo log on disk, the transaction is considered committed at the database. From the client perspective, the transaction is considered committed when an Oracle message (called the *commit outcome*), generated after that redo is written, is received by the client. However, if a `COMMIT` has been issued, then the `COMMIT` failure message cannot be retrieved if it is not received by the client or the application.

Restoring original function results

Restoring original function results are non-deterministic that can obtain a new value every time they are called, and thus their results can change frequently. Restoring original function results cause a problem for replay because the results can change at replay. Consider *sequence*`.NEXTVAL` and `SYSDATE`, often used in key values. If a primary key is built with values from these function calls, and is used in later foreign keys or other binds, at replay the same function result must be returned.

Support for keeping the original results of Oracle functions is provided for `SYSDATE`, `SYSTIMESTAMP`, `SYS_GUID`, *sequence*`.NEXTVAL`, `CURRENT_TIMESTAMP`, and `LOCALTIMESTAMP`. Identity sequences are supported for owned sequences in SQL.

Application Continuity keeps the original values for SQL automatically. If you are using PL/SQL, then grant `KEEP` for application users, and the `KEEP` clause for a sequence owner.

Session state consistency

After a `COMMIT` statement has processed, if state was changed in that transaction, it is not possible to replay the transaction to reestablish that state if the session is lost. Transparent Application Continuity creates a new checkpoint of the session state to reestablish a new starting point for replay. When configuring Transparent Application Continuity, use session state consistency set to `AUTO`.

- A session has **dynamic** state if the session state changes during the request. When using Application Continuity, failover is internally disabled until the next request begins.
- With Transparent Application Continuity, the state is managed for you when you set the session state consistency service attribute to `AUTO`. The session states are tracked and verified at failover. After a disable, when session state consistency service attribute is set to `AUTO`, failover is re-enabled automatically when possible.
- Use session state `AUTO` so that the session state is managed as the application evolves. Session state `AUTO` is available with TAC. When using TAC, both the client visible and server visible session states must match for a successful failover or session migration.

Copyright © 1999, 2025, Oracle and/or its affiliates.

When there is an unrestorable session state at the beginning of a request, TAC does not enable because the state cannot be proven at the beginning of the request. Use ACCHK and the protection statistics to report unrestorable session state at the beginning of the request. Use `RESET_STATE` to clean session state automatically by Oracle AI Database between requests.

-

> **Note**
>
> The service attribute values `FAILOVER_TYPE = TRANSACTION` with `SESSION_STATE_CONSISTENCY = STATIC` are no longer a supported service attribute combination.

Stateless applications

A stateless application is an application program that does not use session state in one request – such as context and PL/SQL states that were set by a prior usage of that session by another web request or similar usage. The necessary state to handle the request is contained within the request itself, whether as part of the URL, query-string parameters, body, or headers. In a cloud environment, it is preferable that applications be stateless for the sake of scalability and portability. Statelessness enables greater scalability because the server does not have to maintain, update, or communicate that session state. Additionally, load balancers do not have to consider session affinity for stateless systems. Most modern Java Web applications are stateless. The service attribute `RESET_STATE` is recommended for all stateless applications to prevent leakage of session state to later reuses.

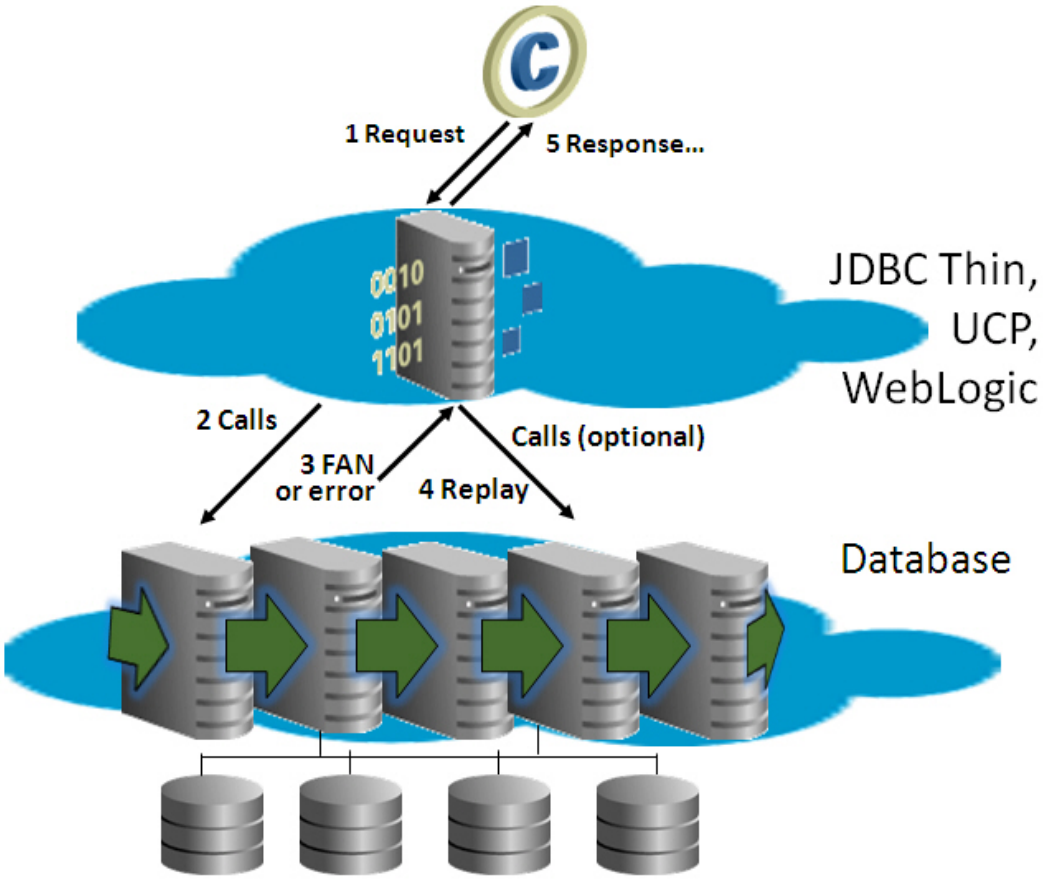
How Application Continuity Works for Applications

If a recoverable error occurs and if you enabled replay, then Application Continuity attempts recovery of the database session.

The following figure is a graphical representation of how Application Continuity works.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Figure 6-1 Application Continuity



To attempt to recover a database session following a recoverable error, Application Continuity performs the following steps:

> **Note**
>
> The steps to recover a database session apply for both unplanned and planned outages, although specific steps vary depending on the type of outage.

1. The client application makes a request, which is passed to a middle tier (such as the Universal Connection Pool (UCP), ODP.NET, WebLogic Server, OCI session pool, Tuxedo, or third-party pool using UCP) and forwarded to the database. The application could also make a request directly to the database using the JDBC replay driver or ODP.NET managed provider 23ai.
2. The middle tier, or Oracle drivers (JDBC replay driver, OSI, or ODPM 23ai).
3. A planned or unplanned `DOWN` Fast Application Notification (FAN) event or recoverable error is received. The driver stops the terminated session.
4. Application Continuity begins the replay and does the following:
   a. Replaces the terminated physical session with a new clean session.
   b. Prepares for replay by using Transaction Guard to determine the outcome of the in-flight transaction, if one was open.

Copyright © 1999, 2025, Oracle and/or its affiliates.

c. If `FAILOVER_RESTORE=LEVEL2`, `FAILOVER_RESTORE=LEVEL1`, or `FAILOVER_TYPE=AUTO`, then Application Continuity restores the initial session state. Use Wallets with `FAILOVER_RESTORE` to restore all modifiable parameters. Application Continuity uses a label callback or initial callback if an application also sets session states that are not provided by `FAILOVER_RESTORE` in the callback

d. Rebuilds the database session, recovering the transactional and non-transactional states, and validating at each step that the data and messages seen by the client driver are the same as those that the client may have seen and used to make a decision.

e. Ends the replay and returns to run-time mode.

f. Submits the last queued call.

   This is the last call made when the outage was discovered. During replay, only this call can run a `COMMIT`. A `COMMIT` midway through rebuilding the session terminates replay (excluding autonomous transactions).

5. The response is returned to the application.

   If replay succeeded, then the application can continue with the problem masked. If not, then the application must handle the original error.

The behavior of Application Continuity after a communication failure depends on the Oracle products and technologies involved. For example:

- If you use Oracle RAC or an Oracle Active Data Guard farm, then, after the connection is reestablished on another running instance, Application Continuity attempts to rebuild the session and replay the last transaction if there is one in flight.
- If you use Oracle Active Data Guard and fail over to a standby site, then Application Continuity connects to the failover instance and attempts to rebuild the session and replay the last transaction there, if a transaction was in-flight. Application Continuity does not replay if the Oracle Active Data Guard switchover or failover has lost data.
- If you are using Oracle RAC or Oracle RAC One Node and not using Oracle Active Data Guard, and if an outage causes a break in all public networks or causes the database or database session to shut down briefly, then Application Continuity attempts to rebuild the session and replay the last transaction (if a transaction was in flight) against the database after connectivity is restored.

Related Topics

- FAILOVER_RESTORE
  Setting `FAILOVER_RESTORE` to `LEVEL1` (for AC with older 19c versions), `LEVEL2` (for AC with database templates), or `LEVEL1` or `AUTO` (for TAC) automatically restores all modifiable parameters before replaying the request.
- Using Application Continuity for Planned Maintenance
  For planned maintenance, Oracle recommends that you drain requests from Oracle connection pools in combination with Application Continuity for those requests that do not complete.

Support for Oracle Application Continuity and Transparent Application Continuity

Support for Application Continuity (AC) is integrated into many Oracle applications.

Application Continuity is available for general use with the following Oracle technologies:

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Pro*C/C++ and Pro*COBOL 23.9 or later, when used with embedded SQL request boundaries.
- Oracle JDBC Replay Driver 12c or later. This is a JDBC driver feature provided with Oracle Database 12c for Application Continuity.
- The Oracle Database 21c or later JDBC Consolidated Data Source automatically enables replay when Application Continuity is enabled. There is no longer a need to choose the replay driver.
- Oracle Universal Connection Pool (UCP) 12c or later
- Oracle WebLogic Server Active GridLink or third-party JDBC application servers using UCP with Oracle JDBC Replay Driver 19c or later, or JDBC Consolidated Data Source.
- Java connection pools or standalone Java applications using Oracle JDBC Replay Driver 12c or later with Request Boundaries.
- Applications and language drivers using the session pool with Oracle Call Interface (OCI) 12c Release 2 (12.2) or later, including Python with thick mode.
- SQL*Plus 19c or later.
- ODP.NET Unmanaged Provider 12c Release 2 (12.2) or later (Set "`pooling=true`" and "`Application Continuity=true`" as default in 12.2 and later).
- Oracle Data Provider for .NET (ODP.NET), Managed Driver 23ai and later.

Transparent Application Continuity (TAC) is available for general use with the following Oracle technologies:

- Pro*C/C++ and Pro*COBOL 23.9 or later.
- Oracle JDBC Replay Driver 19c or later. This is a JDBC driver feature provided with Oracle Database 19c for Application Continuity
- The Oracle Database 21c or later JDBC Consolidated Data Source automatically enables replay when Application Continuity is enabled. There is no longer a need to choose the replay driver.
- Oracle Universal Connection Pool (UCP) 19c or later with Oracle JDBC Replay Driver 19c or later
- Oracle WebLogic Server Active GridLink or third-party JDBC application servers using UCP with Oracle JDBC Replay Driver 19c or later, or JDBC Consolidated Data Source.
- Java connection pools or standalone Java applications using Oracle JDBC Replay Driver 19c or later
- Oracle Call Interface (OCI) 19c or later
- Python (`python-oracledb`) and Node.js (`node-oracledb`) drivers in thick mode
- SQL*Plus 19.3 or later
- ODP.NET pooled, Unmanaged Driver 18c or later (Set "`pooling=true`" as default in 12.2 and later)
- OCI-based applications using OCI 19c or later
- Oracle Data Provider for .NET (ODP.NET), Managed Driver 23ai and later

Application Continuity for Java is embedded in the Universal Connection Pool, WebLogic data sources, including non-XA and XA data sources, and is available with the thin JDBC replay driver, standalone (which is a JDBC replay driver without Oracle connection pools, such as Apache Tomcat or a custom Java connection pool). Application Continuity for OCI is embedded in SQL*Plus, OCI 12.2 or later, ODP.NET, Unmanaged Provider including Python

Copyright © 1999, 2025, Oracle and/or its affiliates.

with thick mode, (ODP.NET), Managed Provider 23ai and later, and Python when using thick mode. With AC and TAC, JDBC applications auto enable starting with Oracle Database 21c, and OCI applications starting with Oracle Database 19c (19.3).

If a connection pool or container does not use an Oracle connection pool, then many third-party applications fully support replacing the connection pool with the Universal Connection Pool. This includes IBM WebSphere and Apache Tomcat. C applications that implement their own pooling can use `OCIRequestBegin` and `OCIRequestEnd` calls. Alternatively—for Java applications, only—an application can add its own request boundaries.

Request Boundaries

Request boundaries are embedded in Oracle connection pools. Request boundaries are also embedded for third party Java Application Servers that are standard with JDK9 or later. When you use the Oracle connection pools, request boundaries are marked explicitly at check-out and check-in, delimiting the size of each replay. When using third-party connection pools, use UCP if Java, or use Transparent Application Continuity, or add request boundaries, or use third party Java Application Servers that are standard with JDK9 or later. Request boundaries are discovered using state tracking when using Transparent Application Continuity. This type of request boundary is called an implicit request boundary. This functionality is available starting with the Oracle Database 19c Java replay driver, and the Oracle Database 19c OCI driver, which includes open source and ODP.NET Unmanaged Provider.

Related Topics

- Introducing Oracle Data Provider for .NET
- Introducing JDBC

Restrictions and Other Considerations for Application Continuity

Be aware of these restrictions and considerations when using Application Continuity.

Application Continuity is not supported for:

- JDBC OCI driver (type 2)
- OLE DB
- ODBC
- OCCI
- Pro*FORTRAN
- XA
- python-oracledb in thin mode
- node-oracledb in thin mode

> **Note**
>
> If applications in these environments require support for planned maintenance, then consider draining using connection tests.

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> Oracle Data Provider for .NET (ODP.NET), Unmanaged Driver is deprecated in Oracle AI Database.
> ODP.NET provides ADO.NET-based data access to Oracle Database. There are two primary Oracle data access drivers for Microsoft .NET Framework: ODP.NET, Managed Driver and ODP.NET, Unmanaged Driver. In Oracle AI Database, ODP.NET, Managed Driver supports all major features available in ODP.NET, Unmanaged Driver with the same application programming interfaces and configuration settings. Code migration from unmanaged ODP.NET to managed ODP.NET is straightforward for the vast majority of existing .NET applications. Oracle recommends that you migrate existing unmanaged ODP.NET applications to ODP.NET, Managed Driver. The ODP.NET, Unmanaged Driver can be desupported in a future release.

For OCI and ODP.NET Unmanaged drivers, Application Continuity does not work for ADTs, Advanced Queues (AQ), and some LOB APIs. These exclusions do not apply to Java.

For applications using JDBC, there is no support for `oracle.sql` deprecated concrete classes: `OPAQUE`, `ANYDATA`, or `STRUCT`.

If a statement cache at the application server level is enabled (for example, the WebLogic or third-party application server statement cache), this cache must be disabled when replay is used. Instead, configure the JDBC statement cache, which supports Application Continuity and is optimized for JDBC and Oracle AI Database (`oracle.jdbc.implicitstatementcachesize=`*`nnn`*).

Note the following restrictions when replay of transactions can occur:

- Replay is disabled intentionally if a request issues an `ALTER SYSTEM` or `ALTER DATABASE` statement.
- Replay is disabled at a request level for `ALTER SESSION` statements that are deemed unsafe to rebuild the session. This includes `SERIALIZABLE` mode. Starting with Oracle AI Database 26ai, statements for events and disabling and enabling `COMMIT IN PROCEDURE` and `GUARD` are supported.

  Note that other `ALTER SESSION` statements at an application level are supported for replay. These include statements for globalization support (NLS) settings, stored and private outlines, setting the container (CDB/PDB), SQL trace, and PL/SQL warnings.
- The replay target database must be in the same database cluster (Oracle RAC, Oracle Data Guard, Oracle Active Data Guard, or Oracle Multitenant) as the source database. To protect the integrity of business transactions, Application Continuity does not replay if the target is a different database. Application Continuity also does not replay if the target database is the same as the source database (or Pluggable Database) but with data loss, such as one flashed back, recovered incompletely by media recovery, or opened by Oracle Data Guard to an earlier point in time.
- For streams arguments, replay is on a "best effort" basis. For example, if the application is using physical addresses, the address has gone with the outage and cannot be repositioned. JDBC stream setters (such as `setBinaryStream`), for example, cause replay to be disabled.
- Application Continuity is not currently supported for Oracle GoldenGate, Logical Standby, third-party replication solutions, or DML redirection if using Oracle Active Data Guard.
- Replay does not support `ISOLATION_LEVEL=SERIALIZABLE`.

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> If you are creating clones of databases by splitting disk images (for example, BCVs) or by cloning so it is a "different" database for the purpose of making a logical standby or logical copy that is not a physical or Oracle Active Data Guard database, then the `nid` utility ***must*** be used to change the DBID to differentiate the databases.

Related Topics

- When Application Continuity in OCI Can Fail Over
- How to Change the DBID, DBNAME Using NID Utility (My Oracle Support Doc ID 863800.1)

Application Continuity for Various Applications

Transparent Application Continuity covers applications that belong to three different groups, automatically tracked by the state tracking system.

- Applications That Use Containers with Request Boundaries
  Applications that use containers with request boundaries enable Application Continuity to manage replay between explicit boundaries.
- Applications that are Database Agnostic
  Database-agnostic applications set a state when the connection is established, and do not change non-transactional session states again, or change it rarely.

Applications That Use Containers with Request Boundaries

Applications that use containers with request boundaries enable Application Continuity to manage replay between explicit boundaries.

A **request boundary** is a tag that marks the beginning and end of a database request. Connection pools that embed request boundaries include Oracle Universal Connection Pool, all WebLogic server data sources, Tuxedo, Oracle Call Interface, ODP.NET Unmanaged Provider, and standard third-party application servers and standalone Java pools that use the JDBC drivers `PooledConnection` interface, in addition to SQL*Plus.

When Oracle AI Database is aware of request boundaries:

- The database can process web requests effectively and with no performance overhead, including when to attach and release connections. It can multiplex, drain, rebalance, shed, and allow complex states inside requests. Without request boundaries, the lower layers of the database are not aware of web requests. Subsequently, the database relies on Oracle Client actions, advisory methods and heuristics, such as fast connection failover, connection validation, and state advice.
- The length of replay is limited to the initial state, followed by the user calls in that request less those that are purged by Application Continuity. Request boundaries enable you to control the length of replay. You can also determine where to drain for planned maintenance (at the end of the request), and where to fail over for planned maintenance (at the beginning of the request).
-
- When using Application Continuity, the replay driver detects safe places to move the request boundaries forward automatically. This feature is available when `FAILOVER_TYPE` is set to `AUTO`.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Applications deployed using middle-tier containers that set request boundaries have access to the full set of transparency features that the database server provides. The database detects when a client sets request boundaries and uses the boundaries to mark safe points for draining, failover, concentration, and throughput measures.

Request boundaries enable an application to use all complex, non-transactional session states within a request. The request boundary specification requires that these states are not dependent across boundaries.

Applications that are Database Agnostic

Database-agnostic applications set a state when the connection is established, and do not change non-transactional session states again, or change it rarely.

Database-agnostic applications (applications with no request boundaries) set basic, non-transactional states. These applications do not use features or sequences proprietary to Oracle AI Database. For these applications, Application Continuity identifies implicit boundaries. These applications often set state once when a connection is created, and then do not change state again, or change the state infrequently. This category of applications includes those applications that use anonymous PL/SQL that does not create server-side session states.

When using Transparent Application Continuity with Oracle AI Database, explicit request boundaries are not required, but they are recommended. (For Oracle Database 18c ONLY: Java requires an initial `beginRequest`.) This allows support for SQL*Plus and third-party connection pools. When explicit request boundaries are present, they are used. Explicit request boundaries continue to be needed for Application Continuity. Oracle recommends that you return your connections to the connection pools, when not in use.

Transparent Application Continuity

A version of Application Continuity for simple applications that uses discovery to detect request boundaries.

- About Transparent Application Continuity
  In Oracle AI Database both Application Continuity and Transparent Application Continuity transparently tracks and records session and transactional state so that a database session can be recovered following recoverable outages.
- Deciding on Transparent Application Continuity
  Transparent Application Continuity (TAC) is an **application failover solution** that is safe to enable by default.
- Using Transparent Application Continuity in Oracle Cloud Environments
  Transparent Application Continuity is enabled by default on TP and TPURGENT services in an Oracle Cloud environment for Oracle Autonomous AI Database–Dedicated, and is available for Oracle Autonomous AI Database–Serverless.

About Transparent Application Continuity

In Oracle AI Database both Application Continuity and Transparent Application Continuity transparently tracks and records session and transactional state so that a database session can be recovered following recoverable outages.

Transparent Application Continuity uses this information to discover boundaries for basic applications. Recovery of the user database session is done safely and with no need for a DBA or developer to have any knowledge of the application or make application code changes.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Transparency is achieved by using a state-tracking infrastructure that categorizes session state usage as an application issues user calls.

Transparent Application Continuity is enabled when `FAILOVER_TYPE=AUTO`.

You can enable Application Continuity or Transparent Application Continuity to protect applications during planned maintenance and when unplanned outages occur. For planned maintenance, database sessions that reach a safe place (such as a connection test or a known recoverable point) are drained at the database. For database sessions that do not drain, the database determines when to fail the database session over and triggers Transparent Application Continuity or Application Continuity to do so. Application Continuity hides unplanned outages for Java-based applications, OCI and ODP.NET applications including SQL*Plus, all Oracle connection pools, Tuxedo, WebLogic Server, and third-party application servers using Universal Connection Pool.

Starting with Oracle AI Database 26ai, Transparent Application Continuity (TAC) includes Transparent Application Failover (TAF) style cursors, known as Resumable Cursors. TAC supports basic `SELECT` cursors in `FETCH`. To support `SELECT` cursors in `FETCH`, not in transactions, set session state consistency to `AUTO`. In addition, the database repositions these cursors at failover for much faster failover. TAC provides higher protection because cursors can remain open across transaction boundaries. This provides better-unplanned outages and planned maintenance support for applications. It allows TAC to re-enable faster to provide higher protection and achieves speedier failover.

For unplanned outages, Transparent Application Continuity is invoked for outages that result in recoverable errors, typically related to underlying software, foreground, hardware, communications, network, or storage layers, hiding most failures from applications and users.

TAC supports basic applications using SQL and basic `ALTER SESSION` statements. With Transparent Application Continuity, DBAs no longer need to have knowledge of an application to:

- Restore preset states—At run time, Transparent Application Continuity records the initial preset session states, monitors further states, and records session signatures sufficient to detect deviation in the state of a session at failover for monitored states. At failover, Transparent Application Continuity restores the preset session states before replay starts, and verifies that these session states fully match the original before replay starts. This also allows for session state that has been restored using both Application Continuity and other mechanisms, such as log-on triggers, labels, and connection call backs. You will continue to add log-on triggers, call backs, or labels if the state is outside the preset states.
- Recognize and disable application-level side effects when recovering a session—Transparent Application Continuity records the transactions and session state from the client, which is referred to as *capture*, to enable replay. During normal run-time, Transparent Application Continuity detects *side effects*, or changes that occur as a result of the transaction, but are not part of the transaction itself. The type of side effect is distinguished between those that relate to an application's logic and those that are internal, relating to database housekeeping. For applications that use statements that have side effects, capture is disabled when the statement is running. Once a new request starts, capture is re-enabled automatically.
- Restoring original function values for owned functions—Such functions can return a new value each time they are run. Oracle provides support for keeping the original results of these functions `SYSDATE`, `SYSTIMESTAMP`, `LOCAL_TIMESTAMP`, `CURRENT_TIMESTAMP`, `SYS_GUID`, and `sequence.NEXTVAL`. If the original values are not kept and if different values are returned to the application at replay, then Transparent Application Continuity rejects replay. Use grants to keep your sequences, dates, and times. When an application is using its own schema, you can assign the grants for keeping to a role and then grant this role to users.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Know about request boundaries—Request Boundaries demarcate where applications and application servers borrow and return connections from their connection pools. For applications using Application Continuity with the JDBC thin driver, OCI, and ODP.NET Unmanaged Provider, DBAs do not need to know about request boundaries but when they are in use, Transparent Application Continuity takes advantage of them. As a best practice, use explicitly delineated request boundaries in your application because it is not always possible for the database to identify a checkpoint where a request boundary can be inserted.

  Using Transparent Application Continuity, the server and the drivers are tracking transaction and session state usage. This allows the driver to detect implicit request boundaries. For an *implicit boundary* with SSS cursors, no objects may be open, cursors must have been returned to the statement cache, no transaction may be active, and the session state must have been recognized as fully restorable. The driver either discards the currently tracked information, and starts tracking again from this point, or it re-enables tracking if there had been a disabling event. On the next call to the server, the server verifies and, if applicable, creates a request boundary where there was previously no explicit boundary.
- To use Transparent Application Continuity for more complex applications with unrestorable session states, use a pool like you use for Application Continuity and set `RESET_STATE` to clean the session state between usages. `RESET_SATE` is suitable for applications that must not leak session state from one pooled usage to a later usage.

Deciding on Transparent Application Continuity

Transparent Application Continuity (TAC) is an **application failover solution** that is safe to enable by default.

TAC provides failover for planned events, called planned failover, as well as for unplanned events. Refer to *Oracle AI Database High Availability Overview and Best Practices* to know more about planned and unplanned failover and Oracle Maximum Availability Architecture (MAA) levels.

A successful failover is a delayed execution, with no error returned to the application. If TAC cannot produce the same logical session, TAC returns an error to the application. The correctness guarantees for a successful failover includes the following, but not limited to:

- Within the SLA set for the service under `replay_initiation_timeout`- Failover starts if the timestamp for the failover is less than or equal to `replay_initiation_timeout`, where `replay_initiation_timeout` is set at the start of the request.
- Same database forward in time- The failover checks that the target database is at the correct database (`DBID` or `PDBID`) and is at the same time or is forward in time (called SCN ancestry) from the original.
- No duplication of transactions- The failover checks whether the last request committed and if the request committed returns `COMMITTED` to the client.
- Same session state at the start of replay as the original session- If replay is needed, after restoring the session state using `FAILOVER_RESTORE` and before replaying, the session states are checked to ensure that they match the original session at the start of the current request.
- Same results that the application has already processed- Cursors are repositioned and the results of that repositioning are checked to be the same as before. This is similar to TAF with the difference that the reposition in Oracle AI Database 26ai is at the server with the rows no longer returned to the client. To provide higher success of same results, TAC keep original function results for sequences, date, time, and GUIDs.

Copyright © 1999, 2025, Oracle and/or its affiliates.

TAC does this automatically for your application and declares success when it is correct to do so. This enables application developers to focus on business functionality while TAC handles the failover in the best possible way.

Know about planned failover?

With TAC enabled, the planned failover engine in the Oracle AI Database looks for places where it is confident to failover and initiates the failover. This bounds your drain time and aims to prevent sessions from being aborted. This is automatic with both TAC and AC, but it is frequently with TAC.

Difference between TAC and AC (original)

AC needs request boundaries, which means AC depends on Oracle or third-party pools that embed request boundaries when connections are borrowed and returned to the pool. Whereas, TAC uses provided request boundaries and creates its own. With TAC request boundaries advance automatically when there is no transaction, the session state is restorable, and cursors are closed.

AC provides similar correctness guarantees for a successful failover as TAC, with a small variation. AC enforces same client visible session states at the start of replay as the original session. AC replays side effects, such as sending email by default. TAC does not replay side effects as the default. The action for side effects is customizable in Oracle AI Database 26ai.

TAC request boundaries

TAC creates its own request boundaries, when:

- There is no transaction.
- The session state is restorable (no PL/SQL global variables, temporary tables, OJVM, and session duration LOBs)
- The cursors are resumable in Oracle AI Database 26ai or closed

Cleaning sessions for next usage

The service attribute `RESET_STATE` cleans your session state and cursors at the end of each request so these states do not leak to later requests. `RESET_STATE` saves you work and is future proofed, meaning that later changes to your application are also cleaned. It allows TAC to re-enable if TAC was unable to clean a session due to not restorable states.

Using Transparent Application Continuity in Oracle Cloud Environments

Transparent Application Continuity is enabled by default on TP and TPURGENT services in an Oracle Cloud environment for Oracle Autonomous AI Database–Dedicated, and is available for Oracle Autonomous AI Database–Serverless.

In Oracle Cloud environments, the use of `FAILOVER_RESTORE` and wallets means that you should not have to add callbacks to set initial state.

There are two features that work together to enable Transparent Application Continuity automatically:

- For planned outages, sessions that reach a safe place for transactions are drained from the instance and automatically failed over to another instance. For sessions that do not drain, Oracle AI Database determines where to fail over the session and invokes Application Continuity to fail over the session.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- For unplanned outages, Transparent Application Continuity transfers the user sessions to a remaining instance, hiding the outage from users automatically without having to understand or change the application.

Enable Application Continuity on Your Service

You can change the service attributes specified on your service by using the generic package `DBMS_APP_CONT_ADMIN`. Use this procedure to enable Application Continuity or Transparent Application Continuity, or to disable failover. New sessions will use the new failover type. You must have the `PDBADMIN` user permissions to use these procedures. Use the full service name in these examples.

- Use the `dbms_app_cont_admin.enable_tac('TPURGENT')` procedure to enable Transparent Application Continuity for your service:

```
SQL> execute dbms_app_cont_admin.enable_tac('TPURGENT');
```

- Use the `dbms_app_cont_admin.enable_ac('TPURGENT')` procedure to enable Application Continuity for your service:

```
SQL> execute dbms_app_cont_admin.enable_ac('TPURGENT');
```

- Use the `dbms_app_cont_admin.disable_failover('HIGH')` procedure to disable failover for your service:

```
SQL> execute dbms_app_cont_admin.disable_failover('HIGH');
```

- Use the `dbms_app_cont_admin.acchk_set` procedure to enable ACCHK for your service:

```
SQL> execute dbms_app_cont_admin.acchk_set(true);
```

- Use the `dbms_app_cont_admin.acchk_set_filter` procedure to set ACCHK filter for your service:

```
SQL> execute
dbms_app_cont_admin.acchk_set_filter(DBMS_APP_CONT_ADMIN.SERVICE_FIL
TER, 'TPURGENT');
```

- Use the `dbms_app_cont_admin.enable_reset_state` procedure to enable RESET STATE for your service:

```
SQL> execute dbms_app_cont_admin.enable_reset_state('TPURGENT', 'LEVEL1');
```

- Use the `dbms_app_cont_admin.set_draining` procedure to configure draining for your service:

```
SQL> execute dbms_app_cont_admin.set_draining('TPURGENT', 300,
'IMMEDIATE');
```

Configuring Application Continuity

Application Continuity (AC) requires using an Oracle Pool or an application that allows Transparent Application Continuity (TAC) to discover request boundaries.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Overview of Application Continuity Configuration Tasks
  The Application Continuity features in various Oracle applications are used automatically if you set the required service attributes.
- Configuring Connections for High Availability and Application Continuity
  These are general recommendations for configuring the connections used by applications for high availability.
- Configuring Oracle AI Database for Application Continuity
  Before you can use Application Continuity, you must ensure that your system is configured correctly.
- Establishing the Initial State Before Application Continuity Replays
  At the beginning of replay, Application Continuity restores the initial session state.
- Potential Side Effects of Application Continuity
  When you use Application Continuity with the service attribute `FAILOVER_TYPE` set to `TRANSACTION`, statements that perform side effects are replayed.

Overview of Application Continuity Configuration Tasks

The Application Continuity features in various Oracle applications are used automatically if you set the required service attributes.

The main actions for ensuring transparent replay for an application are the following:

1. Ensure that you have the necessary CPU and memory resources.
   - **CPU**: Application Continuity is managed on the client and server sides and requires minimal CPU overhead to operate.

     At the client, CPU is used to build proxy objects and for garbage collection (GC).

     At the server, CPU is used for validation. CPU overhead is reduced for platforms with current Intel and SPARC chips where validation is assisted in the hardware.
   - **Memory**: When using Application Continuity, the replay driver requires more memory than the base driver because the calls are retained until the end of a request. At the end of the request, the calls are released to the garbage collector. This action differs from the base driver that releases closed calls.

     The memory consumption of the replay driver depends on the number of calls per request. If this number is small, then the memory consumption of the replay driver is less, and comparable to the base driver.

     To obtain the best performance, you must set the same value for both the `-Xmx` and `-Xms` parameters on the client. For example, if there is sufficient memory, then allocate 4 to 8 GB (or more) of memory for the Virtual Machine (VM), for example, by setting `-Xms4g` for 4 GB. If the `-Xms` parameter has a lower value, then the VM also uses a lower value from the operating system, and performance might suffer and garbage collection operations increase.
2. Determine whether the application borrows and returns connections from the connection pool, for example WebLogic Server Pool, Universal Connection Pool, OCI Session Pool, Oracle Tuxedo request, or ODP.NET connection pool, for each request, or whether to add `beginRequest` and `endRequest` APIs to the application's own connection pool to identify request boundaries for Java, only.

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **⚠ Caution**
>
> Do not use the `beginRequest` and `endRequest` Java API calls or the `OCIRequestBegin` and `OCIRequestEnd` OCI API calls anywhere other than at request boundaries (borrow and return connections from your connection pool). `endRequest` indicates that the request is complete, and that it is now stateless. Replay starts from the next `beginRequest`. If there is prior state, it must be reestablished using `FAILOVER_RESTORE` or callback.

3. Application Continuity replays all states in a request. If the application sets states before vending connections, `FAILOVER_RESTORE` or a callback is needed. When using Oracle WebLogic Server or the Universal Connection Pool, use `FAILOVER_RESTORE`, connection labeling, or triggers. When using Oracle Call Interface (OCI) session pool, Oracle Tuxedo or ODP.NET with Oracle Database 18c or later clients, use `FAILOVER_RESTORE`, and only add the Transparent Application Failover (TAF) callback if it is needed. The labeling is used for both runtime and replay. Oracle strongly recommends setting `FAILOVER_RESTORE` to `AUTO`, `LEVEL1`, or `LEVEL2` with wallets, to use database templates.
4. Determine whether the application requires, and therefore needs to configure keeping original values for, `SYSDATE`, `SYSTIMESTAMP`, and `SYS_GUID` and sequences during failover.
5. Assess the application style for the `session_state_consistency` value, and set the appropriate value on the service:
   - If `session_state_consistency` is set to `AUTO`, then Transparent Application Continuity monitors the session state and decides what to do. If you are unsure about state usage or know that states can change in the future, then use Transparent Application Continuity. With Oracle AI Database 26ai templates and wallets, modifiable parameters are restored. For .NET applications, use ODP.NET, Managed Driver 23ai or later, or ODP.NET Unmanaged Provider or later.

     Use session state `AUTO` so that the session state is managed as the application evolves. Session state `AUTO` is available with TAC. When using TAC, both the client visible and server visible session states must match for a successful failover or session migration. When there is unrestorable session state at the beginning of a request, TAC does not enable as the state cannot be proven at the beginning of the request. Use ACCHK and the protection statistics to detect unrestorable session state at the beginning of request. Use `RESET_STATE` to clean session state automatically by Oracle AI Database between requests.
   - If `session_state_consistency` is set to `DYNAMIC`, then the application changes the environment or settings during the request. Replay is disabled after the first `COMMIT` until the beginning of the next request. `DYNAMIC` is the default mode, appropriate for most applications.

   > **ⓘ Note**
   >
   > The service attribute values `FAILOVER_TYPE = TRANSACTION` with `SESSION_STATE_CONSISTENCY = STATIC` are no longer a supported service attribute combination.
6. Determine if any requests in the application should not be replayed.

   For example, replay may need to be disabled for requests using external PL/SQL actions.
7. Follow these configuration guidelines:

Copyright © 1999, 2025, Oracle and/or its affiliates.

- For Java, use Oracle Database 12c release 1 (12.1.0.1), or later. For OCI-based applications, use Oracle Database 12c release 2 (12.2), or later.
- For .NET applications, use ODP.NET, Managed Driver 23ai or later, or ODP.NET, Unmanaged Provider 12.2, or later, connecting to an Oracle Database 12c Release 2 (12.2) or later. By default, Application Continuity is enabled for ODP.NET applications in this configuration. When using OCI-based applications that do not use the OCI Session Pool, including SQL*Plus, use Transparent Application Continuity that adds boundaries for you.

  > **Note**
  >
  > Oracle Data Provider for .NET (ODP.NET), Unmanaged Driver is deprecated in Oracle AI Database.

- Custom Java pools and standalone Java applications can also use the JDBC replay data source directly. When using custom Java pools and standalone applications, Oracle recommends that you use Transparent Application Continuity which adds boundaries for you. You can also add `beginRequest` and `endRequest` Java APIs to your application.
- If the application does not borrow and return from the Oracle connection pools, explicitly mark request boundaries. For example, if using custom JDBC pools, or other pools, Oracle recommends that you use Transparent Application Continuity which adds boundaries for you. You can also add `beginRequest` and `endRequest` Java APIs to your application. These APIs can also be used for standalone JDBC applications without a connection pool.
- Enable FAN for fast interrupt on errors. This is essential to eliminate a TCP delay occurring before the failover can start. In 12.2 FAN is built into the JDBC and OCI drivers and is on by default for Java.
- Use a database service to connect; never use a SID or an instance name, or the administration service that is the `DB_NAME` or `DB_UNIQUE_NAME`.
- Use a connection string that sets retries for new incoming connections and a delay between these retries.
- For the service, set `FAILOVER_TYPE` to `TRANSACTION` for Application Continuity or set `FAILOVER_TYPE` to `AUTO` for Transparent Application Continuity. Set `COMMIT_OUTCOME` to `TRUE` and, for OCI FAN, set `NOTIFICATION` to `TRUE`. Optionally to find the best connections to use, set `GOAL` to `SERVICE_TIME` and `CLB_GOAL` to `LONG`.
- Use the statistics for request boundaries and protection level to monitor the level of coverage. If you need more details, then use Application Continuity Check Coverage (with the ORAchk utility) to report the percentage of requests that are fully protected by Application Continuity, and the location of those requests that are not fully protected. Use this coverage check before deployment and after application changes. Developers and management will know how well protected an application release is from failures of the underlying infrastructure. If there is a problem, then it can be fixed before the application is released, or waived knowing the level of coverage.

Related Topics

- Session State Consistency
  Session state consistency describes how non-transactional state is changed during a request.
- *Oracle Data Provider for .NET Developer's Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

- *Oracle AI Database JDBC Developer's Guide*
- *Oracle Universal Connection Pool Developer's Guide*

Configuring Connections for High Availability and Application Continuity

These are general recommendations for configuring the connections used by applications for high availability.

If you are using Java, then you must use the `oracle.jdbc.datasource.impl.OracleDataSource`, `oracle.jdbc.replay.OracleConnectionPoolDataSourceImpl`, or `oracle.jdbc.replay.driver.OracleXADataSourceImpl` data source to obtain JDBC connections. These data sources support all the properties and configuration parameters of all the Oracle JDBC data sources, for example, the `oracle.jdbc.pool.OracleDataSource`.

For OCI based applications including SQL*Plus and ODP.NET, the OCI driver 12.2, and later, supports Application Continuity.

You must remember the following points while using the connection URL:

- If the `REMOTE_LISTENER` setting for the database does not match the addresses in the `ADDRESS_LIST` at the client, then it does not connect, showing `services cannot be found`. So, the `REMOTE_LISTENER` setting for the database *must* match the addresses in the `ADDRESS_LIST` at the client:
  - If the connect string uses the SCAN Name, then `REMOTE_LISTENER` must be set to the SCAN name.
  - If the connect string uses an `ADDRESS_LIST` of host VIPs, then `REMOTE_LISTENER` must be set to an address list that includes all SCAN VIPs and all host VIPs

  > **Note**
  >
  > Use SCAN for location independence, to avoid having to reconfigure the client when you add or delete nodes, or when databases change to running on different nodes.

- Set `RETRY_COUNT`, `RETRY_DELAY`, `CONNECT_TIMEOUT`, and `TRANSPORT_CONNECT_TIMEOUT` parameters in the connection string. These settings improve acquiring new connections at runtime, at replay, and during work drains for planned outages.

  The `CONNECT_TIMEOUT` parameter is equivalent to the `SQLNET.OUTBOUND_CONNECT_TIMEOUT` parameter in the `sqlnet.ora` file and applies to the full connection. The `TRANSPORT_CONNECT_TIMEOUT` parameter applies per address.
- Set `CONNECT_TIMEOUT` to a high value to prevent an overabundance of log ins. Low values can result in *log in storms* to the application. Do not set `(RETRY_COUNT+1)*RETRY_DELAY` or `CONNECT_TIMEOUT` larger than your response time SLA. The application must either connect or receive an error within the response time SLA.
- You are not recommended to use Easy*Connect for High Availability deployments.

Example 6-1 Recommended TNS Entry for High Availability

The following is an example of a Transparent Network Substrate (TNS entry). This is the required TNS format for Oracle Notification Service (ONS) to be auto configured. ONS is the

Copyright © 1999, 2025, Oracle and/or its affiliates.

transport system used for Fast Application Notification (FAN). Oracle recommends using FAN with Application Continuity to provide fast outage detection.

```
myAlias=(DESCRIPTION=
   (CONNECT_TIMEOUT=90)(RETRY_COUNT=30)(RETRY_DELAY=3)
(TRANSPORT_CONNECT_TIMEOUT=3)
   (ADDRESS_LIST=
      (LOAD_BALANCE=ON)
      (ADDRESS=(PROTOCOL=TCP)(HOST=RAC-scan)(PORT=1521)))
   (ADDRESS_LIST=
      (LOAD_BALANCE=ON)
      (ADDRESS=(PROTOCOL=TCP)(HOST=DG-Scan)(PORT=1521)))
   (CONNECT_DATA=(SERVICE_NAME=service_name)))
```

Related Topics

- Local Naming Parameters in the tnsnames.ora File
- Installing and Configuring Oracle Data Provider for .NET

Configuring Oracle AI Database for Application Continuity

Before you can use Application Continuity, you must ensure that your system is configured correctly.

Your Oracle AI Database configuration must include the following to use Application Continuity:

- If you are using Oracle Real Application Clusters (Oracle RAC) or Oracle RAC One Node, Oracle Data Guard, or Oracle Active Data Guard, then ensure that Fast Application Notification (FAN) is configured with Oracle Notification Service (ONS) to communicate with pools and drivers that are Oracle Database 12c or later.
- Set the service attributes on the service for replay and load balancing. For example, set:
  - `FAILOVER_TYPE = AUTO | TRANSACTION`: Use `FAILOVER_TYPE=AUTO` for Transparent Application Continuity or `FAILOVER_TYPE=TRANSACTION` for Application Continuity. This attribute enables the replay functionality for the replay drivers and Application Continuity. Oracle drivers keep track of all replayable statements issued during a database session. If all of the statements are replayable, and any in-flight transactions did not commit or the session is in conversation, then Oracle replays the uncommitted work following a planned or unplanned database outage. This mode re-establishes transactional and non-transaction states automatically with no additional application steps.
  - `REPLAY_INITIATION_TIMEOUT = n`: For setting the duration, in seconds, to allow replay to start. For example, you might set the value of *n* to 300.
  - `FAILOVER_RETRIES = 3`: For specifying the number of connection retries for each replay. Oracle recommends that you set `retry_count` in the `tnsnames.ora` file, rather than on the service.
  - `FAILOVER_DELAY = 3`: For specifying the delay in seconds between connection retries. Oracle recommends that you set `retry_count` in the `tnsnames.ora` file, rather than on the service.
  - `COMMIT_OUTCOME = TRUE`: If you are using Transaction Guard, the `COMMIT_OUTCOME` service parameter determines whether the transaction **commit outcome** is accessible after the `COMMIT` has run and an outage has occurred. Starting with Oracle AI Database 26ai, `COMMIT_OUTCOME` defaults to using Native Transaction Guard. Native Transaction

Copyright © 1999, 2025, Oracle and/or its affiliates.

Guard is used whenever the XID can be sent to the driver in advance of the `COMMIT`. While Oracle AI Database has always made the `COMMIT` action durable, Transaction Guard makes the outcome of the `COMMIT` durable.

– `FAILOVER_RESTORE = AUTO | LEVEL1 | LEVEL2`: Use `FAILOVER_RESTORE=AUTO` or `FAILOVER_RESTORE=LEVEL1` for Transparent Application Continuity and `FAILOVER_RESTORE=LEVEL1 | LEVEL2` for Application Continuity. To automatically restore client states that are preset on the connection pool or driver before replay begins—including NLS states, TAGS (MODULE, ACTION, ECID, CLIENT_ID, CLIENT_INFO) states, and all server modifiable session states that use `FAILOVER_RESTORE=LEVEL1 | LEVEL2` with Wallets.

> **⚠ Caution**
>
> *Do not use* the default database service corresponding to the `DB_NAME` or `DB_UNIQUE_NAME`. Also, *do not use* the default database service for high availability, because this service cannot be enabled or disabled, and cannot be relocated on Oracle RAC or switched over to Oracle Data Guard. This service is reserved for Oracle Enterprise Manager Cloud Control (Cloud Control) and for DBAs.

Establishing the Initial State Before Application Continuity Replays

At the beginning of replay, Application Continuity restores the initial session state.

The topics in this section describe how to configure the service and database to successfully restore the session state before replaying.

- Checking Initial States for Application Continuity
  Learn about the session state that is restored.
- FAILOVER_RESTORE
  Setting `FAILOVER_RESTORE` to `LEVEL1` (for AC with older 19c versions), `LEVEL2` (for AC with database templates), or `LEVEL1` or `AUTO` (for TAC) automatically restores all modifiable parameters before replaying the request.
- States Restored with FAILOVER_RESTORE
  The following session states are restored when `FAILOVER_RESTORE` is set to `LEVEL1`, `LEVEL2`, or `AUTO`, with or without wallets.
- Full FAILOVER_RESTORE with Database Templates
  Oracle AI Database 26ai supports database templates. A database template is a set of session states with their values that describe a database session.
- FAILOVER_RESTORE with Database Templates
  Starting with Oracle AI Database 26ai, Application Continuity uses database templates to checkpoint session state, restore the session state at the start of replay, and support session migration during planned maintenance.
- Configuring a Keystore using WALLET_ROOT for FAILOVER_RESTORE
  Use these steps to configure encryption of dictionary credentials by using a software keystore (wallet) and Transparent Data Encryption (TDE) for use with `FAILOVER_RESTORE`.
- Configuring a Keystore using SQLNET.ORA for FAILOVER_RESTORE
  Use these steps to configure encryption of dictionary credentials by using `SQLNET.ORA` to point to the wallet location for use with `FAILOVER_RESTORE`.
- FAILOVER_RESTORE = NONE and No Callback
  `FAILOVER_RESTORE = NONE` does not restore session state at failover.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Connection Labeling
  Connection Labeling is not required unless you want to restore more complex states, such as proxying a session.
- Connection Initialization Callback
  Use the Connection Initialization Callback only if you need to restore session states such as `SYS_CONTEXT` and password protected roles. `FAILOVER_RESTORE` with wallets restores all modifiable parameters.

Checking Initial States for Application Continuity

Learn about the session state that is restored.

If your application sets an initial state for the connection before allowing applications to use the connection, then Application Continuity must establish this initial state before replay starts. With common states, which are listed in the topics in this section, `FAILOVER_RESTORE` restores the states. However, you must review the topics that describe common states. If the states that your application presets are not listed, and the application needs initial states, then you must add an additional callback.

Examples of states that can be preset include:

- PL/SQL package state
- NLS Setting
- Optimizer setting

During a request, Application Continuity reestablishes the entire state for the request. This prerequisite is for the initial state before Application Continuity starts replaying.

A callback is not required if `FAILOVER_RESTORE` restores all required states, which is the case for most applications.

> **See Also**
>
> *Oracle AI Database Release Notes* for your platform, because more parameters are restored in each release

FAILOVER_RESTORE

Setting `FAILOVER_RESTORE` to `LEVEL1` (for AC with older 19c versions), `LEVEL2` (for AC with database templates), or `LEVEL1` or `AUTO` (for TAC) automatically restores all modifiable parameters before replaying the request.

`FAILOVER_RESTORE` is a setting on your service. Available with Oracle Database 12.2 and later, `FAILOVER_RESTORE` automatically restores all session states available for your application at the client-side.

Oracle recommends setting `FAILOVER_RESTORE` to `LEVEL2`, `LEVEL1`, or `AUTO` for all applications.

Refer to States Restored with FAILOVER_RESTORE for the client-side session states that are restored.

Copyright © 1999, 2025, Oracle and/or its affiliates.

States Restored with FAILOVER_RESTORE

The following session states are restored when `FAILOVER_RESTORE` is set to `LEVEL1`, `LEVEL2`, or `AUTO`, with or without wallets.

Session States That Are Restored

- NLS_CALENDAR
- NLS_CURRENCY
- NLS_DATE_FORMAT
- NLS_DATE_LANGUAGE
- NLS_DUAL_CURRENCY
- NLS_ISO_CURRENCY
- NLS_LANGUAGE
- NLS_LENGTH_SEMANTICS
- NLS_NCHAR_CONV_EXCP
- NLS_NUMERIC_CHARACTER
- NLS_SORT
- NLS_TERRITORY
- NLS_TIME_FORMAT
- NLS_TIME_TZ_FORMAT
- TIME_ZONE
- NLS_TIMESTAMP_FORMAT
- NLS_TIMESTAMP_TZ_FORMAT
- CURRENT_SCHEMA
- MODULE
- ACTION
- CLIENT_ID
- AUTOCOMMIT states (for Java and SQL*Plus)
- CONTAINER (PDB) and SERVICE
- ROLES (excludes secure roles, which continue to require a call back)
- ROW_ARCHIVAL
- EDITION
- ERROR_ON_OVERLAP_TIME
- SQL_TRANSLATION_PROFILE
- CLIENT_INFO. (JDBC)

Full FAILOVER_RESTORE with Database Templates

Oracle AI Database 26ai supports database templates. A database template is a set of session states with their values that describe a database session.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Database templates are created or assigned by the database automatically as the session state changes. TAC and AC use templates to restore the session state at replay.

Using database templates, a larger subset of sessions failover during planned maintenance and unplanned outages because their session states are restored and verified automatically, and clients use less memory. Database templates are enabled by default for TAC and you are recommended to use `FAILOVER_RESTORE=LEVEL2` for AC.

At failover, `FAILOVER_RESTORE` restores session parameters that were altered in your session. Examples of session parameters restored include `optimizer_capture_sql_plan_baselines` and `create_stored_outlines` that were set in the session.

If you are already using a logon trigger, connection label, or callback to restore session parameters, you can continue to use them. Labels and callbacks are fully supported with and without extended `FAILOVER_RESTORE`. Using extended `FAILOVER_RESTORE` has the advantage that you do not need to update it as the application changes.

To use this feature, you must set `FAILOVER_RESTORE` to `LEVEL2` or `AUTO` and ensure that the dictionary credentials are encrypted on your system. You can also set `FAILOVER_RESTORE` to `LEVEL1` for AC and TAC that also restores all modifiable session parameters when wallets are used.

There are two methods of adding the wallet or keystore for dictionary credentials encryption:

> **Note**
>
> You must use either one of the following methods to configure a keystore for `FAILOVER_RESTORE`.

- **Recommended**: Use the `WALLET_ROOT` database instance initialization parameter to specify the wallet location. Using an initialization parameter for the wallet location ensures consistency across Oracle Real Application Clusters (Oracle RAC) and Oracle Data Guard. This method requires a rolling restart of the database.
- Modify the `sqlnet.ora` file in your `TNS_ADMIN` directory on the database server to point to the wallet location. This method does not require a database restart,unless your database runs on the Microsoft Windows operating system. You are responsible for ensuring that the `sqlnet.ora` files are consistent in all `ORACLE_HOME` directories. Also, the `sqlnet.ora` might require additional maintenance when performing database upgrades.

Related Topics

- Recommendations for Oracle Net Services When Upgrading Oracle AI Database
- Using Application Contexts to Retrieve User Information
- Connection Initialization Callback
  Use the Connection Initialization Callback only if you need to restore session states such as `SYS_CONTEXT` and password protected roles. `FAILOVER_RESTORE` with wallets restores all modifiable parameters.

FAILOVER_RESTORE with Database Templates

Starting with Oracle AI Database 26ai, Application Continuity uses database templates to checkpoint session state, restore the session state at the start of replay, and support session migration during planned maintenance.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Database templates restore server-side and client-visible session states at the beginning of the Application Continuity replay, thus increasing Application Continuity protection. To use this feature set `FAILOVER_RESTORE=AUTO` for Transparent Application Continuity or set `FAILOVER_RESTORE=LEVEL2` for Application Continuity.

`FAILOVER_RESTORE` with database templates improves coverage for applications with a wider set of session states and lowers client memory usage. It enables you to migrate more sessions during planned maintenance and to have a higher number of successful and faster failovers during unplanned outages as protection by Application Continuity is higher.

Database templates for Application Continuity ensures the migration of sessions containing server-side session states that have not reached a boundary to drain during the draining period, and improves the Application Continuity protection for unplanned outages. Database templates for Application Continuity enables migration of sessions using secure and large session states for planned maintenance and for load balancing.

To enable this feature, you must set `FAILOVER_RESTORE` to `LEVEL2` or `AUTO`. To retain the Oracle Database 19c restore functionality, set `FAILOVER_RESTORE` to `LEVEL1` for both Application Continuity and Transparent Application Continuity. To restore all session states at failover, the database must have encryption enabled, for example using TDE wallets.

Related Topics

- Configuring a Keystore using WALLET_ROOT for FAILOVER_RESTORE
  Use these steps to configure encryption of dictionary credentials by using a software keystore (wallet) and Transparent Data Encryption (TDE) for use with `FAILOVER_RESTORE`.

Configuring a Keystore using WALLET_ROOT for FAILOVER_RESTORE

Use these steps to configure encryption of dictionary credentials by using a software keystore (wallet) and Transparent Data Encryption (TDE) for use with `FAILOVER_RESTORE`.

1. If you are using Oracle Autonomous AI Database, you do not need to perform these steps.

   For Oracle Autonomous AI Database, a software keystore already exists and dictionary credentials are encrypted

2. If you are not using Oracle Autonomous AI Database, then check if your system is already configured to enforce dictionary credential encryption.

   a. Verify a wallet (a Keystore) exists using the following SQL query:

   ```
   SELECT con_id, wrl_type, status , wallet_type FROM V$ENCRYPTION_WALLET
   ORDER BY con_id;
       CON_ID WRL_TYPE     STATUS   WALLET_TYPE
   ---------- ------------ -------- -----------
            0 FILE         OPEN     PASSWORD
   ```

   If no rows are returned by this SQL query, then a wallet, or keystore, does not exist.

   b. Verify that dictionary credentials are encrypted using the following SQL query:

   ```
   SQL> SELECT enforcement FROM DICTIONARY_CREDENTIALS_ENCRYPT;
   ENFORCEMENT
   ---------------
   ENABLED
   ```

   If this SQL query returns `DISABLED`, then the dictionary is not encrypted.

Copyright © 1999, 2025, Oracle and/or its affiliates.

If you have a wallet and dictionary credentials encrypted, you can use extended FAILOVER_RESTORE by setting the attribute on your service. You do not need to complete any more of the steps in this procedure.

If you do not have an existing wallet, or if you need to enable dictionary credentials encryption, then continue with the following steps.

3. Configure the database to use a software keystore.

   a. If necessary, create a directory to store the wallet.

      The location selected needs to be shared across Oracle RAC nodes and replicated to Oracle Data Guard sites. For Oracle RAC, the directory must be on shared storage.

   b. Change the initialization parameter WALLET_ROOT.

      The parameter value should be the directory where the wallet is stored.

      ```
      SQL> CONNECT / AS SYSDBA
      SQL> ALTER SYSTEM SET WALLET_ROOT='/myOracleBase/admin/wallet/'
      SCOPE=spfile;
      ```

      You can also change the WALLET_ROOT initialization parameter using the init.ora file on all RAC nodes by adding wallet_root='/myOracleBase/admin/wallet/'.

      Starting with Oracle AI Database, the parameter ENCRYPTION_WALLET_LOCATION is desupported.
      To store and retrieve the TDE wallet, use the WALLET_ROOT structure (introduced with Oracle Database 18c).

      > **Note**
      >
      > If Transparent Data Encryption (TDE) is enabled, but WALLET_ROOT is not configured, then you will be blocked from upgrading to Oracle AI Database. This block for upgrades of databases using TDE is to prevent the possibility of not being able to open the database after the upgrade.

   c. Change the initialization parameter TDE_CONFIGURATION to specify a software keystore.

      ```
      SQL> CONNECT / AS SYSDBA
      SQL> ALTER SYSTEM SET TDE_CONFIGURATION="KEYSTORE_CONFIGURATION=FILE"
      SCOPE=BOTH SID='*'
      ```

      You can also change the TDE_CONFIGURATION initialization parameter using the init.ora file on all RAC nodes by adding TDE_CONFIGURATION="KEYSTORE_CONFIGURATION=FILE" .

   d. Perform a rolling restart of the database instances to activate the new initialization parameters.

      For example, for a two node clustered database named orcl, where the instances are named orcl1 and orcl2, you would use the following commands to stop and restart each instance individually to avoid a complete outage of your database.

      ```
      $ srvctl stop instance -db orcl -instance orcl1 -drain_timeout 600 -
      stopoption IMMEDIATE
      $ srvctl start instance -db orcl -instance orcl1
      ```


Copyright © 1999, 2025, Oracle and/or its affiliates.

```
$ srvctl stop instance -db orcl -instance orcl2 -drain_timeout 600 -
stopoption IMMEDIATE
$ srvctl start instance -db orcl -instance orcl2
```

> **Note**
>
> Fleet Patching and Provisioning, if used, automates this process and can be used instead if you are modifying the parameters during a patch upgrade.

e. Verify that the parameters are set to the correct values after restarting the instances.

```
SQL> SHOW PARAMETER WALLET_ROOT;
SQL> SHOW PARAMETER TDE_CONFIGURATION;
```

4. Create a keystore with a password, if one does not already exist.

   In the following example *password* is the password for the keystore. The password is case sensitive. Keystore passwords adhere to the same rules as database user passwords.

```
SQL> CONNECT / AS SYSKM
SQL> ADMINISTER KEY MANAGEMENT CREATE KEYSTORE IDENTIFIED BY "password";
```

5. Open a keystore and set an encryption key.

   If your database is configured as an Oracle Multitenant database, then a keystore and encryption key must be set for each PDB using the `CONTAINER=all` clause. In the following example *password* is the password for the keystore.

```
SQL> CONNECT / AS SYSKM
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "password"
CONTAINER=all;
SQL> ADMINISTER KEY MANAGEMENT SET ENCRYPTION KEY IDENTIFIED BY "password"
SQL> WITH BACKUP CONTAINER=all;
```

   If your database is not configured as an Oracle Multitenant database, then use the following SQL commands, where *password* is the password for the keystore:

```
SQL> CONNECT / AS SYSKM
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "password";
SQL> ADMINISTER KEY MANAGEMENT SET ENCRYPTION KEY IDENTIFIED BY "password"
WITH BACKUP;
```

6. Encrypt the database dictionary credentials.

   Use an operator with the SYSKM role to run the following SQL command from within the Container Database (CDB) root and each PDB.

```
SQL> CONNECT / AS SYSKM
SQL> ALTER DATABASE DICTIONARY ENCRYPT CREDENTIALS;
```

Encryption and decryption of the information occurs automatically at the server during failover restoration.

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Warning**
>
> It is recommended to backup the software keystore and the wallet location. Do not lose your TDE software keystore or `WALLET_ROOT` location. If you do, for Application Continuity or Transparent Application Continuity, a new keystore can be created but encrypted dictionary credentials will need to be re-instantiated. Failover will not succeed while there is a mismatch in the wallet keys.

Related Topics

- Configuring a Software Keystore
- Managing the Keystore and the Master Encryption Key
- Permitted Features, Options, and Management Packs by Oracle AI Database Offering

Configuring a Keystore using SQLNET.ORA for FAILOVER_RESTORE

Use these steps to configure encryption of dictionary credentials by using `SQLNET.ORA` to point to the wallet location for use with `FAILOVER_RESTORE`.

> **Note**
>
> Oracle recommends that you configure a Keystore using `WALLET_ROOT` for `FAILOVER_RESTORE` in preference to `SQLNET.ORA`.

This method does not require a database restart, unless your database runs on the Microsoft Windows operating system. You are responsible for ensuring that the `sqlnet.ora` files are consistent in all `ORACLE_HOME` directories.

1. If you are using Oracle Autonomous AI Database, you do not need to perform these steps.

   For Oracle Autonomous AI Database, a software keystore already exists and dictionary credentials are encrypted

2. If you are not using Oracle Autonomous AI Database, then check if your system is already configured to enforce dictionary credential encryption.
   a. Verify a wallet exists using the following SQL query:

   ```
   SQL> SELECT con_id, wrl_type, status , wallet_type FROM
   V$ENCRYPTION_WALLET
   ORDER BY con_id;
       CON_ID WRL_TYPE     STATUS   WALLET_TYPE
   --------- ------------ -------- -----------
             0 FILE         OPEN     PASSWORD
   ```

   If no rows are returned by this SQL query, then a wallet, or keystore, does not exist.

   b. Verify that dictionary credentials are encrypted using the following SQL query:

   ```
   SQL> SELECT enforcement FROM DICTIONARY_CREDENTIALS_ENCRYPT;
   ENFORCEMENT
   --------------
   ENABLED
   ```


Copyright © 1999, 2025, Oracle and/or its affiliates.

If this SQL query returns `DISABLED`, then the dictionary is not encrypted.

If you have a wallet and dictionary credentials encrypted, you can use extended `FAILOVER_RESTORE` by setting the attribute on your service. You do not need to complete any more of the steps in this procedure.

If you do not have an existing wallet, or if you need to enable dictionary credentials encryption, then continue with the following steps.

3. Configure the database to use a wallet.

   a. View the `TNS_ADMIN` environment variable to find the location of the network configuration files used by your database.

      - On Linux and UNIX systems, as the Oracle Home software owner, view the current setting of the `TNS_ADMIN` environment variable.

        ```
        $ env | grep TNS_ADMIN
        ```

      - On Microsoft Windows systems, check the value set for `TNS_ADMIN` as both an environment variable and in the registry in the path `Computer\HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_`*`HOME_NAME`*.

      If the `TNS_ADMIN` variable is not set, then the default location of `$ORACLE_BASE_HOME/network/admin` is used for the Oracle Net configuration files with read-only Oracle homes.

   b. If necessary, create a directory to store the wallet.

      The location selected needs to be shared across Oracle RAC nodes and replicated to Oracle Data Guard sites. For Oracle RAC, the directory must be on shared storage.

   c. Change the initialization parameter `TDE_CONFIGURATION` to specify a software keystore.

      ```
      SQL> CONNECT / AS SYSDBA
      SQL> ALTER SYSTEM SET TDE_CONFIGURATION="KEYSTORE_CONFIGURATION=FILE"
      SCOPE=BOTH SID='*'
      ```

4. Create a keystore with a password, if one does not already exist.

   In the following example *myOracleWalletLoc* is the full path name of the directory created to store the wallet (or keystore) and *password* is the password for the keystore. The password is case sensitive. Keystore passwords adhere to the same rules as database user passwords.

   ```
   SQL> CONNECT / AS SYSKM
   SQL> ADMINISTER KEY MANAGEMENT CREATE KEYSTORE '/myOracleWalletLoc'
   IDENTIFIED BY "password";
   ```

5. Open a keystore and set an encryption key.

   If your database is configured as an Oracle Multitenant database, then a keystore and encryption key must be set for each PDB using the `CONTAINER=all` clause. In the following example *password* is the password for the keystore.

   ```
   SQL> CONNECT / AS SYSKM
   SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "password"
   CONTAINER=all;
   SQL> ADMINISTER KEY MANAGEMENT SET ENCRYPTION KEY IDENTIFIED BY "password"
   WITH BACKUP CONTAINER=all;
   ```


Copyright © 1999, 2025, Oracle and/or its affiliates.

If your database is not configured as an Oracle Multitenant database, then use the following SQL commands, where *password* is the password for the keystore:

```
SQL> CONNECT / AS SYSKM
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "password";
SQL> ADMINISTER KEY MANAGEMENT SET ENCRYPTION KEY IDENTIFIED BY "password"
WITH BACKUP;
```

6. Encrypt the database dictionary credentials.

   Use an operator with the SYSKM role to run the following SQL command from within the Container Database (CDB) root and each PDB.

   ```
   SQL> CONNECT / AS SYSKM
   SQL> ALTER DATABASE DICTIONARY ENCRYPT CREDENTIALS;
   ```

Encryption and decryption of the information occurs automatically at the server during failover restoration.

> **⚠ Caution**
>
> It is recommended to backup the wallet location. Do not lose your wallet or location. If you do, for Application Continuity or Transparent Application Continuity, a new wallet can be created but encrypted dictionary credentials will need to be re-instantiated.
>
> Failover will not succeed while there is a mismatch in the wallet keys.

Related Topics

- Locating Oracle Net Services Configuration Files
- Using sqlnet.ora to Configure Transparent Data Encryption Keystores
- Permitted Features, Options, and Management Packs by Oracle AI Database Offering

FAILOVER_RESTORE = NONE and No Callback

`FAILOVER_RESTORE = NONE` does not restore session state at failover.

Oracle recommends setting `FAILOVER_RESTORE` to `LEVEL2`, `LEVEL1`, or `AUTO` for all applications.

Connection Labeling

Connection Labeling is not required unless you want to restore more complex states, such as proxying a session.

This scenario is applicable to Universal Connection Pool (UCP) and Oracle WebLogic server. Connection Labeling APIs are required to match secure session states- `SYS_CONTEXT` and password protected roles. They are also useful in reporting a failover on the driver side. When a connection is borrowed, Connection Labeling populates the gap by using a callback.

Related Topics

- *Oracle Universal Connection Pool Developer's Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

Connection Initialization Callback

Use the Connection Initialization Callback only if you need to restore session states such as `SYS_CONTEXT` and password protected roles. `FAILOVER_RESTORE` with wallets restores all modifiable parameters.

If your replaying driver uses an application callback to set the initial state of the session during runtime and replay, then the callback interface depends on whether the driver is a Java Database Connectivity (JDBC) driver or an Oracle Call Interface (OCI) driver.

In this scenario, with either JDBC or OCI, the driver uses an application callback to set the initial state of the session during runtime and replay. With the JDBC replay driver, the driver provides a connection initialization callback interface and methods to register and unregister connection initialization callbacks in the `oracle.jdbc.replay.OracleDataSource` interface. With OCI and Oracle Data Provider for .NET (ODP.NET), you register the Transparent Application Failover (TAF) callback.

When registered, the initialization callback is run every time a connection is borrowed from the pool, and at each successful reconnection following a recoverable error. (This is true for the JDBC/UCP initialization callback, and should be the same for TAF.) Using the same callback during both runtime and replay ensures that the same initialization is established at replay, as it was when the session was first established. An application is responsible for ensuring that the initialization actions are the same as that on the original connection before failover. If the callback invocation fails, then replay is disabled on that connection. Use the connection initialization callback only when the application has not implemented UCP and WebLogic Connection Labeling and the state cannot be restored automatically by setting either `FAILOVER_RESTORE=AUTO` or `FAILOVER_RESTORE=LEVEL1` for Transparent Application Continuity or `FAILOVER_RESTORE=LEVEL2` for Application Continuity.

Potential Side Effects of Application Continuity

When you use Application Continuity with the service attribute `FAILOVER_TYPE` set to `TRANSACTION`, statements that perform side effects are replayed.

When you use Transparent Application Continuity with the `FAILOVER_TYPE` service attribute set to `AUTO`, then replay is disabled when a side effect is detected. Replay re-enables at the next discovered or explicit boundary.

> **Note**
>
> As an application owner you can elect to disable replay for requests that contain side effects that you do not want to repeat. The simplest way to disable side effects is to use Transparent Application Continuity (set the service attribute `FAILOVER_TYPE` to `AUTO`, which disables side effects for you.

Application Continuity replays PL/SQL chronologically to restore database state. This serves to rebuild the session as if the user submission was delayed. Most applications want the full state rebuilt as if the submission was repeated, such as writing a report or completing some auditing. However, the actions that are replayed to build the state might include some for which you want to take action to accommodate or mitigate the effects of the replay. Some applications elect to disable replay for requests that contain calls that they do not want to repeat.

Examples of actions that create side effects include the following:

Copyright © 1999, 2025, Oracle and/or its affiliates.

- DBMS_ALERT calls (email or other notifications)
- DBMS_FILE_TRANSFER calls (copying files)
- DBMS_PIPE and RPC calls (to external sources)
- UTL_FILE calls (writing text files)
- UTL_HTTP calls (making HTTP callouts)
- UTL_MAIL calls (sending email)
- UTL_SMTP calls (sending SMTP messages)
- UTL_TCP calls (sending TCP messages)
- UTL_URL calls (accessing URLs)

For applications with external actions (such as autonomous transactions or using UTL_HTTP to issue a service-oriented application (SOA) call), Application Continuity is transparent when the application is satisfied with replaying external actions, such as resending email, auditing, and transferring a file.

Allowing and Disallowing Side Effects for 26ai

Replay disabled for side effect by default for TAC, use DBMS_APP_CONT_ADMIN.SET_REPLAY_RULES or DBMS_APP_CONT.APPLY_REPLAY_RULE to allow or disallow side effects.

Replayable Interface Side Effect Use Cases

The following use cases cover various scenarios for replayable interface side effects:

- Use the following interface to allow side effects with TAC for entire service:

```
SQL> execute
DBMS_APP_CONT_ADMIN.SET_REPLAY_RULES('Service_name',TRUE,dbms_app_cont.side
_effects);
PL/SQL procedure successfully completed.

SQL> execute DBMS_APP_CONT_ADMIN.CHECK_REPLAY_RULES('Service_name');
PL/SQL procedure successfully completed.
ResultSet #1
TARGETS                     REPLAY
-----------------------     --------------------------------
PL/SQL Side Effects         enabled
Autonomous Transactions     disabled
Database Links              disabled
```

- Use the following interface to allow side effects with TAC:

```
begin
insert into file_tab values (1,'file1.txt');
DBMS_APP_CONT.APPLY_REPLAY_RULE(REPLAYABLE => TRUE, TARGETS =>
DBMS_APP_CONT.SIDE_EFFECTS);
DBMS_FILE_TRANSFER.COPY_FILE(source_directory_object=>'SOURCEDIR',
source_file_name=>'file1.txt', destination_directory_object=>'DGROUP',
destination_file_name=>'file1.txt');
commit;
end
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

- Use the following interface to disallow side effects with AC:

```
begin
insert into file_tab values (1,'file1.txt');
DBMS_APP_CONT.APPLY_REPLAY_RULE(REPLAYABLE => FALSE, TARGETS =>
DBMS_APP_CONT.SIDE_EFFECTS);
DBMS_FILE_TRANSFER.COPY_FILE(source_directory_object=>'SOURCEDIR',
source_file_name=>'file1.txt', destination_directory_object=>'DGROUP',
destination_file_name=>'file1.txt');
commit;
end
```

- Use the following statement to wrap a replayable interface function for TAC:

```
create or replace procedure mark_replayable as
begin
DBMS_APP_CONT.APPLY_REPLAY_RULE(REPLAYABLE => TRUE, TARGETS =>
DBMS_APP_CONT.SIDE_EFFECTS, SCOPE => DBMS_APP_CONT.SCOPE_PARENT);
end mark_replayable;

begin
insert into file_tab values (1,'file1.txt');
mark_replayable;
DBMS_FILE_TRANSFER.COPY_FILE(source_directory_object=>'SOURCEDIR',
source_file_name=>'file1.txt', destination_directory_object=>'DGROUP',
destination_file_name=>'file1.txt');
commit;
end
```

Replayable Interface Autonomous Transaction Use Cases

Replay is also disabled for autonomous transaction and DB link by default for TAC. Use `DBMS_APP_CONT_ADMIN.SET_REPLAY_RULES` or `DBMS_APP_CONT.APPLY_REPLAY_RULE` to allow or disallow autonomous transaction and DB link.

- Use the following interface to allow autonomous transaction with TAC for entire service:

```
SQL> execute
DBMS_APP_CONT_ADMIN.SET_REPLAY_RULES('Service_name',TRUE,dbms_app_cont.auto
nomous_transactions);
PL/SQL procedure successfully completed.

SQL> execute DBMS_APP_CONT_ADMIN.CHECK_REPLAY_RULES('Service_name');
PL/SQL procedure successfully completed.
ResultSet #1
TARGETS                    REPLAY
-----------------------    --------------------------------
PL/SQL Side Effects        disabled
Autonomous Transactions    enabled
Database Links             disabled
```

- Use the following interface to allow autonomous transaction with TAC:

```
create or replace procedure log_update(description VARCHAR2) as
    PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

```
    INSERT INTO update_history VALUES (description, sysdate);
    COMMIT;
END log_update;

begin
DBMS_APP_CONT.APPLY_REPLAY_RULE(REPLAYABLE => TRUE, TARGETS =>
DBMS_APP_CONT.AUTONOMOUS_TRANSACTIONS);
insert into emp_tab('Andy', 100);
log_update('ID updated for user Andy');
commit;
end
```

- Use the following interface to disallow autonomous transaction with AC:

```
create or replace procedure log_update(description VARCHAR2) as
    PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
    INSERT INTO update_history VALUES (description, sysdate);
    COMMIT;
END log_update;

begin
DBMS_APP_CONT.APPLY_REPLAY_RULE(REPLAYABLE => FALSE, TARGETS =>
DBMS_APP_CONT.AUTONOMOUS_TRANSACTIONS);
insert into emp_tab('Andy', 100);
log_update('ID updated for user Andy');
commit;
end
```

- Use the following interface to allow DB Link with TAC for entire service:

```
SQL> execute
DBMS_APP_CONT_ADMIN.SET_REPLAY_RULES('Service_name',TRUE,dbms_app_cont.data
base_links);
PL/SQL procedure successfully completed.

SQL> execute DBMS_APP_CONT_ADMIN.CHECK_REPLAY_RULES('Service_name');
PL/SQL procedure successfully completed.
ResultSet #1
TARGETS                    REPLAY
-----------------------    --------------------------------
PL/SQL Side Effects        disabled
Autonomous Transactions    disabled
Database Links             enabled
```

- Use the following interface to allow DB Link with TAC for the current code block:

```
create database link dblink connect to scott identified by tiger using
'dblink_service';
begin
DBMS_APP_CONT.APPLY_REPLAY_RULE(REPLAYABLE => TRUE, TARGETS =>
DBMS_APP_CONT.DATABASE_LINKS);
select emp_id,emp_name from emp@dblink_servie;
commit;
end
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

- Use the following interface to disallow DB Link with AC for the current code block:

```
create database link dblink connect to scott identified by tiger using
'dblink_service';
begin
DBMS_APP_CONT.APPLY_REPLAY_RULE(REPLAYABLE => FALSE, TARGETS =>
DBMS_APP_CONT.DATABASE_LINKS);
select emp_id,emp_name from emp@dblink_service;
commit;
end
```

Related Topics

- Understanding Enabling and Disabling Replay in Application Continuity
  Replay occurs following a recoverable error, but you can disable replay.

Administering Application Continuity Operation and Usage

Learn how to manage the use of Application Continuity, and how you can use it in applications.

- Using Application Continuity for Planned Maintenance
  For planned maintenance, Oracle recommends that you drain requests from Oracle connection pools in combination with Application Continuity for those requests that do not complete.
- Administering Restoring Original Function Results
  To manage restoring original function results, you need to grant certain privileges.
- Protection-Level Statistics
  Use the statistics for request boundaries and protection level to monitor the level of coverage.
- Session State Consistency
  Session state consistency describes how non-transactional state is changed during a request.
- Application Continuity Statistics
  Once Application Continuity is configured, you can use the statistics to verify Application Continuity usage and to check how well Application Continuity is protecting user workloads.
- Application Continuity Protection Check
  The Application Continuity Protection Check (ACCHK) feature generates Application Continuity coverage reports and views that describe the protection of your application by Application Continuity.
- Delaying the Reconnection in Application Continuity
  Learn about how you can set parameters to manage reconnects with Application Continuity, or Transparent Application Continuity, and see examples on single-instance and Oracle Real Application Clusters (Oracle RAC) databases.
- Running Without Application Continuity
  Sometimes Application Continuity is not in effect because a disabling call has been issued.
- Disabling Replay in Application Continuity
  Learn about how you can disable replay with applications, and about specific rules and guidelines for disabling replay.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Terminating or Disconnecting a Session Without Replay
  Learn how to disable replay when a DBA terminates or disconnects a session by using the `ALTER SYSTEM KILL SESSION` or `ALTER SYSTEM DISCONNECT SESSION` statement.

Using Application Continuity for Planned Maintenance

For planned maintenance, Oracle recommends that you drain requests from Oracle connection pools in combination with Application Continuity for those requests that do not complete.

This procedure has the least impact when there is minimal recovery to complete. Instances do need to be stopped to switch over to the patched software.

1. FAN-aware pool, such as OCI, UCP, WebLogic Server, or ODP.NET Managed and Unmanaged Drivers.

   The FAN planned event drains at request boundaries.

   > **Note**
   > - ODP.NET Managed Driver does not support Application Continuity.
   > - Oracle Data Provider for .NET (ODP.NET), Unmanaged Driver is deprecated in Oracle AI Database.

2. Use the `srvctl relocate service` command to relocate the service from the instance without disrupting the sessions or, for a uniform service, use the `srvctl stop service` command on the instance (do not use the `-force` parameter).

   The FAN planned event clears the idle sessions immediately and marks the active sessions to be released at check-in (end of request). This drains the sessions from the instance without disrupting work.

3. If not all sessions have checked in and the time to stop the instance has been reached, then stop the instance (terminate).

   For Application Continuity-enabled pools (UCP, WebLogic, Tuxedo, ODP.NET, and OCI), and any Java pool that adds `beginRequest`/`endRequest` , Application Continuity attempts to recover those remaining sessions.

4. Restart the instance and service.

   Runtime load balancing, when enabled, balances the sessions back to the restored instance at the next request boundaries.

Related Topics

- *Oracle Autonomous Health Framework User's Guide*
- Transparent Application Continuity
  A version of Application Continuity for simple applications that uses discovery to detect request boundaries.
- *Oracle Orachk and Exachk User's Guide*

Administering Restoring Original Function Results

To manage restoring original function results, you need to grant certain privileges.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Restoring Original Oracle Function Values and Application Continuity
  When a request is replayed, the default and desired treatment of restoring original Oracle function values can vary.
- Checking Your Keep Permissions
  You must ensure that you have required KEEP permissions to keep function results at replay.
- Granting and Revoking Keep Permissions for Restoring Original Oracle Function Values
  To keep function results at replay, you must grant KEEP privileges to the user invoking the function.
- Granting Permission to Keep Restoring Original Oracle Function Values for Oracle Sequences
  To keep the original values of sequence.nextval for replaying so that keys match, you must grant permissions on the sequence.
- Rules for Grants on Restoring Original Oracle Function Values
  These considerations apply to granting and revoking privileges on restoring original Oracle function values.

Restoring Original Oracle Function Values and Application Continuity

When a request is replayed, the default and desired treatment of restoring original Oracle function values can vary.

By default, for SQL the original value received is restored for sequences. This is the value that the application owns. For PL/SQL, DATE and TIME, and SYSGUID restoring original Oracle function values, the KEEP clause must be granted as a part of the schema.

Support for restoring original Oracle function values is currently provided for SYSDATE, SYSTIMESTAMP, LOCALTIMESTAMP, CURRENT_TIMESTAMP, SYS_GUID, and *sequence*.NEXTVAL. If the original values are not kept and if different values for these objects are returned to the client, then replay is rejected because the client observes different results. If the application can use original values, then configure restoring original Oracle function values using the KEEP clause for owned sequences and GRANT KEEP for other users. (Most applications need sequence values to be kept at replay, for bind variable consistency.)

> **Note**
>
> Keeping SYS_GUID values is supported only for serial processing plans. When parallel query is used, Application Continuity is not able to restore original values for SYS_GUID.

The following table shows examples of the treatment of restoring original Oracle function values by products during replay. (Actual implementation depends on specific products and releases.)

Table 6-1 Example Treatment of Restoring Original Oracle Function Values by Products During Replay

Function	Product 1	Product 2	Product 3
SYSDATE, SYSTIMESTAMP	Original	Original	Current
Sequence NEXTVAL and CURRVAL	Original	Original	(Not applicable)
SYS_GUID	Original	(Not applicable)	(Not applicable)

Copyright © 1999, 2025, Oracle and/or its affiliates.

To allow Application Continuity to keep and use original function results at replay:

- The database user running the application might have the `KEEP DATE TIME` and `KEEP SYSGUID` privileges granted, and the `KEEP SEQUENCE` object privilege on each sequence whose value is to be kept. For example:

```
GRANT KEEP DATE TIME TO user2;
GRANT KEEP SYSGUID TO user2;
GRANT KEEP SEQUENCE ON sales.seq1 TO user2;
```

> **Notes**
>
> – Starting with Oracle Database 19c, grants are not required for restoring original Oracle function values for SQL for sequences.
>
> – `GRANT ALL ON` *`object`* *does not* include (that is, does not grant the access provided by) the `KEEP DATE TIME` and `KEEP SYSGUID` privileges, and the `KEEP SEQUENCE` object privilege.
>
> – Grant privileges related to restoring original Oracle function values support only to application users, and to each application user grant only the necessary privileges.
>
> – Do ***not*** grant DBA privileges to database users running applications for which you want replay to be enabled.

- Sequences in the application can use the `KEEP` attribute, which keeps the original values of *`sequence`*`.NEXTVAL` for the sequence owner, so that the keys match during replay. Most applications need sequence values to be kept at replay. The following example sets the `KEEP` attribute for a sequence (in this case, one owned by the user running the statement; for others, use `GRANT KEEP SEQUENCE`):

```
SQL> CREATE SEQUENCE my_seq KEEP;
SQL> -- Or, if the sequence already exists but without KEEP:
SQL> ALTER SEQUENCE my_seq KEEP;
```

> **Note**
>
> Specifying `ALTER SEQUENCE ... KEEP/NOKEEP` applies to the owner of the sequence. It does not affect other users (not the owner) that have the `KEEP SEQUENCE` object privileges. If you want `KEEP` for all users, then be sure to grant the `KEEP SEQUENCE` object privilege to these users (or to revoke it from each user if the privilege has been granted).

- To keep function results (for named functions) at replay, the DBA must grant `KEEP` privileges to the user invoking the function. This security restriction ensures that it is valid for replay to save and restore function results for code that is not owned by that user.

For identity sequences, restoring original Oracle function values is supported for owned sequences. Restoring original Oracle function values at SQL level is automatic for identity

Copyright © 1999, 2025, Oracle and/or its affiliates.

sequences. To restore original Oracle function values in PL/SQL for identity sequences use the KEEP clause. The definition of procedure and table are as follows:

```
create table tab_identity_mine( id NUMBER GENERATED ALWAYS AS IDENTITY keep,
content varchar2(50));
```

Use the following statement to create a procedure:

```
insert_identity(cnt in varchar2,newid out number) as
begin
insert into tab_identity_mine(content) values(cnt) returning id into newid;
end insert_identity;
```

Related Topics

- Rules for Grants on Restoring Original Oracle Function Values
  These considerations apply to granting and revoking privileges on restoring original Oracle function values.
- ALTER SEQUENCE
- GRANT

Checking Your Keep Permissions

You must ensure that you have required KEEP permissions to keep function results at replay.

- To check permission to keep SYSDATE and SYSGUID:

  ```
  SELECT * FROM USER_SYS_PRIVS WHERE PRIVILEGE LIKE '%KEEP%';
  ```

  This query returns output similar to the following:

  ```
  USERNAME   PRIVILEGE        ADM   COM   INH
  --------   --------------   ---   ---   ---
  SOE1       KEEP SYSGUID     NO    NO    NO
  SOE1       KEEP DATE TIME   NO    NO    NO
  ```

- To check permission to keep SEQUENCES:

  ```
  SELECT SEQUENCE_NAME, KEEP_VALUE FROM USER_SEQUENCES;
  ```

  This query returns output similar to the following:

  ```
  SEQUENCE_NAME   KEEP_VALUE
  -------------   ----------
  SEQ_PERSON      N
  SEQ_PLSQL       N
  SEQ_PRODUCTS    Y
  SEQ_PRODUCT_ID  Y
  ```

  The KEEP_VALUE in the above example is Y or N.

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> For all sequences grants, run the `SELECT SEQUENCE_NAME, KEEP_VALUE FROM ALL_SEQUENCES;` statement.

Granting and Revoking Keep Permissions for Restoring Original Oracle Function Values

To keep function results at replay, you must grant `KEEP` privileges to the user invoking the function.

- To grant permission to keep restoring original Oracle function values for `SYSDATE` and `SYSTIMESTAMP`, or `SYSGUID`:

  ```
  GRANT [KEEP DATE TIME | KEEP SYSGUID]...[to USER]
  ```

  For example, for possible Oracle E-Business Suite usage with original dates:

  ```
  GRANT KEEP DATE TIME, KEEP SYSGUID to [custom user];
  GRANT KEEP DATE TIME, KEEP SYSGUID to [apps user];
  ```

- To revoke permission to keep restoring original Oracle function values for `SYSDATE` and `SYSTIMESTAMP`, or `SYSGUID`:

  ```
  REVOKE [KEEP DATE TIME | KEEP SYSGUID]...[from USER]
  ```

Granting Permission to Keep Restoring Original Oracle Function Values for Oracle Sequences

To keep the original values of `sequence.nextval` for replaying so that keys match, you must grant permissions on the sequence.

- To grant permission as the owner of the sequence:

  ```
  CREATE SEQUENCE [sequence object] [KEEP|NOKEEP];
  ALTER SEQUENCE [sequence object] [KEEP|NOKEEP];
  ```

- To grant and revoke permission for others using the sequence:

  ```
  GRANT KEEP SEQUENCE on sequence.object to [myUser|role];
  REVOKE KEEP SEQUENCE on sequence.object from [myUser|role];
  ```

  For example, for possible Oracle E-Business Suite usage with original sequence values:

  ```
  GRANT KEEP SEQUENCE on sequence.object to apps-user;
  REVOKE KEEP SEQUENCE on sequence.object from my-user ;
  ```


Copyright © 1999, 2025, Oracle and/or its affiliates.

For example, for identity sequences, use the KEEP clause on the table create or alter statements:

```
CREATE TABLE tab_identity_mine(id NUMBER GENERATED ALWAYS AS IDENTITY
keep,
content varchar2(50));
```

Rules for Grants on Restoring Original Oracle Function Values

These considerations apply to granting and revoking privileges on restoring original Oracle function values.

- If you grant all on an object for a user, then restoring original Oracle function values are excluded. Restoring original Oracle function values require explicit grants. Oracle does not support granting restoring original Oracle function values to the users supplied or created by Oracle AI Database, such as SYS, AUDSYS, GSMUSER, and SYSTEM.
- The DBA role includes restoring original Oracle function value permission.
- If a user has restoring original Oracle function values granted, then the objects inherit restoring original Oracle function value access when the functions are called (in SYS_GUID, SYSDATE and SYSTIMESTAMP).
- If keeping restoring original Oracle function values on a sequence object is revoked, then SQL or PL/SQL commands using that object does not allow restoring original Oracle function value collection or application for that sequence.
- If grants are revoked between run time and failover, then the restoring original Oracle function values that were collected are not applied.
- If grants are granted between run time and failover, then restoring original Oracle function values are not collected and so none are applied.

Protection-Level Statistics

Use the statistics for request boundaries and protection level to monitor the level of coverage.

Application Continuity collects statistics from the system, the session, and the service, enabling you to monitor your protection levels. The statistics are available in V$SESSTAT, V$SYSSTAT, and, when service statistics are enabled, in V$SERVICE_STATS. For example, if you query V$SESSTAT and join with V$STATNAME, you can view output like the following:

```
NAME                                                             VALUE
---------------------------------------------------------------- ----------
cumulative begin requests                                           731
cumulative end requests                                             739
cumulative user calls in requests                                  7285
cumulative user calls protected by Application Continuity          7228
cumulative time in requests                                  2665167909
```

These statistics are saved in the Automatic Workload Repository (AWR) and are available in AWR reports. Statistics include:

- Requests completed per second
- User calls in a request
- Protected user calls

Copyright © 1999, 2025, Oracle and/or its affiliates.

The AWR report output is similar to the following:

```
Statistic                                     Total    per Second    per Trans
---------------------------------------- -------- ------------ ---------
cumulative requests                        177,406          49.2       5.0
cumulative user calls in request           493,329         136.8      13.8
cumulative user calls protected            493,329         136.8      13.8
```

To enable protection-level statistics, use (`_request_boundaries = 3`).

Session State Consistency

Session state consistency describes how non-transactional state is changed during a request.

- About Session State Consistency
  To ensure session state consistency, Oracle recommends that you set the service parameter `session_state` to `AUTO`, which is available with Transparent Application Continuity.
- Auto Session State Consistency
  When you set the service parameter `session_state` to `AUTO`, Transparent Application Continuity tracks and records session and transactional states so the database session can be recovered following recoverable outages.
- Dynamic Session State Consistency
  Application Continuity uses **Dynamic** Session State Consistency. The session state is tracked as it is for TAC when using Session State Consistency = `AUTO`.

About Session State Consistency

To ensure session state consistency, Oracle recommends that you set the service parameter `session_state` to `AUTO`, which is available with Transparent Application Continuity.

Transparent Application Continuity tracks and manages session states. If you choose to use Transparent Application Continuity, then you do not have to do anything else to ensure session state consistency.

You can set `session_state` to `DYNAMIC` for Application Continuity. You should only set `session_state` to `DYNAMIC` if you fully understand the application, and the application is not expected to change from the value set.

Examples of session state are NLS settings (globalization support), optimizer preferences, event settings, PL/SQL global variables, temporary tables, advanced queues, Large Objects (LOBs), and result cache. If non-transactional values change in committed transactions, then use the default value, `DYNAMIC`.

Using `DYNAMIC` mode, after a `COMMIT` has run, if the state was changed in that transaction, then it is not possible to replay the transaction to reestablish that state if the session is lost. Applications can be categorized depending on whether the session state after the initial setup is dynamic, and hence whether it is correct to continue past a `COMMIT` operation.

`AUTO` mode is appropriate for almost all applications. If your customers or users can modify your application, then you must use `AUTO` or `DYNAMIC` mode. `AUTO` mode is a newer version of `DYNAMIC` mode with the additional feature that it re-enables automatically when possible.

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> Set `session_state` to `AUTO` for long-running, stateless applications. Unless you require Application Continuity, Oracle recommends setting `session_state` to `AUTO`.

Auto Session State Consistency

When you set the service parameter `session_state` to `AUTO`, Transparent Application Continuity tracks and records session and transactional states so the database session can be recovered following recoverable outages.

Setting `session_state` to `AUTO` is the only value permitted for Transparent Application Continuity. When set to `AUTO`, a state-tracking infrastructure categorizes session state usage as the application issues user calls. Tracked session states are monitored and verified.

> **Note**
>
> If you set `session_state` to `AUTO`, then you must also set `failovertype` to `AUTO`.

Replay (that is, Transparent Application Continuity) is enabled at an explicit begin request (from an Oracle pool, which is the usual source, or added by OCI or JDBC thin APIs), and is disabled on a `COMMIT`, at the end of a request (most often return to the pool), or a restricted call. Transparent Application Continuity automatically re-enables replay when the session state is describable after a disable within a request. Following is the step logic for three application scenarios:

- No transaction
- A transaction with `COMMIT` as the last statement
- A transaction with an embedded `COMMIT` statement

> **Note**
>
> Enable `RESET_STATE` if the session state set should not leak to the next request. This also ensures that TAC is enabled in the next request when PL/SQL is used.

For the request with **no transaction**, the logical steps are as follows:

1. Check out from a connection pool.
2. Begin request and enable replay.
3. Issue one or more `SELECT` statements and perhaps other PL/SQL statements.
4. Other actions.
5. Check in.
6. End request.

For the request with **a transaction with COMMIT as the last statement**, the logical steps are as follows:

1. Check out from a connection pool.
2. Begin request and enable replay.

Copyright © 1999, 2025, Oracle and/or its affiliates.

3. Issue one or more `SELECT` statements and perhaps other PL/SQL statements.
4. The transaction begins.
5. Other actions.
6. Commit (which disables replay).
7. Check in.
8. End request.

For the request with **a transaction with an embedded COMMIT statement**, the logical steps are as follows:

1. Check out from a connection pool.
2. Begin request and enable replay.
3. Issue one or more `SELECT` statements and perhaps other PL/SQL statements.
4. The transaction begins.
5. Other actions.
6. Commit (inside PL/SQL).
7. Next SQL statement, which re-enables Transparent Application Continuity when possible.
8. Check in.
9. End request.

Dynamic Session State Consistency

Application Continuity uses **Dynamic** Session State Consistency. The session state is tracked as it is for TAC when using Session State Consistency = `AUTO`.

Replay (that is, Application Continuity) is enabled at the `beginRequest` call, and is disabled on a `COMMIT` , an `endRequest` call, or a restricted call. Following is the step logic for three application scenarios:

- No transaction
- A transaction with `COMMIT` as the last statement
- A transaction with an embedded `COMMIT` statement

For the request with **no transaction**, the logical steps are as follows:

1. Check out from a connection pool.
2. Begin request and enable replay.
3. Issue one or more `SELECT` statements and perhaps other PL/SQL statements.
4. Other actions.
5. Check in.
6. End request and disable replay.

For the request with **a transaction with COMMIT as the last statement**, the logical steps are as follows:

1. Check out from a connection pool.
2. Begin request and enable replay.
3. Issue one or more `SELECT` statements and perhaps other PL/SQL statements.

Copyright © 1999, 2025, Oracle and/or its affiliates.

4. The transaction begins.
5. Other actions.
6. Commit (which disables replay).
7. Check in.
8. End request.

For the request with **a transaction with an embedded COMMIT statement**, the logical steps are as follows:

1. Check out from a connection pool.
2. Begin request and enable replay.
3. Issue one or more `SELECT` statements and perhaps other PL/SQL statements.
4. The transaction begins.
5. Other actions.
6. Commit (which disables replay).
7. Other actions, during which Application Continuity is not covering the application.
8. Check in.
9. End request.

Application Continuity Statistics

Once Application Continuity is configured, you can use the statistics to verify Application Continuity usage and to check how well Application Continuity is protecting user workloads.

You can read the following statistics from the `V$SESSTATS` and `V$SYSSTAT` views:

- `cumulative begin requests`
- `cumulative end requests`
- `cumulative user calls in requests`
- `cumulative user calls protected by Application Continuity`
- `cumulative time in requests`

You can read the following statistics only from the `V$SESSTATS` view:

- `cumulative DB time in requests`
- `cumulative DB time protected in requests`
- `successful replays by Application Continuity`
- `rejected replays by Application Continuity`

The following example shows you how to read Application Continuity statistics values for each active session.

```
SQL> SELECT sn.name, s.value FROM V$SESSTAT s, V$STATNAME sn WHERE
sn.statistic# = s.statistic# AND sn.name in(
    'cumulative begin requests','cumulative end requests','cumulative user
calls in requests','cumulative user calls protected by Application
    Continuity','cumulative time in requests');
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

The following example shows you how to read Application Continuity statistics from the V$SYSSTAT view.

```
SQL> SELECT name, value FROM V$SYSSTAT WHERE name in ('cumulative begin
requests','cumulative end requests',
    'cumulative user calls in requests','cumulative user calls protected by
Application Continuity','cumulative time in requests',
    'cumulative DB time in requests','cumulative DB time protected in
requests','successful replays by Application Continuity',
    'rejected replays by Application Continuity');
```

Application Continuity Protection Check

The Application Continuity Protection Check (ACCHK) feature generates Application Continuity coverage reports and views that describe the protection of your application by Application Continuity.

- About Application Continuity Protection Check
  The Application Continuity Protection Check (ACCHK) utility provides protection guidance for applications that use Application Continuity.
- Enabling and Disabling Application Continuity Protection Check
  You can manually enable or disable the Application Continuity Protection Check (ACCHK) feature for applications that use Application Continuity.
- Generating an Application Continuity Protection Check Report
  Generate the Application Continuity Protection Check (ACCHK) report to get guidance for the level of protection, reason for incomplete protection, and methods to increase the protection level.
- Filtering Statistics and Events
  ACCHK can gather the statistics and events by filtering the data through a service, module, program, or machine name.
- Cleaning Up ACCHK Collected Information
  You can purge the information collected by ACCHK after the ACCHK reports are generated and data gathered from ACCHK views is analyzed.

About Application Continuity Protection Check

The Application Continuity Protection Check (ACCHK) utility provides protection guidance for applications that use Application Continuity.

ACCHK provides guidance on the level of protection for each application that uses Application Continuity and helps guide you to increase protection, if required. ACCHK uses Application Continuity traces to collect coverage for a workload and provides detailed information as per your request. You must enable Application Continuity tracing to collect coverage before you run a database workload. ACCHK also provides diagnostics for an unsuccessful failover.

Database views and PL/SQL-based reports show you the level of protection for your applications for failover. If an application is not fully protected, then ACCHK identifies that application, finds out the reason why the application is not fully protected, and guides you how to increase the protection.

For the protected applications, ACCHK also reports which operations of an application are protected, and which operations of an application are not protected. If any operation or configuration of an application is not protected by the Application Continuity, then you can make configuration changes to increase the protection coverage. ACCHK generates a report

Copyright © 1999, 2025, Oracle and/or its affiliates.

with coverage statement and percentage value for the workload. The ACCHK report also shows how many operations were performed, how many operations were fully protected, and how many operations were not fully protected.

Related Topics

- DBA_ACCHK_EVENTS
- DBA_ACCHK_EVENTS_SUMMARY
- DBA_ACCHK_STATISTICS
- DBA_ACCHK_STATISTICS_SUMMARY

Enabling and Disabling Application Continuity Protection Check

You can manually enable or disable the Application Continuity Protection Check (ACCHK) feature for applications that use Application Continuity.

Application Continuity Protection Check is not enabled by default. Follow this procedure to enable or disable ACCHK and generate reports to check protection level for the applications.

1. Grant read access to the users, who will run the Application Continuity Protection Check report and views, using the `ACCHK_READ` role:

   ```
   GRANT ACCHK_READ TO USER;
   ```

2. Enable Application Continuity tracing for your applications using the `dbms_app_cont_admin.acchk_set(true)` procedure:

   ```
   SQL> execute dbms_app_cont_admin.acchk_set(true);
   ```

   By default, ACCHK is disabled automatically after 600 seconds. You can specify a lower number to reduce the auto disable time. For example, use the `dbms_app_cont_admin.acchk_set(true,300)` procedure to disable ACCHK after 300 seconds.

   The `dbms_app_cont_admin.acchk_set(true)` procedure enables Application Continuity tracing at the database level to which you are connected. If you are connected at the CDB level, then tracing is enabled for the CDB, and if you are connected at the PDB level, then tracing is enabled for the PDB.

   > **Note**
   >
   > Set the `COMPATIBLE` parameter to 19c or greater. If the `COMPATIBLE` parameter was previously set to a lower value, then the `acchk_set` procedure creates the ACCHK views and roles when you run the procedure for the first time after updating the `COMPATIBLE` parameter.

3. Use the `dbms_app_cont_admin.acchk_set(false)` procedure to disable Application Continuity tracing for new sessions in your applications:

   ```
   SQL> execute dbms_app_cont_admin.acchk_set(false);
   ```


Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> - The tracing will be disabled for the current sessions after the time expires.
> - The tracing is enabled by default for the entire Oracle Real Application (Oracle RAC) Cluster.

Related Topics

- ACCHK_SET Procedure
- Running Application Continuity Protection Check
  Generate the Application Continuity Protection Check (ACCHK) report to get guidance for the level of protection, reason for incomplete protection, and methods to increase the protection level.

Generating an Application Continuity Protection Check Report

Generate the Application Continuity Protection Check (ACCHK) report to get guidance for the level of protection, reason for incomplete protection, and methods to increase the protection level.

The ACCHK utility is a post-processing tool that uses ACCHK tables in `AUX` to report Application Continuity coverage. Enable the Application Continuity tracing and Application Continuity Protection Check before running a workload and generating the report.
When in read-only mode, traces can record event and statistics. The parameter source in `ACCHK_REPORT` is used to decide which input will use the procedure for displaying the report. You can choose either from tables or traces. When the database is in read-only mode, set this parameter to `DBMS_APP_CONT_REPORT.FROM_TRACES`.

1. Run a set of database options after enabling ACCHK and tracing for your applications.

   ACCHK generates reports only for the Application Continuity sessions.

2. Generate the Application Continuity Protection Check report using the `dbms_app_cont_report.acchk_report` procedure:

   ```
   SQL> SET LINESIZE 350;
   SQL> SET SERVEROUTPUT ON FORMAT WRAPPED;
   SQL> execute dbms_app_cont_report.acchk_report;
   ```

   You can specify the type of the report from `FULL`, `WARNING`, or `SUMMARY`. For example:

   ```
   SQL> SET LINESIZE 350;
   SQL> SET SERVEROUTPUT ON FORMAT WRAPPED;
   SQL> execute dbms_app_cont_report.acchk_report(dbms_app_cont_report.FULL);
   SQL> execute
   dbms_app_cont_report.acchk_report(dbms_app_cont_report.WARNING);
   SQL> execute
   dbms_app_cont_report.acchk_report(dbms_app_cont_report.SUMMARY);
   ```

   The default report type is `SUMMARY`.

Copyright © 1999, 2025, Oracle and/or its affiliates.

3. Analyze the report and increase the protection level for the applications that are not fully protected. For example, a summary report looks like the following:

```
--------------------------------------
------------ ACCHK Report ------------
--------------------------------------
CON_ID SERVICE       FAILOVER PROTECTED_ PROTECTED_ REQUESTS AVG_CALLS/
PROTECTED_    AVG_TIME/  PROTECTED_TIME/ EVENT_  ERROR_ PROGRAM
MODULE            ACTION    SQL_ CALL      TOTAL
                              CALLS %    TIME %              REQUEST
CALLS/REQUEST REQUEST MS REQUEST MS      TYPE
CODE                                        ID
------ ------------ -------- ---------- ---------- -------- ----------
------------ ---------- -------------- ------- ------ ---------
---------------- _________ ---- --------- ------
3      srv_tacr_pdb1 AUTO     98.734     98.432     117      9.453
9.333         2279.751   2244.014        DISABLE 41409  JDBC Thin
AddCustNewOrder   Action-20      COMMIT    1


Client
3      srv_tacr_pdb1 AUTO     98.734     98.432     117      9.453
9.333         2279.751   2244.014        REPLAY_ 41412  JDBC Thin
InsertNewChecksum Action-1       SQL/PLSQL 1

                                       FAILED
Client                                      Execu
End of report.
```

The following examples show how to query detailed information from an ACCHK report using the ACCHK views.

Example 6-2 Using the DBA_ACCHK_EVENTS View

In this example, the last row indicates that the application that is using the `srv_tacr_pdb1` service has an event that caused Application Continuity to fail.

```
SQL> SELECT * FROM DBA_ACCHK_EVENTS ORDER BY TIMESTAMP;
INST_ID CON_ID TIMESTAMP        SESSION_ID SERIAL# SERVICE_NAME  PROGRAM
MODULE            ACTION    SQL_ID CALL_NAME EVENT_TYPE ERROR_CODE
------- ------ ---------------- ---------- ------- ------------ -------
----------------- --------- ------ --------- ---------- ----------
2       3    21-SEP-20          9598       1644    srv_tacr_pdb1 JDBC
AddCustNewOrder   Action-36        COMMIT    DISABLE    41409
             06.54.18.191 PM
Thin
             -07:00
Client
2       3    21-SEP-20          1703       61265   srv_tacr_pdb1 JDBC
InsertNewChecksum  Action-1        SQL/PLSQL REPLAY_    41412
             06.51.07.624 PM
Thin                                       Execution FAILED
             -07:00
Client
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Example 6-3 Using the DBA_ACCHK_EVENTS_SUMMARY View

In this example, the last row indicates that the application that is using the `srv_tacr_pdb1` service has an event that caused Application Continuity to fail.

```
SQL> SELECT * FROM DBA_ACCHK_EVENTS_SUMMARY ORDER BY SERVICE_NAME;
INST_ID CON_ID SERVICE_NAME  FAILOVER_TYPE FAILOVER_RESTORE RESET_STATE
PROGRAM MODULE            ACTION    SQL_ID CALL_NAME EVENT_TYPE ERROR_CODE
FREQUENCY
------- ------ ------------ ------------ --------------- -----------
------- ----------------- --------- ------ --------- ---------- ----------
----------
2       3      srv_tacr_pdb1 AUTO          AUTO             LEVEL1
JDBC    AddCustNewOrder   Action-20        COMMIT    DISABLE    41409      1

Thin                                       Execution

Client
2       3      srv_tacr_pdb1 AUTO          AUTO             LEVEL1
JDBC    InsertNewChecksum Action-1         SQL/PLSQL REPLAY_    41412      1

Thin                                       Execution FAILED

Client
```

Example 6-4 Using the DBA_ACCHK_STATISTICS View

In this example, the first row indicates that the application that is using the `srv_tacr_pdb1` service has 11 implicit requests from JDBC and 31 calls in the application. 30 calls in these requests are protected.

```
SQL> SELECT * FROM DBA_ACCHK_STATISTICS ORDER BY TIMESTAMP;
INST_ID CON_ID TIMESTAMP        SESSION_ID SERIAL# STAT_TYPE  SERVICE_NAME
FAILOVER_ FAILOVER_ RESET_ PROGRAM  BEGIN_   END_     USER_CALLS_
PROTECTED_CALLS_ TIME_IN_ TIME_PROTECTED_

TYPE      RESTORE   STATE           REQUESTS REQUESTS IN_REQUESTS
IN_REQUESTS      REQUESTS IN_REQUEST
------- ------ --------------- ---------- ------- ---------- -------------
--------- --------- ------ -------  -------- -------- -----------
--------------- -------- ---------------
2       3      21-SEP-20        5653       54237   SESSION_   srv_tacr_pdb1
AUTO      AUTO      LEVEL1 JDBC     11       11       31
30              13316750 12415247
              06.54.25.321 PM
STATISTICS
Thin

-07:00
          Client
2       3      21-SEP-20        11291      26560   SESSION_   srv_tacr_pdb1
AUTO      AUTO      LEVEL1 JDBC     3        3        50
49              13094072 13068259
              06.54.24.915 PM
STATISTICS                                        Thin
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

```
-07:00
          Client
```

Example 6-5 Using the DBA_ACCHK_STATISTICS_SUMMARY View

In this example, the application that is using the `srv_tacr_pdb1` service has 144 implicit requests, 99.5688328 percent calls in these requests are protected by Application Continuity or Transparent Application Continuity.

```
SQL> SELECT * FROM DBA_ACCHK_STATISTICS_SUMMARY ORDER BY SERVICE_NAME;
INST_ID CON_ID SERVICE_NAME  FAILOVER_ FAILOVER_ RESET_ TOTAL_
PROTECTED_CALLS_ PROTECTED_TIME_ AVG_USER_CALLS_ AVG_PROTECTED_
AVG_TIME_   AVG_TIME_
                              TYPE      RESTORE   STATE  REQUESTS
PERCENT          PERCENT          IN_REQUESTS     CALLS_IN_REQUESTS
IN_REQUESTS PROTECTED_IN_REQUESTS
------- ------ ------------ --------- --------- ------ --------
--------------- -------------- -------------- -----------------
----------- --------------------
2       3      srv_tacr_pdb1 AUTO      AUTO      LEVEL1 144
99.5688328       99.0130288      22.5486111      22.4513889
3078654.35  3048268.92
```

Related Topics

- DBA_ACCHK_EVENTS
- DBA_ACCHK_STATISTICS
- DBA_ACCHK_STATISTICS_SUMMARY
- ACCHK_REPORT Procedure

Filtering Statistics and Events

ACCHK can gather the statistics and events by filtering the data through a service, module, program, or machine name.

When you provide single service filter or multiple service filters, ACCHK records only the statistics and events related to the specified services. This functionality also applies when you provide only a single or multiple program filters for only a single or multiple module filters. Provide a service, program, module, and machine filter for ACCHK to record only the statistics and events that match the service, program, module, and machine.

The `DBMS_APP_CONT_ADMIN` package contains the following procedures for filtering data:

- `ACCHK_SET_FILTER`
- `ACCHK_CLEAR_FILTER`
- `ACCHK_SHOW_FILTERS`

> **Note**
>
> After adding filters by using `ACCHK_SET_FILTER`, you must run `ACCHK_SET(TRUE)` to load the filters. If you add or clear a filter, but do not run `ACCHK_SET(TRUE)`, then that particular filter is not updated.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Filter Life Span

- Filters are accumulative.
- The `ACCHK_CLEAR_FILTER` procedure clears one or all filters depending on how you run the procedure.

Related Topics

- ACCHK_SET_FILTER Procedure
- ACCHK_CLEAR_FILTER Procedure
- ACCHK_SHOW_FILTERS Procedure

Cleaning Up ACCHK Collected Information

You can purge the information collected by ACCHK after the ACCHK reports are generated and data gathered from ACCHK views is analyzed.

Purging the ACCHK collected information is optional. You can initiate a new ACCHK process with new collected data after purging existing ACCHK information.

Use the following statement to purge all previously collected ACCHK information:

```
SQL> execute dbms_app_cont_admin.acchk_purge(purge_all => TRUE);
```

Related Topics

- DBMS_APP_CONT_ADMIN
- ACCHK_PURGE Procedure

Delaying the Reconnection in Application Continuity

Learn about how you can set parameters to manage reconnects with Application Continuity, or Transparent Application Continuity, and see examples on single-instance and Oracle Real Application Clusters (Oracle RAC) databases.

- Understanding How to Wait for the Service to be Ready for Application Continuity
  To manage planned and unplanned outages, learn about the parameters that you use to manage application continuity.
- Creating Services on Oracle RAC with Application Continuity
  You can create services on Oracle RAC that utilize Transparent Application Continuity or Application Continuity.
- Modifying Services on Single-instance Databases to use Application Continuity
  If you are using a single-instance database, then use the `DBMS_SERVICE` package to modify services.

Understanding How to Wait for the Service to be Ready for Application Continuity

To manage planned and unplanned outages, learn about the parameters that you use to manage application continuity.

By default, when Application Continuity initiates a failover, the driver attempts to recover the in-flight work at an instance where the service is available. For recovering the work, the driver must establish a good connection with the instance. This reconnection can take some time if

Copyright © 1999, 2025, Oracle and/or its affiliates.

the database or the instance must be restarted before the service is relocated and published. For this reason, the failover must be delayed until the service is available from another instance or database.

To manage connecting and reconnecting, you must use the `RETRY_COUNT` and `RETRY_DELAY` parameters in your TNS or URL. These TNS parameters can work well in conjunction with a planned outage, for example, an outage that may make a service unavailable for several seconds. While setting the `RETRY_COUNT` and `RETRY_DELAY` parameters, check the value of the `REPLAY_INITIATION_TIMEOUT` parameter first. A high value for the `RETRY_DELAY` parameter can cause replay to be canceled.

Parameter Name	Possible Value	Default Value
`RETRY_COUNT`	Positive integer zero or above	30
`RETRY_DELAY`	Time in sec, ms, or cs	3

Creating Services on Oracle RAC with Application Continuity

You can create services on Oracle RAC that utilize Transparent Application Continuity or Application Continuity.

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

You can create services that use Transparent Application Continuity, as follows:

```
$ srvctl add service -db mydb -service TACSERVICE -pdb mypdb –preferred inst1
-available inst2
  -failovertype AUTO -session_state AUTO -failover_restore AUTO -
commit_outcome TRUE -replay_init_time 600
  -retention 86400 -notification TRUE -drain_timeout 300 -stopoption
IMMEDIATE -role PRIMARY
```

You can create services that use Application Continuity, as follows:

```
$ srvctl add service -db mydb -service ACSERVICE -pdb mypdb -preferred inst1 -
available inst2
  -failovertype TRANSACTION -session_state DYNAMIC -failover_restore LEVEL1 -
commit_outcome TRUE  -replay_init_time 600
  -retention 86400 - notification TRUE -drain_timeout 300 -stopoption
IMMEDIATE -role PRIMARY
```

Modifying Services on Single-instance Databases to use Application Continuity

If you are using a single-instance database, then use the `DBMS_SERVICE` package to modify services.

Copyright © 1999, 2025, Oracle and/or its affiliates.

For Application Continuity:

```
DECLARE
params dbms_service.svc_parameter_array;
BEGIN
params('FAILOVER_TYPE'):='TRANSACTION';
params('REPLAY_INITIATION_TIMEOUT'):=1800;
params('RETENTION_TIMEOUT'):=86400;
params('FAILOVER_DELAY'):=10;
params('FAILOVER_RETRIES'):=30;
params('FAILOVER_RESTORE'):='LEVEL1';
params('commit_outcome'):='true';
params('aq_ha_notifications'):='true';
dbms_service.modify_service('[your service]',params);
END;
/
```

For Transparent Application Continuity:

```
DECLARE
params dbms_service.svc_parameter_array;
BEGIN
params('FAILOVER_TYPE'):='AUTO';
params('REPLAY_INITIATION_TIMEOUT'):=1800;
params('RETENTION_TIMEOUT'):=86400;
params('FAILOVER_DELAY'):=10;
params('FAILOVER_RETRIES'):=30;
params('FAILOVER_RESTORE'):='AUTO';
params('commit_outcome'):='true';
params('aq_ha_notifications'):='true';
dbms_service.modify_service('[your service]',params);
END;
/
```

Running Without Application Continuity

Sometimes Application Continuity is not in effect because a disabling call has been issued.

Application Continuity is not in effect when it has not been started or when it has been disabled. If it has been disabled, it remains so through to the `endRequest` call.

Application Continuity is not started when the service property `FAILOVER_TYPE` does *not* have the value set to `TRANSACTION` or `AUTO`. For planned maintenance, set the `FAILOVER_TYPE` value to `TRANSACTION` or `AUTO`, beforehand; the setting applies to new connections, and existing connections retain their original service value.

Application Continuity is disabled for the current request when any of the following occurs:

- The application runs a statement that is restricted for Application Continuity (for example, `ALTER SYSTEM`).
- Application Continuity is explicitly disabled using `disableReplay`.
- A `COMMIT` statement is issued when the service parameter `session_state_consistency` is set to `Dynamic` (the default, when not using Transparent Application Continuity).
- An `endRequest` statement is issued until the next `beginRequest` is issued.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- The session is terminated or disconnected and the `NOREPLAY` keyword is specified.

Related Topics

- Transparent Application Continuity
- Understanding Enabling and Disabling Replay in Application Continuity
- Terminating or Disconnecting a Session Without Replay
  Learn how to disable replay when a DBA terminates or disconnects a session by using the `ALTER SYSTEM KILL SESSION` or `ALTER SYSTEM DISCONNECT SESSION` statement.

Disabling Replay in Application Continuity

Learn about how you can disable replay with applications, and about specific rules and guidelines for disabling replay.

The rules described in this section are generic. They apply to all applications that replay work, including Application Continuity, and TAF (release 12.2 and after).

- Understanding Enabling and Disabling Replay in Application Continuity
  Replay occurs following a recoverable error, but you can disable replay.
- Application Calls Autonomous Transactions, External PL/SQL, or Java Actions that Should Not Be Repeated
  Autonomous transactions, external PL/SQL calls, and Java callouts can have side effects that are separate from the main transaction, and these side effects are replayed unless you specify otherwise.
- Application Synchronizes Independent Sessions
  If the application synchronizes independent sessions using volatile entities that are held until COMMIT, ROLLBACK, or session loss, then you must not configure an application for replay.
- Application Uses Time at the Middle Tier in the Processing Logic
  If the application uses the wall clock at the middle tier as part of the processing logic, then you must not configure an application for replay.
- Application Assumes that ROWIds Do Not Change
  If an application caches `ROWID` values, then access to these `ROWID` values can be invalidated due to database changes.
- Application Assumes that Location Values Do Not Change
  If you have applications that use physical identifiers, then review the guidelines and examples here to avoid issues.

Understanding Enabling and Disabling Replay in Application Continuity

Replay occurs following a recoverable error, but you can disable replay.

If an application has requests that you do not want the application to repeat, then the application either can take a connection to a service that does not have Application Continuity enabled, or the application can explicitly call an API to disable replay for those requests. If you use Transparent Application Continuity, then side effects are detected and disabled automatically. You do not need either to understand the application, or to disable requests with side effects.

When using Application Continuity, all calls are replayed. If an application uses `UTL_SMTP`, for example, and you do not want that application to repeat messages, then the application can use a connection to a different service, or use the `disableReplay` API on Java, or use the

Copyright © 1999, 2025, Oracle and/or its affiliates.

`OCIRequestDisableReplay` API for Oracle Call Interface (OCI). All other requests continue to be replayed.

For applications with external actions (for example, autonomous transactions or using `UTL_HTTP` to issue an SOA call), Application Continuity remains transparent if the application's correctness is preserved when these external actions are replayed after a failure.

Related Topics

- Transparent Application Continuity
- Potential Side Effects of Application Continuity
- Restrictions and Other Considerations for Application Continuity

Application Calls Autonomous Transactions, External PL/SQL, or Java Actions that Should Not Be Repeated

Autonomous transactions, external PL/SQL calls, and Java callouts can have side effects that are separate from the main transaction, and these side effects are replayed unless you specify otherwise.

Examples of side effects separate from the main transaction include the following:

- Writing to an external table, sending email, forking sessions out of PL/SQL(including calls to `UTL_HTTP`, `UTL_URL`, `UTL_FILE`, `UTL_FILE_TRANSFER`, `UTL_SMPT, UTL_TCP`, `UTL_MAIL, DBMS_PIPE`, or `DBMS_ALERT`)
- Writing to an external table, sending email, forking sessions out of Java (including running a shell script in the form `Process proc = rt.exec(command);`), transferring files, and accessing external URLs

Actions such as these leave persistent side effects. PL/SQL messaging and Java callouts can leave persistent results behind. For example, if a user walks away partway through some work without committing, and the session times out, or the user issues a Ctrl+C command, then the foreground or a component fails. As a result, the main transaction rolls back, while the side effects may have been applied.

Application developers decide whether to allow replay for external actions. Examples include using `UTL_HTTP` to issue an `SOA` call, or `UTL_SMTP` to send a message, or `UTL_URL` to access a website. If you do not want these kinds of external actions replayed, then use a connection without application continuity, or use one of the disable replay APIs.

Related Topics

- Potential Side Effects of Application Continuity
  When you use Application Continuity with the service attribute `FAILOVER_TYPE` set to `TRANSACTION`, statements that perform side effects are replayed.

Application Synchronizes Independent Sessions

If the application synchronizes independent sessions using volatile entities that are held until COMMIT, ROLLBACK, or session loss, then you must not configure an application for replay.

For example, if an application synchronizes multiple sessions connected to several data sources that are otherwise interdependent using resources such as a database lock, then this synchronization can be acceptable if the application is only serializing these sessions, and understands that any session can fail. However, if the application assumes that a lock or any other volatile resource held by one data source implies exclusive access to data on the same

Copyright © 1999, 2025, Oracle and/or its affiliates.

or a separate data source from other connections, then this assumption can be invalidated when replaying.

During replay, the client driver is not aware that the sessions are dependent on one session holding a lock or other volatile resource. To implement the synchronization lost by failures, you can also use pipes, buffered queues, or stored procedures taking a resource (such as a semaphore, device, or socket).

Application Uses Time at the Middle Tier in the Processing Logic

If the application uses the wall clock at the middle tier as part of the processing logic, then you must not configure an application for replay.

The client driver does not repeat the middle tier time logic, but instead uses the database calls that run as part of this logic. For example, an application using middle-tier time might assume that a statement processed at Time T1 is not re-processed at Time T2, unless the application explicitly does so.

Application Assumes that ROWIds Do Not Change

If an application caches `ROWID` values, then access to these `ROWID` values can be invalidated due to database changes.

Although a `ROWID` uniquely identifies a row in a table, a `ROWID` can change its value in the following situations:

- The underlying table is reorganized.
- An index is created on the table.
- The underlying table is partitioned.
- The underlying table is migrated.
- The underlying table is exported and imported using `EXP/IMP/DUL`.
- The underlying table is rebuilt using Oracle GoldenGate, or other replication technology.
- The database of the underlying table is flashed back or restored.

In general, Oracle does not recommend that an application stores `ROWID` values for later use, because the corresponding row either might not exist, might contain completely different data. Note that using `ROWID` values do not prevent using Application Continuity. Replays can be rejected.

Application Assumes that Location Values Do Not Change

If you have applications that use physical identifiers, then review the guidelines and examples here to avoid issues.

`SYSCONTEXT` options comprise two sets:

- A location-independent set, such as National Language Support (NLS) settings, `ISDBA`, `CLIENT_IDENTIFIER`, `MODULE`, and `ACTION`
- A location-dependent set, which uses physical locators

Typically, an application does not use the physical identifiers, except in testing environments. If physical locators are used in mainline code, then the replay finds the mismatch and rejects it. However, it is acceptable to use physical locators between requests (before `beginRequest`) or in callbacks. A common issue is for QA to modify test applications to select `V$INSTANCE`. As

Copyright © 1999, 2025, Oracle and/or its affiliates.

V$INSTANCE can change, only put this check in the callback, or select the instance locally at the client and not from the database.

Example of Physical Identifier Use

```
select
    sys_context('USERENV','DB_NAME')
    ,sys_context('USERENV','HOST')
    ,sys_context('USERENV','INSTANCE')
    ,sys_context('USERENV','IP_ADDRESS')
    ,sys_context('USERENV','ISDBA')
    ,sys_context('USERENV','SESSIONID')
    ,sys_context('USERENV','TERMINAL')
    ,sys_context('USERENV','SID')
from dual;
```

Terminating or Disconnecting a Session Without Replay

Learn how to disable replay when a DBA terminates or disconnects a session by using the ALTER SYSTEM KILL SESSION or ALTER SYSTEM DISCONNECT SESSION statement.

If Application Continuity is configured and if a DBA terminates or disconnects a session by using the ALTER SYSTEM KILL SESSION or ALTER SYSTEM DISCONNECT SESSION statement, then Application Continuity, by default attempts, to recover the session. However, if you *do not* want the session to be replayed, then use the NOREPLAY keyword. For example:

```
alter system kill session 'sid, serial#, @inst' noreplay;

alter system disconnect session 'sid, serial#, @inst' noreplay

$ srvctl stop service -db orcl -instance orcl2 –drain_timeout 60 -stopoption
immediate -force -noreplay

$ srvctl stop service -db orcl -node myode3 –noreplay -drain_timeout 60 -
stopoption immediate -force

$ srvctl stop instance -node mynode3 -drain_timeout 60 -stopoption immediate -
force –noreplay
```

To terminate all sessions running on the local instance (rather that only one session) and not have the sessions replayed, you can also use the DBMS_SERVICE.DISCONNECT_SESSION PL/SQL procedure, and specify NOREPLAY for the disconnect_option parameter.

Related Topics

- ALTER SYSTEM
- DBMS_SERVICE.DISCONNECT_SESSION

Fast Application Notification (FAN)

The Oracle RAC high availability framework monitors a database and its services, and sends Fast Application Notification (FAN) event notifications.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Oracle AI Database focuses on maintaining the highest possible service availability. In Oracle Real Application Clusters (Oracle RAC), services are designed to be continuously available with loads shared across one or more instances. The Oracle RAC high-availability framework maintains service availability by using Oracle Clusterware and resource profiles. Oracle Clusterware recovers and balances services according to business rules and the service attributes.

- Overview of Fast Application Notification (FAN)
  FAN provides immediate interrupt of clients following outages related to the database, nodes, and networks.
- The Importance of Using Fast Application Notification
  Using Fast Application Notification (FAN) events eliminates applications waiting on TCP timeouts, time wasted processing the last result at the client after a failure has occurred, and time wasted running work on slow, suspended, or terminated nodes.
- How FAN is Used with Oracle AI Database and Applications
  Fast Application Notification (FAN) is essential to prevent applications from stop responding on TCP/IP timeouts.
- Requirements for Using FAN
  Learn what you need to do to take advantage of FAN-aware capabilities in client drivers connecting to Oracle Real Application Clusters (Oracle RAC) databases.
- FAN Callouts
  Fast Application Notification (FAN) callouts are server-side scripts or executables that run whenever a FAN event is generated.
- Fast Application Notification High Availability Events
  Learn how the Fast Application Notification (FAN) event delivers information to a callout program.
- Subscription to High Availability Events
  To monitor and notify applications about services, Oracle Real Application Clusters (Oracle RAC) uses Oracle RAC Fast Application Notification (FAN).
- Using Fast Application Notification Callouts
  Fast Application Notification (FAN) callouts are server-side program files that Oracle RAC runs immediately when high availability events occur.

Overview of Fast Application Notification (FAN)

FAN provides immediate interrupt of clients following outages related to the database, nodes, and networks.

FAN is essential to break clients out of TCP/IP timeouts immediately following failures. FAN notifies clients immediately when resources become available and initiates draining of database sessions so clients experience no outages during planned maintenance. FAN also includes notifying configuration-level and service-level information that includes changes in service status.

The Oracle client drivers and Oracle Real Application Clusters (Oracle RAC) connection pools respond to FAN events, and take immediate action. `FAN UP` and `DOWN` events apply to services, databases, instances, networks, and nodes.

Related Topics

- Enabling Clients for Oracle RAC
  Learn how FAN is integrated with Oracle Clients, and how to enable FAN events for the several specific client development environments.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Server Draining Ahead of Planned Maintenance
  Before planned maintenance, drain or failover database sessions at the database instance so application work is not interrupted.

The Importance of Using Fast Application Notification

Using Fast Application Notification (FAN) events eliminates applications waiting on TCP timeouts, time wasted processing the last result at the client after a failure has occurred, and time wasted running work on slow, suspended, or terminated nodes.

Applications can waste time in many critical ways:

- Waiting for TCP/IP timeouts when a node fails without closing sockets, and for every subsequent connection while that IP address is down.
- Attempting to connect when services are down.
- Not connecting when services resume.
- Processing the last result at the client when the server goes down.
- Attempting to run work on sub-optimal nodes.

When a node fails without closing sockets, all sessions that are blocked in an I/O wait (read or write) wait for `tcp_keepalive`. This wait status is the typical condition for an application connected by a socket. Sessions processing the last result are even worse off, not receiving an interrupt until the next data is requested.

How FAN is Used with Oracle AI Database and Applications

Fast Application Notification (FAN) is essential to prevent applications from stop responding on TCP/IP timeouts.

FAN events are published using Oracle Notification Service starting with Oracle Database 12.2. Advanced Queuing is used for FAN events only for older Oracle Call Interface (OCI) applications (OCI drivers before 12.2). The publication mechanisms are automatically configured as part of your Oracle RAC installation. There are specific settings needed on each client to enable the client to receive FAN events.

- For OCI clients, the service attribute `notification` must be set on the server. For example `srvctl modify service -db EMEA -service GOLD -notification TRUE`. Also, for OCI clients you must set `events` to `TRUE` in the `oraaccess.xml` configuration file.
- For ODP .Net clients, you must set `HA events` to `TRUE` in the `oraaccess.xml` in the connect string.
- For Universal Connection Pool clients, set the pool property Fast Connection Failover to true (`setFastConnectionFailoverEnabled(true)`) in the properties file.

All clients can use the auto-configuration of ONS to receive events, but the clients still need settings to ensure they react to these events.

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

Oracle connection pools use FAN to receive very fast notification of failures, to balance connections following failures, and to balance connections again after the failed components are repaired. So, when a service connecting to an Oracle AI Database instance starts, the connection pool uses the FAN event to route work to that resource, immediately. When a

Copyright © 1999, 2025, Oracle and/or its affiliates.

service for a database instance or node fails, the connection pool uses the FAN event to immediately interrupt applications to recover.

For cluster configuration changes, the Oracle Real Application Clusters (Oracle RAC) high availability framework publishes a FAN event immediately when a state change occurs in the cluster. Instead of waiting for the application to time out against the database and detect a problem, applications can receive FAN events and react immediately. With FAN, in-flight transactions are immediately terminated and the client notified when the instance fails.

FAN also publishes load balancing advisory events. Applications can take advantage of the load balancing advisory FAN events to direct work requests to the instance in the cluster that is currently providing the best service quality.

If you specify the connection load balancing goal `CLB_GOAL_LONG` for a database service, then the listener uses the load balancing advisory when the listener balances the connection loads. When load balancing advisory is enabled, the metrics used for the listener are finer grained.

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

- Applications can use FAN if you use an integrated Oracle client. The integrated clients for FAN events include Oracle JDBC Universal Connection Pool, ODP.NET connection pool, OCI session pool, Oracle WebLogic Server Active Gridlink for Oracle RAC, and OCI, JDBC-thin, and ODP.NET Managed and Unmanaged clients. The pooled clients can take advantage of the load balancing advisory FAN events.
- You can configure third-party application containers, such as those provided by Apache Tomcat and WebSphere, to use the built-in FAN support offered by using the Universal Connection Pool in place of the default pool, which is certified as a connection pool for third-party Java application servers including Apache Tomcat and WebSphere.
- Use the FAN-aware capability of the Oracle drivers by using standard interfaces to test connections on *get* or *release* from the third-party connection pools in use by third-party application servers or custom applications.
  - This solution applies to standard Java applications through the use of the standard TNS connect string and ensures that the `ons.jar` and `simpleFAN.jar` files are available on the application CLASSPATH.
  - For the OCI/OCCI driver, the `OCI_ATTR_SERVER_STATUS` server context handle attribute is sensitive to FAN events and will return `OCI_SERVER_NOT_CONNECTED` if the connection has been affected by a FAN event.
- You can implement FAN with server-side callouts on your database tier.
- Applications can use FAN programmatically by using the JDBC and Oracle RAC FAN application programming interface (API) or by using callbacks with OCI and ODP.NET to subscribe to FAN events and to run event handling actions upon the receipt of an event.

For planned maintenance and applications using OCI or Pro* precompilers (and not using the OCI session pool or Tuxedo), an application must check `OCI_ATTR_SERVER_STATUS`. Add this check when sessions are returned to your own connection pool, and for idle connections, regularly. Following a FAN down event with planned maintenance, this attribute is set to `OCI_SERVER_NOT_CONNECTED`. The application closes the connection after reading this disconnected status. The session remains open for draining of active work until the application closes, providing error-free failover.

If you use one of the integrated clients listed in the first item of the preceding list, then, for `DOWN` events, the disruption to the application is minimized because the FAN-aware client terminates the connections to the failed instance or node before they are reused. Active work can be allowed to complete and, if there is a remaining instance, then continuous service can be maintained for ongoing work. Any sessions active when the instance or service stops are

Copyright © 1999, 2025, Oracle and/or its affiliates.

terminated and the application user is immediately notified. Incomplete transactions can be protected by Application Continuity, if it is enabled. Application users who request connections are directed only to available instances.

For `UP` events, when database services and instances are started, new connections are created so that the application can immediately take advantage of the extra hardware resources or additional capacity.

Requirements for Using FAN

Learn what you need to do to take advantage of FAN-aware capabilities in client drivers connecting to Oracle Real Application Clusters (Oracle RAC) databases.

Client drivers on releases after Oracle Database 12c release 2 (12.2) are FAN-aware, and FAN is enabled by default. This is also true for the JDBC Thin driver (12.2.0.1 and later), and Oracle Data Provider for Net (ODP.NET) drivers. A client driver can detect planned and unplanned FAN events, and take action beneath the application.

To take advantage of FAN-aware capabilities in the drivers, the following is required:

- For the thin Java driver, beginning with release 12.2, FAN is automatically enabled by placing the `ons.jar` and `simpleFAN.jar` files on the `CLASSPATH`, and by using the recommended TNS format. Using the recommended TNS format automatically configures ONS. Also with the Java thin driver, FAN is supported for both planned and unplanned events. For unplanned outages, the FAN interrupt is immediate. For planned maintenance, configure the Java application servers or custom pools using standard interfaces to test connections on *get* or *release* from third-party connection pools. For example, depending on the application server, test `TestConnectionsOnReserve`, `TestOnBorrow`, or `PreTest` connections.

  With this approach, when a FAN event is received during planned maintenance, Fast Connection Failover (FCF) closes sessions when they are tested, because the application does not have a connection to the database at this time, and can retry for a new connection. The connection tests may use `isValid`, `isClosed`, `isUsable`, or `PingDatabase`.
- At the time the SQL command runs, the database will drain the connection, if it is affected by the upcoming planned maintenance. Connection pools, data sources, and, in the programmatic case, customer applications, must all be ready to manage the recoverable error that occurs when the SQL command runs, which usually closes the physical connection.
- Third-party Java application servers and Java applications can use the `PooledConnection` standard interface when developing connection pools.
- Beginning with the 11.2.0.3 release of the Oracle Call Interface (OCI) and Oracle C++ Call Interface (OCCI) drivers, when the `OCI_ATTR_SERVER_STATUS` server context handle attribute returns `OCI_SERVER_NOT_CONNECTED`, the application must terminate the connection. Work will be drained for planned maintenance. Releases of the driver after 12.2.0.1 can also detect `OCISessionRelease` and `OCIRequestEnd` when it receives a planned DOWN event.

FAN Callouts

Fast Application Notification (FAN) callouts are server-side scripts or executables that run whenever a FAN event is generated.

You can design and build callouts to do many things, such as:

- Log status information

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Page DBAs or to open support tickets when resources fail to start
- Automatically start dependent external applications that must be co-located with a service
- Shut down services when the number of available instances for a database decreases, for example, if nodes fail
- Automate the fail back of a service to preferred instances, if the `-failback` parameter is not sufficient

Fast Application Notification High Availability Events

Learn how the Fast Application Notification (FAN) event delivers information to a callout program.

In the following example, FAN event types are listed always as the first entry when you receive FAN information through a callout, as in the following examples:

```
#service UP when the service starts
SERVICEMEMBER VERSION=1.0
   service=HRPDB1.example.com database=ractest
   instance=ractest2 host=prod_host01_1 status=up reason=BOOT
   card=1 timestamp=2019-10-24 09:11:51 timezone=+00:00
   db_domain=example.com
SERVICE VERSION=1.0
   service=HRPDB1.example.com database=ractest instance=ractest2
   host=prod_host01_1 status=up reason=BOOT
   timestamp=2019-10-24 09:11:51 timezone=+00:00
   db_domain=example.com

#service DOWN
SERVICEMEMBER VERSION=1.0 service=HRPDB1.example.com database=ractest
   instance=ractest2 host=prod_host01_1 status=down reason=USER
   timestamp=2019-10-25 17:59:43 timezone=+00:00 db_domain=example.com
   drain_timeout=120
SERVICE VERSION=1.0 service=HRPDB1.example.com database=ractest
   instance=ractest2 host=prod_host01_1 status=down reason=FAILURE
   timestamp=2019-10-24 21:25:57 timezone=+00:00 db_domain=example.com
```

Note that the preceding examples normally display as one line.

FAN event types include:

```
DATABASE
INSTANCE
NODE
SERVICE
SERVICEMEMBER
SERVICEMETRICS
```

The `DATABASE` and `INSTANCE` types list the default database service as `DB_UNIQUE_NAME`.

All events except for `NODE` events include a `db_domain` field.

Events of `SERVICEMETRICS` type are load balancing advisory events.

The following table describes name-value pairs for the event parameters, and provides more information about load balancing events:

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table 6-2 Event Parameter Name-Value Pairs and Descriptions

Parameter	Description
VERSION	Version of the event record. Used to identify release changes.
database	The unique name of the database supporting the service; matches the initialization parameter value for DB_UNIQUE_NAME, which defaults to the value of the DB_NAME initialization parameter.
instance	The name of the instance that supports the service.
host	The name of the node that supports the service or the node that has stopped; matches the node name known to Cluster Synchronization Services (CSS).

service	The service name; matches the name of the service as listed in DBA_SERVICES and is domain-qualified as appropriate. Refer to the following examples:
	
	`SERVICEMEMBER VERSION=1.0 service=swingbench`
	` database=orcl instance=orcl_2 host=dev_host1 status=up`
	` reason=USER card=1 timestamp=2018-05-29 17:26:37`
	` timezone=-07:00 db_domain=`
	
	`SERVICEMEMBER VERSION=1.0`
	`service=swingbench.example.com`
	` database=orcl instance=orcl1 host=dev_host1 status=up`
	` reason=USER card=2 timestamp=2018-05-03 17:29:28`
	` timezone=-07:00 db_domain=example.com`
	
	`SERVICEMEMBER VERSION=1.0`
	`service=swingbench.example.com`
	` database=orcl instance=orcl2 host=dev_host1 status=up`
	` reason=USER card=1 timestamp=2018-07-03 17:29:18`
	` timezone=-07:00 db_domain=example.com`

status	Values are UP, DOWN, NODEDOWN (this only applies to the NODE event type), NOT_RESTARTING, and UNKNOWN. **Notes:** • When the node is down, the status is NODEDOWN, as opposed to DOWN for other event types. • When STATUS=NODEDOWN and REASON=MEMBER_LEAVE, a node has failed and is no longer part of the cluster, or a user has stopped a node. • When STATUS=NODEDOWN and REASON=PUBLIC_NW_DOWN, the node is up but it is unreachable because the public network is down because of either a failure or a user action. • Multiple public networks are supported by Oracle Clusterware. The FAN event reflects this fact.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table 6-2 (Cont.) Event Parameter Name-Value Pairs and Descriptions

Parameter	Description
`reason`	`AUTOSTART`, `BOOT`, `DEPENDENCY`, `FAILURE`, `MEMBER_LEAVE`, `PUBLIC_NW_DOWN`, `USER`. **Notes:** • For `DATABASE` and `SERVICE` event types, `REASON=AUTOSTART` if, when the node started, the `AUTO_START` resource attribute was set to restore, and the resource was offline before the node started. • For `DATABASE` and `SERVICE` event types, `REASON=BOOT` if, when the node started, the resource started because it was online before the node started. • For SRVCTL and Oracle Enterprise Manager operations, `REASON=USER` describes planned actions for such operations as draining work.

`card` (cardinality)	The number of service members that are currently active; included in all `SERVICEMEMBER UP` events.
	Here is an example of a `SERVICEMEMBER UP` event:
	`SERVICEMEMBER VERSION=1.0`
	`service=swingbench.example.com`
	`database=orcl instance=orcl_2 host=dev_host3 status=up`
	`reason=USER card=1`
	`timestamp=2018-07-12 14:46:46  timezone=-07:00`
	`db_domain=example.com`

`incarn` (incarnation)	For `NODEDOWN` events; the new cluster incarnation. This value changes each time a member joins or leaves the cluster.
	Here is an example of a `NODEDOWN` event:
	`VERSION=1.0 event_type=NODE host=dev_host2`
	`incarn=175615351 status=nodedown`
	`reason=member_leave timestamp=2019-10-24 05:55:06`
	`timezone=+00:00`

`timestamp`	The time according to Oracle Clusterware that an event occurs.
`timezone`	The time zone of Oracle Clusterware where the event occurred, given as GMT +/-hh:mm.
`drain_timeout`	Time in seconds during which a service will drain. Appears with SERVICEMEMBER events

`vip_ips`	VIP on a public network that has gone down. Part of a NODE event.
	Here is an example of a `NODEDOWN` event:
	`NODE VERSION=2.0 host=my-exa status=nodedown`
	`reason=public_nw_down`
	`incarn=0 timestamp=2019-10-24 09:02:35 timezone=+00:00`
	`vip_ips=10.1.1.94`

Copyright © 1999, 2025, Oracle and/or its affiliates.

Some of the FAN event record parameters have values that correspond to values returned by the SYS_CONTEXT function using the default namespace USERENV, as shown in the following table:

Table 6-3 FAN Parameters and Matching Session Information

FAN Parameter	Matching Session Information
SERVICE	sys_context('userenv', 'service_name')
DATABASE_UNIQUE_NAME	sys_context('userenv', 'db_unique_name')
INSTANCE	sys_context('userenv', 'instance_name')
CLUSTER_NODE_NAME	sys_context('userenv', 'server_host')

Subscription to High Availability Events

To monitor and notify applications about services, Oracle Real Application Clusters (Oracle RAC) uses Oracle RAC Fast Application Notification (FAN).

Oracle RAC uses FAN to notify applications about configuration changes and the current service level that is provided by each instance where the service is enabled. If you are using an Oracle Call Interface (OCI) client, or an ODP.NET client to receive FAN events, then you must enable the service used by that client to access the alert notification queue by using SRVCTL with the -notification parameter.

Using Fast Application Notification Callouts

Fast Application Notification (FAN) callouts are server-side program files that Oracle RAC runs immediately when high availability events occur.

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

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

To use FAN callouts, place a program file in the *Grid_home*/racg/usrco directory on every node that runs Oracle Clusterware. The program file must be able to run standalone when called, with optional arguments, from another program. The following is an example of a shell script, named callout.sh, which is placed in the *Grid_home*/racg/usrco directory:

```
#! /bin/bash
FAN_LOGFILE= [your_path_name]/admin/log/'hostname'_uptime'.log
echo $* "reported="'date' >> $FAN_LOGFILE &
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

The previous example adds entries similar to the following in the log file, indicated by `$FAN_LOGFILE` in the shell script, each time a FAN event is generated:

```
NODE VERSION=2.0 host=my-exa status=nodedown reason=public_nw_down
incarn=0 timestamp=2019-10-24  09:02:35 timezone=+00:00 vip_ips=10.1.1.94
```

The contents of a FAN event record matches the current session of the user logged on to the database. The user environment (`USERENV`) information is also available using Oracle Call Interface (OCI) connection handle and descriptor attributes (using `OCIAttrGet()`). Use this information to take actions on sessions that match the FAN event data.

In general, events are only posted to user callouts on the node from which the event originated. For example, if the database on `node1` goes down, then the callout is posted to `node1`, only. The only exceptions to this are node down and VIP down events—these events are posted to all nodes, regardless of from where they originated.

Related Topics

- Fast Application Notification High Availability Events
  Learn how the Fast Application Notification (FAN) event delivers information to a callout program.
- *Oracle Call Interface Developer's Guide*

Configure for Unplanned Outages

You can assign services to one or more instances in an administrator-managed Oracle RAC database to hide outages.

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

It is a complex task for application developers to mask outages of a database session (instance, node, storage or network, or any other related component). As a result, errors and timeouts are often exposed to the end users, which can lead to user frustration, lost productivity, and lost opportunities. Together, FAN and Application Continuity mask outages from users and applications by recovering the in-flight work for impacted database sessions following outages. Application Continuity performs this recovery beneath the application, so that the outage appears to the application as a slightly delayed processing of the request.

If Oracle Real Application Clusters (Oracle RAC) detects an outage, then Oracle Clusterware isolates the failed component and recovers the dependent components. For services, if the failed component is an instance, then Oracle Clusterware attempts to maintain the cardinality of the service. If the service definition allows for failover and that is necessary to maintain cardinality, then failover occurs.

Fast Application Notification (FAN) events can occur at various levels within the Oracle AI Database architecture. To provide backward compatibility with earlier release Oracle Call Interface (OCI) clients, they are published through Oracle Notification Service and Advanced Queuing. FAN callouts can also be written to run on the database server in response to FAN events.

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

FAN is published from a remaining node when the failed node is out of service. The location and number of instances in an Oracle RAC environment that provide a service are transparent to applications. Restart and recovery are automatic, including the restarting of the subsystems, such as the listener and the Oracle Automatic Storage Management (Oracle ASM) processes, not only the database. You can use FAN callouts to report faults to your fault management system and to initiate repair jobs.

Related Topics

- About Application Continuity
  Application Continuity provides high availability for your application workloads.
- *Oracle AI Database Net Services Administrator's Guide*

Managing Planned Maintenance

To minimize service disruption to application users, Oracle Real Application Clusters (Oracle RAC) provides interfaces that relocate, disable, and enable services.

- About Planned Maintenance Management
  For repairs, upgrades, and changes, drain your sessions before stopping or relocating the services.
- Planned Maintenance Without User Interruption
  Follow the planned maintenance best practices to avoid the need to restart your application servers.
- Managing a Group of Services for Maintenance
  With Oracle Real Application Clusters (Oracle RAC), you can use SRVCTL to manage groups of services in your cluster.
- Server Draining Ahead of Planned Maintenance
  Before planned maintenance, drain or failover database sessions at the database instance so application work is not interrupted.
- Planned Failover with Application Continuity
  Planned Failover is a failover that is invoked by Oracle AI Database at points where the database knows that the session is replayable using Application Continuity and that the session is expected not to drain.

About Planned Maintenance Management

For repairs, upgrades, and changes, drain your sessions before stopping or relocating the services.

When you relocate a service, you indicate the service should run on another instance. When a service is stopped or relocated, FAN is published with a planned reason code, typically `reason=user`. Once you complete the operation, you can return the service to normal operation or enable the service and then restart it. When a service restarts, FAN is published with `UP` status codes.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Due to dependencies, if you manually shut down your database, then all of your services for that database automatically stop. If you want your services to automatically start when you manually restart the database, then you must set the management policy of the service to automatic. If you want to shut down only one instance of the database and you want the service to remain offered, then you can either relocate the service using `srvctl relocate service` or stop the instance using `srvctl stop instance` with the `-failover` option, which enables the service to automatically failover according to its failover policy.

In either case, Oracle recommends that work running under the service drain at request boundaries. The drain interval is specified as an attribute of the service or you can provide a drain interval on the SRVCTL command line.

Related Topics

- Service Management Policy
  When you use Oracle Clusterware to manage your database, you can configure startup options for each individual database service when you add the service using the `srvctl add service` command with the `-policy` parameter.

Planned Maintenance Without User Interruption

Follow the planned maintenance best practices to avoid the need to restart your application servers.

For planned maintenance, the recommended approach is to provide time for current work to complete before maintenance is started. You do this by draining the current work. Oracle provides several methods for draining. Choose the one that best suits your application:

- Oracle Connection Pools
- Standard driver-side connection tests
- Server-side connection tests and more
- Planned failover with Transparent Application Continuity

Use draining in combination with your chosen failover solution for the requests that do not complete within the allocated time for draining. The failover solution will try to recover sessions that are expected not to drain in the allocated time.

The service attributes `-drain_timeout` and `-stopoption` control the drain time period, and then how the service manages sessions that have not completed once this time period expires. Requests that complete and then check back in to the pool or close, can be directed to a new location that is not affected by the planned maintenance.

Application Continuity provides additional cover, giving continuous service for those requests that do not complete within the allotted drain time. Using any FAN-aware pool allows sessions to drain at request boundaries after receipt of the FAN planned `DOWN` event.

Because not every application uses an Oracle connection pool and not every application is FAN-aware, the database inspects sessions during planned maintenance seeking safe places to stop a session so that the application is not disrupted. After stopping the service, the database looks for safe places where the connection can be closed. When the connection is closed, the database cleans up the session.

Stopping a session at a safe place enables the application to open a new connection with the states that it needs. Draining sessions may take a period of work to flow through each session. There is no requirement that closing a session is immediate, rather the close must occur at a safe place that exposes no errors to the application, and, preferably, before the drain timeout period has expired.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Requests are far more important than transactions because they enable the issued work to complete. For draining requests, the Oracle Universal Connection Pool uses the drain timeout to gradually drain, which prevents an overload of logins on the instances drained, by slowly releasing the original sessions across the time period rather than all at once. Gradual draining has the benefit of not disturbing the other work ongoing at the target instances.

Both `DRAIN_TIMEOUT` and `STOP_OPTION` are service attributes that you can define when you add the service or modify it after creation. You can also specify these attributes using SRVCTL, which will take precedence over what is defined on the service. You can specify the `-drain_timeout` and `-stopoption` parameters when using the following SRVCTL commands:

- `srvctl add service`
- `srvctl modify service`
- `srvctl relocate service`
- `srvctl stop service`
- `srvctl stop database`
- `srvctl stop instance`
- `srvctl stop pdb`
- `srvctl relocate pdb`

To manage planned maintenance without user interruption:

1. Use SRVCTL to relocate a singleton service or a service not running on all nodes. Use the `-force` flag with the previously listed SRVCTL commands, except `add` and `modify`. You must use the `-force` flag if you specify the `-stopoption` parameter on the command line when you run either `srvctl relocate service` or `srvctl stop service`. For example:

   ```
   $ srvctl relocate service –db mycdb01 –service myservice –drain_timeout 120
     –stopoption IMMEDIATE –oldinst mycdb01_01 -force
   ```

   The preceding command relocates the service named `myservice01` from the instance named `mycdb01_01` to any instance on which it is configured to run. Oracle Clusterware chooses the instance if you do not specify a target on the command line, and waits two minutes (in this example) for any active sessions to drain, after which any sessions remaining on `mycdb01_01` are forcibly disconnected. The connection pool automatically releases a connection at a request boundary.

   > **Note**
   >
   > If the service you want to relocate is a uniform service that is currently running on all nodes, then the preceding command returns an error, unless the service is not up on all instances, in which case the preceding command example would succeed for a uniform service.

2. The FAN planned `DOWN` event clears idle sessions from the connection pool immediately and marks active sessions to be released at the next check-in. These FAN actions drain the sessions from the instance without disrupting the users.

   Existing connections on other instances remain usable, and new connections can be opened to these instances if needed. The database also marks the sessions to drain. The database looks for connection tests and, in Oracle Database 19c and later, for safe places

Copyright © 1999, 2025, Oracle and/or its affiliates.

to failover. An implicit connection boundary with Transparent Application Continuity is such a place.

3. Not all sessions, in all cases, will check their connections into the pool. Oracle recommends, as a best practice, to have a timeout period (by setting the `-drain_timeout` parameter), after which the instance is forcibly shut down or the service stopped, evicting any remaining client connections.

   After the drain interval expires, the `-stopoption` parameter is implemented, which you can define against a service or a database, as follows:

   - When stopping a service (`srvctl stop service`), you can specify one of the following stop options using the `-stopoption` parameter: TRANSACTIONAL or IMMEDIATE
   - When stopping a database (`srvctl stop database`), you can specify one of the following stop options using the `-stopoption` parameter: NORMAL, TRANSACTIONAL, IMMEDIATE, or ABORT

   The database stop options correlate to the service stop options, as follows:

   NORMAL=NONE
   TRANSACTIONAL/TRANSACTIONAL LOCAL=TRANSACTIONAL
   IMMEDIATE/ABORT=IMMEDIATE

   For those services that are configured to use Application Continuity, an attempt is made to recover these remaining sessions, after they are terminated, masking the outage from users and applications.
4. Once maintenance is complete, restart the instance and the services on the original node.
5. The FAN `UP` event for the service informs the connection pool that a new instance is available for use, allowing sessions to be created on this instance at next request boundaries.

Related Topics

- About Application Continuity
- Server Draining Ahead of Planned Maintenance
  Before planned maintenance, drain or failover database sessions at the database instance so application work is not interrupted.

Managing a Group of Services for Maintenance

With Oracle Real Application Clusters (Oracle RAC), you can use SRVCTL to manage groups of services in your cluster.

- Stopping a Group of Services Example
  See how you can use `SRVCTL` to stop services by node name, database name, Pluggable Database name, or instance name.
- Starting Services
  You can use the `srvctl start service` command to start all services on a node, all services offered by a database, or all services offered by a Pluggable Database.
- Pluggable Database-Level Operations
  You can use SRVCTL to manage services on Pluggable Databases.
- Relocating Services
  You can use the `srvctl relocate service` command to relocate services to a target destination, which can be an instance, a node, or a database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Stopping Services
  You can use the `srvctl stop service` command to stop all services on a node, all services offered by a database, or all services offered by a Pluggable Database.

Stopping a Group of Services Example

See how you can use `SRVCTL` to stop services by node name, database name, Pluggable Database name, or instance name.

Many enterprises run a large number of services, whether it be many services offered by a single database or instance, or many databases offering a few services running on the same node. You no longer need to run SRVCTL commands for each individual service but need only specify the node name, database name, Pluggable Database name, or the instance name for all affected services.

For example, if you want to stop all of the services running on a node called `racnode01`, then you can use the following command:

```
$ srvctl stop service –node racnode01 –drain_timeout 60 –stopoption IMMEDIATE
```

The command stops all services running on `racnode01`, allowing a drain interval of 60 seconds. After 60 seconds, any remaining sessions are stopped immediately. The 60-second drain timeout interval overrides any attribute setting on any of the services.

The command can also be qualified to stop the databases on a node, as in the following example:

```
$ srvctl stop instance     -node racnode01 -drain_timeout 60 –stopoption
TRANSACTIONAL LOCAL -failover –force
```

When you specify the `-failover` parameter:

- All services are relocated, if possible, respecting the drain timeout interval and the stop option specified.
- Any services that cannot be failed over are stopped, using the stop option specified.
- Wait for the length of the drain timeout interval or until all sessions for targeted services are removed, whichever is sooner.
- All instances stop according to the stop option specified.

When you specify the `–stopoption TRANSACTIONAL LOCAL` parameter:

- Remaining services stop according to the drain timeout interval and stop option specified.
- Wait for the length of the drain timeout interval or until all sessions for targeted services are removed, whichever is sooner.
- The instance stops using the `TRANSACTIONAL LOCAL` stop option.

Starting Services

You can use the `srvctl start service` command to start all services on a node, all services offered by a database, or all services offered by a Pluggable Database.

To start services, you can also supply a list of services (a subset of all services) to the `srvctl start service` command that you want to start. Additionally, you can provide a node restriction, used in conjunction with the database option, for all services that can be started on

Copyright © 1999, 2025, Oracle and/or its affiliates.

a particular node. You can restrict the `srvctl start service` command to start only the parallel query service by specifying the `-pq` parameter.

The following examples illustrate how you can start services:

To start all of the services offered by a single Pluggable Database:

```
$ srvctl start service –db myRACCDB01 –pdb myPDB01 –startoption OPEN
```

To start all services on a given database and any of its Pluggable Databases:

```
$ srvctl start service –db myRACDB
```

To start a list of services on a given database, regardless of any Pluggable Database with which they are associated:

```
$ srvctl start service –db myRACDB –service
"myFirstService,mySecondService,myThirdService"
```

To start all services for a database that can run on a given node:

```
$ srvctl start service –d myRACDB –node racnode01
```

Pluggable Database-Level Operations

You can use SRVCTL to manage services on Pluggable Databases.

To start all services for a Pluggable Database, for all instances or a single instance:

```
$ srvctl start service -db db_name -pdb pdb_name [-instance instance_name]
```

To stop all services for a Pluggable Database, for all instances or a single instance:

```
$ srvctl stop service -db db_name -pdb pdb_name [-node node_name | -instance
   inst_name] [-stopoption stop_option] [-drain_timeout timeout]
   [-force [-noreplay]]
```

> **Note**
>
> The `-pdb pdb_name` parameter is optional. If you omit the Pluggable Database name, then the operation occurs for the entire Container Database (all Pluggable Databases within this container).

Relocating Services

You can use the `srvctl relocate service` command to relocate services to a target destination, which can be an instance, a node, or a database.

In the following command examples, all services are relocated from the named database, Pluggable Database, instance, or node. The services will only relocate if the target can support that service, as defined by the service configuration. Any services that cannot be relocated

Copyright © 1999, 2025, Oracle and/or its affiliates.

remain at the original location. A placement error is recorded against any services that could not be relocated, or were already running at the new target. Services that fail to relocate remain running at their original location, and any sessions remain active.

```
$ srvctl relocate service –db myRACCDB –oldinst RACCDB_01 –newinst RACCDB_03
  -drain_timeout 30 -stopoption immediate
```

or

```
$ srvctl relocate service –db myRACCDB –pdb myPDB01 –currentnode racnode01
  –targetnode racnode02 -drain_timeout 30 -stopoption immediate
```

The relocate operation starts the service in the new location before stopping the service in its existing location.

If you do not specify a target destination, then Oracle Clusterware relocates all services or specific services from the specified database, Pluggable Database, instance, or node, as in the following examples:

```
$ srvctl relocate service –db myRACCDB –service "myService01,myService02"
  -drain_timeout 30 -stopoption immediate
```

or

```
$ srvctl relocate service –db myRACCDB –pdb myPDB01 -drain_timeout 30
  -stopoption transactional
```

If there is no valid target available, then the service remains at the original location and the sessions remain active. You must examine the services and stop them if that is what you want to do.

When you relocate a service, it starts at the new location before it stops at the original location. Oracle Clusterware can start that new instance or Pluggable Database as a dependency. When specified, the `-drain_timeout` and `-stopoption` parameters override the service attributes.

Stopping Services

You can use the `srvctl stop service` command to stop all services on a node, all services offered by a database, or all services offered by a Pluggable Database.

When you want to stop a subset of services, you can also supply a list of services (a subset of all services) that you want to stop to the `srvctl stop service` command. You can also restrict the `srvctl stop service` command to stop only the parallel query service by specifying the `-pq` parameter.

To stop all of the services offered by a single Pluggable Database:

```
$ srvctl stop service –db myRACCDB01 –pdb myPDB01 –drain_timeout 15 –
stopoption TRANSACTIONAL
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

To stop all services on a given database and any of its Pluggable Databases:

```
$ srvctl stop service –db myRACDB –drain_timeout 15 –stopoption IMMEDIATE
```

To stop only a subset of the services offered by a database:

```
$ srvctl stop service –db myRACDB –service "myFirstService,mySecondService,
  myThirdService" –drain_timeout 60 –stopoption IMMEDIATE
```

> **Note**
>
> If you use the `–wait YES SRVCTL` command line parameter, then the `–stopoption` parameter is not enforced until the entire drain timeout interval has expired, even if all of the sessions have exited prior to this interval completing.

Server Draining Ahead of Planned Maintenance

Before planned maintenance, drain or failover database sessions at the database instance so application work is not interrupted.

When you prepare for planned maintenance, you must stop or relocate the services that are using the server infrastructure. Relocating services is done over a period of time prior to the planned outage and is based on the nature of work associated with each service.

The procedure for rolling planned maintenance moves services in advance of maintenance to another database instance, and notifies the client-side drivers, connections pools, the database instance itself, and other subscribers that maintenance is pending, and what needs to be drained (either connections or sessions using this service). Once notified of draining, a Fast Application Notification (FAN) event is sent and the client pools behave as described elsewhere, in addition, the database begins to search for safe places to release connections and, if needed, to migrate the connections.

Moving or stopping a service triggers a FAN notification that is received by the subscribing Oracle drivers and Oracle connection pools. The FAN notification also triggers session draining at the server. Immediately, new work to that service is directed to another functioning instance of that service. Existing sessions are marked for release after their work completes. As work completes and the connections are returned to the connection pool, either the Oracle driver or the connection pool terminates these sessions.

Draining Sessions at the Database

For OLTP applications, application servers, and custom applications, which all have their own connection pools that borrow and return database sessions, it is safe to drain a database session when it is no longer borrowed. The optimal point for the Oracle server infrastructure to close a session is when the application server tests the validity of that connection. No error is returned to the application when the connection pool manager tests the validity of connections when borrowing and releasing, and also finds that the connection is no longer valid.

A safe place is a point where an application is not disturbed. In the case of connection pools, that means connections that are not borrowed (checked-in), and, in the case of applications, the same applies at the point of borrowing or returning a connection. At this time, all work is either complete or not started. The database can also fail over connections when all states can be restored transparent to the application.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Oracle AI Database uses an extensible set of rules and heuristics to detect when to take the database session away. When draining starts, the database session persists at the database until a rule is satisfied. The rules include the following:

- Standard application server tests for validity
- Custom SQL tests for validity
- Request boundaries are in effect and no request is active
- Request boundaries are in effect and the current request has ended
- The session has one or more session states that are recoverable, and can be recreated at failover

In the case of connection tests, for example, it is standard practice for application servers, pooled applications, job schedulers, and others, to test connections when borrowed from connection pools, when returned to the pool, and at batch commits. When draining, the database intercepts the connection test, closes the connection and returns a failed status for the test. The application layer issuing the connection test is ready to handle a failed return status and, typically, issues a further request, to obtain a different connection. The application is not disturbed.

Not all sessions can be drained, such as when a connection is not returned to the pool or when FAN is not in use. If Transparent Application Continuity or Application Continuity is enabled, then the server detects request boundaries where Application Continuity can recover the session fast. The server can interrupt the session, which Application Continuity recovers elsewhere (such as, to another server in the Oracle RAC cluster) with no interruption.

For database sessions that do not drain, the database must finds a break point when the session can be replaced. At a break point, a connection can be failed over transparently when states are known and recoverable. Break points can be transaction boundaries, a request starting (beginRequest), before calls are processed in that request, and patterns, such as an audit call that signals that a request is starting or ending. Break points apply only when states are known to be restorable. Oracle AI Database software determines where to failover the session and invokes Application Continuity to failover the session.

Failing over connections requires that you enable Application Continuity, Transparent Application Continuity, or transparent application failover (TAF), depending on your application.

Draining Sessions Using XA Transactions

When using XA, enable `XA_START_NEW` in `DBA_CONNECTION_TESTS`. You can enable this connection test for services with applications that are using XA transactions.

When a session is marked to drain and the first branch of an XA transaction is received at the database instance, then the following actions take place:

1. The database server closes the session.
2. The database server returns `XAER_RMFAIL` and places the connection back in `R0` state.
3. The client needs to perform `xa_open()` again to continue, which is the expected behavior.

These steps allow XA transactions that are underway to complete and do not allow new XA transactions at the instance.

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> When using the `XA_START_NEW` drain method for XA, do not mix this method with other drain methods, such as client side connection tests or FAN on sessions that are using XA.

Draining XA and Sessionless Transactions

As described in the previous section, the connections are failed over at safe points like when they are not borrowed (checked-in) during the draining period. You cannot do that for XA and sessionless transactions. These type of transactions are not supported by the Application Continuity or Transparent Application Continuity and they do not failover when they are active in the sessions. Such transactions are aborted after the drain timeout.

Starting with Oracle AI Database 26ai, the extensible set of rules and heuristics to detect when to take the database session away are further enhanced by adding two new rules, one for XA and the other for the sessionless transactions.

When you enable the new rules, the borrowed (checked-in) connections are allowed to resume the existing XA and sessionless transactions during the drain period. But, you cannot start new transactions. This increases the transaction success rate by allowing the current transactions to successfully end while disallowing the new transactions to start.

Enabling or Disabling Predefined Connection Tests

By default, predefined connection tests `PING_TEST`, `ENDREQUEST_TEST`, `XA_START_TEST`, and `SESSIONLESS_START_TEST` are disabled and all other predefined connection tests are enabled. To enable predefined connection test rules, which are disabled by default for a particular service, follow these steps.

1. To enable the connection test for a CDB or PDB, use a SQL statement similar to the following:

   ```
   SQL> execute
   dbms_app_cont_admin.enable_connection_test(connection_test_type =>
   dbms_app_cont_admin.xa_start_test);
   ```

2. To selectively disable the connection test for services where it is not needed, use a SQL statement similar to the following:

   ```
   SQL> execute
   dbms_app_cont_admin.disable_connection_test(connection_test_type =>
   dbms_app_cont_admin.xa_start_test, service_name => 'tkfojpm2');
   ```

Adding, Disabling, Enabling, and Removing Connection Tests for Draining at the Server

You can add a SQL connection test to a service, a Pluggable Database. To add a new connection test to your PDB, use `ALTER SESSION SET CONTAINER` to switch to your PDB.

There are four SQL connection tests added for every database service and Pluggable Database service, by default, so, if an application uses these following SQL connection tests on the connection, then you do not to need to add them:

```
SELECT 1 FROM DUAL;
SELECT COUNT(*) FROM DUAL;
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

```
SELECT 1;
BEGIN NULL;END
```

> **Note**
>
> - When you use a connection test, the outcome of the connection test applies to that session only. Do not use connection tests to make general decisions about the instance and to make a determination to stop more than the session to which the test applies. Disable connection pool properties for flushing and destroying the pool on connection test failure. This is important for Oracle WebLogic Server data sources.
> - For external monitors, you are recommended to prefix the SQL used in that monitor with a hint `/*+ MONITOR */` for SQL used inside the monitors, so that the monitor does not drain.

- To add a server-side SQL connection test for a service, use a SQL statement similar to the following:

  ```
  SQL> execute dbms_app_cont_admin.add_sql_connection_test('select dummy
  from dual','sw_orcl');
  ```

  To add a server-side SQL connection test for a Pluggable Database, use a SQL statement similar to the following:

  ```
  SQL> execute dbms_app_cont_admin.add_sql_connection_test('begin
  null;end;');
  ```

  After you add the SQL connection test, it will be enabled by default.
- You can disable a SQL connection test if you do not need it or it is not in use by logging on to a Pluggable Database and using a SQL statement similar to the following:

  ```
  SQL> execute
  dbms_app_cont_admin.disable_connection_test(dbms_app_cont_admin.sql_test,'s
  elect dummy from dual');
  ```

  By default, the ping test and end request test are disabled but, if you want to disable them after enabling them, then you can use either of the following SQL statements:

  If you want to disable the ping test, then use a SQL statement similar to the following:

  ```
  SQL> execute
  dbms_app_cont_admin.disable_connection_test(dbms_app_cont_admin.ping_test);
  ```

  If you want to disable the end request test, then use a SQL statement similar to the following:

  ```
  SQL> execute
  dbms_app_cont_admin.disable_connection_test(dbms_app_cont_admin.endrequest_
  test);
  ```


Copyright © 1999, 2025, Oracle and/or its affiliates.

- You can enable a SQL connection test after you disable it by logging on to the Pluggable Database and using a SQL statement similar to the following:

  ```
  SQL> execute
  dbms_app_cont_admin.enable_connection_test(dbms_app_cont_admin.sql_test,'se
  lect dummy from dual');
  ```

  You can also enable the ping test and end request test if they are disabled by using either of the following SQL statements:

  If you want to run any test that uses ping such as `isValid`, `isUsable`, `OCIping`, or `connection.status`, then use a SQL statement similar to the following:

  ```
  SQL> execute
  dbms_app_cont_admin.enable_connection_test(dbms_app_cont_admin.ping_test);
  ```

  If you want to enable draining at the end of a request, then use a SQL statement similar to the following:

  ```
  SQL> execute
  dbms_app_cont_admin.enable_connection_test(dbms_app_cont_admin.endrequest_t
  est);
  ```

  If you want to disable draining on the end of a request, then use a SQL statement similar to the following:

  ```
  SQL> execute
  dbms_app_cont_admin.disable_connection_test(dbms_app_cont_admin.endrequest_
  test);
  ```

- You can remove a SQL connection test if it is not needed by logging on to the Pluggable Database and running SQL statements similar to the following:

  ```
  SQL> execute dbms_app_cont_admin.delete_sql_connection_test('select dummy
  from dual','sw_orcl');
  SQL> execute dbms_app_cont_admin.delete_sql_connection_test('begin
  null;end;');
  ```

- If you want to modify rules for a service that belongs to a specific PDB, then switch to that PDB and modify the rules. For example, for eBusiness Suite:

  ```
  SQL> alter session set container='VISPRD';
  SQL> execute dbms_app_cont_admin.add_sql_connection_test('Begin null;
  End ;');
  SQL> execute dbms_app_cont_admin.add_sql_connection_test('Begin null;
  End ;', 'VISPRD');
  ```

- If you want to enable XA transactions start test, then use a SQL statement similar to the following:

  ```
  SQL> execute
  dbms_app_cont_admin.enable_connection_test(dbms_app_cont_admin.xa_start_tes
  t);
  ```


Copyright © 1999, 2025, Oracle and/or its affiliates.

- If you want to disable XA transactions start test, then use a SQL statement similar to the following:

```
SQL> execute
dbms_app_cont_admin.disable_connection_test(dbms_app_cont_admin.xa_start_te
st);
```

- If you want to enable sessionless transactions start test, then use a SQL statement similar to the following:

```
SQL> execute
dbms_app_cont_admin.enable_connection_test(dbms_app_cont_admin.sessionless_
start_test);
```

- If you want to disable sessionless transactions start test, then use a SQL statement similar to the following:

```
SQL> execute
dbms_app_cont_admin.disable_connection_test(dbms_app_cont_admin.sessionless
_start_test);
```

Every application server has a feature to test the validity of the connections in their respective connection pools, which is set either by a configuration property or at the administrative console. The purpose of the test is to prevent vending an unusable connection to an application, and when an unusable connection is detected, to remove it when released to the pool.

Across the various application servers, the tests have similar names. The tests offered use various approaches, the most common being a SQL statement. Oracle recommends that Java application servers use the standard Java call `connection.isValid`. Beginning with Oracle Database 18c, these tests are used to drain the database. Also beginning with Oracle Database 18c, the database drains sessions without using FAN by inspecting sessions for safe draining points.

The following table describes the standard connection tests available for several of the more common application servers:

Table 6-4 Standard Connection Tests for Some Common Application Servers

Application Server	Connection Test to Database
Oracle WebLogic Server	The tests offered include: • `dbms_app_cont_admin.enable_connection_test(dbms_app_cont_admin.sql_test,'select 1 from dual');` • `TestConnectionsonReserve`: `isUsable`, `isValid`, or `PingDatabase` • `TestConnectionsOnCreate` (SQL syntax) for server draining: `Select 1 from dual;`

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table 6-4 (Cont.) Standard Connection Tests for Some Common Application Servers

Application Server	Connection Test to Database
Oracle WebLogic Server Active Gridlink	The test is embedded: `isUsable`

IBM WebSphere	`dbms_app_cont_admin.enable_connection_test(dbms_app_cont_admin.sql_test,'select 1 from dual');`
	
	Pretest connections (SQL syntax) for server draining:
	
	`Select 1 from dual;`

RedHat JBoss	`check-valid-connection-sql` (SQL syntax): `dbms_app_cont_admin.enable_connection_test(dbms_app_cont_admin.sql_test,'select 1 from dual');`

Apache Tomcat	There are two tests available—`testOnBorrow` and `testOnReturn`—and they both use SQL syntax to test the connection to the database:
	
	`dbms_app_cont_admin.enable_connection_test(dbms_app_cont_admin.sql_test,'select 1 from dual');`
	
	Application server uses:
	
	`Select 1 from dual;`

Oracle recommends that you use the following format for supporting automatic configuration of Oracle Notification Services (ONS), so that you can receive FAN events (over ONS):

Example 6-6 Automatic Configuration of FAN

```
alias =(DESCRIPTION =
 (CONNECT_TIMEOUT=90)(RETRY_COUNT=20)(RETRY_DELAY=3)
(TRANSPORT_CONNECT_TIMEOUT=3)
   (ADDRESS_LIST =
   (LOAD_BALANCE=on)
   ( ADDRESS = (PROTOCOL = TCP)(HOST=primary-scan)(PORT=1521)))
   (ADDRESS_LIST =
   (LOAD_BALANCE=on)
   ( ADDRESS = (PROTOCOL = TCP)(HOST=secondary-scan)(PORT=1521)))
  (CONNECT_DATA=(SERVICE_NAME = gold-cloud)))
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Related Topics

- Planned Maintenance Without User Interruption
  Follow the planned maintenance best practices to avoid the need to restart your application servers.

Related Topics

- Transparent Application Continuity
  A version of Application Continuity for simple applications that uses discovery to detect request boundaries.

Planned Failover with Application Continuity

Planned Failover is a failover that is invoked by Oracle AI Database at points where the database knows that the session is replayable using Application Continuity and that the session is expected not to drain.

Planned failover is an automatic solution that is used for relocating sessions during planned maintenance for batch and long running operations that are not expected to complete in the specified drain timeout period. Session migration is also used with `ALTER SYSTEM DISCONNECT SESSION` statement.

Planned failover is activated when draining starts. A rules engine decides when to invoke a failover.

- The database maintains statistics regarding the rate and size of requests, and when a failover feature such as Application Continuity is enabled, then the database also maintains statistics for the level of protection for calls for replay, and the session state that needs to be recovered.
- The database knows when Transparent Application Continuity or Application Continuity are enabled on a session and whether session state is tracked and recoverable, and when failover is enabled, if that failover is likely to be disabled before the drain timeout expires.
- The database knows when a session is not expected to drain, and that a session is likely to recover, and how much replay it would need to run if it needs to replay.
- The database knows if Fast Application Notification is enabled.
- The database knows when request boundaries are discovered for Transparent Application Continuity.

A session failed over by the database is marked in the alert log so that you can find more information about the failed over session.

To use planned failover, follow these steps:

1. Enable Application Continuity or Transparent Application Continuity.
2. Set the service attributes `-drain_timeout` and `-stopoption` on your services.
3. During maintenance, drain your services by relocating or stopping them. For Data Guard, you can use switchover wait with Data Guard Broker.

Transaction Guard for Improving Client Failover

Transaction Guard is a developer feature that your applications can use to determine `COMMIT_OUTCOME` following recoverable errors.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- About Transaction Guard
  Transaction Guard provides a fully integrated tool for applications to use to achieve idempotence automatically and transparently, and in a manner that scales.
- Database Native Transaction Guard
  Database Native Transaction Guard enhances the existing Transaction Guard protocols to use the database Transaction ID (DB XID) to impose at-most-once processing.
- Transaction Guard Support During Major Database Version Upgrades
  Starting with Oracle AI Database 26ai, Transaction Guard works during `DBMS_ROLLING` operations to ensure continuous application functions during switchover, issued by `DBMS_ROLLING` to Transient Logical Standby.
- Transaction Guard Configuration Checklist
  Oracle recommends that you use this configuration checklist before you configure services for Transaction Guard.
- Configuring Services for Transaction Guard
  To configure services to use Transaction Guard, review and set the required service parameters.

About Transaction Guard

Transaction Guard provides a fully integrated tool for applications to use to achieve idempotence automatically and transparently, and in a manner that scales.

Transaction Guard uses Logical Transaction ID (LTXID) to avoid submitting duplicate transactions. This function is referred to as **transaction idempotence**. The LTXID persists on commit, and is reused following a rollback. During normal runtime, a LTXID is automatically held in the session at both the client and server for each database transaction. At commit, the LTXID is persisted as part of committing the transaction and the next LTXID to use is returned to the client.

> **Note**
>
> Application Continuity protects outages only for normal applications. Administrative tasks through AC-enabled services are not supported and you must avoid them.

Applications have a problem if they fail to recognize that the last submission has committed, or that it will commit sometime soon, or that the last submission has not run to completion. Applications failing to recognize these submission states can result in users who resubmit or applications that use their own replay to issue duplicate requests, repeating changes that are already committed to the database, and other forms of logical corruption. Transaction Guard can be used to solve this problem.

Application Continuity automatically enables and uses Transaction Guard, but you can also enable Transaction Guard independently. If the application has implemented an application-level replay, then it requires the application to be integrated with Transaction Guard to provide idempotence.

Transaction Guard for XA Transactions

Transaction Guard also supports XA-based transactions, which are transactions that are an option for transaction managers, such as Oracle WebLogic Server, Oracle Tuxedo, and Microsoft Transaction Server (exposed to Oracle AI Database through Oracle ODP.NET) when using one-phase commit with XA.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Related Topics

- *Oracle AI Database Development Guide*
- *Oracle AI Database Development Guide*
- *Oracle AI Database JDBC Developer's Guide*

Database Native Transaction Guard

Database Native Transaction Guard enhances the existing Transaction Guard protocols to use the database Transaction ID (DB XID) to impose at-most-once processing.

A server-side transaction is identified by a database transaction identifier or DB XID. Persisting the LTXID in the LTXID_TRANS table as part of the commit of each DB XID introduces overheads in normal transaction operation. This overhead with respect to performance and redo generation is visible in CPU overhead in the case of small transactions because the extra work required by Transaction Guard is not insignificant relative to the work done by the user transaction.

> **Note**
>
> Database Native Transaction Guard is the default option when you use Application Continuity.

The DB XID is unique within a Pluggable Database (PDB) with local undo and uniquely identifies a transaction in the database. Using the DB XID does not require persistence in a separate table, thus does not incur extra redo generation or performance overheads. The `FORCE_OUTCOME` procedure takes advantage of the DB XID, instead of relying on the persistence of the LTXID in the separate table.

Database Native Transaction Guard provides the Transaction Guard functionality with lower overheads by avoiding extra writes to the `LTXID_TRANS` table. This feature does not affect any client-side APIs and does not require administrative changes.

By default, the `COMMIT_OUTCOME_FAST_PATH` service parameter is set to `TRUE` if `COMMIT_OUTCOME` is set to `TRUE`. Set the `COMMIT_OUTCOME_FAST_PATH` service parameter to `FALSE` to disable Database Native Transaction Guard.

Transaction Guard Support During Major Database Version Upgrades

Starting with Oracle AI Database 26ai, Transaction Guard works during `DBMS_ROLLING` operations to ensure continuous application functions during switchover, issued by `DBMS_ROLLING` to Transient Logical Standby.

Transaction Guard returns the commit outcome of the current in-flight transaction when an error or outage occurs. Applications embed the Transaction Guard APIs in their error handling procedures to ensure that work continues without any in-flight work lost or duplicate submissions after an outage. Transaction Guard provides idempotence support to ensure that a commit occurs not more than once when a transaction is re-processed (replay) after an outage.

Transaction Guard ensures continuous application operation during the `DBMS_ROLLING` switchover operation to Transient Logical Standby. Transaction Guard ensures that the last commit outcome of transactions in the in-flight sessions during a switchover outage is used to protect the applications from duplicate submissions of the transactions on replay.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Transaction Guard maintains a transaction history table called LTXID_TRANS that has the mapping of logical transaction identifiers (LTXIDs) to database transactions. For a failover to succeed after an outage, the changes to LTXID_TRANS from the primary database must first replicate and apply to Transient Logical Standby. With supplemental logging enabled for the DBMS_ROLLING procedure, Transaction Guard uses SQL to allow supplemental capture of LTXID_TRANS at CDB and PDB levels. The capture process replicates the LTXID_TRANS table and the apply process reads and recreates the LTXID_TRANS tables for the logical standby, along with the committed user transactions.

As a part of its support for the DBMS_ROLLING procedure, Transaction Guard performs the following functions:

- Tracks when the primary database is in DBMS_ROLLING mode (when the database upgrade is initiated)
- Checks that supplemental logging is in use
- Records the redo vector for the primary key (PK) at runtime while in supplemental logging mode
- Waits for all current updates to finish and replicate to the logical standby before performing the LTXID replication
- Replicates the LTXID_TRANS tables and applies the redo to Transient Logical Standby for each PDB
- Provides a mechanism for failover to know about successful LTXID replication
- Enforces last commit outcome for inflight sessions on replay after an outage
- Handles new users during supplemental capture and apply process to ensure that any apply does not create mismatched logged-in UIDs (user IDs) at the target database

Transaction Guard Configuration Checklist

Oracle recommends that you use this configuration checklist before you configure services for Transaction Guard.

Before configuring services for Transaction Guard, complete each check on this list:

- Grant permission to the application user who will call GET_LTXID_OUTCOME, as follows:

  ```
  GRANT EXECUTE ON DBMS_APP_CONT to user_name;
  ```

  > **Note**
  >
  > *Do not* run this statement if you use Application Continuity.

- Locate and define the transaction history table for optimal performance.

  The transaction history table (LTXID_HIST) is created, by default, in the SYSAUX tablespace when you create or upgrade an Oracle AI Database. New partitions are added when you add instances, using the storage of the last partition. If the location of transaction history table is not optimal for performance, then you can move it to another tablespace

Copyright © 1999, 2025, Oracle and/or its affiliates.

and create partitions there. For example, the following statement moves the transaction history table to a tablespace named FastPace:

```
ALTER TABLE LTXID_TRANS move partition LTXID_TRANS_1 tablespace FastPace
   storage ( initial 10G next 10G minextents 1 maxextents 121 );
```

- Set values for the `COMMIT_OUTCOME` and `RETENTION_TIMEOUT` service parameters.
- If you are using Oracle Real Application Clusters (Oracle RAC), Oracle Data Guard, or Oracle Active Data Guard, then to obtain rapid notification of an outage, Oracle recommends that you use Fast Application Notification (FAN).

Configuring Services for Transaction Guard

To configure services to use Transaction Guard, review and set the required service parameters.

Review and set the following parameters:

- `COMMIT_OUTCOME`: Set the `COMMIT_OUTCOME` service parameter to `TRUE`.

  The `COMMIT_OUTCOME` service parameter determines whether the transaction **commit outcome** is accessible after the `COMMIT` has run and an outage has occurred. While Oracle AI Database has always made COMMIT durable, Transaction Guard makes the outcome of the `COMMIT` durable. Applications use this durability of a commit to enforce the status of the last transaction run before an outage.
- `RETENTION_TIMEOUT`: Use the `RETENTION_TIMEOUT` service parameter with `COMMIT_OUTCOME`. The `RETENTION_TIMEOUT` service parameter determines the amount of time, in seconds, that the `COMMIT` outcome is retained. Oracle recommends that most installations use the default value.

In the following example, the `SRVCTL` command configures a service named `sales` for Transaction Guard:

```
$ srvctl add service -db crm -service sales -preferred crm_1,crm_2
  -available crm_3,crm_4 -commit_outcome TRUE -retention 86400
  -notification TRUE
```

You can also modify an existing service to configure it for Transaction Guard by using the `srvctl modify service` command.

> **Note**
>
> Do not use the default database service, the service which has the name set to the value of `db_name` or `db_unique_name`. The default service is used for administrative purposes and does not have the same properties as user-created services.

Related Topics

- srvctl add service
  Adds services to a database and assigns them to instances.
- srvctl modify service
  Modifies a service configuration.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- About Application Continuity
  Application Continuity provides high availability for your application workloads.
- *Oracle AI Database JDBC Developer's Guide*
- *Oracle Call Interface Developer's Guide*

Application Continuity During Major Database Version Upgrades with DBMS_ROLLING

Starting with Oracle AI Database 26ai, Application Continuity and draining of database sessions are supported during major database version upgrades and database restructuring when using Application Continuity with `DBMS_ROLLING`.

Application Continuity with `DBMS_ROLLING` has full support within Oracle AI Database 26ai. This is the strategic approach for applying non-rolling patches. This feature also supports the upgrade from one major release to another major release. For the major release upgrade from Oracle Database 19c to Oracle AI Database 26ai, it is supported for Oracle Autonomous Database - Dedicated ***only***.

Applications are continuously available during `DBMS_ROLLING` when you use Application Continuity. This means that your applications can continue through the database upgrade process.

`DBMS_ROLLING` support for Application Continuity enables transparent failover of database sessions across different major versions of a database so that the interruption is minimized during major database upgrades and restructuring of that database. This feature relocates the database sessions in real time to a different logical version of the database, where it is the same database in terms of data but is a different major version or major database restructuring.

This feature provides failover of the application during the transition period, termed switchover, to the new logical database. Although the transition to the new database is a planned maintenance operation, but the applications are unaware of when the switchover step will occur to the new logical database. The infrastructure monitors and starts the switchover automatically when the lag between the databases is sufficiently small enough for the switchover to run in low numbers of seconds to minutes.

The database sessions failover to the new primary database thereby keeping the application operational. The failover minimizes the interruption and thus enable business continuity during major upgrades and major restructures. This feature integrates support for Oracle Application Continuity with a Logical Standby when using `DBMS_ROLLING`.

Reset Database Session State

`RESET_STATE` is an Oracle-exclusive feature that automatically prevents session states from leaking between application usage.

> **Note**
>
> Starting with Oracle AI Database 26ai, `RESET_STATE` does not have a dependency on the Application Continuity feature.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Setting session state in a request leaves the session with the current state, meaning that subsequent usages of the same session can see the previously set state. For example, when an application borrows and returns a connection to a connection pool, if the sessions state is not cleared, the next user of that connection can see the session state left behind by the previous request. Leaving session state in a session after it is returned to a pool creates a vulnerability for unintended or malicious use of the leftover state.

Consider pooled applications where multiple users share the same connection. Application states such as Local Application Context may store sensitive information like routing numbers, account numbers, addresses, or medical and healthcare details.

`RESET_STATE` is a critical database feature that ensures developers can rely on a clean session state whenever a session is returned to a connection pool with request boundaries. This applies to both Oracle connection pools and custom connection pools with added request boundaries. `RESET_STATE` guarantees that the Oracle AI Database clears session states.

Additionally, `RESET_STATE` enhances protection when using Transparent Application Continuity (TAC), as it ensures the session state is clean at the beginning of each new request.

The `RESET_STATE` service property also improves protection when using Transparent Application Continuity because the session state is clean at the beginning of the next request.

> **Note**
>
> `RESET_STATE` supersedes `RESET_PACKAGE`.

Related Topics

- ENABLE_RESET_STATE Procedure

Copyright © 1999, 2025, Oracle and/or its affiliates.

7

Configuring Recovery Manager and Archiving

You can configure Oracle Recovery Manager (Oracle RMAN) to support your Oracle RAC environment.

This chapter provides procedures for using for archiving in Oracle RAC environments and discusses online redo log and archived redo log considerations.

- Overview of Configuring RMAN for Oracle RAC
  Oracle Recovery Manager (Oracle RMAN) enables you to back up, restore, and recover files and archived logs.
- Archiving Mode in Oracle RAC
  To archive redo log files, the Oracle RAC database must be in `ARCHIVELOG` mode.
- Configuring the RMAN Snapshot Control File Location
  The snapshot control file is a copy of a database control file that RMAN creates in an operating system-specific location.
- Configuring RMAN to Automatically Backup the Control File and SPFILE
  If you set `CONFIGURE CONTROLFILE AUTOBACKUP` to `ON`, then RMAN automatically creates a control file and an SPFILE backup after you run the `BACKUP` or `COPY` commands.
- Crosschecking on Multiple Oracle RAC Nodes
  When crosschecking on multiple nodes, and when operating RMAN in general, configure your cluster so that every node can access all of the backups, regardless of which node created the backups.
- Configuring Channels for RMAN in Oracle RAC
  Learn how to configure channels for RMAN in Oracle RAC.
- Managing Archived Redo Logs Using RMAN in Oracle RAC
  Learn about managing archived redo logs using RMAN in Oracle RAC.
- Archived Redo Log File Conventions in Oracle RAC
  For any archived redo log configuration, uniquely identify the archived redo logs with the `LOG_ARCHIVE_FORMAT` parameter.
- RMAN Archiving Configuration Scenarios
  Learn about the various RMAN archiving scenarios.
- Monitoring the Archiver Processes
  Learn how to monitor the archiver processes.

Overview of Configuring RMAN for Oracle RAC

Oracle Recovery Manager (Oracle RMAN) enables you to back up, restore, and recover files and archived logs.

RMAN enables you to back up, restore, and recover data files, control files, server parameter files (SPFILEs) and archived redo log files. RMAN is included with Oracle AI Database and does not require separate installation. You can run RMAN from the command line or use RMAN in the Backup Manager in Oracle Enterprise Manager.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Archiving Mode in Oracle RAC

To archive redo log files, the Oracle RAC database must be in `ARCHIVELOG` mode.

You can run the `ALTER DATABASE` SQL statement to change the archiving mode in Oracle RAC, because the database is mounted by the local instance but not open in any instances. You do not need to modify parameter settings to run this statement.

> **Note**
>
> - The `ARCHIVELOG` mode is set at the database level, not the instance level. Either all instances archive or none do.
> - You can also change the archive log mode by using the Recovery Settings page in the **Maintenance** tab of the Oracle Enterprise Manager Oracle RAC Database Home Page.

Related Topics

- Managing Archived Redo Log Files

Configuring the RMAN Snapshot Control File Location

The snapshot control file is a copy of a database control file that RMAN creates in an operating system-specific location.

RMAN creates the snapshot control file so that it has a consistent version of a control file to use when either resynchronizing the recovery catalog or backing up the control file.

For effective backup and recovery operations, the RMAN snapshot control file must be on shared storage that is accessible by all database nodes in a cluster. Run the following RMAN command to determine the configured location of the snapshot control file:

```
SHOW SNAPSHOT CONTROLFILE NAME;
```

You can change the configured location of the snapshot control file. For example, on Linux and UNIX systems you can specify the snapshot control file location in Oracle ASM, for example, by entering the following at the RMAN prompt:

```
CONFIGURE SNAPSHOT CONTROLFILE NAME TO '+RECODG/Oracle SID/snap_prod.cf';
```

This command sets the configuration for the location of the snapshot control file for every instance of your cluster database. Therefore, ensure that the directory location is shared by all nodes that perform backups.

The `CONFIGURE` command creates persistent settings across RMAN sessions. Therefore, you do not need to run this command again unless you want to change the location of the snapshot control file.

Copyright © 1999, 2025, Oracle and/or its affiliates.

To delete a snapshot control file you must first change the snapshot control file location, then delete the file at the older location, as follows:

```
CONFIGURE SNAPSHOT CONTROLFILE NAME TO 'new_name';
DELETE COPY OF CONTROLFILE;
```

Related Topics

- RMAN Commands: @ (at sign) to QUIT

Configuring RMAN to Automatically Backup the Control File and SPFILE

If you set `CONFIGURE CONTROLFILE AUTOBACKUP` to `ON`, then RMAN automatically creates a control file and an SPFILE backup after you run the `BACKUP` or `COPY` commands.

RMAN can also automatically restore an SPFILE, if this is required to start an instance to perform recovery, because the default location for the SPFILE must be available to all nodes in your Oracle RAC database.

> **Note**
>
> If you back up the control file using the SQL*Plus `ALTER DATABASE` command, then you must also create the control file backup on a device shared by all nodes.

These features are important in disaster recovery because RMAN can restore the control file even without a recovery catalog. RMAN can restore an autobackup of the control file even after the loss of both the recovery catalog and the current control file. You can change the default name that RMAN gives to this file with the `CONFIGURE CONTROLFILE AUTOBACKUP FORMAT` command. Note that if you specify an absolute path name in this command, then this path must exist identically on all nodes that participate in backups.

RMAN performs the control file autobackup on the first allocated channel. Therefore, when you allocate multiple channels with different parameters, especially when you allocate a channel with the `CONNECT` command, determine which channel will perform the control file autobackup. Always allocate the channel for this node first.

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

Related Topics

- *Oracle AI Database Backup and Recovery User's Guide*

Crosschecking on Multiple Oracle RAC Nodes

When crosschecking on multiple nodes, and when operating RMAN in general, configure your cluster so that every node can access all of the backups, regardless of which node created the backups.

When the cluster is configured this way, you can allocate channels to any node in the cluster during restore or crosscheck operations.

Copyright © 1999, 2025, Oracle and/or its affiliates.

If you cannot configure the cluster so that each node can access all backups, then during restore and crosscheck operations, you must allocate channels on multiple nodes by providing the `CONNECT` option to the `CONFIGURE CHANNEL` command, so that every backup can be accessed by at least one node. If some backups are not accessible during crosscheck because no channel was configured on the node that can access those backups, then those backups are marked `EXPIRED` in the RMAN repository after the crosscheck.

For example, you can use `CONFIGURE CHANNEL ... CONNECT` in an Oracle RAC configuration in which tape backups are created on various nodes in the cluster and each backup is only accessible on the node on which it is created.

Related Topics

- *Oracle AI Database Backup and Recovery User's Guide*

Configuring Channels for RMAN in Oracle RAC

Learn how to configure channels for RMAN in Oracle RAC.

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

- Configuring Channels to Use Automatic Load Balancing
  Learn how to configure channels to use automatic load balancing.

Configuring Channels to Use Automatic Load Balancing

Learn how to configure channels to use automatic load balancing.

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

```
CONFIGURE DEVICE TYPE [disk | sbt] PARALLELISM number_of_channels;
...
```

Where *number_of_channels* is the number of channels that you want to use for the operation. After you complete this one-time configuration, you can run the `BACKUP` or `RESTORE` commands.

Managing Archived Redo Logs Using RMAN in Oracle RAC

Learn about managing archived redo logs using RMAN in Oracle RAC.

When a node generates an archived redo log, Oracle AI Database always records the file name of the log in the control file of the target database. If you are using a recovery catalog, then RMAN also records the archived redo log filenames in the recovery catalog when a resynchronization occurs.

The archived redo log naming scheme that you use is important because when a node writes to a log with a specific file name on its file system, the file must be readable by any node that must access this archived redo log. For example, if `node1` archives a log to `/oracle/arc_dest/log_1_100_23452345.arc`, then `node2` can back up this archived redo log only if it can read `/oracle/arc_dest/log_1_100_23452345.arc` on its own file system.

The backup and recovery strategy that you choose depends on how you configure the archiving destinations for each node. Whether only one node or all nodes perform archived redo log backups, you must ensure that all archived redo logs are backed up. If you use RMAN parallelism during recovery, then the node that performs recovery must have read access to all archived redo logs in your cluster.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Multiple nodes can restore archived logs in parallel. However, during recovery, only one node applies the archived logs. Therefore, the node that is performing the recovery must be able to access all of the archived logs that are needed for the recovery operation. By default, the database determines the optimum number of parallel threads to use during the recovery operation. You can use the `PARALLEL` clause in the `RECOVER` command to change the number of parallel threads.

Guidelines and Considerations for Archived Redo Logs

The primary consideration is to ensure that all archived redo logs can be read from every node during recovery, and, if possible, during backups. During recovery, if the archived log destinations are visible from the node that performs the recovery, then Oracle AI Database can successfully recover the archived log data.

Archived Redo Log File Conventions in Oracle RAC

For any archived redo log configuration, uniquely identify the archived redo logs with the `LOG_ARCHIVE_FORMAT` parameter.

The format of this parameter is operating system-specific and the format can include text strings, one or more variables, and a file name extension.

Table 7-1 Archived Redo Log File Name Format Parameters

Parameter	Description	Example
%r	Resetlogs identifier, not padded	log_1_62_23452345
%R	Resetlogs identifier, left-zero-padded	log_1_62_0023452345
%s	Log sequence number, not padded	log_251
%S	Log sequence number, left-zero-padded	log_0000000251
%t	Thread number, not padded	log_1
%T	Thread number, left-zero-padded	log_0001

All of the file name format parameters for the archive redo logs, in either upper or lowercase, are mandatory for Oracle RAC. These parameters enable Oracle AI Database to create unique names for archive logs across the incarnation. This requirement is in effect when the `COMPATIBLE` parameter is set to `10.0` or greater.

Use the `%R` or `%r` parameters to include the `resetlogs` identifier to avoid overwriting the logs from a previous incarnation. If you do not specify a log format, then the default is operating system-specific and includes `%t`, `%s`, and `%r`.

As an example, if the instance associated with redo thread number 1 sets `LOG_ARCHIVE_FORMAT` to `log_%t_%s_%r.arc`, then its archived redo log files are named:

```
log_1_1000_23435343.arc
log_1_1001_23452345.arc
log_1_1002_23452345.arc
...
```

Related Topics

- *Oracle AI Database Administrator's Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

RMAN Archiving Configuration Scenarios

Learn about the various RMAN archiving scenarios.

This section describes the archiving scenarios for an Oracle RAC database. The two configuration scenarios in this chapter describe a three-node UNIX cluster for an Oracle RAC database. For both scenarios, the `LOG_ARCHIVE_FORMAT` that you specify for the instance performing recovery must be the same as the format that you specified for the instances that archived the redo log files.

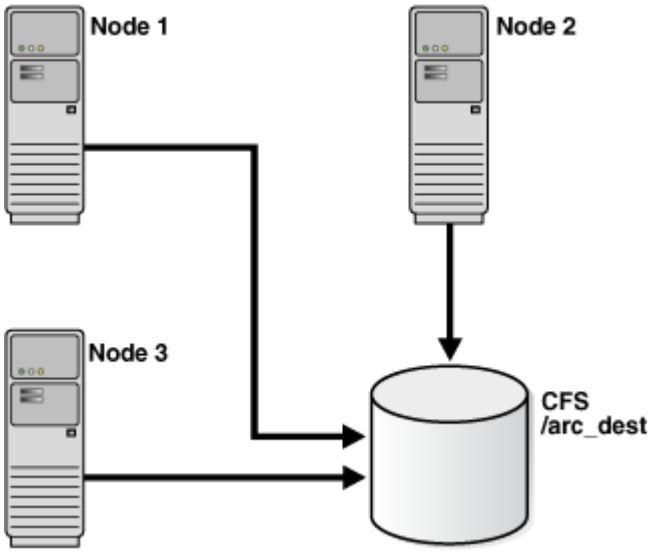
- Oracle Advanced Cluster File System Archiving Scheme
  The preferred configuration for Oracle RAC is to use Oracle Automatic Storage Management (Oracle ASM) for a recovery area using a disk group for your recovery set that is different from the disk group used for your data files.
- Noncluster File System Local Archiving Scheme
  Learn about the noncluster file system local archiving scheme.

Oracle Advanced Cluster File System Archiving Scheme

The preferred configuration for Oracle RAC is to use Oracle Automatic Storage Management (Oracle ASM) for a recovery area using a disk group for your recovery set that is different from the disk group used for your data files.

When you use Oracle ASM, it uses an Oracle Managed Files naming format. Alternatively, you can use a cluster file system archiving scheme. If you use a cluster file system, then each node writes to a single location on the cluster file system when archiving the redo log files. Each node can read the archived redo log files of the other nodes. For example, as shown in the following image, if Node 1 archives a redo log file to `/arc_dest/log_1_100_23452345.arc` on the cluster file system, then any other node in the cluster can also read this file.

Figure 7-1 Cluster File System Archiving Scheme



Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> The archive log naming format in this example is only for a cluster file system example.

If you do not use a cluster file system, then the archived redo log files cannot be on raw devices. This is because raw devices do not enable sequential writing of consecutive archive log files.

- Advantages of the Cluster File System Archiving Scheme
  The main advantage of the cluster file system arcvhiving scheme is that any node can read the achive logs.
- Initialization Parameter Settings for the Cluster File System Archiving Scheme
  Learn about parameter settings for cluster file system archiving.
- Location of Archived Logs for the Cluster File System Archiving Scheme
  Any node can read the archive logs, regarless of which node created the logs.

Related Topics

- *Oracle AI Database Backup and Recovery User's Guide*

Advantages of the Cluster File System Archiving Scheme

The main advantage of the cluster file system arcvhiving scheme is that any node can read the achive logs.

The advantage of this scheme is that none of the nodes uses the network to archive logs. Because the file name written by a node can be read by any node in the cluster, RMAN can back up all logs from any node in the cluster. Backup and restore scripts are simplified because each node has access to all archived redo logs.

Initialization Parameter Settings for the Cluster File System Archiving Scheme

Learn about parameter settings for cluster file system archiving.

In the cluster file system scheme, each node archives to a directory that is identified with the same name on all instances within the cluster database (`/arc_dest`, in the following example). To configure this directory, set values for the `LOG_ARCH_DEST_1` parameter, as shown in the following example:

```
*.LOG_ARCHIVE_DEST_1="LOCATION=/arc_dest"
```

The following list shows archived redo log entry examples that would appear in the RMAN catalog or in the control file based on the previous example. Note that any node can archive logs using any of the threads:

```
/arc_dest/log_1_999_23452345.arc
/arc_dest/log_1_1000_23435343.arc
/arc_dest/log_1_1001_23452345.arc <- thread 1 archived in node3
/arc_dest/log_3_1563_23452345.arc <- thread 3 archived in node2
/arc_dest/log_2_753_23452345.arc <- thread 2 archived in node1
/arc_dest/log_2_754_23452345.arc
/arc_dest/log_3_1564_23452345.arc
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Location of Archived Logs for the Cluster File System Archiving Scheme

Any node can read the archive logs, regarless of which node created the logs.

Because the file system is shared and because each node is writing its archived redo logs to the `/arc_dest` directory in the cluster file system, each node can read the logs written by itself and any other node.

Noncluster File System Local Archiving Scheme

Learn about the noncluster file system local archiving scheme.

When archiving locally to a noncluster file system, each node archives to a uniquely named local directory. If recovery is required, then you can configure the recovery node so that it can access directories on the other nodes remotely. For example, use NFS on Linux and UNIX computers, or mapped drives on Windows systems. Therefore, each node writes only to a local destination, but each node can also read archived redo log files in remote directories on the other nodes.

- Considerations for Using Noncluster File System Local Archiving
  If you use noncluster file system local archiving for media recovery, then you must configure the node that is performing recovery for remote access to the other nodes so that the node can read the archived redo log files in the archive directories on the other nodes.
- Initialization Parameter Settings for Non-Cluster File System Local Archiving
  You can set the archiving destination values as follows in the initialization parameter file.
- Location of Archived Logs for Noncluster File System Local Archiving
  Learn about the location of archived logs for noncluster file system local archiving.
- File System Configuration for Noncluster File System Local Archiving
  Use NFS to perform recovery using a remaining instance to read logs that are not yet backed up.

Considerations for Using Noncluster File System Local Archiving

If you use noncluster file system local archiving for media recovery, then you must configure the node that is performing recovery for remote access to the other nodes so that the node can read the archived redo log files in the archive directories on the other nodes.

In addition, if you are performing recovery and you do not have all of the available archive logs, then you must perform an incomplete recovery up to the first missing archived redo log sequence number. You do not have to use a specific configuration for this scheme. However, to distribute the backup processing onto multiple nodes, the easiest method is to configure channels as described in the backup scenarios in "Managing Backup and Recovery".

> **Note**
>
> Because different file systems are used in a noncluster case, the archive log directories must be unique on each node. For example, `/arc_dest_1` is only available on `node1`, `/arc_dest_2` is only directly mounted on `node2`, and so on.
>
> Then `node1` mounts `/arc_dest_2` from `node2` and `/arc_dest_3` from `node3` through NFS.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Related Topics

- Managing Backup and Recovery
  Learn how to use Recovery Manager (RMAN) to back up and restore Oracle Real Application Clusters (Oracle RAC) databases, and about Oracle RAC instance recovery, parallel backup, recovery with SQL*Plus, and using the Fast Recovery Area in Oracle RAC.

Initialization Parameter Settings for Non-Cluster File System Local Archiving

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

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

Set the *SID*.LOG_ARCH_DEST parameter for each instance using the SID designator, as shown in the following example:

```
sid1.LOG_ARCHIVE_DEST_1="LOCATION=/arc_dest_1"
sid2.LOG_ARCHIVE_DEST_1="LOCATION=/arc_dest_2"
sid3.LOG_ARCHIVE_DEST_1="LOCATION=/arc_dest_3"
```

The following list shows the possible archived redo log entries in the database control file. Note that any node can read archived redo logs from any of the threads, which must happen in order for the database to recover after a failure.

```
/arc_dest_1/log_1_1000_23435343.arc
/arc_dest_2/log_1_1001_23452345.arc <- thread 1 archived in node2
/arc_dest_2/log_3_1563_23452345.arc <- thread 3 archived in node2
/arc_dest_1/log_2_753_23452345.arc <- thread 2 archived in node1
/arc_dest_2/log_2_754_23452345.arc
/arc_dest_3/log_3_1564_23452345.arc
```

Location of Archived Logs for Noncluster File System Local Archiving

Learn about the location of archived logs for noncluster file system local archiving.

As illustrated in the following table, each of three nodes has a directory containing the locally archived redo logs. Additionally, if you mount directories on the other nodes remotely through NFS or mapped drives, then each node has two remote directories through which RMAN can read the archived redo log files that are archived by the remaining nodes.

> **Note**
>
> The archive log destinations, similar to those shown in the following table, must be different on each node so that if you mount the NFS directory on a different node, then it does not conflict with an existing archive log directory

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table 7-2 UNIX/NFS Location Log Examples, Noncluster File System Local Archiving

Node	Reads the archived redo log files in the directory	For logs archived by node
1	/arc_dest_1	1
1	/arc_dest_2	2 (through NFS)
1	/arc_dest_3	3 (through NFS)
2	/arc_dest_1	1 (through NFS)
2	/arc_dest_2	2
2	/arc_dest_3	3 (through NFS)
3	/arc_dest_1	1 (through NFS)
3	/arc_dest_2	2 (through NFS)
3	/arc_dest_3	3

File System Configuration for Noncluster File System Local Archiving

Use NFS to perform recovery using a remaining instance to read logs that are not yet backed up.

If you are performing recovery and a remaining instance must read all of the logs that are on disk but not yet backed up, then you should configure NFS as shown in the following table.

Table 7-3 UNIX/NFS Configuration for Shared Read Local Archiving Examples

Node	Directory...	Is configured...	And mounted on...	On node...
1	/arc_dest_1	Local read/write	n/a	n/a
1	/arc_dest_2	NFS read	/arc_dest_2	2
1	/arc_dest_3	NFS read	/arc_dest_3	3
2	/arc_dest_1	NFS read	/arc_dest_1	1
2	/arc_dest_2	Local read/write	n/a	n/a
2	/arc_dest_3	NFS read	/arc_dest_3	3
3	/arc_dest_1	NFS read	/arc_dest_1	1
3	/arc_dest_2	NFS read	/arc_dest_2	2
3	/arc_dest_3	Local read/write	n/a	n/a

> **Note**
>
> Microsoft Windows users can achieve the same results depicted in the examples in this section by using mapped drives.

Monitoring the Archiver Processes

Learn how to monitor the archiver processes.

Copyright © 1999, 2025, Oracle and/or its affiliates.

After your RMAN configuration is operative in your Oracle RAC environment, use the `GV$ARCHIVE_PROCESSES` and `V$ARCHIVE_PROCESSES` views to determine the status of the archiver processes. Depending on whether you query the global or local views, these views display information for all database instances, or for only the instance to which you are connected.

> **Note**
>
> If you use the `kill` command to stop the archiver process, then the database instance will fail.

Related Topics

- *Oracle AI Database Administrator's Guide*
- *Oracle AI Database Reference*

Copyright © 1999, 2025, Oracle and/or its affiliates.

8

Managing Backup and Recovery

Learn how to use Recovery Manager (RMAN) to back up and restore Oracle Real Application Clusters (Oracle RAC) databases, and about Oracle RAC instance recovery, parallel backup, recovery with SQL*Plus, and using the Fast Recovery Area in Oracle RAC.

- Managing Backup and Recovery in Clusters
  In a cluster, any node in the cluster can restore archived redo log files.
- RMAN Backup Scenario for Noncluster File System Backups
  Learn about using RMAN for noncluster file system backups.
- RMAN Restore Scenarios for Oracle RAC
  Learn about the RMAN restore scenarios for Oracle RAC.
- Instance Recovery in Oracle RAC
  Learn about instance recovery in Oracle RAC.
- Media Recovery in Oracle RAC
  Learn about media recovery in Oracle RAC.
- Parallel Recovery in Oracle RAC
  Oracle AI Database automatically selects the optimum degree of parallelism for instance, crash, and media recovery.
- Using a Fast Recovery Area in Oracle RAC
  To use a fast recovery area in Oracle RAC, place the recovery area on an Oracle ASM disk group, on a Cluster File System, or on a shared directory that is configured through a network file system file for each Oracle RAC instance.

Managing Backup and Recovery in Clusters

In a cluster, any node in the cluster can restore archived redo log files.

For restore and recovery in Oracle Real Application Clusters (Oracle RAC) database environments, you do not have to configure the instance that performs the recovery to also be the sole instance that restores all of the data files. In Oracle RAC, data files are accessible from every node in the **cluster**, so any node can restore archived redo log files.

Related Topics

- Managing Oracle Cluster Registry and Voting Files

RMAN Backup Scenario for Noncluster File System Backups

Learn about using RMAN for noncluster file system backups.

In a noncluster file system environment, each node can back up only to a locally-mounted noncluster file system directory. For example, `node1` cannot access the archived redo log files on `node2` or `node3` unless you configure the network file system for remote access. If you configure a network file system file for backups, then each node backs up its archived redo logs to a local directory.

Copyright © 1999, 2025, Oracle and/or its affiliates.

RMAN Restore Scenarios for Oracle RAC

Learn about the RMAN restore scenarios for Oracle RAC.

- Restoring Backups from a Cluster File System
  Learn how to restore backups from cluster file systems.
- Restoring Backups from a Noncluster File System
  Learn how to restore backups from noncluster file systems.
- Using RMAN or Oracle Enterprise Manager to Restore the Server Parameter File (SPFILE)
  You can restore SPFILEs with RMAN or Oracle Enterprise Manager.

Restoring Backups from a Cluster File System

Learn how to restore backups from cluster file systems.

The scheme that this section describes assumes that you are using the "Oracle Advanced Cluster File System Archiving Scheme". In this scheme, assume that `node3` performed the backups to a cluster file system. If `node3` is available for the restore and recovery operation, and if all of the archived logs have been backed up or are on disk, then run the following commands to perform complete recovery:

```
RESTORE DATABASE;
RECOVER DATABASE;
```

If `node3` performed the backups but is unavailable, then configure a media management device for one of the remaining nodes and make the backup media from `node3` available to this node.

> **Note**
>
> If you configured RMAN as described in "Configuring Channels to Use Automatic Load Balancing", then, to load balance the channels across nodes, note that channels cannot be load balanced before at least one instance has successfully opened the database. This means that the channels will not be load balanced across the nodes during a full database restore. To achieve load balancing of channels for `RESTORE` and `RECOVER` commands, you can temporarily reallocate channels by running commands similar to the following:
>
> ```
> run {
> ALLOCATE CHANNEL DEVICE TYPE sbt C1 CONNECT '@racinst_1'
> ALLOCATE CHANNEL DEVICE TYPE sbt C2 CONNECT '@racinst_2'
> ...
> }
> ```

Related Topics

- Oracle Advanced Cluster File System Archiving Scheme
  The preferred configuration for Oracle RAC is to use Oracle Automatic Storage Management (Oracle ASM) for a recovery area using a disk group for your recovery set that is different from the disk group used for your data files.
- Configuring Channels to Use Automatic Load Balancing
  Learn how to configure channels to use automatic load balancing.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Restoring Backups from a Noncluster File System

Learn how to restore backups from noncluster file systems.

The scheme that this section describes assumes that you are using the "Noncluster File System Local Archiving Scheme". In this scheme, each node archives locally to a different directory. For example, `node1` archives to `/arc_dest_1`, `node2` archives to `/arc_dest_2`, and `node3` archives to `/arc_dest_3`. You must configure a network file system file so that the recovery node can read the archiving directories on the remaining nodes.

If all nodes are available and if all archived redo logs have been backed up, then you can perform a complete restore and recovery by mounting the database and running the following commands from any node:

```
RESTORE DATABASE;
RECOVER DATABASE;
```

Because the network file system configuration enables each node read access to the redo log files on other nodes, then the recovery node can read and apply the archived redo logs located on the local and remote disks. No manual transfer of archived redo logs is required.

Related Topics

- Noncluster File System Local Archiving Scheme
  Learn about the noncluster file system local archiving scheme.

Using RMAN or Oracle Enterprise Manager to Restore the Server Parameter File (SPFILE)

You can restore SPFILEs with RMAN or Oracle Enterprise Manager.

RMAN can restore the server parameter file either to the default location or to a location that you specify.

You can also use Oracle Enterprise Manager to restore the SPFILE. From the Backup/Recovery section of the **Maintenance** tab, click **Perform Recovery**. The Perform Recovery link is context-sensitive and navigates you to the SPFILE restore only when the database is closed.

Instance Recovery in Oracle RAC

Learn about instance recovery in Oracle RAC.

Instance failure occurs when software or hardware problems disable an instance. After instance failure, Oracle AI Database automatically uses the online redo logs to perform recovery as described in this section.

- Single Node Failure in Oracle RAC
  Learn about single node failures in Oracle RAC.
- Multiple-Node Failures in Oracle RAC
  Learn how to manage multi-node failures in Oracle RAC.
- Using RMAN to Create Backups in Oracle RAC
  Oracle AI Database provides RMAN for backing up and restoring the database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Channel Connections to Cluster Instances with RMAN
  Learn about using RMAN for channel connections to cluster instances.
- Node Affinity Awareness of Fast Connections
  Learn about fast connection node affinity awareness.
- Deleting Archived Redo Logs after a Successful Backup
  Learn how to delete archived redo logs after backups.
- Autolocation for Backup and Restore Commands
  Learn about autolocation for the backup and restore commands.

Single Node Failure in Oracle RAC

Learn about single node failures in Oracle RAC.

Instance recovery in Oracle RAC does not include the recovery of applications that were running on the failed instance. Oracle Clusterware restarts the instance automatically.

Applications that were running on a node before it failed continue running by using failure recognition and recovery. This provides consistent and uninterrupted service if hardware or software fails. When one instance performs recovery for another instance, the remaining instance reads online redo logs generated by the failed instance and uses that information to ensure that committed transactions are recorded in the database. Thus, data from committed transactions is not lost. The instance performing recovery rolls back transactions that were active at the time of the failure and releases resources used by those transactions.

> **Note**
>
> All online redo logs must be accessible for instance recovery. Therefore, Oracle recommends that you mirror your online redo logs.

Multiple-Node Failures in Oracle RAC

Learn how to manage multi-node failures in Oracle RAC.

When failures occur, if one instance remains active, then Oracle RAC performs instance recovery for any other instances that fail. If all instances of an Oracle RAC database fail, then Oracle AI Database automatically recovers the instances the next time one instance opens the database. The instance performing recovery can mount the database in either cluster database or exclusive mode from any node of an Oracle RAC database. This recovery procedure is the same for Oracle AI Database running in shared mode as it is for Oracle AI Database running in exclusive mode, except that one instance performs instance recovery for all of the failed instances.

Using RMAN to Create Backups in Oracle RAC

Oracle AI Database provides RMAN for backing up and restoring the database.

RMAN enables you to back up, restore, and recover data files, control files, SPFILEs, and archived redo logs. RMAN is included with the Oracle AI Database server and it is installed by default. You can run RMAN from the command line or you can use it from the Backup Manager in Oracle Enterprise Manager. In addition, RMAN is the recommended backup and recovery tool if you are using Oracle Automatic Storage Management (Oracle ASM). The procedures for

Copyright © 1999, 2025, Oracle and/or its affiliates.

using RMAN in Oracle RAC environments do not differ substantially from those for Oracle noncluster environments.

Related Topics

- *Oracle AI Database Backup and Recovery User's Guide*

Channel Connections to Cluster Instances with RMAN

Learn about using RMAN for 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 `dbauser/pwd@service_name`. If you configure the SQL Net service name with load balancing turned on, then the channels are allocated at a node as decided by the load balancing algorithm.

```
CONFIGURE DEVICE TYPE sbt PARALLELISM 3;
CONFIGURE DEFAULT DEVICE TYPE TO sbt;
CONFIGURE CHANNEL DEVICE TYPE SBT CONNECT 'dbauser/pwd@service_name'
```

However, if the service name used in the connect string is not for load balancing, then you can control at which instance the channels are allocated using separate connect strings for each channel configuration, as follows:

```
CONFIGURE DEVICE TYPE sbt PARALLELISM 3;
CONFIGURE CHANNEL 1.. CONNECT 'dbauser/pwd@mydb_1';
CONFIGURE CHANNEL 2.. CONNECT 'dbauser/pwd@mydb_2';
CONFIGURE CHANNEL 3.. CONNECT 'dbauser/pwd@mydb_3';
```

In the previous example, it is assumed that `mydb_1`, `mydb_2` and `mydb_3` are SQL*Net service names that connect to pre-defined nodes in your Oracle RAC environment. Alternatively, you can also use manually allocated channels to backup your database files. For example, the following command backs up the SPFILE, control file, data files and archived redo logs:

```
RUN
{
    ALLOCATE CHANNEL CH1 CONNECT 'dbauser/pwd@mydb_1';
    ALLOCATE CHANNEL CH2 CONNECT 'dbauser/pwd@mydb_2';
    ALLOCATE CHANNEL CH3 CONNECT 'dbauser/pwd@mydb_3';
    BACKUP DATABASE PLUS ARCHIVED LOG;
}
```

During a backup operation, if at least one channel allocated has access to the archived log, then RMAN automatically schedules the backup of the specific log on that channel. Because the control file, SPFILE, and data files are accessible by any channel, the backup operation of these files is distributed across the allocated channels.

For a local archiving scheme, there must be at least one channel allocated to all of the nodes that write to their local archived logs. For a cluster file system archiving scheme, if every node writes the archived logs in the same cluster file system, then the backup operation of the archived logs is distributed across the allocated channels.

Copyright © 1999, 2025, Oracle and/or its affiliates.

During a backup, the instances to which the channels connect must be either all mounted or all open. For example, if the instance on `node1` has the database mounted while the instances on `node2` and `node3` have the database open, then the backup fails.

Related Topics

- *Oracle AI Database Backup and Recovery Reference*

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

Node Affinity Awareness of Fast Connections

Learn about fast connection node affinity awareness.

In some cluster database configurations, some nodes of the cluster have faster access to certain data files than to other data files. RMAN automatically detects this situation, which is known as node affinity awareness. When deciding which channel to use to back up a particular data file, RMAN gives preference to the nodes with faster access to the data files that you want to back up. For example, if you have a three-node cluster, and if `node1` has faster read/write access to data files 7, 8, and 9 than the other nodes, then `node1` has greater node affinity to those files than `node2` and `node3`.

Deleting Archived Redo Logs after a Successful Backup

Learn how to delete archived redo logs after backups.

If you have configured the automatic channels as defined in section "Channel Connections to Cluster Instances with RMAN", then you can use the following example to delete the archived logs that you backed up *n* times. The device type can be `DISK` or `SBT`:

```
DELETE ARCHIVELOG ALL BACKED UP n TIMES TO DEVICE TYPE device_type;
```

During a delete operation, if at least one channel allocated has access to the archived log, then RMAN automatically schedules the deletion of the specific log on that channel. For a local archiving scheme, there must be at least one channel allocated that can delete an archived log. For a cluster file system archiving scheme, if every node writes to the archived logs on the same cluster file system, then the archived log can be deleted by any allocated channel.

If you have not configured automatic channels, then you can manually allocate the maintenance channels as follows and delete the archived logs.

```
ALLOCATE CHANNEL FOR MAINTENANCE DEVICE TYPE DISK CONNECT 'SYS/oracle@node1';
ALLOCATE CHANNEL FOR MAINTENANCE DEVICE TYPE DISK CONNECT 'SYS/oracle@node2';
ALLOCATE CHANNEL FOR MAINTENANCE DEVICE TYPE DISK CONNECT 'SYS/oracle@node3';
DELETE ARCHIVELOG ALL BACKED UP n TIMES TO DEVICE TYPE device_type;
```

Related Topics

- Channel Connections to Cluster Instances with RMAN
  Learn about using RMAN for channel connections to cluster instances.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Autolocation for Backup and Restore Commands

Learn about autolocation for the backup and restore commands.

RMAN automatically performs autolocation of all files that it must back up or restore. If you use the noncluster file system local archiving scheme, then a node can only read the archived redo logs that were generated by an instance on that node. RMAN never attempts to back up archived redo logs on a channel it cannot read.

During a restore operation, RMAN automatically performs the autolocation of backups. A channel connected to a specific node only attempts to restore files that were backed up to the node. For example, assume that log sequence 1001 is backed up to the drive attached to `node1`, while log 1002 is backed up to the drive attached to `node2`. If you then allocate channels that connect to each node, then the channel connected to `node1` can restore log 1001 (but not 1002), and the channel connected to `node2` can restore log 1002 (but not 1001).

Media Recovery in Oracle RAC

Learn about media recovery in Oracle RAC.

Media recovery must be user-initiated through a client application, whereas instance recovery is automatically performed by the database. In these situations, use RMAN to restore backups of the data files and then recover the database. The procedures for RMAN media recovery in Oracle RAC environments do not differ substantially from the media recovery procedures for noncluster environments.

The node that performs the recovery must be able to restore all of the required data files. That node must also be able to either read all of the required archived redo logs on disk or be able to restore them from backups.

When recovering a database with encrypted tablespaces (for example after a `SHUTDOWN ABORT` or a catastrophic error that brings down the database instance), you must open the Oracle Wallet after database mount and before you open the database, so the recovery process can decrypt data blocks and redo.

Parallel Recovery in Oracle RAC

Oracle AI Database automatically selects the optimum degree of parallelism for instance, crash, and media recovery.

Oracle AI Database applies archived redo logs using an optimal number of parallel processes based on the availability of CPUs. You can use parallel instance recovery and parallel media recovery in Oracle RAC databases as described under the following topics:

- Parallel Recovery with RMAN
  Learn how to use parallel recovery with RMAN.
- Disabling Parallel Recovery
  Learn how to disable parallel recovery.

Related Topics

- *Oracle AI Database Backup and Recovery User's Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

Parallel Recovery with RMAN

Learn how to use parallel recovery with RMAN.

With RMAN's `RESTORE` and `RECOVER` commands, Oracle AI Database automatically makes parallel the following three stages of recovery:

Restoring Data Files

When restoring data files, the number of channels you allocate in the RMAN recover script effectively sets the parallelism that RMAN uses. For example, if you allocate five channels, you can have up to five parallel streams restoring data files.

Applying Incremental Backups

Similarly, when you are applying incremental backups, the number of channels you allocate determines the potential parallelism.

Applying Archived Redo Logs

With RMAN, the application of archived redo logs is performed in parallel. Oracle AI Database automatically selects the optimum degree of parallelism based on available CPU resources.

Disabling Parallel Recovery

Learn how to disable parallel recovery.

You can override parallel recovery using the procedures under the following topics:

- Disabling Instance and Crash Recovery Parallelism
  Learn how to disable instance and crash recovery parallelism.
- Disabling Media Recovery Parallelism
  Learn how to disable media recovery parallelism.

Disabling Instance and Crash Recovery Parallelism

Learn how to disable instance and crash recovery parallelism.

To disable parallel instance and crash recovery on a system with multiple CPUs, set the `RECOVERY_PARALLELISM` parameter in the database initialization parameter file, SPFILE, to `0` or `1`.

Disabling Media Recovery Parallelism

Learn how to disable media recovery parallelism.

Use the `NOPARALLEL` clause of the RMAN `RECOVER` command or the `ALTER DATABASE RECOVER` statement to force Oracle AI Database to use non-parallel media recovery.

Using a Fast Recovery Area in Oracle RAC

To use a fast recovery area in Oracle RAC, place the recovery area on an Oracle ASM disk group, on a Cluster File System, or on a shared directory that is configured through a network file system file for each Oracle RAC instance.

Copyright © 1999, 2025, Oracle and/or its affiliates.

In other words, the fast recovery area must be shared among all of the instances of an Oracle RAC database. In addition, set the parameter `DB_RECOVERY_FILE_DEST` to the same value on all instances.

Oracle Enterprise Manager enables you to set up a fast recovery area. To use this feature:

1. From the Cluster Database home page, click the **Maintenance** tab.
2. Under the Backup/Recovery options list, click **Configure Recovery Settings**.
3. Specify your requirements in the Fast Recovery Area section of the page.
4. Click **ui** on this page for more information.

Related Topics

- *Oracle AI Database Backup and Recovery User's Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

9

Cloning Oracle RAC to Nodes in a New Cluster

Learn how to clone Oracle Real Application Clusters (Oracle RAC) database homes on Linux and Unix systems to nodes in a new cluster.

- Introduction to Cloning Oracle RAC
  Learn how to use cloning in Oracle RAC to simplify your administrative tasks.
- Preparing to Clone Oracle RAC
  Use this overview to understand the Oracle RAC cloning procedures.
- Deploying Oracle RAC Clones to Nodes in a Cluster
  Learn about deploying Oracle RAC clones from one node to other nodes in a cluster.
- Locating and Viewing Log Files Generated During Cloning
  The cloning script runs multiple tools, each of which may generate its own log files.

Introduction to Cloning Oracle RAC

Learn how to use cloning in Oracle RAC to simplify your administrative tasks.

You can implement a noninteractive cloning technique using scripts. These cloning techniques are best suited for performing multiple simultaneous cluster installations. Creating the scripts is a manual process and can be error-prone. If you only have one cluster to install, then you should use the traditional automated and interactive installation methods, such as Oracle Universal Installer, or the Provisioning Pack feature of Oracle Enterprise Manager.

Cloning is the process of copying an existing Oracle RAC installation to a different location and updating the copied bits to work in the new environment. The changes made by one-off patches applied on the source Oracle home, would also be present after the clone operation. The source and the destination path (host to be cloned) need not be the same.

Some situations in which cloning is useful are:

- Cloning provides a way to prepare an Oracle home once and deploy it to many hosts simultaneously. You can complete the installation silently, as a noninteractive process. You do not need to use a graphical user interface (GUI) console and you can perform cloning from a Secure Shell (SSH) terminal session, if required.
- Cloning enables you to create an installation (copy of a production, test, or development installation) with all patches applied to it in a single step. Once you have performed the base installation and applied all patch sets and patches on the source system, the clone performs all of these individual steps as a single procedure. This is in contrast to going through the installation process to perform the separate steps to install, configure, and patch the installation on each node in the cluster.
- Installing Oracle RAC by cloning is a very quick process. For example, cloning an Oracle home to a new cluster of more than two nodes requires a few minutes to install the Oracle base software, plus a few minutes more for each node (approximately the amount of time it takes to run the `root.sh` script).

Copyright © 1999, 2025, Oracle and/or its affiliates.

The cloned installation behaves the same as the source installation. For example, the cloned Oracle home can be removed using Oracle Universal Installer. You can also use the cloned Oracle home as the source for another cloning operation. You can create a cloned copy of a test, development, or production installation by using the command-line cloning scripts. The default cloning procedure is adequate for most usage cases. However, you can also customize various aspects of cloning, for example, to specify custom port assignments, or to preserve custom settings.

The cloning process works by copying all of the files from the source Oracle home to the destination Oracle home. Thus, any files used by the source instance that are located outside the source Oracle home's directory structure are not copied to the destination location.

The size of the binaries at the source and the destination may differ because these are relinked as part of the clone operation and the operating system patch levels may also differ between these two locations. Additionally, the number of files in the cloned home would increase because several files copied from the source, specifically those being instantiated, are backed up as part of the clone operation.

> **Note**
>
> Cloning is not a replacement for Oracle Enterprise Manager cloning that is a part of the Provisioning Pack. During Oracle Enterprise Manager cloning, the provisioning process interactively asks you the details about the Oracle home (such as the location to which you want to deploy the clone, the name of the Oracle AI Database home, a list of the nodes in the cluster, and so on).
>
> The Provisioning Pack feature of Oracle Enterprise Manager Cloud Control provides a framework to automate the provisioning of new nodes and clusters. For data centers with many Oracle RAC clusters, the investment in creating a cloning procedure to easily provision new clusters and new nodes to existing clusters is worth the effort.

Related Topics

- Using Cloning to Extend Oracle RAC to Nodes in the Same Cluster
  Learn how to use cloning to extend Oracle RAC nodes within a cluster.

Preparing to Clone Oracle RAC

Use this overview to understand the Oracle RAC cloning procedures.

In the preparation phase, you create a copy of an Oracle home that you then use to perform the cloning procedure on one or more nodes. You also install Oracle Grid Infrastructure.

Install Oracle RAC

Use the detailed instructions in *Oracle Real Application Clusters Installation Guide* for your platform for your platform to install the Oracle RAC software and patches:

1. Install Oracle RAC and choose the **Software only** installation option.
2. Patch the release to the required Release Update (RU).
3. Apply one-off patches, if necessary.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Create a backup of the source home

Create a copy of the Oracle RAC home. Use this file to copy the Oracle RAC home to each node in the cluster.

When creating the backup file, the best practice is to include the release number in the name of the file. For example:

```
# cd /u01/app/oracle/product/23.0.0/db_1
# tar –zcvf /pathname/db2300.tgz .
```

Install and start Oracle Clusterware

Before you can use cloning to create an Oracle RAC home, you must first install and start Oracle Clusterware on the node or nodes to which you want to copy a cloned Oracle RAC home. In other words, you configure an Oracle RAC home that you cloned from a source cluster onto the nodes in a target cluster in the same order that you installed the Oracle Clusterware and Oracle RAC software components on the original nodes.

Related Topics

- *Oracle Real Application Clusters Installation Guide for Linux and UNIX*
- Deploying Oracle RAC Clones to Nodes in a Cluster
  Learn about deploying Oracle RAC clones from one node to other nodes in a cluster.
- *Oracle Clusterware Administration and Deployment Guide*

Deploying Oracle RAC Clones to Nodes in a Cluster

Learn about deploying Oracle RAC clones from one node to other nodes in a cluster.

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

1. Perform any Oracle RAC preinstallation tasks, as described in your platform-specific Oracle RAC installation guide, to prepare the new cluster nodes, such things as:
   - Specify the kernel parameters.
   - Ensure Oracle Clusterware is active.
   - Ensure that Oracle ASM is active and that at least one Oracle ASM disk group exists and is mounted.
2. Deploy the Oracle RAC database software, as follows:
   a. Copy the clone of the Oracle home to all nodes. For example:

      ```
      [root@node1 root]# mkdir -p /u01/app/oracle/product/23.0.0/db
      [root@node1 root]# cd /u01/app/oracle/product/23.0.0/db
      [root@node1 db]# tar –zxvf /path_name/db2300.tgz
      ```

      When providing the home location and *path_name*, the home location can be in the same directory path or in a different directory path from the source home that you used to create the tar.

Copyright © 1999, 2025, Oracle and/or its affiliates.

b. If either the `oracle` user or the `oinstall` group, or both is different between the source and destination nodes, then change the ownership of the Oracle Inventory files, as follows:

```
[root@node1]# chown -R oracle:oinstall /u01/app/oracle/product/23.0.0/db
```

When you run the preceding command on the Oracle RAC home, it clears `setuid` and `setgid` information from the Oracle binary.

> **Note**
>
> You can perform this step at the same time you perform Step 5 and Step 6 to run the `clone.pl` and `$ORACLE_HOME/root.sh` scripts on each cluster node.

3. Change the directory to the unzipped Oracle home directory, and remove all the `.ora` (`*.ora`) files present in the unzipped `$ORACLE_HOME/network/admin` directory.
4. Delete unnecessary files from the unzipped Oracle home directory.
The unzipped Oracle home directory contains files that are relevant only to the source Oracle home. The following example shows how to remove these unnecessary files from the unzipped Oracle home directory:

   Remove the `.ora` files from the `network/admin` directory, and remove the old database entries from the `dbs` directory.

```
# cd $ORACLE_HOME
# rm -rf network/admin/*.ora
# rm dbs/old_database_entries
```

5. Use any of the following methods to perform the main Oracle RAC cloning tasks:
   a. Perform a software-only Oracle RAC installation on each cluster node to register the Oracle RAC Database.
      i. From this new destination Oracle home directory, run the `runInstaller` command to start the Oracle AI Database Setup Wizard and register Oracle RAC Database.

```
$ cd /u01/app/oracle/product/23.0.0/db
$ ./runInstaller
```

      ii. In the Select Configuration Option screen, select **Set Up Software Only** to perform a software-only Oracle RAC installation.
      iii. Select your installation type and respond to the configuration prompts as needed.
   b. Run the `clone.pl` script on each node:
      i. Supply the environment variables and cloning parameters in the `start.sh` script, as described in Table 9-2 and Table 9-3. Because the `clone.pl` script is sensitive to the parameters being passed to it, you must be accurate in your use of brackets, single quotation marks, and double quotation marks.
      ii. Run the script as `oracle` or the user that owns the Oracle RAC software.

      The following table lists and describes the `clone.pl` script parameters:

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table 9-1 clone.pl Script Parameters

Parameter	Description
ORACLE_HOME=*Oracle_home*	The complete path to the Oracle home you want to clone. If you specify an invalid path, then the script exits. This parameter is required.
ORACLE_BASE=*ORACLE_BASE*	The complete path to the Oracle base you want to clone. If you specify an invalid path, then the script exits. This parameter is required.
ORACLE_HOME_NAME=*Oracle_home_name* | -defaultHomeName	The Oracle home name of the home you want to clone. Optionally, you can specify the -defaultHomeName flag. This parameter is optional.
ORACLE_HOME_USER=*Oracle_home_user*	The OracleHomeUser for Windows services. This parameter is applicable to Windows only and is optional.
OSDBA_GROUP=*group_name*	Specify the operating system group you want to use as the OSDBA privileged group. This parameter is optional.
OSOPER_GROUP=*group_name*	Specify the operating system group you want to use as the OSOPER privileged group. This parameter is optional.
OSRACDBA_GROUP=*group_name*	Specify the operating system group you want to use as the OSRACDBA privileged group. This parameter is optional.
OSASM_GROUP=*group_name*	Specify the operating system group you want to use as the OSASM privileged group. This parameter is optional.
OSBACKUPDBA_GROUP=*group_name*	Specify the operating system group you want to use as the OSBACKUPDBA privileged group. This parameter is optional.
OSDGDBA_GROUP=*group_name*	Specify the operating system group you want to use as the OSDGDBA privileged group. This parameter is optional.
OSKMDBA_GROUP=*group_name*	Specify the operating system group you want to use as the OSKMDBA privileged group. This parameter is optional.
-debug	Specify this option to run the clone.pl script in debug mode
-help	Specify this option to obtain help for the clone.pl script.

The following example shows an excerpt from the start.sh script that calls the clone.pl script:

```
ORACLE_BASE=/u01/app/oracle
ORACLE_HOME=/u01/app/oracle/product/23.0.0/db
cd $ORACLE_HOME/clone
THISNODE='host_name'

E01=ORACLE_HOME=/u01/app/oracle/product/23.0.0/db
E02=ORACLE_HOME_NAME=OraDBRAC
E03=ORACLE_BASE=/u01/app/oracle
C01="-O CLUSTER_NODES={node1,node2}"
C02="-O LOCAL_NODE=$THISNODE"

perl $ORACLE_HOME/clone/bin/clone.pl $E01 $E02 $E03 $C01 $C02
```

The following table lists and describes the environment variables E01, E02, and E03 that are shown in bold typeface in the preceding example:

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table 9-2 Environment Variables Passed to the clone.pl Script

Symbol	Variable	Description
E01	ORACLE_HOME	The location of the Oracle RAC database home. This directory location must exist and must be owned by the Oracle operating system group: oinstall.
E02	ORACLE_HOME_NAME	The name of the Oracle home for the Oracle RAC database. This is stored in the Oracle Inventory.
E03	ORACLE_BASE	The location of the Oracle Base directory.

The following table lists and describes the cloning parameters C01 and C02, that are shown in bold typeface in the preceding example:

Table 9-3 Cloning Parameters Passed to the clone.pl Script.

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

The following example shows an excerpt from the start.bat script that the user must create that calls the clone.pl script:

```
set ORACLE_home=C:\oracle\product\23.0.0\db1
cd %ORACLE_home%\clone\bin
set THISNODE=%hostname%
set E01=ORACLE_HOME=%ORACLE_home%
set E02=ORACLE_HOME_NAME=OraDBRAC
set E03=ORACLE_BASE=Oracle_Base
set C01="CLUSTER_NODES={node1,node2}"
set C02="-O LOCAL_NODE=%THISNODE%"
perl clone.pl %E01% %E02% %E03% %C01% %C02%
```

6. **Note**

   This step applies to Linux and UNIX installations, only.

Run the $ORACLE_HOME/root.sh as the root operating system user as soon as the cloning procedure completes on the node.

```
[root@node1 root]# /u01/app/oracle/product/23.0.0/db/root.sh -silent
```

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

```
[root@node2 root]# /u01/app/oracle/product/23.0.0/db/root.sh -silent
```

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

7. **Note**

   You need only run Oracle DBCA on one node in the cluster to create Oracle RAC instances on all nodes.

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

   The following example creates an Oracle RAC database named ERI on each node with `AUTOMATIC` management policy, creates database instances on each node, registers the instances in OCR, and creates the database files in the Oracle ASM disk group called `DATA`. It also configures Oracle Machine Learning for Python in the database and sets the `SYS` and `SYSTEM` passwords to *`password`*, which is the password for each account:

```
[oracle@node1 oracle]$ export ORACLE_HOME=/u01/app/oracle/product/23.0.0/db
[oracle@node1 oracle]$ cd $ORACLE_HOME/bin/
[oracle@node1 bin]$./dbca -silent -createDatabase -templateName
"General_Purpose.dbc" \
-gdbName gdb_name -sid SID_prefix \
-storageType ASM -diskGroupName DATA -nodelist node1,node2 -characterset
WE8MSWIN1252 \
-recoveryAreaDestination RECO -databaseType "MULTIPURPOSE" \
-createAsContainerDatabase true -numberofPDBs 2 -pdbName rracwpdb
```

Related Topics

- Preparing to Clone Oracle RAC
  Use this overview to understand the Oracle RAC cloning procedures.
- *Oracle Real Application Clusters Installation Guide*

See Also

Oracle Multitenant Administrator's Guide for a complete description of Oracle DBCA commands and options.

Locating and Viewing Log Files Generated During Cloning

The cloning script runs multiple tools, each of which may generate its own log files.

After the `clone.pl` script finishes running, you can view log files to obtain more information about the cloning process.

The following log files that are generated during cloning are the key log files of interest for diagnostic purposes:

- *`Central_Inventory`*`/logs/cloneActions`*`timestamp`*`.log`

  Contains a detailed log of the actions that occur during the Oracle Universal Installer part of the cloning.

- *`Central_Inventory`*`/logs/oraInstall`*`timestamp`*`.err`

  Contains information about errors that occur when Oracle Universal Installer is running.

- *`Central_Inventory`*`/logs/oraInstall`*`timestamp`*`.out`

Copyright © 1999, 2025, Oracle and/or its affiliates.

Contains other miscellaneous messages generated by Oracle Universal Installer.

- `$ORACLE_HOME/clone/logs/clone`*`timestamp`*`.log`

  Contains a detailed log of the actions that occur before cloning and during the cloning operations.

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

  Contains information about errors that occur before cloning and during cloning operations.

The following table describes how to find the location of the Oracle inventory directory.

Table 9-4 Finding the Location of the Oracle Inventory Directory

Type of System...	Location of the Oracle Inventory Directory
All UNIX computers except Linux and IBM AIX	`/var/opt/oracle/oraInst.loc`
IBM AIX and Linux	`/etc/oraInst.loc` file.
Windows	`C:\Program Files\Oracle\Inventory`

Copyright © 1999, 2025, Oracle and/or its affiliates.

10

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

Learn how to use cloning to extend Oracle RAC nodes within a cluster.

This chapter provides information about using cloning to extend Oracle Real Application Clusters (Oracle RAC) to nodes in an existing cluster.

To add Oracle RAC to nodes in a new cluster, see Cloning Oracle RAC to Nodes in a New Cluster.

- About Adding Nodes Using Cloning in Oracle RAC Environments
  You can use cloning to add nodes in Oracle RAC environments.
- Cloning Local Oracle Homes on Linux and UNIX Systems
  Add nodes to Oracle RAC environments by cloning a local, non-shared Oracle home in Linux and UNIX environments.
- Cloning Shared Oracle Homes on Linux and UNIX Systems
  Add nodes to Oracle RAC environments by cloning a shared Oracle home in Linux and UNIX systems.
- Cloning Oracle Homes on Windows Systems
  Add nodes to Oracle RAC environments by cloning a shared or local Oracle home in Microsoft Windows environments.

Related Topics

- Introduction to Cloning Oracle RAC
  Learn how to use cloning in Oracle RAC to simplify your administrative tasks.
- *Oracle Clusterware Administration and Deployment Guide*

About Adding Nodes Using Cloning in Oracle RAC Environments

You can use cloning to add nodes in Oracle RAC environments.

The cloning procedures assume that you have successfully installed and configured an Oracle RAC environment to which you want to add nodes and instances. To add nodes to an Oracle RAC environment using cloning, first extend the Oracle Clusterware configuration, then extend the Oracle AI Database software with Oracle RAC, and then add the listeners and instances by running the Oracle assistants

The cloning script runs multiple tools, each of which may generate its own log files. After the `clone.pl` script finishes running, you can view log files to obtain more information about the cloning process.

Related Topics

- Locating and Viewing Log Files Generated During Cloning
  The cloning script runs multiple tools, each of which may generate its own log files.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Cloning Local Oracle Homes on Linux and UNIX Systems

Add nodes to Oracle RAC environments by cloning a local, non-shared Oracle home in Linux and UNIX environments.

Complete the following steps to clone Oracle AI Database with Oracle RAC:

1. Follow the steps in the topic "Preparing to Clone Oracle RAC" to create a copy of an Oracle home that you then use to perform the cloning procedure on one or more nodes.
2. Use the `tar` utility to create an archive of the Oracle home on the existing node and copy it to the new node. If the location of the Oracle home on the source node is `$ORACLE_HOME`, then you must use this same directory as the destination location on the new node.
3. On the new node, configure the environment variables `ORACLE_HOME` and `ORACLE_BASE`. Then go to the *`Grid_home`*`/clone/bin` directory on the new node and run the following command, where *`existing_node`* is the name of the node that you are cloning, *`new_node2`* and *`new_node3`* are the names of the new nodes, and *`Oracle_home_name`* is the name of the Oracle home:

   ```
   perl clone.pl ORACLE_HOME=$ORACLE_HOME ORACLE_HOME_NAME=Oracle_home_name
     ORACLE_BASE=$ORACLE_BASE
   "'CLUSTER_NODES={existing_node,new_node2,new_node3}'"
     "'LOCAL_NODE=new_node2'" CRS=TRUE INVENTORY_LOCATION=/u01/app/
   oraInventory
   ```

4. Run the following command to run the configuration assistants to configure Oracle RAC on the new nodes:

   ```
   $ORACLE_HOME/cfgtoollogs/configToolFailedCommands
   ```

   This script contains all commands that failed, were skipped, or were canceled during the installation. You can use this script to run the database configuration assistants outside of Oracle Universal Installer. Note that before you run the script you should check the script to see if any passwords within it need to be updated.
5. Run the following command on the existing node from the `$ORACLE_HOME/oui/bin` directory to update the inventory in the Oracle AI Database home with Oracle RAC, specified by *`Oracle_home`*, where *`existing_node`* is the name of the original node that you are cloning and *`new_node2`* and *`new_node3`* are the names of the new nodes:

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

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

   ```
   ./root.sh
   ```

7. From the node that you cloned, run Oracle DBCA to add Oracle RAC database instances on the new nodes.

Related Topics

- Preparing to Clone Oracle RAC
  Use this overview to understand the Oracle RAC cloning procedures.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Cloning Shared Oracle Homes on Linux and UNIX Systems

Add nodes to Oracle RAC environments by cloning a shared Oracle home in Linux and UNIX systems.

Complete the following steps to clone Oracle AI Database with Oracle RAC:

1. Follow the steps in the "Preparing to Clone Oracle RAC" to create a copy of an Oracle home that you then use to perform the cloning procedure on one or more nodes.
2. On the new node, configure the environment variables `ORACLE_HOME` and `ORACLE_BASE`. Then go to the `$ORACLE_HOME/clone/bin` directory and run the following command, where *`existing_node`* is the name of the node that you are cloning, *`new_node2`*, and *`new_node3`* are the names of the new nodes, *`Oracle_home_name`* is the name of the Oracle home, and the `-cfs` option indicates the Oracle home is shared:

   ```
   perl clone.pl -O 'CLUSTER_NODES={existing_node,new_node2,new_node3}'
   -O LOCAL_NODE=new_node2 ORACLE_BASE=$ORACLE_BASE ORACLE_HOME=$ORACLE_HOME
    ORACLE_HOME_NAME=Oracle_home_name [-cfs]
   ```

   > **Note**
   >
   > In the preceding command:
   >
   > - Use the `-cfs` option for a shared Oracle AI Database home with Oracle RAC.
   > - The value for the `ORACLE_HOME_NAME` parameter must be that of the node you are cloning.

3. Run the following command on the existing node from the `$ORACLE_HOME/oui/bin` directory to update the inventory in the Oracle AI Database home with Oracle RAC, specified by *`Oracle_home`*, where *`existing_node`* is the name of the original node that you are cloning and *`new_node2`* and *`new_node3`* are the names of the new nodes:

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

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

   ```
   ./root.sh
   ```

5. From the node that you cloned, run Oracle DBCA to add Oracle RAC database instances to the new nodes.

Related Topics

- Preparing to Clone Oracle RAC
  Use this overview to understand the Oracle RAC cloning procedures.

Cloning Oracle Homes on Windows Systems

Add nodes to Oracle RAC environments by cloning a shared or local Oracle home in Microsoft Windows environments.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Complete the following steps to clone Oracle AI Database with Oracle RAC:

1. If you have a local Oracle home, then use the ZIP utility to create an archive of the Oracle AI Database home with Oracle RAC on the existing node and copy it to the new node. Otherwise, proceed to the next step.

   Extract the Oracle AI Database with Oracle RAC home files from the ZIP file on the new node in the same directory in which the Oracle AI Database home with Oracle RAC resided on the existing node. For example, assume that the location of the destination Oracle RAC home on the new node is `%ORACLE_HOME%`.

2. On the new node, go to the `%ORACLE_HOME%\clone\bin` directory and run the following command, where *`Oracle_Home`* is the Oracle AI Database home, *`Oracle_Home_Name`* is the name of the Oracle AI Database home, *`Oracle_Base`* is the Oracle base directory, *`user_name`* is the name of the Oracle home user (a non-Administrator user) for the Oracle home being cloned, *`existing_node`* is the name of the existing node, and *`new_node`* is the name of the new node:

   ```
   perl clone.pl ORACLE_HOME=Oracle_Home ORACLE_BASE=Oracle_Base
   ORACLE_HOME_NAME=Oracle_Home_Name ORACLE_HOME_USER=user_name
   -O 'CLUSTER_NODES={existing_node,new_node}'
   -O LOCAL_NODE=new_node
   ```

   If you have a shared Oracle AI Database home with Oracle RAC, then append the `-cfs` option to the command to indicate that the Oracle home is shared as shown in the following example:

   ```
   perl clone.pl ORACLE_HOME=Oracle_Home ORACLE_BASE=Oracle_Base
   ORACLE_HOME_NAME=Oracle_Home_Name ORACLE_HOME_USER=user_name
   -O 'CLUSTER_NODES={existing_node,new_node}' -O LOCAL_NODE=new_node
   [-cfs -noConfig]
   ```

   > **Note**
   >
   > - The `ORACLE_HOME_USER` is required *only* if you are cloning a secured Oracle home.
   > - Use the `-cfs` and `-noConfig` options for a shared Oracle AI Database home with Oracle RAC.
   > - The value for the `ORACLE_HOME_NAME` parameter must be that of the node you are cloning. To obtain the `ORACLE_HOME_NAME`, look in the registry on the node you cloning for the `ORACLE_HOME_NAME` parameter key under `HKEY_LOCAL_MACHINE\SOFTWARE\oracle\KEY_OraCRs2300_home1`.

3. On the existing node, from the `%ORACLE_HOME%\oui\bin` directory run the following command to update the inventory in the Oracle AI Database home with Oracle RAC, specified by *`Oracle_home`*, where *`existing_node`* is the name of the existing node, and *`new_node`* is the name of the new node:

   ```
   setup.exe -updateNodeList ORACLE_HOME=Oracle_home "CLUSTER_NODES=
   {existing_node,new_node}" LOCAL_NODE=existing_node
   ```

4. From the node that you cloned, run Oracle DBCA to add Oracle RAC database instances to the new nodes.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Copyright © 1999, 2025, Oracle and/or its affiliates.

11

Adding and Deleting Oracle RAC from Nodes on Linux and Unix Systems

Extend an existing Oracle Real Application Clusters (Oracle RAC) home to other nodes and instances in the cluster, and delete Oracle RAC from nodes and instances in the cluster.

- About Adding and Deleting Nodes
  Adding and deleting nodes is the process of adding or modifying your Oracle Real Application Clusters database cluster
- Adding Oracle RAC to Nodes with Oracle Grid Infrastructure Installed
  To add Oracle Real Application Clusters (Oracle RAC) with Oracle Grid Infrastructure installed, your procedure depends on the storage you use, and your Oracle home configuration,
- Adding Administrator-Managed Oracle RAC Database Instances to Target Nodes
  Learn how to configure Administrator-Managed Oracle Real Application Clusters (Oracle RAC) instances.
- Deleting Oracle RAC from a Cluster Node
  You can delete Oracle RAC from cluster nodes using this procedure.

About Adding and Deleting Nodes

Adding and deleting nodes is the process of adding or modifying your Oracle Real Application Clusters database cluster

If your goal is to clone an existing Oracle RAC home to create multiple new Oracle RAC installations across the cluster, then use the cloning procedures that are described in "Cloning Oracle RAC to Nodes in a New Cluster".

> **Note**
>
> - Ensure that you have a current backup of Oracle Cluster Registry (OCR) before adding or deleting Oracle RAC by running the `ocrconfig -showbackup` command.
> - The phrase "target node" in the context of configuring Oracle RAC nodes refers to the node to which you plan to extend the Oracle RAC environment.

Related Topics

- Cloning Oracle RAC to Nodes in a New Cluster
  Learn how to clone Oracle Real Application Clusters (Oracle RAC) database homes on Linux and Unix systems to nodes in a new cluster.
- Adding and Deleting Oracle RAC from Nodes on Windows Systems
  Use these procedures to extend an existing Oracle Real Application Clusters (Oracle RAC) home on Microsoft Windows to other nodes and instances in the cluster, or delete Oracle RAC from nodes and instances in the cluster.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Adding Oracle RAC to Nodes with Oracle Grid Infrastructure Installed

To add Oracle Real Application Clusters (Oracle RAC) with Oracle Grid Infrastructure installed, your procedure depends on the storage you use, and your Oracle home configuration,

Before beginning the procedure on your system, ensure that your existing nodes have the correct path to the *Grid_home* and that the `$ORACLE_HOME` environment variable is set to the Oracle RAC home.

Procedure for Local (Non-Shared) Oracle Home

If you are using a local (non-shared) Oracle home, then you must extend the Oracle RAC database home that is on an existing node (`node1` in this procedure) to a target node (`node3` in this procedure).

1. Navigate to the *Oracle_home*`/addnode` directory on `node1` and run the `addnode.sh` script.
2. If you want to perform a silent installation, run the `addnode.sh` script using the following syntax:

   ```
   $ ./addnode.sh -silent "CLUSTER_NEW_NODES={node3}"
   ```

3. Run the `Oracle_home/root.sh` script on `node3` as `root`.
4. Manually copy the `tnsnames.ora` file from your old node to the newly added node.

   ```
   $ cd $ORACLE_BASE_HOME/network/admin
   $ sftp tnsnames.ora new_node:$ORACLE_BASE_HOME/network/admin/
   ```

5. Open the Pluggable Databases (PDBs) on the newly added node using the following commands in your SQL*Plus session:

   ```
   SQL> CONNECT / AS SYSDBA
   SQL> ALTER PLUGGABLE DATABASE pdb_name OPEN;
   ```

Procedure for Shared Oracle Home Using Oracle ACFS

If you have a shared Oracle home that is shared using Oracle Advanced Cluster File System (Oracle ACFS), then do the following to extend the Oracle AI Database Oracle home to `node3`:

1. Start the Oracle ACFS resource on the new node by running the following command as `root` from the *Grid_home*`/bin` directory:

   ```
   # srvctl start filesystem -device volume_device [-node node_name]
   ```

> **Note**
>
> Make sure the Oracle ACFS resources, including Oracle ACFS registry resource and Oracle ACFS file system resource where the Oracle home is located, are online on the newly added node.

Copyright © 1999, 2025, Oracle and/or its affiliates.

2. Run the following command as the user who installed Oracle RAC from the *Oracle_home*/oui/bin directory on the node you are adding to add the Oracle RAC database home:

```
$ ./runInstaller -attachHome ORACLE_HOME="ORACLE_HOME"
"CLUSTER_NODES={node3}"
  LOCAL_NODE="node3" ORACLE_HOME_NAME="home_name" -cfs
```

3. Navigate to the *Oracle_home*/addnode directory on node1 and run the addnode.sh script as the user who installed Oracle RAC using the following syntax:

```
$ ./addnode.sh -noCopy "CLUSTER_NEW_NODES={node3}"
```

> **Note**
>
> Use the -noCopy option because the Oracle home on the destination node is already fully populated with software.

4. Run the *Oracle_home*/root.sh script on node3 as root.

Procedure for Shared Oracle Home Not Using Oracle ACFS

If you have a shared Oracle home on a shared file system that *is not* Oracle ACFS, then you must first create a mount point for the Oracle RAC database home on the target node, mount and attach the Oracle RAC database home, and update the Oracle Inventory:

1. Run the srvctl config database -db *db_name* command on an existing node in the cluster to obtain the mount point information.
2. Run the following command as root on node3 to create the mount point:

```
# mkdir -p mount_point_path
```

3. Mount the file system that hosts the Oracle RAC database home.
4. Run the following command as the user who installed Oracle RAC from the *Oracle_home*/oui/bin directory on the node you are adding to add the Oracle RAC database home:

```
$ ./runInstaller -attachHome ORACLE_HOME="ORACLE_HOME" "CLUSTER_NODES=
  {local_node_name}" LOCAL_NODE="node_name" ORACLE_HOME_NAME="home_name"
```

5. Update the Oracle Inventory as the user who installed Oracle RAC, as follows:

```
$ ./runInstaller -updateNodeList ORACLE_HOME=mount_point_path
"CLUSTER_NODES=
  {node_list}"
```

   In the preceding command, *node_list* refers to a list of all nodes where the Oracle RAC database home is installed, including the node you are adding.

6. Run the *Oracle_home*/root.sh script on node3 as root.

You can now add an Oracle RAC database instance to the target node.

draf

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> Oracle recommends that you back up the OCR after you complete the node addition process.

Related Topics

- *Oracle Clusterware Administration and Deployment Guide*

Adding Administrator-Managed Oracle RAC Database Instances to Target Nodes

Learn how to configure Administrator-Managed Oracle Real Application Clusters (Oracle RAC) instances.

- About Adding Administrator-Managed Oracle RAC Database Instances
  To add Oracle Real Application Clusters (Oracle RAC) database instances, you have several tools available.
- Using Oracle DBCA in Interactive Mode to Add Database Instances to Target Nodes
  To add a database instance to a target node with Oracle DBCA in interactive mode, perform the steps described here.
- Using Oracle DBCA in Silent Mode to Add Database Instances to Target Nodes
  You can use Oracle DBCA in silent mode to add instances to nodes on which you have extended an Oracle Clusterware home and an Oracle AI Database home.

About Adding Administrator-Managed Oracle RAC Database Instances

To add Oracle Real Application Clusters (Oracle RAC) database instances, you have several tools available.

You can use either Oracle Enterprise Manager or Oracle DBCA to add Oracle RAC database instances to the target nodes.

This section describes using Oracle DBCA to add Oracle RAC database instances.

These tools guide you through the following tasks:

- Creating a new database instance on each target node
- Creating and configuring high availability components
- Creating the Oracle Net configuration for a non-default listener from the Oracle home
- Starting the new instance
- Creating and starting services if you entered services information on the Services Configuration page

After adding the instances to the target nodes, you should perform any necessary service configuration procedures, as described in "Workload Management with Dynamic Database Services".

Copyright © 1999, 2025, Oracle and/or its affiliates.

Related Topics

- Workload Management with Dynamic Database Services
  Workload management includes load balancing, enabling clients for Oracle Real Application Clusters (Oracle RAC), distributed transaction processing, and services.

Using Oracle DBCA in Interactive Mode to Add Database Instances to Target Nodes

To add a database instance to a target node with Oracle DBCA in interactive mode, perform the steps described here.

1. Ensure that your existing nodes have the `$ORACLE_HOME` environment variable set to the Oracle RAC home.
2. Start Oracle DBCA by entering `dbca` at the system prompt from the *`Oracle_home`*`/bin` directory.

   Oracle DBCA performs certain CVU checks while running. However, you can also run CVU from the command line to perform various verifications.

   Oracle DBCA displays the Welcome page for Oracle RAC. Click **Help** on any Oracle DBCA page for additional information.
3. Select **Instance Management**, click **Next**, and Oracle DBCA displays the Instance Management page.
4. Select **Add Instance** and click **Next**. Oracle DBCA displays the List of Cluster Databases page that shows the databases and their current status, such as `ACTIVE` or `INACTIVE`.
5. From the List of Cluster Databases page, select the active Oracle RAC database to which you want to add an instance. Click **Next** and Oracle DBCA displays the List of Cluster Database Instances page showing the names of the existing instances for the Oracle RAC database that you selected.
6. Click **Next** to add a new instance and Oracle 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 Oracle DBCA provides does not match your existing instance naming scheme.
8. Review the information on the Summary dialog and click **OK** or click **Cancel** to end the instance addition operation. Oracle DBCA displays a progress dialog showing Oracle DBCA performing the instance addition operation.
9. After you terminate your Oracle DBCA session, run the following command to verify the administrative privileges on the target node and obtain detailed information about these privileges where *`node_list`* consists of the names of the nodes on which you added database instances:

   ```
   cluvfy comp admprv -o db_config -d Oracle_home -n node_list [-verbose]
   ```

10. Perform any necessary service configuration procedures, as described in "Workload Management with Dynamic Database Services".

Related Topics

- *Oracle Clusterware Administration and Deployment Guide*
- Workload Management with Dynamic Database Services
  Workload management includes load balancing, enabling clients for Oracle Real Application Clusters (Oracle RAC), distributed transaction processing, and services.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Using Oracle DBCA in Silent Mode to Add Database Instances to Target Nodes

You can use Oracle DBCA in silent mode to add instances to nodes on which you have extended an Oracle Clusterware home and an Oracle AI Database home.

Before you run the `dbca` command, ensure that you have set the `ORACLE_HOME` environment variable correctly on the existing nodes. Run Oracle DBCA, supplying values for the variables using the following syntax:

```
dbca -silent -addInstance -nodeName node_name -gdbName gdb_name
  [-instanceName instance_name -sysDBAUserName sysdba -sysDBAPassword
  password]
```

The following table describes the values that you need to supply for each variable.

Table 11-1 Variables in the Oracle DBCA Silent Mode Syntax

Variable	Description
node_name	The node on which you want to add (or delete) the instance.
gdb_name	Global database name.
instance_name	Name of the instance. Provide an instance name only if you want to override the Oracle naming convention for Oracle RAC instance names.
sysdba	Name of the Oracle user with SYSDBA privileges.
password	Password for the SYSDBA user.

Perform any necessary service configuration procedures, as described in "Workload Management with Dynamic Database Services".

Related Topics

- Workload Management with Dynamic Database Services
  Workload management includes load balancing, enabling clients for Oracle Real Application Clusters (Oracle RAC), distributed transaction processing, and services.

Deleting Oracle RAC from a Cluster Node

You can delete Oracle RAC from cluster nodes using this procedure.

To remove Oracle RAC from cluster nodes, delete the database instance and Oracle RAC before removing the node from the cluster.

> **Note**
>
> If there are no database instances on the node you want to delete, then proceed to "Removing Oracle RAC".

- Deleting Instances from Oracle RAC Databases
  You can use the SRVCTL utility to delete an Oracle RAC database instance.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Removing Oracle RAC
  This procedure removes Oracle RAC from the node that you are deleting from a cluster and updates the inventories on the remaining nodes.
- Deleting Nodes from A Cluster
  Use this procedure to delete nodes from a cluster.

Related Topics

- Removing Oracle RAC
  This procedure removes Oracle RAC from the node that you are deleting from a cluster and updates the inventories on the remaining nodes.

Deleting Instances from Oracle RAC Databases

You can use the SRVCTL utility to delete an Oracle RAC database instance.

Deleting Instances from Administrator-Managed Databases

> **Note**
>
> Before deleting an instance from an Oracle RAC database using SRVCTL to do the following:
>
> - If you have services configured, then relocate the services
> - Modify the services so that each service can run on one of the remaining instances
> - Ensure that the instance to be removed from an administrator-managed database is neither a preferred nor an available instance of any service

- Using Oracle DBCA in Interactive Mode to Delete Instances from Nodes
  This procedure explains how to use Oracle DBCA in interactive mode to delete instances from Oracle RAC databases.
- Using Oracle DBCA in Silent Mode to Delete Instances from Nodes
  Learn how to use Oracle DBCA in silent mode to delete instances from nodes.

Related Topics

- Removing Oracle RAC
  This procedure removes Oracle RAC from the node that you are deleting from a cluster and updates the inventories on the remaining nodes.
- Administering Services with SRVCTL
  Learn how to use SRVCTL to perform service administration on an Oracle Real Application Clusters (Oracle RAC) database.
- Using Oracle DBCA in Interactive Mode to Delete Instances from Nodes
  This procedure explains how to use Oracle DBCA in interactive mode to delete instances from Oracle RAC databases.

Using Oracle DBCA in Interactive Mode to Delete Instances from Nodes

This procedure explains how to use Oracle DBCA in interactive mode to delete instances from Oracle RAC databases.

To delete an instance using Oracle DBCA in interactive mode, perform the following steps:

Copyright © 1999, 2025, Oracle and/or its affiliates.

1. Start Oracle DBCA.

   Start Oracle DBCA on a node *other than* the node that hosts the instance that you want to delete. The database and the instance that you plan to delete should be running during this step.

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

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

4. On the List of Cluster Databases page, select the Oracle RAC database from which to delete the instance, as follows:

   a. On the List of Cluster Database Instances page, Oracle DBCA displays the instances that are associated with the Oracle RAC database that you selected and the status of each instance. Select the cluster database from which you will delete the instance.

   b. Click **OK** on the Confirmation dialog to proceed to delete the instance.

      Oracle DBCA displays a progress dialog showing that Oracle DBCA is deleting the instance. During this operation, Oracle DBCA removes the instance and the instance's Oracle Net configuration.

      Click **No** and exit Oracle DBCA or click **Yes** to perform another operation. If you click **Yes**, then Oracle DBCA displays the Operations page.

5. Verify that the dropped instance's redo thread has been removed by using SQL*Plus on an existing node to query the `GV$LOG` view. If the redo thread is not disabled, then disable the thread. For example:

   ```
   SQL> ALTER DATABASE DISABLE THREAD 2;
   ```

6. Verify that the instance has been removed from OCR by running the following command, where *`db_unique_name`* is the database unique name for your Oracle RAC database:

   ```
   $ srvctl config database -db db_unique_name
   ```

7. If you are deleting more than one node, then repeat these steps to delete the instances from all the nodes that you are going to delete.

Using Oracle DBCA in Silent Mode to Delete Instances from Nodes

Learn how to use Oracle DBCA in silent mode to delete instances from nodes.

Run the following command, where the variables are the same as those shown in Table 11-1 for the Oracle DBCA command to remove an instance. Provide a node name only if you are deleting an instance from a node other than the one on where Oracle DBCA is running as shown in the following example where *password* is the password:

```
dbca -silent -deleteInstance [-nodeName node_name] -gdbName gdb_name
-instanceName instance_name [-sysDBAUserName sysdba -sysDBAPassword password]
```

At this point, you have accomplished the following:

- Deregistered the selected instance from its associated Oracle Net Services listeners
- Deleted the selected database instance from the instance's configured node
- Removed the Oracle Net configuration

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Deleted the Oracle Flexible Architecture directory structure from the instance's configured node.

Removing Oracle RAC

This procedure removes Oracle RAC from the node that you are deleting from a cluster and updates the inventories on the remaining nodes.

1. If there is a listener in the Oracle RAC home on the node you are deleting, then you must disable and stop it before deleting the Oracle RAC software. Run the following commands on any node in the cluster, specifying the name of the listener and the name of the node you are deleting:

   ```
   $ srvctl disable listener -l listener_name -n name_of_node_to_delete
   $ srvctl stop listener -l listener_name -n name_of_node_to_delete
   ```

2. Deinstall the Oracle home—only if the Oracle home is *not* shared—from the node that you are deleting by running the following command from the *Oracle_home*`\deinstall` directory:

   ```
   deinstall -local
   ```

> ⚠ **Caution**
>
> If the Oracle home is shared, then *do not* run this command because it will remove the shared software. Proceed to the next step instead.

Deleting Nodes from A Cluster

Use this procedure to delete nodes from a cluster.

After you delete the database instance and Oracle RAC, you can delete the node from the cluster. Do this by running scripts on the node that you want to delete to remove Oracle Clusterware. Then run scripts on the remaining nodes to update the node list.

Related Topics

- *Oracle Clusterware Administration and Deployment Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

12

Adding and Deleting Oracle RAC from Nodes on Windows Systems

Use these procedures to extend an existing Oracle Real Application Clusters (Oracle RAC) home on Microsoft Windows to other nodes and instances in the cluster, or delete Oracle RAC from nodes and instances in the cluster.

In these topics, the entries for *Grid_home* refer to the full path name for the Oracle Grid Infrastructure home, and the entries for *Oracle_home* refer to substitutes for environment variables for the Oracle home with Oracle RAC.

If your goal is to clone an existing Oracle RAC home to create multiple new Oracle RAC installations across the cluster, then use the cloning procedures that are described in "Cloning Oracle RAC to Nodes in a New Cluster".

> **Note**
>
> - Ensure that you have a current backup of Oracle Cluster Registry (OCR) before adding or deleting Oracle RAC by running the `ocrconfig -showbackup` command.
> - For all of the add node and delete node procedures, temporary directories such as `%TEMP%` or `C:\Temp` *should not be* shared directories. If your temporary directories are shared, then set your temporary environment variable, such as `%TEMP%`, to a location on a local node. In addition, use a directory path that exists on all of the nodes.

- Adding Oracle RAC to Nodes with Oracle Grid Infrastructure Installed
  To add Oracle Real Application Clusters to Microsoft Windows nodes that are Oracle Clusterware cluster nodes, review these procedures.
- Adding Administrator-Managed Oracle RAC Database Instances to Target Nodes
  Learn about how to use Oracle DBCA to add Oracle RAC database instances.
- Deleting Oracle RAC from a Cluster Node

Adding Oracle RAC to Nodes with Oracle Grid Infrastructure Installed

To add Oracle Real Application Clusters to Microsoft Windows nodes that are Oracle Clusterware cluster nodes, review these procedures.

Before beginning these procedures, ensure that your existing nodes have the correct path to the *Grid_home* and that the *Oracle_home* environment variables are set correctly.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Extending the Oracle RAC Home On an Existing Node

To add Oracle RAC database instances to nodes that already have Oracle Grid Infrastructure installed, you must extend the Oracle RAC home that is on an existing node (`node1` in this procedure) of the cluster to the target nodes.

1. Navigate to the *`Oracle_home`*`\addnode` directory on `node1` and run the `addnode.bat` script using the following syntax, where `node2` is the name of the node you are adding:

   ```
   addnode.bat "CLUSTER_NEW_NODES={node2}"
   ```

   To run this command in silent mode:

   ```
   addNode.bat -silent "CLUSTER_NEW_NODES={node2}"
   ```

   For the Oracle home directory you use, if an Oracle home user was specified when the Oracle AI Database software was installed, then OUI requires the password for the Oracle home user. OUI checks the wallet (stored in the OCR) for the user and extracts the password from there. If the user information is not contained in the wallet, then the `addnode.bat` script generates an error unless you specify the `-promptPasswd` flag on the command line.

Creating a Mount Point for the Oracle Home On a Shared File System

If you have a shared Oracle home on a shared file system that *is not* Oracle ACFS, then you must first create a mount point for the Oracle RAC database home on the target node, mount and attach the Oracle RAC database home, and update the Oracle Inventory, as follows:

1. Run the `srvctl config database -db` *`db_name`* command on an existing node in the cluster to obtain the mount point information.
2. Mount the file system that hosts the Oracle RAC database home.
3. Run the following command as the user who installed Oracle RAC from the *`Oracle_home`*`\oui\bin` directory on the node you are adding to add the Oracle RAC database home:

   ```
   setup.exe -attachHome ORACLE_HOME="ORACLE_HOME" "CLUSTER_NODES=
     local_node_name}" LOCAL_NODE="node_name" ORACLE_HOME_NAME="home_name"
   ```

4. Update the Oracle Inventory as the user who installed Oracle RAC, as follows:

   ```
   setup.exe -updateNodeList ORACLE_HOME=mount_point_path
   "CLUSTER_NODES={node_list}"
   ```

   In the preceding command, *`node_list`* refers to a list of all nodes where the Oracle RAC database home is installed, including the node you are adding.

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

Copyright © 1999, 2025, Oracle and/or its affiliates.

Related Topics

- *Oracle Clusterware Administration and Deployment Guide*
- Adding Administrator-Managed Oracle RAC Database Instances to Target Nodes
  Learn about how to use Oracle DBCA to add Oracle RAC database instances.

Adding Administrator-Managed Oracle RAC Database Instances to Target Nodes

Learn about how to use Oracle DBCA to add Oracle RAC database instances.

To add Oracle RAC database instances to the target nodes. you can use either Oracle Enterprise Manager or Oracle DBCA. These topics describe using Oracle DBCA

- About Using Oracle DBCA to Add Oracle RAC Instances
  When you use Oracle DBCA to add Oracle Real Application Clusters instances to Oracle Clusterware, it helps you to complete more than only a database deployment.
- Using Oracle DBCA in Interactive Mode to Add Database Instances to Target Nodes
  In these procedures, you first use Oracle DBCA to add a database instance to a target node, and then create a service for Oracle Services for Microsoft Transaction Server (`OraMTS`).
- Using Oracle DBCA in Silent Mode to Add Database Instances to Target Nodes
  Add instances to nodes on which you have extended an Oracle Clusterware home and an Oracle AI Database home.

About Using Oracle DBCA to Add Oracle RAC Instances

When you use Oracle DBCA to add Oracle Real Application Clusters instances to Oracle Clusterware, it helps you to complete more than only a database deployment.

Oracle DBCA guides you through the following tasks:

- Creating a new Oracle AI Database instance on each target node
- Creating and configuring high availability components
- Creating the Oracle Net configuration for a non-default listener from the Oracle home
- Starting the new instance
- Creating and starting services if you entered services information on the Services Configuration page

After adding the instances to the target nodes, you should perform any necessary service configuration procedures.

Related Topics

- Workload Management with Dynamic Database Services
  Workload management includes load balancing, enabling clients for Oracle Real Application Clusters (Oracle RAC), distributed transaction processing, and services.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Using Oracle DBCA in Interactive Mode to Add Database Instances to Target Nodes

In these procedures, you first use Oracle DBCA to add a database instance to a target node, and then create a service for Oracle Services for Microsoft Transaction Server (`OraMTS`).

Adding a Database Instance To Target Nodes

To add a database instance to a target node using Oracle DBCA in interactive mode, perform the following steps:

1. Ensure that your existing nodes have the Oracle home environment variable set correctly.
2. Start Oracle DBCA by entering `dbca` at the system prompt from the *`Oracle_home`*`\bin` directory on an existing node.

   Oracle DBCA performs certain CVU checks while running. However, you can also run CVU from the command line to perform various verifications.
3. On the Database Operations page, select **Instance Management**, click **Next**, and Oracle DBCA displays the Instance Management page.
4. Select **Add Instance** and click **Next**. Oracle DBCA displays the List of Cluster Databases page that shows the databases and their current status, such as `ACTIVE` or `INACTIVE`.
5. From the List of Cluster Databases page, select the active Oracle RAC database to which you want to add an instance. Click **Next** and Oracle DBCA displays the List of Cluster Database Instances page showing the names of the existing instances for the Oracle RAC database that you selected.
6. Click **Next** to add a new instance and Oracle 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 Oracle DBCA provides does not match your existing instance naming scheme. Then select the new node name from the list.

   > **Note**
   >
   > If you installed the Oracle home with the Oracle Home User option, then Oracle DBCA prompts you for that password on this page.

8. Review the information on the Summary Page and click **Finish** to initiate instance addition operation. Oracle DBCA displays a progress dialog showing Oracle DBCA performing the instance addition operation.

Creating the OraMTS Service for Microsoft Transaction Server

Oracle Services for Microsoft Transaction Server (`OraMTS`) permits Oracle AI Database to be used as a resource manager in Microsoft application-coordinated transactions. `OraMTS` acts as a proxy for Oracle AI Database to the Microsoft Distributed Transaction Coordinator (`MSDTC`). As a result, `OraMTS` provides client-side connection pooling and allows client components that leverage Oracle to participate in promotable and distributed transactions. In addition, `OraMTS` can operate with Oracle AI Databases running on any operating system, given that the services themselves are run on Windows. You must use a configuration tool to create this service.

Copyright © 1999, 2025, Oracle and/or its affiliates.

To create the `OraMTS` service after adding a node or performing a software-only installation for Oracle RAC, complete this procedure:

1. Open a command window.
2. Change directories to `%ORACLE_HOME%\bin`.
3. Run the `OraMTSCtl` utility to create the `OraMTS` Service, where *`host_name`* is a list of nodes on which the service should be created:

   ```
   C:\..bin> oramtsctl.exe -new -host host_name
   ```

Related Topics

- *Oracle Clusterware Administration and Deployment Guide*
- *Oracle Services for Microsoft Transaction Server Developer's Guide for Microsoft Windows*

Using Oracle DBCA in Silent Mode to Add Database Instances to Target Nodes

Add instances to nodes on which you have extended an Oracle Clusterware home and an Oracle AI Database home.

Use Oracle DBCA in silent mode with the following syntax:

```
dbca -silent -addInstance -nodeName node_name -gdbName gdb_name
[-instanceName instance_name -sysDBAUserName sysdba -sysDBAPassword password]
```

Perform any necessary service configuration procedures.

Related Topics

- Service Management Policy
  When you use Oracle Clusterware to manage your database, you can configure startup options for each individual database service when you add the service using the `srvctl add service` command with the `-policy` parameter.
- Workload Management with Dynamic Database Services
  Workload management includes load balancing, enabling clients for Oracle Real Application Clusters (Oracle RAC), distributed transaction processing, and services.

Deleting Oracle RAC from a Cluster Node

To remove Oracle Real Application Clusters (Oracle RAC) from a cluster node, you must delete the database instance and the Oracle RAC software before removing the node from the cluster.

> **Note**
>
> If there are no database instances on the node that you want to delete, then remove Oracle RAC.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Deleting Instances from Administrator-Managed Databases
  Deleting an administrator-managed database instance involves using Oracle DBCA to delete the Oracle AI Database instance.
- Using Oracle DBCA in Silent Mode to Delete Instances from Nodes
  You can use Oracle DBCA in silent mode to delete a database instance from a node.
- Using Oracle DBCA in Interactive Mode to Delete Instances from Nodes
  To delete an Oracle Real Application Clusters (Oracle RAC) instance using Oracle DBCA in interactive mode, complete this procedure.
- Removing Oracle RAC
  This procedure removes the Oracle RAC software from the node you are deleting from the cluster and updates inventories on the remaining nodes.
- Deleting Nodes from the Cluster
  After you delete the instance, you can begin the process of deleting the node from the cluster.

Deleting Instances from Administrator-Managed Databases

Deleting an administrator-managed database instance involves using Oracle DBCA to delete the Oracle AI Database instance.

> **Note**
>
> Before deleting an instance from an Oracle RAC database using SRVCTL, do the following:
>
> - If you have services configured, then relocate the services
> - Modify the services so that each service can run on one of the remaining instances
> - Ensure that the instance to be removed from an administrator-managed database is neither a preferred nor an available instance of any service

Related Topics

- Removing Oracle RAC
  This procedure removes the Oracle RAC software from the node you are deleting from the cluster and updates inventories on the remaining nodes.
- Administering Services with SRVCTL
  Learn how to use SRVCTL to perform service administration on an Oracle Real Application Clusters (Oracle RAC) database.

Using Oracle DBCA in Silent Mode to Delete Instances from Nodes

You can use Oracle DBCA in silent mode to delete a database instance from a node.

To remove an instance, use the following command syntax. Provide a node name only if you are deleting an instance from a node other than the one on where Oracle DBCA is running as shown in the following example where *password* is the SYSDBA password:

```
dbca -silent -deleteInstance [-nodeName node_name] -gdbName gdb_name
-instanceName instance_name [-sysDBAUserName sysdba] [-sysDBAPassword password]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

The following table describes the values that you need to supply for each variable.

Table 12-1 Variables in the Oracle DBCA Silent Mode Syntax

Variable	Description
node_name	The node on which you want to add (or delete) the instance.
gdb_name	Global database name.
instance_name	Name of the instance. Provide an instance name only if you want to override the Oracle naming convention for Oracle RAC instance names.
sysdba	Name of the Oracle user with SYSDBA privileges.
password	Password for the SYSDBA user.

At this point, you have accomplished the following:

- Deregistered the selected instance from its associated Oracle Net Services listeners
- Deleted the selected database instance from the instance's configured node
- Removed the Oracle Net configuration
- Deleted the Oracle Flexible Architecture directory structure from the instance's configured node.

Using Oracle DBCA in Interactive Mode to Delete Instances from Nodes

To delete an Oracle Real Application Clusters (Oracle RAC) instance using Oracle DBCA in interactive mode, complete this procedure.

1. Verify there is a current backup of OCR.

   Run the `ocrconfig -showbackup` command to ensure there is a valid backup.

2. Start Oracle DBCA.

   Start Oracle DBCA on a node *other than* the node that hosts the instance that you want to delete. The database and the instance that you plan to delete should continue to be started and running during this step.

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

4. On the Instance Management page, select **Delete Instance**, click **Next**, and Oracle DBCA displays the List of Cluster Databases page.

5. Select an Oracle RAC database from which to delete an instance. Click **Next** and Oracle DBCA displays the List of Cluster Database Instances page. The List of Cluster Database Instances page shows the instances that are associated with the Oracle RAC database that you selected and the status of each instance.

6. On the List of Cluster Databases page, select the Oracle RAC database from which to delete the instance, as follows:

   a. On the List of Cluster Database Instances page, Oracle DBCA displays the instances that are associated with the Oracle RAC database that you selected and the status of each instance. Select the cluster database from which you will delete the instance. Click **Finish**.

   b. Click **OK** on the Confirmation dialog to proceed to delete the instance.

Copyright © 1999, 2025, Oracle and/or its affiliates.

c. Click **OK** on the next Confirmation dialog to delete the instance and related Optimal Flexible Architecture (OFA) directory structure.

   Oracle DBCA displays a progress dialog showing that Oracle DBCA is deleting the instance. During this operation, Oracle DBCA removes the instance and the instance's Oracle Net configuration.

   Click **No** and exit Oracle DBCA or click **Yes** to perform another operation. If you click **Yes**, then Oracle DBCA displays the Operations page.

7. Verify that the dropped instance's redo thread has been removed using SQL*Plus to query the `V$LOG` view from an existing instance. If the redo thread is not disabled, then disable the thread. For example:

   ```
   SQL> ALTER DATABASE DISABLE THREAD 2;
   ```

8. Verify that the instance has been removed from OCR by running the following command, where *db_unique_name* is the name of the database:

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

9. If you are deleting more than one node, then repeat these steps to delete the instances from all the nodes that you are going to delete.

Removing Oracle RAC

This procedure removes the Oracle RAC software from the node you are deleting from the cluster and updates inventories on the remaining nodes.

1. If there is a listener in the Oracle RAC home on the node you are deleting, then you must disable and stop it before deleting the Oracle RAC software. Run the following commands on any node in the cluster, specifying the name of the listener and the name of the node you are deleting:

   ```
   C:\srvctl disable listener -listener listener_name -node
   name_of_node_to_delete
   C:\srvctl stop listener -listener listener_name -node
   name_of_node_to_delete
   ```

2. Deinstall the Oracle home from the node that you are deleting by running the following command from the *Oracle_home*`\deinstall` directory:

   ```
   deinstall -local
   ```

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

Deleting Nodes from the Cluster

After you delete the instance, you can begin the process of deleting the node from the cluster.

To delete a node from the cluster, you run scripts on the node that you want to delete to remove the Oracle Grid Infrastructure installation. You then run scripts on the remaining nodes to update the node list.

Related Topics

- *Oracle Clusterware Administration and Deployment Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

13

Design and Deployment Techniques

Learn about methods to design and deploy Oracle RAC.

This chapter briefly describes database design and deployment techniques for Oracle Real Application Clusters (Oracle RAC) environments. It also describes considerations for high availability and provides general guidelines for various Oracle RAC deployments.

- Deploying Oracle RAC for High Availability
  Learn how to deploy Oracle RAC for high availability.
- General Design Considerations for Oracle RAC
  Learn about design considerations for Oracle RAC.
- General Database Deployment Topics for Oracle RAC
  Learn about various Oracle RAC deployment considerations such as tablespace use, object creation, distributed transactions, and more.
- Introduction to Blocker Resolver
  Blocker Resolver is an Oracle Real Application Clusters (Oracle RAC) environment feature that autonomously resolves delays and keeps the resources available.

Deploying Oracle RAC for High Availability

Learn how to deploy Oracle RAC for high availability.

Many customers implement Oracle RAC to provide high availability for their Oracle AI Database applications. For true high availability, you must make the entire infrastructure of the application highly available. This requires detailed planning to ensure there are no single points of failure throughout the infrastructure. Even though Oracle RAC makes your database highly available, if a critical application becomes unavailable, then your business can be negatively affected. For example, if you choose to use the Lightweight Directory Access Protocol (LDAP) for authentication, then you must make the LDAP server highly available. If the database is up but the users cannot connect to the database because the LDAP server is not accessible, then the entire system appears to be down to your users.

- About Designing High Availability Systems
  For mission critical systems, you must be able to perform failover and recovery, and your environment must be resilient to all types of failures.
- Best Practices for Deploying Oracle RAC in High Availability Environments
  You can improve performance in your Oracle RAC environment by following the best practices described here.
- Consolidating Multiple Applications in Cluster Databases
  Learn about consolidating applications in Oracle RAC databases.
- Scalability of Oracle RAC
  Learn about your choices for improving Oracle RAC scalability.

About Designing High Availability Systems

For mission critical systems, you must be able to perform failover and recovery, and your environment must be resilient to all types of failures.

Copyright © 1999, 2025, Oracle and/or its affiliates.

For mission critical systems, you must be able to perform failover and recovery, and your environment must be resilient to all types of failures. To reach these goals, start by defining service level requirements for your business. The requirements should include definitions of maximum transaction response time and recovery expectations for failures within the data center (such as for node failure) or for disaster recovery (if the entire data center fails). Typically, the service level objective is a target response time for work, regardless of failures. Determine the recovery time for each redundant component. Even though you may have hardware components that are running in an active/active mode, do not assume that if one component fails the other hardware components can remain operational while the faulty components are being repaired. Also, when components are running in active/passive mode, perform regular tests to validate the failover time. For example, recovery times for storage channels can take minutes. Ensure that the outage times are within your business' service level agreements, and where they are not, work with the hardware vendor to tune the configuration and settings.

When deploying mission critical systems, the testing should include functional testing, destructive testing, and performance testing. Destructive testing includes the injection of various faults in the system to test the recovery and to make sure it satisfies the service level requirements. Destructive testing also allows the creation of operational procedures for the production system.

To help you design and implement a mission critical or highly available system, Oracle provides a range of solutions for every organization regardless of size. Small work groups and global enterprises alike are able to extend the reach of their critical business applications. With Oracle and the Internet, applications and their data are now reliably accessible everywhere, at any time. The Oracle Maximum Availability Architecture (MAA) is the Oracle best practices blueprint that is based on proven Oracle high availability technologies and recommendations. The goal of the MAA is to remove the complexity in designing an optimal high availability architecture.

Related Topics

- Oracle Maximum Availability Architecture (MAA)

Best Practices for Deploying Oracle RAC in High Availability Environments

You can improve performance in your Oracle RAC environment by following the best practices described here.

Applications can take advantage of many Oracle AI Database, Oracle Clusterware, and Oracle RAC features and capabilities to minimize or mask any failure in the Oracle RAC environment. For example, you can:

- Remove TCP/IP timeout waits by using the VIP address to connect to the database.
- Create detailed operational procedures and ensure you have the appropriate support contracts in place to match defined service levels for all components in the infrastructure.
- Take advantage of the Oracle RAC Automatic Workload Management features such as connect time failover, Fast Connection Failover, Fast Application Notification, and the Load Balancing Advisory.
- Place voting disks on separate volume groups to mitigate outages due to slow I/O throughput. To remain functional after the failure of *x* voting devices, configure 2*x* + 1 mirrors.
- Place OCR with I/O service times in the order of 2 milliseconds (ms) or less.
- Tune database recovery using the `FAST_START_MTTR_TARGET` initialization parameter.
- Use Oracle Automatic Storage Management (Oracle ASM) to manage database storage.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Ensure that strong change control procedures are in place.
- Check the surrounding infrastructure for high availability and resiliency, such as LDAP, NIS, and DNS. These entities affect the availability of your Oracle RAC database. If possible, perform a local backup procedure routinely.
- Use Oracle Enterprise Manager to administer your entire Oracle RAC environment, not only the Oracle RAC database. Use Oracle Enterprise Manager to create and modify services, and to start and stop the cluster database instances and the cluster database.
- Use Recovery Manager (RMAN) to back up, restore, and recover data files, control files, server parameter files (SPFILEs) and archived redo log files. You can use RMAN with a media manager to back up files to external storage. You can also configure parallelism when backing up or recovering Oracle RAC databases. In Oracle RAC, RMAN channels can be dynamically allocated across all of the Oracle RAC instances. Channel failover enables failed operations on one node to continue on another node. You can start RMAN from Oracle Enterprise Manager Backup Manager or from the command line.
- If you use sequence numbers, then always use `CACHE` with the `NOORDER` option for optimal performance in sequence number generation. With the `CACHE` option, however, you may have gaps in the sequence numbers. If your environment cannot tolerate sequence number gaps, then use the `NOCACHE` option or consider pre-generating the sequence numbers. If your application requires sequence number ordering but can tolerate gaps, then use `CACHE` and `ORDER` to cache and order sequence numbers in Oracle RAC. If your application requires ordered sequence numbers without gaps, then use `NOCACHE` and `ORDER`. The `NOCACHE` and `ORDER` combination has the most negative effect on performance compared to other caching and ordering combinations.

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

  You can use scalable sequences to provide better data load scalability instead of configuring a very large sequence cache. Scalable sequences improve the performance of concurrent data load operations, especially when the sequence values are used for populating primary key columns of tables.
- If you use indexes, then consider alternatives, such as reverse key indexes to optimize index performance. Reverse key indexes are especially helpful if you have frequent inserts to one side of an index, such as indexes that are based on insert date.

Related Topics

- Workload Management with Dynamic Database Services
  Workload management includes load balancing, enabling clients for Oracle Real Application Clusters (Oracle RAC), distributed transaction processing, and services.
- Configuring Recovery Manager and Archiving
  You can configure Oracle Recovery Manager (Oracle RMAN) to support your Oracle RAC environment.
- Making a Sequence Scalable

Consolidating Multiple Applications in Cluster Databases

Learn about consolidating applications in Oracle RAC databases.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Many people want to consolidate multiple databases in a single cluster. Oracle Clusterware and Oracle RAC support both types of consolidation.

Creating a cluster with a single pool of storage that is managed by Oracle ASM provides the infrastructure to manage multiple databases whether they are single-instance databases or Oracle RAC databases.

- Managing Capacity During Consolidation
  Learn how to manage capacity during consolidation.
- Managing the Global Cache Service Processes During Consolidation
  Learn how to manage the global cache services processes during consolidation.
- Using Oracle AI Database Cloud for Consolidation
  A database cloud is a set of databases integrated by the Global Data Services framework into a single virtual server that offers one or more global services while ensuring high performance, availability, and optimal use of resources.

Managing Capacity During Consolidation

Learn how to manage capacity during consolidation.

With Oracle RAC databases, you can adjust the number of instances, and which nodes run instances within a given database based, on your workload requirements. Features such as cluster-managed services enable you to manage multiple workloads on a single database or across multiple databases.

It is important to properly manage the capacity in the cluster when adding work. The processes that manage the cluster,including processes both from Oracle Clusterware and the database, must be able to obtain CPU resources in a timely fashion and must be given higher priority in the system. You can use cluster configuration policies to manage resources at the cluster level.

Managing the Global Cache Service Processes During Consolidation

Learn how to manage the global cache services processes during consolidation.

Oracle recommends that the number of real time Global Cache Service Processes (LMS*n*) on a server is less than or equal to the number of processors. (Note that this is the number of recognized CPUs that includes cores. For example, a dual-core CPU is considered to be two CPUs.) It is important that you load test your system when adding instances on a node to ensure that you have enough capacity to support the workload.

If you are consolidating many small databases into a cluster, then you may want to reduce the number of LMS*n* created by the Oracle RAC instance. By default, Oracle AI Database calculates the number of processes based on the number of CPUs it finds on the server. This calculation may result in more LMS*n* processes than is needed for the Oracle RAC instance. One LMS process may be sufficient for up to 4 CPUs.To reduce the number of LMS*n* processes, set the `GCS_SERVER_PROCESSES` initialization parameter minimally to a value of 1. Add a process for every four CPUs needed by the application. In general, it is better to have few busy LMS*n* processes. Oracle AI Database calculates the number of processes when the instance is started, and you must restart the instance to change the value.

Using Oracle AI Database Cloud for Consolidation

A database cloud is a set of databases integrated by the Global Data Services framework into a single virtual server that offers one or more global services while ensuring high performance, availability, and optimal use of resources.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Global Data Services manages these virtualized resources with minimum administration overhead, and allows the database cloud to quickly scale to handle additional client requests. The databases that constitute a cloud can be globally distributed, and clients can connect to the database cloud by specifying only a service name, without needing to know anything about the components and topology of the cloud.

A database cloud can be comprised of multiple database pools. A database pool is a set of databases within a database cloud that provide a unique set of global services and belong to a certain administrative domain. Partitioning of cloud databases into multiple pools simplifies service management and provides higher security by allowing each pool to be administered by a different administrator. A database cloud can span multiple geographic regions. A region is a logical boundary that contains database clients and servers that are considered to be close to each other. Usually a region corresponds to a data center, but multiple data centers can be in the same region if the network latencies between them satisfy the service-level agreements of the applications accessing these data centers.

Global services enable you to integrate locally and globally distributed, loosely coupled, heterogeneous databases into a scalable and highly available private database cloud. This database cloud can be shared by clients around the globe. Using a private database cloud provides optimal utilization of available resources and simplifies the provisioning of database services.

Related Topics

- *Oracle AI Database Global Data Services Concepts and Administration Guide*

Scalability of Oracle RAC

Learn about your choices for improving Oracle RAC scalability.

Oracle RAC provides concurrent, transactionally consistent access to a single copy of your data from multiple systems. It provides scalability beyond the capacity of a single server. If your application scales transparently on symmetric multiprocessing (SMP) servers, then the application should scale well on Oracle RAC without making application code changes.

Traditionally, when a database server runs out of capacity, it is replaced with a new, larger server. As servers grow in capacity, they become more expensive. However, for Oracle RAC databases, you have alternatives for increasing the capacity:

- You can migrate applications that traditionally run on large SMP servers to run on clusters of small servers.
- You can maintain the investment in the current hardware and add a new server to the cluster (or create or add a new cluster) to increase the capacity.

Adding servers to a cluster with Oracle Clusterware and Oracle RAC does not require an outage. As soon as the new instance is started, the application can take advantage of the extra capacity.

All servers in the cluster must run the same operating system and same version of Oracle AI Database but the servers do not have to have the same capacity. With Oracle RAC, you can build a cluster that fits your needs, whether the cluster is made up of servers where each server is a two-CPU commodity server or clusters where the servers have 32 or 64 CPUs in each server. The Oracle parallel processing feature allows a single SQL statement to be divided up into multiple processes, where each process completes a subset of work. In an Oracle RAC environment, you can define the parallel processes to run only on the instance where the user is connected or to run across multiple instances in the cluster.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Related Topics

- Cloning Oracle RAC to Nodes in a New Cluster
  Learn how to clone Oracle Real Application Clusters (Oracle RAC) database homes on Linux and Unix systems to nodes in a new cluster.
- Using Cloning to Extend Oracle RAC to Nodes in the Same Cluster
  Learn how to use cloning to extend Oracle RAC nodes within a cluster.
- Adding and Deleting Oracle RAC from Nodes on Linux and Unix Systems
  Extend an existing Oracle Real Application Clusters (Oracle RAC) home to other nodes and instances in the cluster, and delete Oracle RAC from nodes and instances in the cluster.
- Adding and Deleting Oracle RAC from Nodes on Windows Systems
  Use these procedures to extend an existing Oracle Real Application Clusters (Oracle RAC) home on Microsoft Windows to other nodes and instances in the cluster, or delete Oracle RAC from nodes and instances in the cluster.

General Design Considerations for Oracle RAC

Learn about design considerations for Oracle RAC.

This section briefly describes database design and deployment techniques for Oracle RAC environments. It also describes considerations for high availability and provides general guidelines for various Oracle RAC deployments.

Consider performing the following steps during the design and development of applications that you are deploying on an Oracle RAC database:

1. Tune the design and the application
2. Tune the memory and I/O
3. Tune contention
4. Tune the operating system

> **Note**
>
> If an application does not scale on an SMP system, then moving the application to an Oracle RAC database cannot improve performance.

Consider using hash partitioning for insert-intensive online transaction processing (OLTP) applications. Hash partitioning:

- Reduces contention on concurrent inserts into a single database structure
- Affects sequence-based indexes when indexes are locally partitioned with a table and tables are partitioned on sequence-based keys
- Is transparent to the application

If you use hash partitioning for tables and indexes for OLTP environments, then you can greatly improve performance in your Oracle RAC database. Note that you cannot use index range scans on an index with hash partitioning.

Copyright © 1999, 2025, Oracle and/or its affiliates.

General Database Deployment Topics for Oracle RAC

Learn about various Oracle RAC deployment considerations such as tablespace use, object creation, distributed transactions, and more.

This section describes considerations when deploying Oracle RAC databases. Oracle RAC database performance is not compromised if you do not employ these techniques. If you have an effective noncluster design, then your application will run well on Oracle RAC.

- Tablespace Use in Oracle RAC
  Learn how to optimize tablespace use in Oracle RAC.
- Object Creation and Performance in Oracle RAC
  Learn about object creation and performance in Oracle RAC.
- Node Addition and Deletion and the SYSAUX Tablespace in Oracle RAC
  Learn how adding and deleting nodes affects the SYSAUX tablespace in Oracle RAC.
- Distributed Transactions and Oracle RAC
  Learn about distributed transactions in Oracle RAC.
- Deploying OLTP Applications in Oracle RAC
  Learn about deploying OTLP applications in Oracle RAC.
- Flexible Implementation with Cache Fusion
  Learn about flexible workload implementation with cache fusion in Oracle RAC.
- Deploying Data Warehouse Applications in Oracle RAC
  Learn how to deploy data warehouse applications in Oracle RAC
- Data Security Considerations in Oracle RAC
  Learn about transparent data encryption and Microsoft Windows firewall considerations for Oracle RAC data security.

Tablespace Use in Oracle RAC

Learn how to optimize tablespace use in Oracle RAC.

In addition to using locally managed tablespaces, you can further simplify space administration by using automatic segment space management (ASSM) and automatic undo management.

ASSM distributes instance workloads among each instance's subset of blocks for inserts. This improves Oracle RAC performance because it minimizes block transfers. To deploy automatic undo management in an Oracle RAC environment, each instance must have its own undo tablespace.

Object Creation and Performance in Oracle RAC

Learn about object creation and performance in Oracle RAC.

As a general rule, only use DDL statements for maintenance tasks and avoid running DDL statements during peak system operation periods. In most systems, the amount of new object creation and other DDL statements should be limited. As in noncluster Oracle AI Databases, excessive object creation and deletion can increase performance overhead.

Node Addition and Deletion and the SYSAUX Tablespace in Oracle RAC

Learn how adding and deleting nodes affects the SYSAUX tablespace in Oracle RAC.

Copyright © 1999, 2025, Oracle and/or its affiliates.

If you add nodes to your Oracle RAC database environment, then you may need to increase the size of the SYSAUX tablespace. Conversely, if you remove nodes from your cluster database, then you may be able to reduce the size of your SYSAUX tablespace.

See Also

Your platform-specific Oracle RAC installation guide for guidelines about sizing the SYSAUX tablespace for multiple instances

Distributed Transactions and Oracle RAC

Learn about distributed transactions in Oracle RAC.

If you are running XA Transactions in Oracle RAC environments and the performance is poor, then direct all of the branches of a tightly coupled distributed transaction to the same instance by creating multiple Oracle Distributed Transaction Processing (DTP) services, with one or more on each Oracle RAC instance.

Each DTP service is a singleton service that is available on one and only one Oracle RAC instance. All access to the database server for distributed transaction processing must be done by way of the DTP services. Ensure that all of the branches of a single global distributed transaction use the same DTP service. In other words, a network connection descriptor, such as a TNS name, a JDBC URL, and so on, must use a DTP service to support distributed transaction processing.

Related Topics

- Distributed Transaction Processing in Oracle RAC
  Learn how Oracle Real Application Clusters (Oracle RAC) supports global (XA) transactions and DTP processing
- *Oracle AI Database Development Guide*

Deploying OLTP Applications in Oracle RAC

Learn about deploying OTLP applications in Oracle RAC.

Cache Fusion makes Oracle RAC databases the optimal deployment servers for online transaction processing (OLTP) applications. This is because these types of applications require:

- High availability if there are failures
- Scalability to accommodate increased system demands
- Load balancing according to demand fluctuations

The high availability features of Oracle AI Database and Oracle RAC can re-distribute and load balance workloads to remaining instances without interrupting processing. Oracle RAC also provides excellent scalability so that if you add or replace a node, then Oracle AI Database updates resources and re-distributes processing loads.

Flexible Implementation with Cache Fusion

Learn about flexible workload implementation with cache fusion in Oracle RAC.

Copyright © 1999, 2025, Oracle and/or its affiliates.

To accommodate the frequently changing workloads of online transaction processing systems, Oracle RAC remains flexible and dynamic despite changes in system load and system availability. Oracle RAC addresses a wide range of service levels that, for example, fluctuate due to:

- Varying user demands
- Peak scalability issues like trading storms (bursts of high volumes of transactions)
- Varying availability of system resources

Deploying Data Warehouse Applications in Oracle RAC

Learn how to deploy data warehouse applications in Oracle RAC

This section discusses how to deploy data warehouse systems in Oracle RAC environments by briefly describing the data warehouse features available in shared disk architectures.

- Parallelism for Data Warehouse Applications on Oracle RAC
  Learn about parallelism for data warehouse applications in Oracle RAC.
- Parallel Processing in Data Warehouse Systems and Oracle RAC
  Use parallel processing to improve data warehouse performance in Oracle RAC.

Parallelism for Data Warehouse Applications on Oracle RAC

Learn about parallelism for data warehouse applications in Oracle RAC.

Oracle RAC is ideal for data warehouse applications because it augments the noncluster benefits of Oracle AI Database. Oracle RAC does this by maximizing the processing available on all of the nodes that belong to an Oracle RAC database to providespeed-up for data warehouse systems.

The query optimizer considers parallel processing when determining the optimal processing plans. The default cost model for the query optimizer is **CPU+I/O** and the cost unit is **time**. In Oracle RAC, the query optimizer dynamically computes intelligent defaults for parallelism based on the number of processors in the nodes of the cluster. An evaluation of the costs of alternative access paths, table scans versus indexed access, for example, takes into account the degree of parallelism available for the operation. This results in Oracle AI Database selecting the processing plans that are optimized for your Oracle RAC configuration.

Parallel Processing in Data Warehouse Systems and Oracle RAC

Use parallel processing to improve data warehouse performance in Oracle RAC.

Parallel processing uses multiple processes to run SQL statements on one or more CPUs and is available on both noncluster Oracle AI Databases and Oracle RAC databases.

Oracle RAC takes full advantage of parallel processing by distributing parallel processing across all available instances. The number of processes that can participate in parallel operations depends on the degree of parallelism assigned to each table or index.

Related Topics

- *Oracle AI Database Performance Tuning Guide*
- *Oracle AI Database Concepts*

Copyright © 1999, 2025, Oracle and/or its affiliates.

Data Security Considerations in Oracle RAC

Learn about transparent data encryption and Microsoft Windows firewall considerations for Oracle RAC data security.

- Transparent Data Encryption and Keystores
  Learn about transparent data encryption and keystores in Oracle RAC.
- Windows Firewall Considerations
  Learn about Microsoft Windows firewall considerations.
- Securely Run ONS Clients Using Wallets
  You can configure and use SSL certificates to set up authentication between the ONS server in the database tier and the notification client in the middle tier.

Transparent Data Encryption and Keystores

Learn about transparent data encryption and keystores in Oracle RAC.

Oracle AI Database enables Oracle RAC nodes to share the keystore (wallet). This eliminates the need to manually copy and synchronize the keystore across all nodes. Oracle recommends that you create the keystore on a shared file system. This allows all instances to access the same shared keystore.

Oracle RAC uses keystores in the following ways:

1. Any keystore operation, such as opening or closing the keystore, performed on any one Oracle RAC instance is applicable for all other Oracle RAC instances. This means that when you open and close the keystore for one instance, then it opens and closes the keystore for all Oracle RAC instances.

   Starting with Oracle AI Database, the parameter `ENCRYPTION_WALLET_LOCATION` is desupported.
   To store and retrieve the TDE wallet, use the `WALLET_ROOT` structure (introduced with Oracle Database 18c).
2. A master key rekey performed on one instance is applicable for all instances. When a new Oracle RAC node comes up, it is aware of the current keystore open or close status.
3. Do not issue any keystore `ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN` or `CLOSE` SQL statements while setting up or changing the master key.

Oracle does not support the use of individual TDE wallets for each Oracle RAC node. Instead, use shared wallets for TDE in the Oracle RAC environment. This enables all of the instances to access the same shared software keystore.

Related Topics

- *Oracle AI Database Advanced Security Guide*

Windows Firewall Considerations

Learn about Microsoft Windows firewall considerations.

By default, all installations of Windows Server 2003 Service Pack 1 and higher enable the Windows Firewall to block virtually all TCP network ports to incoming connections. As a result, any Oracle products that listen for incoming connections on a TCP port will not receive any of those connection requests, and the clients making those connections will report errors.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Depending upon which Oracle products you install and how they are used, you may need to perform additional Windows post-installation configuration tasks so that the Firewall products are functional on Windows Server 2003.

Securely Run ONS Clients Using Wallets

You can configure and use SSL certificates to set up authentication between the ONS server in the database tier and the notification client in the middle tier.

JDBC or Oracle Universal Connection Pools, and other Oracle RAC features, such as Fast Connection Failover, subscribe to notifications from the Oracle Notification Service (ONS) running on Oracle RAC nodes. These connections are not usually authenticated.

1. Starting with Oracle Database 18c, a default wallet is created during the installation of Oracle Grid Infrastructure.
2. If you are running a client-side ONS daemon on the middle tier, then there are two possible configurations:
   - ONS started from OPMN (as in OracleAS 10.1.3.x), which uses `opmn.xml` for its configuration.
   - ONS started standalone (as when using ONSCTL), which uses `ons.config` for its configuration.

   For the first configuration, refer to the OPMN Administrator's Guide for the Oracle Application Server release. This involves modifying the `opmn.xml` file to specify the wallet location.

   For the second configuration, the client-side ONS daemon can, potentially, run on different servers. Copy the wallet from step 1 to those client-side servers and specify the path on that client-side server in either the `ons.config` file or in the `opmn.xml` file.
3. If you are running a remote ONS configuration without a client-side ONS daemon, then configure the client-side server.
   a. Export the ONS resource to the client cluster.

      Use a command similar to the following, where *cluster_name* is the name of the remote cluster, and *filename* is the name of the file to which the credentials data will be written.

      ```
      $ srvctl export ons -clientcluster cluster_name -clientdata filename
      ```

   b. Specify the path on the client-side server.

      Modify either the `ons.config` file or the `opmn.xml` file to point to the location of the copied file.

Related Topics

- Overview of ONS Configuration File

Related Topics

- Remote Configuration of ONS

Related Topics

- Client Side ONS Daemon Configuration

Copyright © 1999, 2025, Oracle and/or its affiliates.

Introduction to Blocker Resolver

Blocker Resolver is an Oracle Real Application Clusters (Oracle RAC) environment feature that autonomously resolves delays and keeps the resources available.

Enabled by default, Blocker Resolver:

- Reliably detects database delays and deadlocks
- Autonomously resolves database delays and deadlocks
- Logs all detections and resolutions
- Provides SQL interface to configure sensitivity (Normal/High) and trace file sizes

A database delays when a session blocks a chain of one or more sessions. The blocking session holds a resource such as a lock or latch that prevents the blocked sessions from progressing. The chain of sessions has a root or a final blocker session, which blocks all the other sessions in the chain. Blocker Resolver resolves these issues autonomously by detecting and resolving the delays.

- Blocker Resolver Architecture
  Blocker Resolver autonomously runs as a `DIA0` task within the database.
- Optional Configuration for Blocker Resolver
  You can adjust the sensitivity, and control the size and number of the log files used by Blocker Resolver.
- Blocker Resolver Diagnostics and Logging
  Blocker Resolver autonomously resolves delays and continuously logs the resolutions in the database alert logs and the diagnostics in the trace files.

Blocker Resolver Architecture

Blocker Resolver autonomously runs as a `DIA0` task within the database.

Blocker Resolver works in the following three phases:

- **Detect:** In this phase, Blocker Resolver collects the data on all the nodes and detects the sessions that are waiting for the resources held by another session.
- **Analyze:** In this phase, Blocker Resolver analyzes the sessions detected in the **Detect** phase to determine if the sessions are part of a potential delay. If the sessions are suspected as delayed, Blocker Resolver then waits for a certain threshold time period to ensure that the sessions are delayed.
- **Verify:** In this phase, after the threshold time period is up, Blocker Resolver verifies that the sessions are delayed and selects a session that's causing the delay.

After selecting the session that's causing the delay, Blocker Resolver applies resolution methods on that session. If the chain of sessions or the delay resolves automatically, then Blocker Resolver does not apply delay resolution methods. However, if the delay does not resolve by itself, then Blocker Resolver resolves the delay by terminating the session that's causing the delay. If terminating the session fails, then Blocker Resolver terminates the process of the session. This entire process is autonomous and does not block resources for a long period and does not affect the performance.

For example, if a high rank session is included in the chain of delayed sessions, then Blocker Resolver expedites the termination of the session that's causing the delay. Termination of the

Copyright © 1999, 2025, Oracle and/or its affiliates.

session that's causing the delay prevents the high rank session from waiting too long and helps to maintain performance objective of the high rank session.

Optional Configuration for Blocker Resolver

You can adjust the sensitivity, and control the size and number of the log files used by Blocker Resolver.

> **Note**
>
> The `DBMS_HANG_MANAGER` package is deprecated in Oracle AI Database 26ai. Use `DBMS_BLOCKER_RESOLVER` instead. The `DBMS_HANG_MANAGER` package provides a method of changing some configuration parameters and constraints to address session issues. This package is being replaced with `DBMS_BLOCKER_RESOLVER`. `DBMS_HANG_MANAGER` can be removed in a future release.

Sensitivity

If Blocker Resolver detects a delay, then Blocker Resolver waits for a certain threshold time period to ensure that the sessions are delayed. Change threshold time period by using `DBMS_BLOCKER_RESOLVER` to set the `sensitivity` parameter to either `Normal` or `High`. If the `sensitivity` parameter is set to `Normal`, then Blocker Resolver waits for the default time period. However, if the sensitivity is set to `High`, then the time period is reduced by 50%.

By default, the `sensitivity` parameter is set to `Normal`. To set Blocker Resolver sensitivity, run the following commands in SQL*Plus as `SYS` user:

- To set the `sensitivity` parameter to `Normal`:

  ```
  exec dbms_blocker_resolver.set(dbms_blocker_resolver.sensitivity,
  dbms_blocker_resolver.sensitivity_normal);
  ```

- To set the `sensitivity` parameter to `High`:

  ```
  exec dbms_blocker_resolver.set(dbms_blocker_resolver.sensitivity,
  dbms_blocker_resolver.sensitivity_high);
  ```

Size of the Trace Log File

The Blocker Resolver logs detailed diagnostics of the delays in the trace files with `_base_` in the file name. Change the size of the trace files in bytes with the `base_file_size_limit` parameter. Run the following command in SQL*Plus, for example, to set the trace file size limit to 100 MB:

```
exec dbms_blocker_resolver.set(dbms_blocker_resolver.base_file_size_limit,
104857600);
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Number of Trace Log Files

The base Blocker Resolver trace files are part of a trace file set. Change the number of trace files in trace file set with the `base_file_set_count` parameter. Run the following command in SQL*Plus, for example, to set the number of trace files in trace file set to 6:

```
exec dbms_blocker_resolver.set(dbms_blocker_resolver.base_file_set_count,6);
```

By default, `base_file_set_count` parameter is set to 5.

Blocker Resolver Diagnostics and Logging

Blocker Resolver autonomously resolves delays and continuously logs the resolutions in the database alert logs and the diagnostics in the trace files.

Blocker Resolver logs the resolutions in the database alert logs as Automatic Diagnostic Repository (ADR) incidents with incident code `ORA-32701`.

You also get detailed diagnostics about the delay detection in the trace files. Trace files and alert logs have file names starting with *`database instance`*`_dia0_`.

- The trace files are stored in the `$ `*`ADR_BASE`*`/diag/rdbms/`*`database name/database instance`*`/incident/`*`incdir_xxxxxx`* directory
- The alert logs are stored in the `$ `*`ADR_BASE`*`/diag/rdbms/`*`database name/database instance`*`/trace` directory

Example 13-1 Blocker Resolver Trace File for a Local Instance

This example shows an example of the output you see for Blocker Resolver for the local database instance

```
Trace Log File .../oracle/log/diag/rdbms/hm1/hm11/incident/incdir_111/
hm11_dia0_11111_i111.trc
Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production
...
*** 2016-07-16T12:39:02.715475-07:00
HM: Hang Statistics - only statistics with non-zero values are listed

            current number of active sessions 3
              current number of hung sessions 1
  instance health (in terms of hung sessions) 66.67%
       number of cluster-wide active sessions 9
         number of cluster-wide hung sessions 5
   cluster health (in terms of hung sessions) 44.45%

*** 2016-07-16T12:39:02.715681-07:00
Resolvable Hangs in the System
                     Root       Chain Total               Hang
   Hang Hang         Inst Root  #hung #hung  Hang   Hang  Resolution
     ID Type Status  Num  Sess   Sess  Sess  Conf   Span  Action
  ----- ---- -------- ---- ----- ----- ----- ------ ------ -------------------
      1 HANG RSLNPEND    3    44     3     5   HIGH GLOBAL Terminate Process
  Hang Resolution Reason: Although hangs of this root type are typically
    self-resolving, the previously ignored hang was automatically resolved.
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Example 13-2 Error Message in the Alert Log Indicating a Delayed Session

This example shows an example of a Blocker Resolver alert log on the primary instance

```
2016-07-16T12:39:02.616573-07:00
Errors in file .../oracle/log/diag/rdbms/hm1/hm1/trace/hm1_dia0_i1111.trc
(incident=1111):
ORA-32701: Possible hangs up to hang ID=1 detected
Incident details in: .../oracle/log/diag/rdbms/hm1/hm1/incident/incdir_1111/
hm1_dia0_11111_i1111.trc
2016-07-16T12:39:02.674061-07:00
DIA0 requesting termination of session sid:44 with serial # 23456
(ospid:34569) on instance 3
     due to a GLOBAL, HIGH confidence hang with ID=1.
     Hang Resolution Reason: Although hangs of this root type are typically
    self-resolving, the previously ignored hang was automatically resolved.
DIA0: Examine the alert log on instance 3 for session termination status of
hang with ID=1.
```

Example 13-3 Error Message in the Alert Log Showing a Session Delay Resolved by Blocker Resolver

This example shows an example of a Blocker Resolver alert log on the local instance for resolved delays

```
2016-07-16T12:39:02.707822-07:00
Errors in file .../oracle/log/diag/rdbms/hm1/hm11/trace/hm11_dia0_11111.trc
(incident=169):
ORA-32701: Possible hangs up to hang ID=1 detected
Incident details in: .../oracle/log/diag/rdbms/hm1/hm11/incident/incdir_169/
hm11_dia0_30676_i169.trc
2016-07-16T12:39:05.086593-07:00
DIA0 terminating blocker (ospid: 30872 sid: 44 ser#: 23456) of hang with ID =
1
     requested by master DIA0 process on instance 1
     Hang Resolution Reason: Although hangs of this root type are typically
    self-resolving, the previously ignored hang was automatically resolved.
     by terminating session sid:44 with serial # 23456 (ospid:34569)
...
DIA0 successfully terminated session sid:44 with serial # 23456 (ospid:34569)
with status 0.
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

14

Monitoring Performance

Learn how to monitor and tune the performance of your Oracle Real Application Clusters (Oracle RAC) database.

- Monitoring and Tuning Oracle RAC Databases
  Learn about monitoring and tuning Oracle Real Application Clusters (Oracle RAC) databases, and about how you can use the Database Reliability Framework to assist you in these tasks.
- Verifying the Interconnect Settings for Oracle RAC
  To verify the interconnect settings for Oracle Real Application Clusters (Oracle RAC), you can use SQL statements.
- Influencing Interconnect Processing
  After your interconnect is operative, you cannot significantly influence its performance. However, you can influence an interconnect protocol's efficiency by adjusting the interprocess communication (IPC) buffer sizes.
- Performance Views in Oracle RAC
  To obtain performance information about your Oracle Real Application Clusters (Oracle RAC) database, you can query either instance-specific views, or dynamic performance views for the entire cluster.
- Creating Oracle RAC Data Dictionary Views with CATCLUST.SQL
  If you did not create your Oracle RAC database with Oracle DBCA, then you must run the `CATCLUST.SQL` script to create views and tables related to Oracle RAC.
- Oracle RAC Performance Statistics
  Oracle Real Application Clusters (Oracle RAC) statistics appear either as message request counters, or as timed statistics.
- Automatic Workload Repository in Oracle RAC Environments
  You can use Automatic Workload Repository to monitor performance statistics related to Oracle RAC databases.
- Active Session History Reports for Oracle RAC
  Learn about the ways that you can check the status of your Oracle Real Application Clusters (Oracle RAC) database by using Active Session History (ASH) reports.
- Monitoring Oracle RAC Statistics and Wait Events
  Learn about wait events and statistics specific to Oracle RAC and how to interpret them when assessing performance data generated by the Automatic Workload Repository (AWR), Statspack, or by ad-hoc queries of the dynamic performance views.

Monitoring and Tuning Oracle RAC Databases

Learn about monitoring and tuning Oracle Real Application Clusters (Oracle RAC) databases, and about how you can use the Database Reliability Framework to assist you in these tasks.

- Overview of Monitoring Oracle RAC and Oracle Clusterware
  Learn about the monitoring capabilities of Oracle Enterprise Manager, including the Cluster Database Homepage, the Interconnects page, and the Cluster Database Performance page.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Tuning Oracle RAC Databases
  All of the noncluster tuning practices for Oracle AI Database also apply to Oracle Real Application Clusters (Oracle RAC) databases.
- Database Reliability Framework
  The Database Reliability Framework (DRF) is a proactive and automatic monitoring and correction framework for Oracle Real Application Clusters (Oracle RAC) databases.

Overview of Monitoring Oracle RAC and Oracle Clusterware

Learn about the monitoring capabilities of Oracle Enterprise Manager, including the Cluster Database Homepage, the Interconnects page, and the Cluster Database Performance page.

- Monitoring Oracle RAC and Oracle Clusterware with Oracle Enterprise Manager
  Using Oracle Enterprise Manager is the preferred method for monitoring Oracle Real Application Clusters (Oracle RAC) and Oracle Clusterware.
- The Cluster Database Home Page
  Using the Oracle Enterprise Manager Cluster Database Home page, you can use a client browser to monitor the status of both Oracle Clusterware and Oracle Real Application Clusters (Oracle RAC) environments.
- The Interconnects Page
  Using the Oracle Enterprise Manager Interconnects page, you can use a client browser to monitor private network status, and troubleshoot cluster wait events.
- The Cluster Database Performance Page
  The Oracle Enterprise Manager Cluster Database Performance page provides a quick glimpse of the performance statistics for an Oracle Real Application Clusters (Oracle RAC) database.

Monitoring Oracle RAC and Oracle Clusterware with Oracle Enterprise Manager

Using Oracle Enterprise Manager is the preferred method for monitoring Oracle Real Application Clusters (Oracle RAC) and Oracle Clusterware.

Oracle Enterprise Manager is an Oracle Web-based integrated management solution for monitoring and administering your computing environment. From any location where you can access a web browser, you can manage Oracle RAC databases, application servers, host computers, and Web applications, in addition to related hardware and software. For example, you can monitor your Oracle RAC database performance from your office, home, or a remote site, if you have access to a Web browser.

Oracle Enterprise Manager Cloud Control is cluster-aware and provides a central console to manage your cluster database. From the Cluster Database Home page, you can do all of the following:

- View the overall system status, such as the number of nodes in the cluster and their current status. This high-level view capability means that you do not have to access each individual database instance for details if you only want to see inclusive, aggregated information.
- View alert messages aggregated across all the instances with lists for the source of each alert message. An **alert message** is an indicator that signifies that a particular metric condition has been encountered. A **metric** is a unit of measurement used to report the system's conditions.
- Review issues that are affecting the entire cluster and those issues that are affecting individual instances.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Monitor cluster cache coherency statistics to help you identify processing trends and optimize performance for your Oracle RAC environment. Cache coherency statistics measure how well the data in caches on multiple instances is synchronized. If the data caches are completely synchronized with each other, then reading a memory location from the cache on any instance will return the most recent data written to that location from any cache on any instance.

Oracle Enterprise Manager accumulates data over specified periods of time, called collection-based data. Oracle Enterprise Manager also provides current data, called real-time data.

Related Topics

- *Oracle AI Database Get Started with Performance Tuning*
- *Oracle Clusterware Administration and Deployment Guide*

The Cluster Database Home Page

Using the Oracle Enterprise Manager Cluster Database Home page, you can use a client browser to monitor the status of both Oracle Clusterware and Oracle Real Application Clusters (Oracle RAC) environments.

The Oracle Enterprise Manager Cluster Database Home Page monitors your entire cluster environment. Monitoring can include such things as:

- Notification if there are any VIP relocations
- Status of the Oracle Clusterware on each node of the cluster using information obtained through the Configuration Verification Utility (cluvfy)
- Notification if node applications (`nodeapps`) start or stop
- Notification of issues in the Oracle Clusterware alert log for OCR, voting disk issues (if any), and node evictions

The Cluster Database Home page is similar to a noncluster Database Home page. However, on the Cluster Database Home page, Oracle Enterprise Manager displays the system state and availability. The system state includes a summary about alert messages and job activity, and links to all the database and Oracle Automatic Storage Management (Oracle ASM) instances. For example, you can track problems with services on the cluster including when a service is not running on all of the preferred instances or when a service response time threshold is not being met.

The Interconnects Page

Using the Oracle Enterprise Manager Interconnects page, you can use a client browser to monitor private network status, and troubleshoot cluster wait events.

You can use the Oracle Enterprise Manager Interconnects page to monitor the Oracle Clusterware environment. The Interconnects page shows the public and private interfaces on the cluster and the load contributed by database instances on the interconnect, including:

- Overall throughput across the private interconnect
- Notification if a database instance is using public interface due to misconfiguration
- Throughput and errors (if any) on the interconnect
- Throughput contributed by individual instances on the interconnect

All of this information is also available as collections that have a historic view, which is useful with cluster cache coherency, such as when diagnosing problems related to cluster wait events. You can access the Interconnects page by clicking the Interconnect tab on the Cluster

Copyright © 1999, 2025, Oracle and/or its affiliates.

Database home page, or by clicking the Interconnect Alerts link under Diagnostic Findings on the Oracle RAC database home page.

The Cluster Database Performance Page

The Oracle Enterprise Manager Cluster Database Performance page provides a quick glimpse of the performance statistics for an Oracle Real Application Clusters (Oracle RAC) database.

Statistics are rolled up across all the instances in the cluster database in charts. Using the links next to the charts, you can get more specific information and perform any of the following tasks:

- Identify the causes of performance issues.
- Decide whether resources need to be added or redistributed.
- Tune your SQL plan and schema for better optimization.
- Resolve performance issues

The charts on the Cluster Database Performance page include the following:

- **Chart for Cluster Host Load Average**: The Cluster Host Load Average chart in the Cluster Database Performance page shows potential problems that are outside the database. The chart shows maximum, average, and minimum load values for available nodes in the cluster for the previous hour.
- **Chart for Global Cache Block Access Latency**: Each cluster database instance has its own buffer cache in its System Global Area (SGA). Using Cache Fusion, Oracle RAC environments logically combine each instance's buffer cache to enable the database instances to process data as if the data resided on a logically combined, single cache.
- **Chart for Average Active Sessions**: The Average Active Sessions chart in the Cluster Database Performance page shows potential problems inside the database. Categories, called wait classes, show how much of the database is using a resource, such as CPU or disk I/O. Comparing CPU time to wait time helps to determine how much of the response time is consumed with useful work rather than waiting for resources that are potentially held by other processes.
- **Chart for Database Throughput**: The Database Throughput charts summarize any resource contention that appears in the Average Active Sessions chart, and also show how much work the database is performing on behalf of the users or applications. The Per Second view shows the number of transactions compared to the number of logons, and the amount of physical reads compared to the redo size per second. The Per Transaction view shows the amount of physical reads compared to the redo size per transaction. Logons is the number of users that are logged on to the database.

In addition, the **Top Activity** drill down menu on the Cluster Database Performance page enables you to see the activity by wait events, services, and instances. Plus, you can see the details about SQL/sessions by going to a prior point in time by moving the slider on the chart.

The **Cluster Database Performance** page provides a quick glimpse of the performance statistics for an Oracle RAC database. Statistics are rolled up across all of the instances in the cluster database so that users can identify performance issues without going through all the instances. To help triage the performance issues related to services, Oracle Enterprise Manager aggregates the activity data at the following levels:

- Aggregate by waits

  All the activity data is presented in 12 categories: CPU, Scheduler, User I/O, System I/O, Concurrency, Application, Commit, Configuration, Administrative, Network, Cluster and Other. The data presented is rolled up from all of the running instances.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Aggregate by services

  All the activity data is rolled up for each service. When the activity data is presented in this way, it is effortless to identify which service is most active, and needs more analysis.

- Aggregate by instances

  As a similar effort, the activity data is rolled up for each instance, if services are not the interested ones.

The aggregates are provided on the pages where the activity data is presented, including: Database Performance Page, Top Activity Page, Wait Details Page, and Service Details Page.

Tuning Oracle RAC Databases

All of the noncluster tuning practices for Oracle AI Database also apply to Oracle Real Application Clusters (Oracle RAC) databases.

Related Topics

- *Oracle AI Database Get Started with Performance Tuning*
- *Oracle AI Database Performance Tuning Guide*

Database Reliability Framework

The Database Reliability Framework (DRF) is a proactive and automatic monitoring and correction framework for Oracle Real Application Clusters (Oracle RAC) databases.

The Database Reliability Framework (DRF) monitors various metrics across different layers of the database continuously to detect problems before any disruption of service occurs. DRF improves database availability by monitoring critical events across all of the database instances in the cluster to identify root causes. It then can take corrective actions when these critical events hit certain thresholds.

After a problem is identified, an action is implemented automatically. Automatic actions include resizing internal memory structures or changing the priority of Oracle RAC processes, depending on the identified problem. For example, consider a system that has high `redo waits` with no I/O contention based on the metrics collected over time. If there is enough CPU resources available, then a possible action plan for reducing the `redo waits` is to move the LGWR process to higher priority to ensure that enough CPU resources are available. DRF can take this action automatically, drawing from metrics across the entire cluster to reach the best solution available. This capability results in problem resolution with minimal service disruption, and it performs these corrective actios before the problem multiplies over time, and affects database availability.

Verifying the Interconnect Settings for Oracle RAC

To verify the interconnect settings for Oracle Real Application Clusters (Oracle RAC), you can use SQL statements.

The interconnect and internode communication protocols can affect Cache Fusion performance. In addition, the interconnect bandwidth, its latency, and the efficiency of the IPC protocol determine the speed with which Cache Fusion processes block transfers.

To verify the interconnect settings of the Oracle RAC database instance to which you are connected, query the `V$CLUSTER_INTERCONNECTS` and `V$CONFIGURED_INTERCONNECTS` views. For example:

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example 14-1 Verify Interconnect Settings with V$CLUSTER_INTERCONNECTS

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

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

> **Note**
>
> You can query the `GV$CLUSTER_INTERCONNECTS` view to display the entries for all of the instances in the cluster.

Example 14-2 Verify Interconnect Settings with V$CONFIGURED_INTERCONNECTS

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

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

Influencing Interconnect Processing

After your interconnect is operative, you cannot significantly influence its performance. However, you can influence an interconnect protocol's efficiency by adjusting the interprocess communication (IPC) buffer sizes.

In Oracle Clusterware, the Oracle Cluster Registry (OCR) stores your system's interconnect information. To identify the interconnect for your cluster, use the Oracle Interface Configuration (`OIFCFG`) command-line utility `oifcfg getif` command, or the `OCRDUMP` utility. 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 a network interface card (NIC). For 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.

> **Notes**
>
> - You can also use `OIFCFG` command to assign private network or private IP addresses.
> - With Oracle Clusterware releases after Oracle Clusterware 12c release 2 (12.2), you can assign either IPv4 or IPv6 addresses to multiple private networks. However, you must choose one or the other protocol, and you must and use that protocol for all of the private networks in the cluster.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Related Topics

- *Oracle Clusterware Administration and Deployment Guide*
- *Oracle Clusterware Administration and Deployment Guide*
- *Oracle AI Database Reference*

Performance Views in Oracle RAC

To obtain performance information about your Oracle Real Application Clusters (Oracle RAC) database, you can query either instance-specific views, or dynamic performance views for the entire cluster.

Each instance in an Oracle Real Application Clusters (Oracle RAC) database has a set of instance-specific views, which are prefixed with `V$`.You can also query global dynamic performance views to retrieve performance information from all of the qualified instances. Global dynamic performance view names are prefixed with `GV$`.

Querying a `GV$` view retrieves the `V$` view information from all qualified instances. In addition to the `V$` information, each `GV$` view contains an extra column named `INST_ID` of data type `NUMBER`. The `INST_ID` column displays the instance number from which the associated `V$` view information was obtained.

You can use the `INST_ID` column as a filter to retrieve `V$` information from a subset of available instances. For example, the following query retrieves the information from the `V$LOCK` view for instances 2 and 5:

```
SQL> SELECT * FROM GV$LOCK WHERE INST_ID = 2 OR INST_ID = 5;
```

Related Topics

- *Oracle AI Database Reference*

Creating Oracle RAC Data Dictionary Views with CATCLUST.SQL

If you did not create your Oracle RAC database with Oracle DBCA, then you must run the `CATCLUST.SQL` script to create views and tables related to Oracle RAC.

If you did not create your Oracle Real Application Clusters (Oracle RAC) database by using Oracle DBCA, then the data dictionary setup for Oracle RAC is incomplete. To create the views and tables related to Oracle RAC, you must run the `CATCLUST.SQL` script. To run the `CATCLUST.SQL` script, the user account you use must be granted `SYSDBA` privileges.

Oracle RAC Performance Statistics

Oracle Real Application Clusters (Oracle RAC) statistics appear either 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.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Automatic Workload Repository in Oracle RAC Environments

You can use Automatic Workload Repository to monitor performance statistics related to Oracle RAC databases.

Automatic Workload Repository (AWR) automatically generates snapshots of the performance data once every hour and collects the statistics in the workload repository. In Oracle RAC environments, each AWR snapshot captures data from all active instances in the cluster. The data for each snapshot set is captured from the same point in time. AWR stores the snapshot data for all instances in the same table and the data is identified by an instance qualifier. For example, the `BUFFER_BUSY_WAIT` statistic shows the number of buffer waits on each instance. AWR does not store data that is aggregated from across the entire cluster. In other words, the data is stored for each individual instance.

Using the Automatic Database Diagnostic Monitor (ADDM), you can analyze the information collected by AWR for possible performance problems with Oracle AI Database. ADDM presents performance data from a cluster-wide perspective, thus enabling you to analyze performance on a global basis. In an Oracle RAC environment, ADDM can analyze performance using data collected from all instances and present it at different levels of granularity, including:

- Analysis for the entire cluster
- Analysis for a specific database instance
- Analysis for a subset of database instances

To perform these analyses, you can run the ADDM Advisor in ADDM for Oracle RAC mode to perform an analysis of the entire cluster; in Local ADDM mode to analyze the performance of an individual instance; or in Partial ADDM mode to analyze a subset of instances. Activate ADDM analysis using the advisor framework through Advisor Central in Oracle Enterprise Manager, or through the `DBMS_ADVISOR` and `DBMS_ADDM` PL/SQL packages.

Related Topics

- *Oracle AI Database Performance Tuning Guide*
- *Oracle AI Database PL/SQL Packages and Types Reference*

Active Session History Reports for Oracle RAC

Learn about the ways that you can check the status of your Oracle Real Application Clusters (Oracle RAC) database by using Active Session History (ASH) reports.

- Overview of ASH Reports for Oracle RAC
  To diagnose performance issues, Active Session History (ASH) reports provide information about all active sessions in Oracle Real Application Clusters (Oracle RAC) databases.
- ASH Report for Oracle RAC: Top Cluster Events
  To identify which events and instances cause a high percentage of cluster wait events, use the Active Sessions History (ASH) Top Cluster Events report.
- ASH Report for Oracle RAC: Top Remote Instance
  To identify specific instances that cause extended cluster wait periods, use the Active Sessions History (ASH) Top Remote Instance report.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Overview of ASH Reports for Oracle RAC

To diagnose performance issues, Active Session History (ASH) reports provide information about all active sessions in Oracle Real Application Clusters (Oracle RAC) databases.

ASH is an integral part of the Oracle AI Database self-management framework and is useful for diagnosing performance problems in Oracle RAC environments. ASH report statistics provide details about Oracle AI Database session activity. Oracle AI Database records information about active sessions for all active Oracle RAC instances, and stores this data in the System Global Area (SGA). Any session that is connected to the database and using CPU is considered an active session. The exception to this is sessions that are waiting for an event that belongs to the idle wait class.

ASH reports present a manageable set of data by capturing only information about active sessions. The amount of the data is directly related to the work being performed, rather than the number of sessions allowed on the system.

ASH statistics that are gathered over a specified duration can be put into ASH reports. Each ASH report is divided into multiple sections to help you identify short-lived performance problems that do not appear in the ADDM analysis. Two ASH report sections that are specific to Oracle RAC are Top Cluster Events and Top Remote Instance as described in the next two sections.

Related Topics

- *Oracle AI Database Performance Tuning Guide*

ASH Report for Oracle RAC: Top Cluster Events

To identify which events and instances cause a high percentage of cluster wait events, use the Active Sessions History (ASH) Top Cluster Events report.

The ASH report Top Cluster Events section is part of the Top Events report that is specific to Oracle RAC. The Top Cluster Events report lists events that account for the highest percentage of session activity in the cluster wait class event along with the instance number of the affected instances. You can use this information to identify which events and instances caused a high percentage of cluster wait events.

ASH Report for Oracle RAC: Top Remote Instance

To identify specific instances that cause extended cluster wait periods, use the Active Sessions History (ASH) Top Remote Instance report.

The ASH report Top Remote Instance section is part of the Top Load Profile report that is specific to Oracle Real Application Clusters (Oracle RAC). The Top Remote Instance report shows cluster wait events along with the instance numbers of the instances that accounted for the highest percentages of session activity. You can use this information to identify the instance that caused the extended cluster wait period.

Monitoring Oracle RAC Statistics and Wait Events

Learn about wait events and statistics specific to Oracle RAC and how to interpret them when assessing performance data generated by the Automatic Workload Repository (AWR), Statspack, or by ad-hoc queries of the dynamic performance views.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Oracle RAC Statistics and Events in AWR and Statspack Reports
  To evaluate statistics snapshots generated by AWR and `Statspack`, you can produce summary data reports.
- Oracle RAC Wait Events
  Analyzing and interpreting what causes sessions to wait is an important method to determine where time is spent.
- Monitoring Performance by Analyzing GCS and GES Statistics
  Learn how to determine the amount of work and cost related to inter-instance messaging and contention, examine block transfer rates and remote requests made by each transaction, and determine the number and time waited for global cache events.
- Analyzing Cache Fusion Transfer Impact Using GCS Statistics
  Learn how to monitor and tune GCS performance by identifying objects read and modified frequently and the service times imposed by the remote access.
- Analyzing Response Times Based on Wait Events
  Learn how to analyze global cache wait events that may present themselves as the top database time consumers without actually indicating a problem.

Oracle RAC Statistics and Events in AWR and Statspack Reports

To evaluate statistics snapshots generated by AWR and `Statspack`, you can produce summary data reports.

The statistics snapshots generated by Automatic Workload Repository (AWR) and the `Statspack` package set of SQL, PL/SQL, and SQL*Plus scripts can be evaluated by producing reports displaying summary data. For example, you can produce summary reports that show such data as load and cluster profiles based on regular statistics, and wait events gathered on each instance.

Most of the relevant data is summarized on the Oracle Real Application Clusters (Oracle RAC) Statistics Page. This information includes:

- Global cache load profile
- Global cache efficiency percentages—workload characteristics
- Global cache and Enqueue Service (GES)—messaging statistics

Additional Oracle RAC sections appear later in the report:

- Global enqueue statistics
- Global CR statistics
- Global `CURRENT` served statistics
- Global cache transfer statistics.

Related Topics

- *Oracle AI Database Performance Tuning Guide*

Oracle RAC Wait Events

Analyzing and interpreting what causes sessions to wait is an important method to determine where time is spent.

In Oracle RAC, the wait time is attributed to an event which reflects the exact outcome of a request. For example, when a session on an instance is looking for a block in the global cache, it does not know whether it will receive the data cached by another instance or whether it will

Copyright © 1999, 2025, Oracle and/or its affiliates.

receive a message to read from disk. The wait events for the global cache convey precise information and waiting for global cache blocks or messages is:

- Summarized in a broader category called Cluster Wait Class
- Temporarily represented by a placeholder event which is active while waiting for a block, for example:
  - `gc current block request`
  - `gc cr block request`
- Attributed to precise events when the outcome of the request is known, for example:
  - `gc current block 3-way`
  - `gc current block busy`
  - `gc cr block grant 2-way`
- Multi-block read request events when all disk reads are preferred, for example:
  - `gc cr multi block grant`
  - `gc cr multi block mixed`

In summary, the wait events for Oracle RAC convey information valuable for performance analysis. They are used in Automatic Database Diagnostic Monitor (ADDM) to enable precise diagnostics of the effect of cache fusion.

Monitoring Performance by Analyzing GCS and GES Statistics

Learn how to determine the amount of work and cost related to inter-instance messaging and contention, examine block transfer rates and remote requests made by each transaction, and determine the number and time waited for global cache events.

- Analyzing the Effect of Cache Fusion in Oracle RAC
  Learn about Global Cache Service (GCS) statistics, GCS wait events, and their relation to Cache Fusion in Oracle Real Application Clusters (Oracle RAC) databases.
- Analyzing Performance Using GCS and GES Statistics
  You can monitor GCS performance by identifying data blocks and objects which are frequently used (*hot*) by all instances.

Analyzing the Effect of Cache Fusion in Oracle RAC

Learn about Global Cache Service (GCS) statistics, GCS wait events, and their relation to Cache Fusion in Oracle Real Application Clusters (Oracle RAC) databases.

The effect of accessing blocks in the global cache and maintaining coherency is represented by:

- The Global Cache Service (GCS) statistics for `current` and `cr` blocks. For example: `gc current blocks received`, `gc cr blocks received`, and so on
- The GCS wait events, for `gc current block 3-way`, `gc cr grant 2-way`, and so on

The response time for cache fusion transfers is determined by the messaging and processing times imposed by the physical interconnect components, the IPC protocol and the GCS protocol. Cache Fusion response time is not affected by disk I/O factors, other than occasional log writes. The Cache Fusion protocol requires no I/O resources to guarantee **cache coherency** for data files (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

Copyright © 1999, 2025, Oracle and/or its affiliates.

through any other cache). Oracle RAC inherently requires no more I/O to disk than a nonclustered instance requires.

Analyzing Performance Using GCS and GES Statistics

You can monitor GCS performance by identifying data blocks and objects which are frequently used (*hot*) by all instances.

High concurrency on certain blocks may be identified by GCS wait events and times.

The `gc current block busy` wait event indicates that the access to cached data blocks was delayed because they were busy either in the remote or the local cache. This could be caused by any of the following:

- The blocks were pinned
- The blocks were held up by sessions
- The blocks were delayed by a log write on a remote instance
- A session on the same instance was already accessing a block which was in transition between instances and the current session needed to wait behind it (for example, `gc current block busy`)

Use the `V$SESSION_WAIT` view to identify objects and data blocks with contention. The GCS wait events contain the file and block number for a block request in p1 and p2, respectively.

An additional segment statistic, `gc buffer busy`, has been added to quickly determine the busy objects without having to query the `V$SESSION_WAIT` view mentioned earlier.

The AWR infrastructure provides a view of active session history which can also be used to trace recent wait events and their arguments. It is therefore useful for hot block analysis. Most of the reporting facilities used by AWR and Statspack contain the object statistics and cluster wait class category, so that sampling of the views mentioned earlier is largely unnecessary.

> **Note**
>
> Oracle recommends using ADDM and AWR. However, Statspack is available for backward compatibility. Statspack provides reporting only. You must run Statspack at level 7 to collect statistics related to block contention and segment block waits.

It is advisable to run ADDM on the snapshot data collected by the AWR infrastructure to obtain an overall evaluation of the impact of the global cache. The advisory will also identify the busy objects and SQL highest cluster wait time.

Analyzing Cache Fusion Transfer Impact Using GCS Statistics

Learn how to monitor and tune GCS performance by identifying objects read and modified frequently and the service times imposed by the remote access.

Waiting for blocks to arrive can constitute a significant portion of the response time, in the same way that reading from disk can increase the block access delays. The difference is that Cache Fusion transfers are usually faster than disk access latencies.

The following wait events indicate that the remotely cached blocks were shipped to the local instance without having been busy, pinned or requiring a log flush:

Copyright © 1999, 2025, Oracle and/or its affiliates.

- `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 Automatic Database Diagnostic Monitor (ADDM) analysis usually points you to the SQL statements and database objects that can be impacted by inter-instance contention.

Any increases in the average wait times for the events mentioned in the preceding list can 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 you can diagnose no interconnect or load issues, then the accumulated time waited can usually be attributed to a few SQL statements that 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. By reviewing this column, you can identify the SQL that may need to be tuned.

Analyzing Response Times Based on Wait Events

Learn how to analyze global cache wait events that may present themselves as the top database time consumers without actually indicating a problem.

- Understanding Normal and Problem Wait Event Response Times
  To distinguish between normal wait events, and wait events that indicate a problem, review routine performance statistics, and then look for the frequent wait events that you should be aware of when interpreting performance data.
- Block-Related Wait Events
  Learn about the main wait events associated with block-related waits.
- Message-Related Wait Events
  Learn about the main wait events associated with message-related waits.
- Contention-Related Wait Events
  Learn about the main wait events associated with contention-related waits.
- Load-Related Wait Events
  Learn about the main wait events associated with load-related waits.

Understanding Normal and Problem Wait Event Response Times

To distinguish between normal wait events, and wait events that indicate a problem, review routine performance statistics, and then look for the frequent wait events that you should be aware of when interpreting performance data.

When you review the Automatic Workload Repository (AWR) reports and the `Statspack` set of SQL, PL/SQL, and SQL*Plus reports, or review the dynamic performance views, most global cache wait events that show a high total time in these reports are normal. These normal wait events can present themselves as the top database time consumers, without actually indicating a problem.

Copyright © 1999, 2025, Oracle and/or its affiliates.

If user response times increase and a high proportion of time waited is for global cache, then you should determine the cause. Most reports include a breakdown of events sorted by percentage of the total time.

It is useful to start with an Automatic Database Diagnostic Monitor (ADDM) report. The ADDM report analyzes the routinely collected performance statistics with respect to their impact, points to the objects and SQL contributing most to the time waited, and moves on to the more detailed reports produced by AWR and `Statspack`.

Block-Related Wait Events

Learn about the main wait events associated with block-related waits.

The main wait events for block-related waits are:

- `gc current block 2-way`
- `gc current block 3-way`
- `gc cr block 2-way`
- `gc cr block 3-way`

The block-related wait event statistics indicate that a block was received as either the result of a 2-way or a 3-way message, that is, the block was sent from either the primary resource requiring 1 message and 1 transfer, or was forwarded to a third node from which it was sent, requiring 2 messages and 1 block transfer.

Message-Related Wait Events

Learn about the main wait events associated with message-related waits.

The main wait events for message-related waits are:

- `gc current grant 2-way`
- `gc cr grant 2-way`

The message-related wait event statistics indicate that no block was received, because it was not cached in any instance. Instead, a global grant was given, enabling the requesting instance to read the block from disk, or to modify it.

If the time consumed by these events is high, then you can assume that the frequently used SQL causes a lot of disk I/O (in the event of the `cr grant`), or that the workload inserts a lot of data, and needs to find and format new blocks frequently (in the event of the current grant).

Contention-Related Wait Events

Learn about the main wait events associated with contention-related waits.

The main wait events for contention-related waits are:

- `gc current block busy`
- `gc cr block busy`
- `gc buffer busy acquire/release`

The contention-related wait event statistics indicate that a block was received that was pinned by a session on another node, or was deferred because a change had not yet been flushed to disk, or was deferred because of high concurrency, and therefore could not be shipped immediately. A buffer can also be busy locally when a session has already initiated a Cache

Copyright © 1999, 2025, Oracle and/or its affiliates.

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 can exacerbate the contention, which can be caused by frequent concurrent read and write accesses to the same data.

The `gc current block busy` and `gc cr block busy` wait events indicate that the local instance that is making the request did not immediately receive a current or consistent read block. The term **busy** in these events names indicates that the sending of the block was delayed on a remote instance. For example, a block cannot be shipped immediately if Oracle AI Database has not yet written the redo for the block's changes to a log file.

In comparison to `block busy` wait events, a `gc buffer busy` event indicates that Oracle AI Database cannot immediately grant access to data that is stored in the local buffer cache. This is because a global operation on the buffer is pending and the operation has not yet completed. In other words, the buffer is busy and all other processes that are attempting to access the local buffer must wait to complete.

The existence of `gc buffer busy` events also means that there is block contention that is resulting in multiple requests for access to the local block. Oracle AI Database must queue these requests. The length of time that Oracle AI Database needs to process the queue depends on the remaining service time for the block. The service time is affected by the processing time that any network latency adds, the processing time on the remote and local instances, and the length of the wait queue.

The average wait time and the total wait time should be considered when being alerted to performance issues where these particular waits have a high impact. Usually, either interconnect or load issues or SQL processing against a large shared working set can be found to be the root cause.

Load-Related Wait Events

Learn about the main wait events associated with load-related waits.

The main wait events for load-related waits are:

- `gc current block congested`
- `gc cr block congested`

The load-related wait events indicate that a delay in processing has occurred in the GCS, which is usually caused by high load or CPU saturation. To solve this kind of wait event, you add additional CPUs, provide greater load-balancing, or offload processing to different times, or to a new cluster node. For the two events mentioned here, 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.

Copyright © 1999, 2025, Oracle and/or its affiliates.

15

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

Learn about procedures for converting from Oracle AI Database single-instance databases to Oracle Real Application Clusters (Oracle RAC) and Oracle RAC One Node databases.

- Converting to Oracle RAC and Oracle RAC One Node Using Oracle DBCA
  Learn about guidelines and procedures for using Oracle DBCA to convert from single-instance Oracle AI Database to Oracle Real Application Clusters (Oracle RAC) or Oracle RAC One Node databases.
- Preparing to Convert with rconfig and Oracle Enterprise Manager
  You can use `rconfig` or Oracle Enterprise Manager to assist with converting a single-instance database installation to an Oracle RAC database.
- Converting Databases to Oracle RAC Using rconfig
  You can use the command-line utility `rconfig` to convert a single-instance database to an Oracle RAC database, or to convert it to an Oracle RAC One Node database, depending on the values you provide in the `ConvertToRAC.xml` file.
- Example of rconfig XML Input Files for ConvertToRAC
  Review this example of an XML `ConvertToRAC` input file for the `rconfig` utility.
- Postconversion Steps
  After completing the conversion of your single instance database to an Oracle Real Application Clusters (Oracle RAC) database, follow these recommendations for configuring Oracle RAC environments.

Converting to Oracle RAC and Oracle RAC One Node Using Oracle DBCA

Learn about guidelines and procedures for using Oracle DBCA to convert from single-instance Oracle AI Database to Oracle Real Application Clusters (Oracle RAC) or Oracle RAC One Node databases.

- Overview of Converting Databases to Oracle RAC Using Oracle DBCA
  Oracle DBCA provides certain benefits that can assist you with converting from single-instance Oracle AI Database to Oracle Real Application Clusters (Oracle RAC) or Oracle RAC One Node databases.
- Converting Oracle AI Database Installations to Oracle RAC Using Oracle DBCA
  To convert from a single-instance Oracle AI Database that is on a non-clustered computer to Oracle Real Application Clusters (Oracle RAC), complete each of the procedures described here.
- Converting Single Instance on a Cluster to Oracle RAC One Node Using Oracle DBCA
  To convert a single-instance Oracle AI Database to Oracle RAC One Node, use this Oracle DBCA procedure.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Converting Single Instance on a Cluster to Oracle RAC Using Oracle DBCA
  Learn when you can convert a single instance Oracle AI Database to an Oracle Real Application Clusters instance, and find out how to perform the conversion.
- Converting Single Instance on a Single Server to Oracle RAC One Node Using Oracle DBCA
  To convert a single-instance Oracle AI Database on a single server to Oracle RAC One Node, use this Oracle DBCA procedure.

Overview of Converting Databases to Oracle RAC Using Oracle DBCA

Oracle DBCA provides certain benefits that can assist you with converting from single-instance Oracle AI Database to Oracle Real Application Clusters (Oracle RAC) or Oracle RAC One Node databases.

Oracle DBCA automates the configuration of the control file attributes, creates the undo tablespaces and the redo logs, and creates the initialization parameter file entries for cluster-enabled environments. Oracle DBCA also configures Oracle Net Services, Oracle Clusterware resources, and the configuration for Oracle RAC database management using Oracle Enterprise Manager or the Server Control utility (SRVCTL).

Before you use Oracle DBCA to convert a single-instance database to an Oracle RAC or an Oracle RAC One Node database, ensure that your system meets the following conditions:

- Your system uses supported hardware and operating system software. Your system is configured properly to support an Oracle RAC database.
- The nodes have access to shared storage; for example, either Oracle Cluster File System or Oracle ASM is available and accessible from all nodes. On Linux on POWER systems, ensure that GPFS is available and accessible from all nodes.

  > **Note**
  >
  > If your cluster is a single node cluster installed with Oracle Grid Infrastructure to Manage Generic Applications, the database can also be on local file systems.

- Your applications have no design characteristics that preclude their use with cluster database processing.

If your platform supports a cluster file system, then you can use it for Oracle RAC. You can also convert to Oracle RAC and use a non-shared file system. In either case, Oracle strongly recommends that you use Oracle Universal Installer to install Oracle AI Database, which sets up the Oracle home and inventory in an identical location on each of the selected nodes in your cluster.

Related Topics

- Converting Databases
  Using `SRVCTL`, you can convert an Oracle Real Application Clusters (Oracle RAC) database with one instance to an Oracle RAC One Node database, or back to an Oracle RAC database instance.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Converting Oracle AI Database Installations to Oracle RAC Using Oracle DBCA

To convert from a single-instance Oracle AI Database that is on a non-clustered computer to Oracle Real Application Clusters (Oracle RAC), complete each of the procedures described here.

> ⚠ **Caution**
>
> You must perform each of the procedures described in the following sections, and in the order shown.

- Use Oracle DBCA to Create an Image of the Single-Instance Database
  To create a preconfigured image of your single-instance database as part of your conversion process from a single instance database to Oracle Real Application Clusters (Oracle RAC) database, use this Oracle DBCA procedure.
- Perform the Oracle Grid Infrastructure Installation
  You must complete the installation of Oracle Grid Infrastructure before you can proceed with the conversion of the single-instance Oracle AI Database to Oracle Real Application Clusters (Oracle RAC).
- Validate the Cluster
  After you install Oracle Grid Infrastructure, validate the cluster configuration by using the Configuration Verification Utility (CVU).
- Copy the Preconfigured Database Image
  After you validate the cluster, copy the preconfigured database image.
- Install the New Oracle AI Database Software with Oracle RAC
  Install the new Oracle AI Database release, selecting **Cluster Installation Mode**.

Use Oracle DBCA to Create an Image of the Single-Instance Database

To create a preconfigured image of your single-instance database as part of your conversion process from a single instance database to Oracle Real Application Clusters (Oracle RAC) database, use this Oracle DBCA procedure.

1. Navigate to the `bin` directory in `$ORACLE_HOME`, and start Oracle DBCA.
2. At the Welcome page, click **Next**.
3. On the Operations page, select **Manage Templates**, and click **Next**.
4. On the Template Management page, select **Create a database template** and **From an existing database (structure as well as data)**, then click **Next**.
5. On the Source Database page, select the database name in the Database instance list, and click **Next**.
6. Use SQL to ensure that all Pluggable Databases (PDBs) are open:

   ```
   SQL> SELECT name, open_mode FROM v$pdbs;
   ```

   If any of the PDBs are in a state other than `OPEN`, then open them using SQL.

Copyright © 1999, 2025, Oracle and/or its affiliates.

7. On the Template Properties page, enter a name for your template in the **Name** field. Oracle recommends that you use the database name.

   By default, the template files are generated in the directory `$ORACLE_HOME/assistants/dbca/templates`. You can enter a description of the file in the **Description** field, and change the template file location in the **Template** data file field.

   When you have completed the entries, click **Next**.

8. On the Location of Database Related Files page, select **Maintain the file locations**, so that you can restore the database to the current directory structure, and click **Finish**.

Oracle DBCA generates two files: a database structure file (*template_name*`.dbc`), and a database preconfigured image file (*template_name*`.dfb`).

Perform the Oracle Grid Infrastructure Installation

You must complete the installation of Oracle Grid Infrastructure before you can proceed with the conversion of the single-instance Oracle AI Database to Oracle Real Application Clusters (Oracle RAC).

To complete the installation of Oracle Grid Infrastructure, refer to the documentation for your operating system.

> **Note**
>
> The option of Installing Oracle Grid Infrastructure to Manage Generic Applications on the single server can support single server rolling database maintenance.

After Oracle Grid Infrastructure is installed on the server running the single-instance Oracle AI Database, you must register the database to Oracle Clusterware:

1. As the software owner of the Oracle AI Database home, register the single-instance Oracle AI Database to Oracle Clusterware.

   ```
   $ srvctl add database -dbname db_name -oraclehome Oracle_home
   -spfile spfile_path_name -dbtype SINGLE
   ```

2. Create and start the database service for your registered database using the SRVCTL commands.

   ```
   $ srvctl add service -db db_name -pdb pdb_name -service service_name
   [additional_options]
   $ srvctl start service -db db_name -s service_name
   ```

Related Topics

- *Oracle Grid Infrastructure Installation and Upgrade Guide for Linux*

Validate the Cluster

After you install Oracle Grid Infrastructure, validate the cluster configuration by using the Configuration Verification Utility (CVU).

Related Topics

- *Oracle Clusterware Administration and Deployment Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

Copy the Preconfigured Database Image

After you validate the cluster, copy the preconfigured database image.

When you copy the preconfigured database image, this includes copying the database structure `*.dbc` file, and the database preconfigured image `*.dfb` file (the one that you used Oracle DBCA to create earlier in the conversion process) to a temporary location on the node in the cluster from which you plan to run Oracle DBCA.

Related Topics

- Use Oracle DBCA to Create an Image of the Single-Instance Database
  To create a preconfigured image of your single-instance database as part of your conversion process from a single instance database to Oracle Real Application Clusters (Oracle RAC) database, use this Oracle DBCA procedure.

Install the New Oracle AI Database Software with Oracle RAC

Install the new Oracle AI Database release, selecting **Cluster Installation Mode**.

1. Run Oracle Universal Installer to install an Oracle AI Database with Oracle RAC.
2. Select **Cluster Installation Mode** on the Specify Hardware Cluster Installation page of Oracle Universal Installer, and select the nodes to include in your Oracle RAC database.
3. On the Oracle Universal Installer Database Configuration Types page, select the **Advanced** installation type.

   After installing the Oracle AI Database software, Oracle Universal Installer runs postinstallation configuration tools, such as Net Configuration Assistant (NETCA), Oracle DBCA, and so on.
4. On the Oracle DBCA Template Selection page, use the template that you copied to a temporary location in the previous section. Use the browse option to select the template location.

   Select the option that you want to deploy. Your choices are the following: Oracle RAC database; Oracle RAC One Node database; or Oracle single-instance database.
5. After creating the Oracle RAC database, Oracle DBCA displays the Password Management page on which you must change the passwords for database users who have SYSDBA and SYSOPER privileges. When Oracle DBCA exits, the conversion process is complete.

Converting Single Instance on a Cluster to Oracle RAC One Node Using Oracle DBCA

To convert a single-instance Oracle AI Database to Oracle RAC One Node, use this Oracle DBCA procedure.

1. Change directory to `$ORACLE_HOME/bin`.
2. Start Oracle DBCA:

   ```
   $ dbca
   ```
3. From the Welcome window, select **Oracle RAC One Node** database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

4. Use the Oracle DBCA template that you selected during conversion of the single-instance Oracle AI Database to Oracle RAC to deploy Oracle RAC One Node.

Converting Single Instance on a Cluster to Oracle RAC Using Oracle DBCA

Learn when you can convert a single instance Oracle AI Database to an Oracle Real Application Clusters instance, and find out how to perform the conversion.

- Scenarios for Converting Single Instance on a Cluster to Oracle RAC
  There are three scenarios where you can use Oracle DBCA to convert an Oracle AI Database single instance on a cluster to an Oracle Real Application Clusters (Oracle RAC) instance.
- Single-Instance Database on a Cluster Running from an Oracle RAC-Enabled Home
  To convert a single-instance database on a cluster node running from an Oracle home that has the Oracle RAC option enabled, complete these procedures.
- Single-Instance Database on a Cluster Running from an Oracle RAC-Disabled Home
  You can create a single-instance database on a cluster running from an Oracle home with the Oracle Real Application Clusters (Oracle RAC) option disabled.

Scenarios for Converting Single Instance on a Cluster to Oracle RAC

There are three scenarios where you can use Oracle DBCA to convert an Oracle AI Database single instance on a cluster to an Oracle Real Application Clusters (Oracle RAC) instance.

- Scenario 1: The Oracle home for the single-instance database was installed on a cluster node, and has Oracle RAC enabled.
- Scenario 2: The Oracle home for the single-instance database was installed on a cluster node, but the Oracle RAC feature is disabled for this Oracle home.
- Scenario 3: The Oracle home for the single-instance database was installed only on the local node in a cluster. This configuration happens when you select the **Local Installation** option on the Oracle Universal Installer Specify Hardware Cluster Installation page during Oracle AI Database installation.

Related Topics

- Setting up the Cluster to Convert a Single-Instance Database on a Cluster
  To convert a single-instance database on a cluster node running from an Oracle home that has the Oracle RAC option enabled, you first use Oracle DBCA to set up the cluster.
- Single-Instance Database on a Cluster Running from an Oracle RAC-Disabled Home
  You can create a single-instance database on a cluster running from an Oracle home with the Oracle Real Application Clusters (Oracle RAC) option disabled.
- Converting Oracle AI Database Installations to Oracle RAC Using Oracle DBCA
  To convert from a single-instance Oracle AI Database that is on a non-clustered computer to Oracle Real Application Clusters (Oracle RAC), complete each of the procedures described here.
- Install the New Oracle AI Database Software with Oracle RAC
  Install the new Oracle AI Database release, selecting **Cluster Installation Mode**.

Single-Instance Database on a Cluster Running from an Oracle RAC-Enabled Home

To convert a single-instance database on a cluster node running from an Oracle home that has the Oracle RAC option enabled, complete these procedures.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Setting up the Cluster to Convert a Single-Instance Database on a Cluster
  To convert a single-instance database on a cluster node running from an Oracle home that has the Oracle RAC option enabled, you first use Oracle DBCA to set up the cluster.
- Automated Conversion Procedure Using Oracle DBCA
  To complete conversion automatically from a single instance Oracle AI Database to an Oracle Real Application Clusters (Oracle RAC) database, you can us this procedure.
- Manual Conversion Procedure
  Use this procedure to manually convert a single instance Oracle AI Database to an Oracle Real Application Clusters (Oracle RAC) database.

Setting up the Cluster to Convert a Single-Instance Database on a Cluster

To convert a single-instance database on a cluster node running from an Oracle home that has the Oracle RAC option enabled, you first use Oracle DBCA to set up the cluster.

1. Use Oracle DBCA to create a preconfigured image of your single-instance database. To perform the conversion manually, shut down the single-instance database.
2. Add nodes to your cluster. Ensure that all nodes can access the shared storage used by Oracle Clusterware and Oracle RAC.
3. From the existing Oracle home, extend this home to the new nodes.
4. From a newly added node, configure the listeners on the additional nodes using NETCA. Choose the same port number and protocol that you used on the existing node. If NETCA displays the existing node in the node list page, then do not select this node, because the listener is already configured on it.
5. Convert the database using one of the following procedures:

After you prepare the cluster, you are ready to convert the Oracle AI Database, either by using an automated conversion with Oracle DBCA, or by performing a manual conversion. Select the procedure that you prefer.

Related Topics

- Use Oracle DBCA to Create an Image of the Single-Instance Database
  To create a preconfigured image of your single-instance database as part of your conversion process from a single instance database to Oracle Real Application Clusters (Oracle RAC) database, use this Oracle DBCA procedure.
- *Oracle Clusterware Administration and Deployment Guide*
- Adding Oracle RAC to Nodes with Oracle Grid Infrastructure Installed
  To add Oracle Real Application Clusters (Oracle RAC) with Oracle Grid Infrastructure installed, your procedure depends on the storage you use, and your Oracle home configuration,

Automated Conversion Procedure Using Oracle DBCA

To complete conversion automatically from a single instance Oracle AI Database to an Oracle Real Application Clusters (Oracle RAC) database, you can us this procedure.

If you used Oracle DBCA to create a preconfigured image of your single-instance database, then perform the following steps to complete the conversion to an Oracle RAC database

1. Start Oracle DBCA from the initial node. Select the names of the nodes to include as part of your cluster database. On the Template Selection page, select the preconfigured template that you created previously with Oracle DBCA. Enter the database name, and respond to the remaining Oracle DBCA prompts.

Copyright © 1999, 2025, Oracle and/or its affiliates.

2. Specify the shared storage location for the Oracle AI Database data files.

After creating the Oracle RAC database, Oracle DBCA displays the Password Management page on which you must change the passwords for the database users who have SYSDBA and SYSOPER privileges. When Oracle DBCA exits, the conversion process is complete.

Manual Conversion Procedure

Use this procedure to manually convert a single instance Oracle AI Database to an Oracle Real Application Clusters (Oracle RAC) database.

If you did not use Oracle DBCA to create a preconfigured image of your single-instance database as described in a previous section, then perform the following steps to complete the conversion:

1. Create the Optimal Flexible Architecture directory structure on each of the nodes that you have added.
2. Shut down the database instance.
3. Set the `CLUSTER_DATABASE` parameter to `TRUE`, and set the `INSTANCE_NUMBER` parameter to a unique value for each instance, using the `sid.parameter=value` syntax.

   ```
   SQL> ALTER SYSTEM SET CLUSTER_DATABASE=TRUE SCOPE=SPFILE;
   SQL> ALTER SYSTEM SET INSTANCE_NUMBER=instance_name.instance_number
   ```

   Oracle RAC requires about 350 bytes for each buffer for Oracle Cache Fusion. Adjust the size of the System Global Area (SGA) to 350 bytes for each buffer before converting to Oracle RAC. For example, if you have 10,000 buffers, then Oracle RAC requires approximately 350*10,000= 35,00,000 bytes more memory. Therefore, adjust the size of the SGA by changing the `DB_CACHE_SIZE` and `DB_nK_CACHE_SIZE` parameters accordingly.
4. Start the database instance.
5. Create an undo tablespace for each additional instance using the `CREATE UNDO TABLESPACE` SQL statement.
6. Create redo threads that have at least two redo logs for each additional instance. Enable the new redo threads by using an `ALTER DATABASE` SQL statement. Then, shut down the database instance.
7. Copy the Oracle Password file to an Oracle ASM disk group.
8. On the local node, use SQL*Plus to run `catclust.sql`. This script creates the dictionary views needed for Oracle RAC databases. For example:

   ```
   SQL> start ?/rdbms/admin/catclust.sql
   ```

9. Add the configuration for the Oracle RAC or Oracle RAC One Node database and its instance-to-node mapping using SRVCTL.
   a. Add the configuration of an Oracle RAC database.

      ```
      $ srvctl add database -dbname db_name -oraclehome Oracle_home -spfile
      spfile_path_name
      $ srvctl add instance -dbname db_name -instance inst1_name -node
      node1_name
      $ srvctl add instance -dbname db_name -instance inst2_name -node
      node2_name
      ```


Copyright © 1999, 2025, Oracle and/or its affiliates.

b. Create a new service for the database.

```
$ srvctl add service -dbname db_name –service service_name
```

c. Add the configuration of an Oracle RAC One Node database.

```
$ srvctl add database -dbname db_name -dbtype RACONENODE -oraclehome
Oracle_home -spfile spfile_path_name
```

10. Start the Oracle RAC or Oracle RAC One Node database using SRVCTL.

```
$ srvctl start database -d db_name
```

After starting the database with SRVCTL, your conversion process is complete. You can run the following SQL statement to see the status of all the instances in your Oracle RAC database:

```
SQL> SELECT * FROM v$active_instances;
```

Related Topics

- Use Oracle DBCA to Create an Image of the Single-Instance Database
  To create a preconfigured image of your single-instance database as part of your conversion process from a single instance database to Oracle Real Application Clusters (Oracle RAC) database, use this Oracle DBCA procedure.

Single-Instance Database on a Cluster Running from an Oracle RAC-Disabled Home

You can create a single-instance database on a cluster running from an Oracle home with the Oracle Real Application Clusters (Oracle RAC) option disabled.

To create an Oracle home on a cluster with Oracle RAC disabled, you can select **local** and **non-cluster** on the Node Selection Page of Oracle Universal Installer when installing the Oracle AI Database software. You can also performed a one-node cluster (with Oracle RAC) installation, but later disable the Oracle RAC option.

Perform the following procedures to convert this type of single-instance database to an Oracle RAC or Oracle RAC One Node database:

1. Use Oracle DBCA to create a preconfigured image of your single-instance database. To perform the conversion manually, shut down the single-instance database.
2. Change the directory to the `lib` subdirectory in the `rdbms` directory under the Oracle home.
3. Relink the `oracle` binary by running the following commands:

```
export ORACLE_HOME=/u01/app/oracle/product/23.4.0/dbhome_1
make -f ins_rdbms.mk rac_on
make -f ins_rdbms.mk ioracle
```

4. Add nodes to your cluster. Ensure that all nodes can access the shared storage used by Oracle Clusterware and Oracle RAC.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Related Topics

- Use Oracle DBCA to Create an Image of the Single-Instance Database
  To create a preconfigured image of your single-instance database as part of your conversion process from a single instance database to Oracle Real Application Clusters (Oracle RAC) database, use this Oracle DBCA procedure.
- *Oracle Clusterware Administration and Deployment Guide*

Converting Single Instance on a Single Server to Oracle RAC One Node Using Oracle DBCA

To convert a single-instance Oracle AI Database on a single server to Oracle RAC One Node, use this Oracle DBCA procedure.

1. If your single-instance database is running, then stop it.

   ```
   $ export ORACLE_SID=<DB SID>
   $ sqlplus "/as sysdba"
   SQL> shutdown immediate
   SQL> exit
   ```

2. If you have a listener running on port 1521, the stop it.

   ```
   $ lsnrctl stop listener
   ```

3. If you have the ONS server running, then stop it.

   ```
   $ onsctl stop
   ```

4. Perform the Oracle Grid Infrastructure installation by using the procedure described in *Oracle Grid Infrastructure Installation and Upgrade Guide* for your platform.

   > **Note**
   >
   > The option of installing Oracle Grid Infrastructure to Manage Generic Applications on the single server also supports single server rolling database maintenance.

5. Make your existing Oracle AI Database home RAC-enabled.
   a. Change the directory to the `lib` subdirectory in the `rdbms` directory under the Oracle home.
   b. Relink the oracle binary by running the following commands:

      ```
      export ORACLE_HOME=/u01/app/oracle/product/23.4.0/dbhome_1
      make -f ins_rdbms.mk rac_on
      make -f ins_rdbms.mk ioracle
      ```

6. As the software owner of the Oracle AI Database home, register the single-instance Oracle AI Database to Oracle Clusterware.

   ```
   $ srvctl add database -dbname db_name -oraclehome Oracle_home -spfile
   spfile_path_name
   $ srvctl start database -db db_unique_name
   ```


Copyright © 1999, 2025, Oracle and/or its affiliates.

7. Use Oracle DBCA in silent mode to convert to the single instance Oracle AI Database to Oracle RAC One Node database.

```
$ dbca -silent -convertToRac -sourceDB db_name
$ srvctl start database -db db_unique_name
```

8. Create and start database service for your registered database using SRVCTL commands.

```
$ srvctl add service -db db_name -pdb pdb_name -service service_name -
notification true [additional_options]
$ srvctl start service -db db_name -service service_name
```

> **Note**
>
> When creating a service, add options as required by your applications. Using Transparent Application Continuity with the service will minimize the impact to the transactions during local rolling database maintenance.

Related Topics

- Transparent Application Continuity
  A version of Application Continuity for simple applications that uses discovery to detect request boundaries.
- Use Oracle DBCA to Create an Image of the Single-Instance Database
  To create a preconfigured image of your single-instance database as part of your conversion process from a single instance database to Oracle Real Application Clusters (Oracle RAC) database, use this Oracle DBCA procedure.

Preparing to Convert with rconfig and Oracle Enterprise Manager

You can use `rconfig` or Oracle Enterprise Manager to assist with converting a single-instance database installation to an Oracle RAC database.

The first of these, `rconfig`, is a command-line utility. Oracle Enterprise Manager Cloud Control database administration option, Convert to Cluster Database, provides a GUI-based conversion tool. Before you start the conversion, back up your existing database; you should take a backup of your database before starting any major change.

> **Note**
>
> Starting with Oracle AI Database, the `RCONFIG` utility is deprecated.

- Prerequisites for Converting to Oracle RAC Databases
  Your Oracle Real Application Clusters (Oracle RAC) environment must meet these prerequisites before you can convert your database.
- Configuration Changes During Oracle RAC Conversion Using rconfig
  These changes occur when you convert a single-instance database to Oracle RAC using the `rconfig` utility.
- Converting Databases to Oracle RAC Using rconfig or Oracle Enterprise Manager
  This list describes scenarios for converting a single-instance Oracle AI Database to an Oracle RAC database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Converting Databases to Oracle RAC Using Oracle Enterprise Manager
  You can use Oracle Enterprise Manager Cloud Control to convert a single-instance database to an Oracle RAC database.

Prerequisites for Converting to Oracle RAC Databases

Your Oracle Real Application Clusters (Oracle RAC) environment must meet these prerequisites before you can convert your database.

Before you convert a single-instance database to an Oracle RAC database, ensure that the following conditions are met for each cluster node that you intend to make an Oracle RAC database node:

- Oracle Clusterware 26ai is installed, configured, and running.
- Oracle RAC 26ai software is installed.
- The Oracle software has the Oracle RAC option enabled.
- Shared storage, either Oracle Cluster File System or Oracle ASM, is available and accessible from all nodes.
- User equivalence exists for the `oracle` account, or the user account used to install the Oracle software.
- If you intend to use Oracle Enterprise Manager, then the Oracle Management Agent on each node is configured and running, and is configured with cluster and host information.
- You have backed up your existing database.

> **Note**
>
> You must use clustered Oracle ASM instances for Oracle RAC databases.

Configuration Changes During Oracle RAC Conversion Using rconfig

These changes occur when you convert a single-instance database to Oracle RAC using the `rconfig` utility.

> **Note**
>
> Starting with Oracle AI Database, the `RCONFIG` utility is deprecated.
> Oracle recommends that you use Database Configuration Assistant (DBCA).
> Beginning with Oracle Enterprise Manager 13c Release 5 Update 23 (13.5.0.23), converting a single-instance database to Oracle RAC through Oracle Enterprise Manager using the `RCONFIG` utility is no longer supported.

- During the conversion, `rconfig` places the target Oracle RAC database into archive log mode, and enables archiving for the database. If you do not plan to use archive log space, then you can disable archive logging after the conversion has completed.
- For the **Shared Storage Type** value, if you enter **CFS**, and you use a cluster file system for your single-instance database storage, then `rconfig` converts the environment to use

Copyright © 1999, 2025, Oracle and/or its affiliates.

Oracle Managed Files for database storage, and places the data files in a subdirectory located under the shared storage location.

- During the conversion, `rconfig` moves database files to a specified shared location, and configures them using Oracle Managed Files.

  To avoid using Oracle Managed Files with your converted database, the single-instance database files must be located in a shared file system, and you must indicate that `rconfig` should not move the files.

Converting Databases to Oracle RAC Using rconfig or Oracle Enterprise Manager

This list describes scenarios for converting a single-instance Oracle AI Database to an Oracle RAC database.

- Converting a single-instance Oracle AI Database 26ai database to an Oracle RAC 26ai database, running out of the *same* Oracle home and using the same data files as the single-instance database.

  In this scenario, run the `rconfig` utility from the Oracle RAC database home, or use the **Convert to RAC** option on the single-instance database target of Oracle Enterprise Manager Cloud Control.

- Converting a single-instance database that uses a release of Oracle AI Database earlier than Oracle AI Database 26ai to an Oracle RAC 26ai database, running out of the same Oracle home and using the same data files as the single-instance database

  In this scenario, use Oracle Universal Installer and Database Upgrade Assistant (DBUA) to update the single-instance database to Oracle AI Database 26ai. Then use `rconfig` or the Oracle Enterprise Manager **Convert to RAC** option, as described in the preceding scenario.

- Converting a single-instance Oracle AI Database 26ai to an Oracle RAC 26ai database, running out of a *different* Oracle home and using the same data files as the single-instance database.

  In this scenario, run the `rconfig` utility in the target database home, or use the **Convert to RAC** option in the single-instance database target of Oracle Enterprise Manager Cloud Control. Provide the file storage location when prompted.

  > **Note**
  >
  > If you specify Oracle home users for both the target database home and the source database home, then the Oracle home user for the target database home must be the same as the Oracle home user for the source database home.

- Converting a single-instance Oracle AI Database 26ai to an Oracle RAC 26ai database, running out of a *different* Oracle home, and where the host on which the single-instance database runs is *not* a node used by the Oracle RAC database.

  In this scenario, create a clone image of the single-instance database, and move the clone image to a host that is one of the nodes used by the Oracle RAC database. Then use `rconfig` or the Oracle Enterprise Manager **Convert to RAC** option, as described in the preceding scenario.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Converting Databases to Oracle RAC Using Oracle Enterprise Manager

You can use Oracle Enterprise Manager Cloud Control to convert a single-instance database to an Oracle RAC database.

To use this feature, complete the following steps:

1. Log in to Oracle Enterprise Manager Cloud Control. From the Home page, click the **Targets** tab.
2. On the Targets page, click the **Databases** secondary tab, and click the link in the Names column of the database to convert to Oracle RAC.
3. On the Database home page, from the Availability menu, select **Convert to Cluster Database**.
4. Log in as the database user `SYS` with `SYSDBA` privileges to the database you want to convert, and click **Next**.
5. On the Convert to Cluster Database: Cluster Credentials page, provide a user name and password for the `oracle` user and password of the target database to convert. If the target database is using Oracle ASM, then also provide the SYSASM user and password, and then click **Next**.
6. On the Hosts page, select the host nodes in the cluster to be cluster members in the installed Oracle RAC database. When you have completed your selection, click **Next**.
7. On the Convert to Database: Options page, select whether you want to use the existing listener and port number, or specify a new listener and port number for the cluster. Also provide a prefix for cluster database instances on the cluster.

   When you have finished entering information, click **Next**, or click **Help** if you need assistance in deciding how to enter information.
8. On the Convert to Cluster Database: Shared Storage page, either select the option to use your existing shared storage area, or select the option to have your database files copied to a new shared storage location. Also, decide if you want to use your existing fast recovery area or copy your recovery files to a new fast recovery area using files managed by Oracle AI Database.

   If you use Oracle ASM, then Oracle recommends that you place the data files and the recovery files in separate failure groups. A failure group is defined by shared hardware, such as a controller shared between two disks, or two disks that are on the same spindle. If two disks share hardware that could fail, making both disks unavailable, then theses disks are said to be in the same failure group. If you do not use Oracle ASM, then Oracle recommends that the data files and the recovery files are stored in separate locations, for example, separate Oracle ASM failure groups, so that a hardware failure does not affect availability.

   When you have finished entering information, click **Next**, or click **Help** if you need assistance in deciding how to enter information.
9. On the Convert to Cluster Database: Review page, review the options you have selected. Click **Submit Job** to proceed with the conversion. To change any options you have selected, click **Back**. To cancel the conversion, click **Cancel**.
10. On the Confirmation page, click **View Job** to check the status of the conversion.

Related Topics

- *Oracle AI Database Upgrade Guide*

Copyright © 1999, 2025, Oracle and/or its affiliates.

Converting Databases to Oracle RAC Using rconfig

You can use the command-line utility `rconfig` to convert a single-instance database to an Oracle RAC database, or to convert it to an Oracle RAC One Node database, depending on the values you provide in the `ConvertToRAC.xml` file.

> **Note**
>
> Starting with Oracle AI Database, the `RCONFIG` utility is deprecated.

1. As the `oracle` user, navigate to the directory `$ORACLE_HOME/assistants/rconfig/sampleXMLs`, and open the file `ConvertToRAC.xml` using a text editor, such as `vi`.
2. Review the `ConvertToRAC.xml` file, and modify the parameters as required for your system. The XML sample file contains comments that provide instructions for how to configure the file. Do NOT put passwords in the XML file. Instead, let the `rconfig` utility prompt you to enter the passwords.

   > **Caution**
   >
   > Set the convert option `Convert verify="ONLY"` to perform a test conversion to ensure that a conversion can be completed successfully.

   When you have finished modifying parameters, save the file with a name of the format *`file_name`*`.xml`. Make a note of the name you select.
3. Navigate to the directory `$ORACLE_HOME/bin`, and use the following command to run the command `rconfig`, where *`input.xml`* is the name of the XML input file you configured in Step 2:

   ```
   rconfig input.xml
   ```

   For example, if you create an input XML file called `convert.xml`, then you would use the following command

   ```
   $ ./rconfig convert.xml
   ```

   The `rconfig` utility will prompt you for the necessary passwords.

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> The `Convert verify` option in the `ConvertToRAC.xml` file has three options:
>
> - `Convert verify="YES"`: rconfig performs checks to ensure that the prerequisites for single-instance to Oracle RAC conversion have been met before it starts conversion
> - `Convert verify="NO"`: rconfig does not perform prerequisite checks, and starts conversion
> - `Convert verify="ONLY"` rconfig only performs prerequisite checks; it does not start conversion after completing prerequisite checks

If performing the conversion fails, then use the following procedure to recover and reattempt the conversion:

1. Attempt to delete the database using the DBCA delete database option.
2. Restore the source database.
3. Review the conversion log, and fix any problems reported by `rconfig` that may have caused the conversion failure. The `rconfig` log files are under the `rconfig` directory in `$ORACLE_BASE/cfgtoollogs`.
4. Reattempt the conversion.

Related Topics

- *Oracle AI Database Upgrade Guide*

Example of rconfig XML Input Files for ConvertToRAC

Review this example of an XML `ConvertToRAC` input file for the `rconfig` utility.

> **Note**
>
> Do not include passwords in the XML file. Instead, let the `rconfig` utility prompt you for the passwords.

Example 15-1 Example rconfig ConvertToRAC XML File for Administrator-Managed Databases

This example shows an XML input file to convert a single-instance database with Oracle ASM to an administrator-managed Oracle RAC database.

```
<?xml version="1.0" encoding="UTF-8"?>
<n:RConfig xmlns:n="http://www.example.com/rconfig"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:schemaLocation="http://www.example.com/rconfig rconfig.xsd">
    <n:ConvertToRAC>
<!-- Verify does a precheck to ensure all pre-requisites are met, before the
 conversion is attempted. Allowable values are: YES|NO|ONLY -->
    <n:Convert verify="YES">
<!--Specify current OracleHome of non-rac database for SourceDBHome -->
          <n:SourceDBHome>/oracle/product/12.1.0/db_1</n:SourceDBHome>
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

```
<!--Specify OracleHome where the rac database should be configured. It can be
same
 as SourceDBHome -->
          <n:TargetDBHome>/oracle/product/12.1.0/db_1</n:TargetDBHome>
<!--Specify SID of non-rac database and credential. User with sysdba role is
 required to perform conversion -->
             <n:SourceDBInfo SID="sales">
            <n:Credentials>
              <n:User>sys</n:User>
              <n:Role>sysdba</n:Role>
            </n:Credentials>
          </n:SourceDBInfo>
<!--Specify the list of nodes that should have rac instances running for the
Admin
 Managed Cluster Database. LocalNode should be the first node in this
nodelist.
 -->
          <n:NodeList>
            <n:Node name="node1"/>
            <n:Node name="node2"/>
          </n:NodeList>
<!--Specify RacOneNode along with servicename to convert database to RACOne
Node -->
<!--n:RacOneNode  servicename="salesrac1service"/-->
<!--Instance Prefix tag is optional starting with 11.2. If left empty, it is
 derived from db_unique_name.-->
          <n:InstancePrefix>sales</n:InstancePrefix>
<!-- Listener details are no longer needed starting 11.2. Database is
registered
 with default listener and SCAN listener running from Oracle Grid
Infrastructure
 home. -->
<!--Specify the type of storage to be used by rac database. Allowable values
are
 CFS|ASM. The non-rac database should have same storage type. ASM credentials
are not needed for conversion. -->
          <n:SharedStorage type="ASM">
<!--Specify Database Area Location to be configured for rac database.If this
field
 is left empty, current storage will be used for rac database. For CFS, this
field will have directory path. -->
             <n:TargetDatabaseArea>+ASMDG</n:TargetDatabaseArea>
<!--Specify Fast Recovery Area to be configured for rac database. If this
field is
 left empty, current recovery area of non-rac database will be configured for
rac
 database. If current database is not using recovery Area, the resulting rac
 database will not have a recovery area. -->
             <n:TargetFlashRecoveryArea>+ASMDG</n:TargetFlashRecoveryArea>
          </n:SharedStorage>
        </n:Convert>
    </n:ConvertToRAC>
</n:RConfig>
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Postconversion Steps

After completing the conversion of your single instance database to an Oracle Real Application Clusters (Oracle RAC) database, follow these recommendations for configuring Oracle RAC environments.

After conversion, Oracle recommends that you follow these guidelines:

- Follow the recommendations for using load balancing and Application Failover using AC or TAC as described in a previous chapter. Use the ORAchk utility to perform Oracle AI Database health check.
- Use locally managed tablespaces instead of dictionary managed tablespaces to reduce contention and manage sequences in Oracle RAC as described in *Oracle AI Database Administrator's Guide*
- Follow the guidelines for using automatic segment space management as described in *Oracle AI Database Administrator's Guide*

The buffer cache and shared pool capacity requirements in Oracle RAC are slightly greater than the requirements for single-instance Oracle AI Databases. Therefore, you should increase the size of the buffer cache by about 10 percent, and the size of the shared pool by about 15 percent.

Related Topics

- Workload Management with Dynamic Database Services
  Workload management includes load balancing, enabling clients for Oracle Real Application Clusters (Oracle RAC), distributed transaction processing, and services.
- About Locally Managed Tablespaces
- Specifying Segment Space Management in Locally Managed Tablespaces

Copyright © 1999, 2025, Oracle and/or its affiliates.

A

Server Control Utility Reference

Use the Server Control Utility (SRVCTL) to manage Oracle Real Application Clusters (Oracle RAC) configuration information.

> **Note**
>
> SRVCTL commands specific to Oracle Grid Infrastructure administration operations are documented in *Oracle Clusterware Administration and Deployment Guide*

- SRVCTL Usage Information
  SRVCTL is installed on each node in a cluster by default. To use SRVCTL, log in to the operating system of a node and enter the SRVCTL command and its parameters in case-sensitive syntax.
- Specifying Command Parameters as Keywords Instead of Single Letters
  The use of single letter commands is deprecated. Oracle recommends that you use full command words with SRVCTL.
- Character Set and Case Sensitivity of SRVCTL Object Values
  SRVCTL interacts with many different types of objects. The character set and name length limitations, and whether the object name is case sensitive, can vary between object types.
- Summary of Tasks for Which SRVCTL Is Used
  Use SRVCTL to manage databases, instances, cluster databases, cluster database instances, Oracle ASM instances and disk groups, services, listeners, or other clusterware resources.
- Using SRVCTL Help
  Learn about how to use context sensitive help with SRVCTL commands.
- SRVCTL Privileges and Security
  To use SRVCTL to change your Oracle RAC database configuration, log in to the operating system as the software owner of the home that you want to manage.
- Additional SRVCTL Topics
  You can use SRVCTL to manage Oracle-supplied resources, but Oracle strongly advises you to follow the guidelines provided here.
- Deprecated SRVCTL Subprograms or Commands
  Oracle recommends that you use alternatives for several SRVCTL commands and parameters deprecated with Oracle Database 12c.
- SRVCTL Command Reference
  Use this comprehensive list of SRVCTL commands to manage Oracle Real Application Clusters (Oracle RAC) environments.

SRVCTL Usage Information

SRVCTL is installed on each node in a cluster by default. To use SRVCTL, log in to the operating system of a node and enter the SRVCTL command and its parameters in case-sensitive syntax.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Use the version of SRVCTL that is provided with the current Oracle AI Database release from the Oracle home of the database that you are managing. The version of SRVCTL must be the same as the version of the object (listeners, Oracle ASM instances, Oracle AI Database, Oracle AI Database instances, and Oracle AI Database services) being managed.
- SRVCTL does not support concurrent processing of commands on the same object. Therefore, run only one SRVCTL command at a time for each database, service, or other object.
- When specifying a comma-delimited list as part of a SRVCTL command, there should not be any spaces between the items in the list.

  When you specify a comma-delimited list in a Windows environment, you must enclose the list in double quotation marks (""). You can enclose a comma-delimited list in double quotation marks in a Linux or UNIX environment but they will be ignored.
- If you are entering a SRVCTL command, and you want to continue the input on a new line, then you can use the operating system continuation character. In Linux, this is the backslash (\) symbol.
- A SRVCTL command that produces no output is a successful command. Not all SRVCTL commands return a message when it completes, successfully. However, if a SRVCTL command fails, then it always returns an error message.
- SRVCTL returns `0` on success, `1` on failure, and `2` on warnings. Some commands, such as `start`, `stop`, `enable`, and `disable`, can return 2 for a warning when the request would not change anything. In other words, the object of the command is already started, already stopped, already disabled, and so on. In warning cases, SRVCTL also prints a message about what was already done.
- You can use the `-eval` parameter with several SRVCTL commands. This parameter, when you use it, enables you to simulate running a command without making any changes to the system. SRVCTL returns output that informs you what will happen if you run a particular command. For example, to know what might happen if you relocate a server:

  The `-eval` parameter is available with the following commands:

  - `srvctl add database`
  - `srvctl modify database`
  - `srvctl relocate service`
  - `srvctl start database`
  - `srvctl start service`
  - `srvctl stop database`
  - `srvctl stop service`

Specifying Command Parameters as Keywords Instead of Single Letters

The use of single letter commands is deprecated. Oracle recommends that you use full command words with SRVCTL.

SRVCTL command parameters use full words instead of single letters, such as `-multicastport` and `-subdomain`.

Copyright © 1999, 2025, Oracle and/or its affiliates.

To support backward compatibility, you can use a mix of single-letter parameters and new keyword parameters. New parameters introduced with keywords can be used with single letter parameters.

> **Note**
>
> The use of single letter parameters are deprecated. Oracle recommends that you use the keyword parameters, so that you avoid using the same letter to implement different functionality, depending on the command.
>
> You can obtain the single-letter equivalents, where applicable, by adding the `-compatible` parameter after the `-help` parameter.

Character Set and Case Sensitivity of SRVCTL Object Values

SRVCTL interacts with many different types of objects. The character set and name length limitations, and whether the object name is case sensitive, can vary between object types.

Table A-1 String Restrictions for SRVCTL Object Names

Object Type	Character Set Limitations	Case Sensitive?	Maximum Length
`db_domain`	Alpha-numeric characters, underscore (_), and number sign (#)	No	128 characters
`db_unique_name`	Alpha-numeric characters, underscore (_), number sign (#), and dollar sign ($); the first 8 characters must be unique because those characters are used to form instance names	No	30 characters but the first 12 characters must be unique relative to any other database in the same cluster
`pdb_name`	Alpha-numeric characters and underscore (_); the first character must be an alphabet character	No	30 characters
`diskgroup_name`	Naming disk groups have the same limitations as naming other database objects.	No (all names are converted to uppercase)	30 characters
`instance_name`	Alphanumeric characters	Depends on the platform	15 characters
`listener_name`	Alphanumeric characters	Depends on the platform	15 characters
`node_name`	Alphanumeric characters	No	15 characters
`scan_name`	The first character must be an alphabetic character	No	15 characters
`service_name`	Alphanumeric characters, underscore (_), number sign (#), period (.), and dollar sign ($); the name cannot begin with a period, nor contain single quotation marks (`''`)	No	250 characters

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-1 (Cont.) String Restrictions for SRVCTL Object Names

Object Type	Character Set Limitations	Case Sensitive?	Maximum Length
`volume_name`	Alphanumeric characters; dashes (-) are not allowed and the first character must be an alphabetic character.	No	11 characters

Summary of Tasks for Which SRVCTL Is Used

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

- Cluster Database Configuration Tasks

Tasks	Commands

Add, modify, and delete cluster database configuration information	`srvctl add database`
	`srvctl modify database`
	`srvctl remove database`

Add an instance to or delete an instance from the configuration of a cluster database	`srvctl add instance` `srvctl remove instance`
Add a service to or delete a service from the configuration of a cluster database	`srvctl add service` `srvctl remove service`

Move instances and services in a cluster database configuration and modify service configurations	`srvctl relocate database`
	`srvctl relocate service`
	`srvctl modify instance`
	`srvctl modify service`

Set and unset the environment for an instance or service in a cluster database configuration	`srvctl modify instance` `srvctl modify service`
Set and unset the environment for an entire cluster database in a cluster database configuration	`srvctl setenv database` `srvctl unsetenv database`

- General Cluster Database Administration Tasks

Tasks	Commands
Start and stop cluster databases	`srvctl start database` `srvctl stop database`
Start and stop cluster database instances	`srvctl start instance` `srvctl stop instance`

Copyright © 1999, 2025, Oracle and/or its affiliates.

Tasks	Commands

Start, stop, and relocate cluster database services	`srvctl start service`
	`srvctl stop service`
	`srvctl relocate service`

Obtain statuses of cluster databases, cluster database instances, or cluster database services	`srvctl status database`
	`srvctl status instance`
	`srvctl status service`

- Node-Level Tasks

Tasks	Commands

Administering VIPs	`srvctl add vip`
	`srvctl config vip`
	`srvctl disable vip`
	`srvctl enable vip`
	`srvctl getenv vip`
	`srvctl modify vip`
	`srvctl relocate vip`
	`srvctl remove vip`
	`srvctl setenv vip`
	`srvctl start vip`
	`srvctl status vip`
	`srvctl stop vip`
	`srvctl unsetenv vip`

Administering node applications	`srvctl add nodeapps`
	`srvctl disable nodeapps`
	`srvctl enable nodeapps`
	`srvctl getenv nodeapps`
	`srvctl modify nodeapps`
	`srvctl remove nodeapps`
	`srvctl setenv nodeapps`
	`srvctl unsetenv nodeapps`

Related Topics

- *Oracle Clusterware Administration and Deployment Guide*

Using SRVCTL Help

Learn about how to use context sensitive help with SRVCTL commands.

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

```
srvctl -help
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

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

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

When you request online help for a command using `-help`, SRVCTL prints the full words for each parameter. You can obtain the single-letter equivalents, where applicable, by adding the `-compatible` parameter after the `-help` parameter. For example:

```
$ srvctl config database -help -compatible
```

The preceding command prints usage information for the `srvctl config database` command, listing all parameters as full words followed by their single-letter equivalents in parentheses, where applicable.

To see the SRVCTL version number enter:

```
$ srvctl -version
```

SRVCTL Privileges and Security

To use SRVCTL to change your Oracle RAC database configuration, log in to the operating system as the software owner of the home that you want to manage.

For example, if different users installed Oracle AI Database and the Oracle Grid Infrastructure, then log in as the database software owner (for example, `ora_db`) to manage databases and log in as the Oracle Grid Infrastructure software owner (for example, `ora_asm`) to manage the Oracle ASM instances.

Users who are members of the OSDBA operating system group can start and stop the database. To stop and start an Oracle ASM instance, you must be a member of the OSASM operating system group.

To create or register objects such as listeners, Oracle Notification Services, and services, you must be logged in to the operating system as the software owner of the Oracle home. The objects you create or register for that Oracle home will run under the user account of the owner of the Oracle home. Databases run as the database installation owner of the home from which they run.

To perform `srvctl add` operations on any object, you must be logged in as the Oracle account owner of the home on which the object runs.

For some SRVCTL commands, to run the commands on Linux and Unix systems, you must be logged in as `root`, and on Windows systems, you must be logged in as a user with Administrator privileges. In this appendix, those commands are preceded by the root prompt (#) in the command examples.

> **Note**
>
> Oracle recommends that you run the SRVCTL commands, which need to be run as the `root` user, from the Oracle Grid Infrastructure home because it is secured with the `root` ownership.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Additional SRVCTL Topics

You can use SRVCTL to manage Oracle-supplied resources, but Oracle strongly advises you to follow the guidelines provided here.

- Use SRVCTL to manage Oracle-supplied resources such as listener, instances, disk groups, and networks, and CRSCTL for managing Oracle Clusterware and its resources.

  > ⚠ **Caution**
  >
  > Oracle strongly discourages you from using CRSCTL to directly manipulate Oracle-supplied resources (resources whose names begin with `ora`). Making manual changes to Oracle resources using CRSCTL can adversely affect the cluster configuration.

- Although you may be able to cancel running SRVCTL commands by pressing the Control-C keys, Oracle strongly advises that you do not attempt to do this, because you can corrupt your configuration data by doing this.

  Do not to attempt to terminate SRVCTL in this manner.

Deprecated SRVCTL Subprograms or Commands

Oracle recommends that you use alternatives for several SRVCTL commands and parameters deprecated with Oracle Database 12c.

- Single Character Parameters Deprecated for all SRVCTL Commands
  Single-character parameters were deprecated in Oracle Database 12c. Use the full keyword for each parameter. Refer to the information here to understand how to update scripts using single-character parameters.
- Miscellaneous SRVCTL Commands and Parameters
  If you have scripts dating from Oracle Database 12.2 or earlier releases, then Oracle recommends that you review the deprecated parameters and update your usage to current forms.

Single Character Parameters Deprecated for all SRVCTL Commands

Single-character parameters were deprecated in Oracle Database 12c. Use the full keyword for each parameter. Refer to the information here to understand how to update scripts using single-character parameters.

Oracle recommends that you use the full keyword for each SRVCTL parameter. To support older tools and scripts that still use single-character parameters, the current version of SRVCTL continues to support both single-character parameters and full keyword parameters. However, deprecated functionality can be desupported in a future release.

The command reference topics for SRVCTL show the keywords for each SRVCTL command. The following table lists the deprecated single-character parameters.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-2 Deprecated Single-Character Parameters for SRVCTL Commands

Single Letter	Long Form	Values	Description	Related Commands
A	address	{*VIP_name* | *IP*}/*netmask*/ [*if1*[| *if2*...]]	VIP address specification for node applications	Node applications, VIP, network, Listener, SCAN VIP, and SCAN listener commands
a	all	none	All resources of that kind	`srvctl config database` Common
a	diskgroup	*diskgroup_list*	Comma-delimited list of Oracle ASM disk groups	Database, instance, Oracle ASM, disk group, and file system commands
a	detail	None	Print detailed configuration information	Common
a	available	*available_list*	A comma-delimited list of available instances	Service commands
a	abort	None	Stop failed online relocation	Relocate database
a	viponly	None	Display VIP configuration	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
B	rlbgoal	{NONE| SERVICE_TIME| THROUGHPUT}	The runtime load balancing goal of a service	Service commands
c	currentnode	*current_node*	Node name from which to relocate the service	Service commands
c	cardinality	{UNIFORM| SINGLETON}	Whether the service should run on every active server (UNIFORM) or only one server (SINGLETON)	Service commands
c	dbtype	*type*	Type of database: Oracle RAC One Node, Oracle RAC, or single instance	Database, instance, Oracle ASM, disk group, and file system commands
d	db or database	*db_unique_name*	Database unique name	Common
d	device	*volume_device*	Volume device path	Database, instance, Oracle ASM, disk group, and file system commands
d	domain	None	Display subdomain served by GNS	OC4J, home, CVU, and GNS commands
e	emport	*em_port_number*	Local listen port for Oracle Enterprise Manager	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-2 (Cont.) Deprecated Single-Character Parameters for SRVCTL Commands

Single Letter	Long Form	Values	Description	Related Commands
e	failovertype	{NONE|SESSION BASIC| TRANSACTION}	The failover type for a service	Service commands
e	server	*server_list*	Candidate server list for Oracle RAC One Node database	Database, instance, Oracle ASM, disk group, and file system commands
f	force	None	Force remove	Common
g	diskgroup	*diskgroup_name*	Disk group name	File system, Diskgroup commands
h	help	None	None	Common
i	instance	*instance_name* *instance_list*	Instance name prefix for administrator-managed Oracle RAC One Node database A comma-delimited list of instance names	Database, instance, Oracle ASM, disk group, and file system commands
I	ip	*ip_address*	VIP address on which GNS is to listen	OC4J, home, CVU, and GNS commands
i	oldinst	*instance_name*	The old instance name	Service commands
i	scannumber	*scan_ordinal_number*	Ordinal number of the IP address for the SCAN	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
i	vip	*vip_name* or "*vip_name_list*"	VIP names	Node applications, GNS, VIP, network, listener, SCAN VIP, and SCAN listener commands
j	acfspath	*acfs_path_list*	Comma-delimited list of Oracle ACFS paths where the dependency on the database will be set	Database, instance, Oracle ASM, disk group, and file system commands
j	clbgoal	{SHORT|LONG}	The connection load balancing goal for a service	Service commands
k	netnum	*network_number*	The network number	Service commands Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands OC4J, home, CVU, and GNS commands
l	list		List all records in GNS	OC4J, home, CVU, and GNS commands
l	listener	*listener_name*	The name of a listener	ASM commands

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-2 (Cont.) Deprecated Single-Character Parameters for SRVCTL Commands

Single Letter	Long Form	Values	Description	Related Commands
l	loglevel	*log_level*	Specify the level (0-6) of logging that GNS should run with	OC4J, home, CVU, and GNS commands
l	onslocalport	*port_number*	Oracle Notification Service listening port for local client connections	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
l	role	*service_role*	Comma-delimited list of server roles within double quotation marks (""), where each role is one of PRIMARY, PHYSICAL_STANDBY, LOGICAL_STANDBY, or SNAPSHOT_STANDBY	Service commands
m	domain	domain_name	The domain for the database	Database, instance, Oracle ASM, disk group, and file system commands
m	failovermethod	{NONE|BASIC}	The failover method of a service	Service commands
m	multicastpost		The port on which the GNS daemon is listening for multicast requests	OC4J, home, CVU, and GNS commands
m	path	*mountpoint_path*	Mountpoint path	Database, instance, Oracle ASM, disk group, and file system commands
n	name		Advertise a name through GNS using the given address	OC4J, home, CVU, and GNS commands
n	node	*node_name*	The name of a specific node	Common
n	nodes	*node_list*	A comma-delimited list of node names	File system commands
n	dbname	*database_name*	The database name (DB_NAME), if different from the unique name specified by the -db parameter	Database, instance, Oracle ASM, disk group, and file system commands
n	scanname	scan_name	Fully-qualified SCAN name (includes the domain)	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
n	servers	*server_list*	A comma-delimited list of candidate server names	Service commands
n	targetnode	*node_name*	Node name to which to relocate the service	Service commands

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-2 (Cont.) Deprecated Single-Character Parameters for SRVCTL Commands

Single Letter	Long Form	Values	Description	Related Commands
o	oraclehome	*oracle_home*	*$ORACLE_HOME* path	Database commands
p	endpoints	[TCP:]*port_number*[/IPC:*key*][/NMP:*pipe_name*][/TCPS:*s_port*][/SDP:*port*]	SCAN listener endpoints	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
p	port		The port which the GNS daemon uses to communicate with the DNS server	OC4J, home, CVU, and GNS commands
p	rmiport	*port_number*	OC4J RMI port number	OC4J, home, CVU, and GNS commands
P	tafpolicy	{NONE|BASIC}	TAF policy specification	Service commands
p	spfile	*spfile_location*	Server parameter file path	Database, instance, Oracle ASM, disk group, and file system commands
q	notification	{TRUE|FALSE}	Whether FAN is enabled for OCI connections	Service commands
q	query		Query GNS for the records belonging to a name	OC4J, home, CVU, and GNS commands
r	preferred	*preferred_list*	A comma-delimited list of preferred instances	Service commands
r	onsremoteport	*port_number*	Oracle Notification Service listening port for connections from remote hosts	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
r	relocate		Relocate the VIP	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
r	revert	None	Remove target node of failed online relocation request from the candidate server list of administrator-managed Oracle RAC One Node database	Relocate database
r	role	*role_type*	Role of the standby database: PRIMARY, PHYSICAL_STANDBY, LOGICAL_STANDBY, or SNAPSHOT_STANDBY	Database, instance, Oracle ASM, disk group, and file system commands
s	onsonly		Display Oracle Notification Service daemon configuration	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-2 (Cont.) Deprecated Single-Character Parameters for SRVCTL Commands

Single Letter	Long Form	Values	Description	Related Commands
s	skip	None	Skip checking the ports	Listener, SCAN, and SCAN listener.
s	statefile	*file_name*	The file path of the *state_file* created by a previously run `srvctl stop home` command	OC4J, home, CVU, and GNS commands
s	status		Display the status of GNS	OC4J, home, CVU, and GNS commands
S	subnet	*subnet/net_mask/*[*if1*[|*if2*...]]	Network address specification for a network	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
s	service	*service_name* *service_name_list*	The name of a service A comma-delimited list of service names	Service commands
s	startoption	*start_options*	Startup options for the database (mount, open, read only)	Database, instance, Oracle ASM, disk group, and file system commands
t	checkinterval	*time_interval*	Interval in minutes between checks	OC4J, home, CVU, and GNS commands
t	edition	*edition_name*	The initial session edition of a service	Service commands
t	envs	"*name_list*"	A list of environment variables	Common
t	namevals	"*name= value,...*"	Names and values of environment variables	Common
T	nameval	"*name=value*"	Name and value of a single environment variable	Common
t	update	*instance_name*	The new instance name	Service commands
t	remoteservers	*host_name*[:*port_number*][,*host_name*[:*port_number*]...]	List of remote host name and port number pairs for Oracle Notification Service daemons outside this cluster	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
t	stopoption	*stop_options*	Stop options for the database (`NORMAL`, `TRANSACTIONAL`, `IMMEDITATE`, or `ABORT`)	Database, instance, Oracle ASM, disk group, and file system commands
t	toversion	*target_version*	Version to which you are downgrading	Database, instance, Oracle ASM, disk group, and file system commands
u	nettype	*network_type*	The network server type, which can be `STATIC`, `DHCP`, or `MIXED`	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-2 (Cont.) Deprecated Single-Character Parameters for SRVCTL Commands

Single Letter	Long Form	Values	Description	Related Commands
u	newinst	None	Add a new instance to the service configuration	Service commands
u	update		Update SCAN listeners to match the number of SCAN VIPs	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
u	user	*oracle_user*	Oracle user or other authorized user to mount and unmount file systems	Database, instance, Oracle ASM, disk group, and file system commands
v	verbose		Verbose output	Common
v	volume	*volume_name*	Name of a volume	Database, instance, Oracle ASM, disk group, and file system commands
V	versions			Common
w	failoverdelay	*number*	Failover delay	Service commands
w	nettype	*network_type*	The network server type, which can be STATIC, DHCP, or MIXED	Node applications, VIP, network, listener, SCAN VIP, and SCAN listener commands
w	timeout	*timeout*	Online relocation timeout in minutes	Database, instance, Oracle ASM, disk group, and file system commands
x	dtp	{TRUE | FALSE}	Whether to enable distributed transaction processing	Service commands
x	node	*node_name*	Node name (use this parameter only with noncluster databases)	Common
y	noprompt		Suppress the confirmation prompt	Common
y	policy	{AUTOMATIC | MANUAL}	Management policy for the resource	Database, instance, Oracle ASM, disk group, file system, and service commands
z	failoverretry	*number*	Number of failover retries	Service commands
z	rmdependisk		To remove a database's dependency upon disk groups	Database, instance, Oracle ASM, disk group, and file system commands

Copyright © 1999, 2025, Oracle and/or its affiliates.

Miscellaneous SRVCTL Commands and Parameters

If you have scripts dating from Oracle Database 12.2 or earlier releases, then Oracle recommends that you review the deprecated parameters and update your usage to current forms.

The following command parameters were deprecated in Oracle Database 12c:

Table A-3 Deprecated Commands and Parameters for SRVCTL

Command	Deprecated Parameters
`srvctl modify asm`	`-node` *`node_name`*
`srvctl modify instance`	`-z` Instead, use the `-node` option with the value set to `""`
`srvctl modify gns`	`[-ip` *`ip_address`*`] [-advertise` *`host_name`* `-address` *`address`*`] [-delete` *`host_name`* `-address` *`address`*`] [-createalias` *`name`* `-alias` *`alias`*`] [-deletealias` *`alias`*`]` Use the `srvctl update gns` command instead.
`srvctl * oc4j`	The `oc4j` noun has been deprecated. SRVCTL still accepts the `oc4j` noun until it is desupported.
`srvctl add service`	The `PRECONNECT`option with the `-tafpolicy` parameter is deprecated.
`srvctl modify service`	The `-failovermethod {NONE | BASIC}` is deprecated. The `PRECONNECT`option with the `-tafpolicy` parameter is deprecated.

SRVCTL Command Reference

Use this comprehensive list of SRVCTL commands to manage Oracle Real Application Clusters (Oracle RAC) environments.

- About Using SRVCTL Commands
  To be able to use SRVCTL commands to obtain the outcome you require, review these guidelines.
- database Commands
  Use commands with the database keyword to manage cluster database.
- diskgroup Commands
  Use commands with the `diskgroup` keyword to manage Oracle ASM disk groups.
- home Commands
  Use commands with the `home` keyword to start, stop, and obtain the status of all clusterware resources related to a Home directory.
- instance Commands
  Use commands with the `instance` keyword to add, modify, enable, disable, start, stop, obtain the status of, and remove database instances.
- listener Commands
  Use commands with the `listener` keyword to add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, and remove listeners.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- network Commands
  Use commands with the `network` keyword to add, modify, list the configuration of, and remove a non-default Network.
- nodeapps Commands
  Use commands with the `nodeapps` keyword to add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, and remove node applications.
- ons Commands
  Use commands with the `ons` keyword to manage only Oracle Notification Service instances for Oracle Restart.
- pdb Commands
  Use commands with the `pdb` keyword to manage the Pluggable Databases (PDBs) in your cluster database.
- scan Commands
  Use commands with the `scan` keyword to add, list the configuration of, modify, enable, disable, start, stop, relocate, obtain the status of, and remove SCAN VIPs.
- scan_listener Commands
  Use commands with the `scan_listener` keyword to add, list the configuration of, modify, enable, disable, start, stop, relocate, obtain the status of, and remove SCAN listeners.
- service Commands
  Use commands with the `service` keyword to add, modify, list the configuration of, enable, disable, start, stop, obtain the status of, relocate, and remove services.
- vip Commands
  Use commands with the `vip` keyword to add, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, and remove a VIP.
- volume Commands
  Use commands with the `volume` keyword to list the configuration of, enable, disable, start, stop, obtain the status of, and remove an Oracle ACFS volume.

About Using SRVCTL Commands

To be able to use SRVCTL commands to obtain the outcome you require, review these guidelines.

SRVCTL commands, object names, and parameters are case-sensitive. Database, instance, listener, and service names are case insensitive and case preserving. You cannot create listener names that differ only in case, such as LISTENER and listener. SRVCTL uses the following command syntax:

```
srvctl command object [parameters]
```

In SRVCTL syntax:

- *command* is a verb, such as `start`, `stop`, or `remove`
- *object* (also known as a *noun*) is the target or object on which SRVCTL performs the command, such as database or instance. You can also use object abbreviations.
- *parameters* extend the use of a preceding command combination to include additional parameters for the command. For example, the `-instances` parameter indicates that a comma-delimited list of preferred instance names follows; the `-instance` parameter only

Copyright © 1999, 2025, Oracle and/or its affiliates.

permits one value and not a list of names. Do not use spaces between the items in a comma-delimited list.

> **Note**
>
> If specifying a comma-delimited list in Windows, then you must enclose the list within double quotation marks ("").

The following table lists the keywords that you can use for the *object* portion of SRVCTL commands. You can use either the full name or the abbreviation for each object keyword. The **Purpose** column describes the object and the actions that can be performed on that object.

Table A-4 Object Keywords and Abbreviations

Object	Keyword	Purpose
Database	`database`	To add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, and obtain the status of databases, remove configuration information for or get behavior predictions for a database, and also to convert, upgrade, downgrade, and relocate databases
Diskgroup	`diskgroup`	To enable, disable, start, stop, obtain the status of, remove, or get behavior predictions for an Oracle ASM disk group
Home	`home`	To start, stop, or obtain the status of resources running from a particular Oracle home directory
Instance	`instance` `inst`	To add, modify, enable, disable, start, stop, obtain the status of, update, and remove database instances
Listener	`listener` `lsnr`	To add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, remove, and get behavior predictions for listeners
Network	`network`	To add, modify, list the configuration of, remove and get behavior predictions for a non-default network resource. **Note:** The node applications object, and the `config` and `modify` commands also manage the default network
Node applications	`nodeapps`	To add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, and remove node applications
Oracle Notification Service	`ons`	To add, configure, enable, start, obtain the status of, stop, disable, and remove Oracle Notification Service instances only for Oracle Restart
Pluggable Database (PDB)	`pdb`	To add, modify, remove, list the configuration of, enable, disable, start, stop, and obtain the status of PDBs
Single client access name (SCAN)	`scan`	To add, list the configuration of, modify, enable, disable, start, stop, relocate, remove, obtain the status of, and get behavior predictions for SCAN VIPs

Copyright © 1999, 2025, Oracle and/or its affiliates.

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

Object	Keyword	Purpose
SCAN listener	`scan_listener`	To add, list the configuration of, modify, enable, disable, start, stop, relocate, obtain the status of, remove, and get behavior predictions for SCAN listeners
Service	`service`	To add, modify, list the configuration of, enable, disable, start, stop, obtain the status of, relocate, remove, and get behavior predictions for services
Virtual IP	`vip`	To add, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, remove, and get behavior predictions for a VIP
Volume	`volume`	To list the configuration of, enable, disable, start, stop, obtain the status of, and remove an Oracle ACFS volume

> **Note**
>
> SRVCTL commands specific to Oracle Grid Infrastructure administration operations are documented in CWADD SRVCTL Command Reference

database Commands

Use commands with the database keyword to manage cluster database.

You can add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, and obtain the status of databases, and also to upgrade, downgrade, and remove database configuration information about databases.

- srvctl add database
  Adds a database configuration to Oracle Clusterware.
- srvctl config database
  Displays the configuration for an Oracle RAC database or lists all configured databases that are registered with Oracle Clusterware.
- srvctl convert database
  Converts a database either to or from an Oracle RAC One Node database.
- srvctl disable database
- srvctl downgrade database
- srvctl enable database
- srvctl getenv database
- srvctl modify database
  Modifies the configuration for a database.
- srvctl predict database
- srvctl relocate database
  Initiates the relocation of an Oracle RAC One Node database from one node to another node.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- srvctl remove database
  Removes database configurations.
- srvctl setenv database
- srvctl start database
- srvctl status database
  This command displays the current state of the of the database.
- srvctl stop database
  Stops a database, its instances, and its services.
- srvctl unsetenv database
- srvctl update database
  Updates the specified database to use the new listener endpoints.
- srvctl upgrade database

srvctl add database

Adds a database configuration to Oracle Clusterware.

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

Syntax

```
srvctl add database -db db_unique_name -oraclehome oracle_home
     [-dbtype  {RACONENODE | RAC | SINGLE} [-server "server_list"]
     [-instance instance_name] [-timeout timeout]] [-domain domain_name]
     [-spfile spfile] [-pwfile password_file_path]
     [-role {PRIMARY | PHYSICAL_STANDBY | LOGICAL_STANDBY | SNAPSHOT_STANDBY
| FAR_SYNC}]
     [-startoption start_options] [-stopoption stop_options]
     [-startconcurrency start_concurrency] [-stopconcurrency stop_concurrency]
     [-dbname db_name] [-policy {AUTOMATIC | MANUAL | NORESTART | USERONLY |
RANK}]
     [-node node_name] [-diskgroup "disk_group_list"] [-acfspath "acfs_path |
acfs_path_list"]
     [-css_critical {yes | no}] [-memorytarget memory_target] [-maxmemory
max_memory]
     [-defaultnetnum network_number] [-verbose]
```

Parameters

Table A-5 srvctl add database Command Parameters

Parameter	Description
`-db db_unique_name`	The unique name of the database.
`-oraclehome oracle_home`	The path for the Oracle AI Database home directory.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-5 (Cont.) srvctl add database Command Parameters

Parameter	Description
`-dbtype {RACONENODE | RAC | SINGLE}`	The type of database you are adding: Oracle RAC One Node, Oracle RAC, or single instance. The default is `RAC`. If you specify the `-node` *`node_name`* parameter, then the `-dbtype` parameter defaults to `SINGLE`.
`-server` *`server_list`*	List of candidate servers for Oracle RAC One Node databases. Oracle Clusterware attempts to start Oracle RAC One instance in the order in which the servers are listed. First, it attempts to start the instance on the first server in the list. If that does not succeed for any reason, then Oracle Clusterware attempts to start the instance on the second server defined in the list, and so on. For example, for a 3-node cluster: • Use "N3,N2,N1" to start a particular instance preferably on the third node. • Use "N3,N1,N2" to make the first node the designated failover node. • Use "N3,N1" to prevent the second node from being used. Oracle Clusterware refers to this list every time an instance starts, which can be either a regular start or a start that is triggered by a failure event.
`-instance` *`instance_name`*	The instance name prefix for Oracle RAC One Node databases. The default value for this parameter is the first 12 characters of the global unique name of the database.
`-timeout` *`timeout`*	The online database relocation timeout, in minutes, for Oracle RAC One Node databases. The default value is `30`.
`-domain` *`db_domain`*	The domain for the database. **Note:** You must use this parameter if you set the `DB_DOMAIN` initialization parameter for the database.
`-spfile` *`spfile`*	The path name of the database server parameter file.
`-pwfile` *`password_file_path`*	The full path to the location of the password file.
`-role {PRIMARY | PHYSICAL_STANDBY | LOGICAL_STANDBY | SNAPSHOT_STANDBY | FAR_SYNC}`	The role of the database in an Oracle Data Guard configuration. The default is `PRIMARY`.
`-startoption` *`start_options`*	Startup options for the database, such as `OPEN`, `MOUNT`, and `NOMOUNT`. The default value is `OPEN`. **Notes:** • For multi-word startup options, such as `read only` and `read write`, separate the words with a space and enclose in double quotation marks (`""`). For example, `"read only"`. • When performing a switchover in an Oracle Data Guard configuration, the `-startoption` for a standby database that becomes a primary database is always set to `OPEN` after the switchover.
`-stopoption` *`stop_options`*	Stop options for the database, such as `NORMAL`, `TRANSACTIONAL`, `IMMEDIATE`, and `ABORT`.
`-startconcurrency` *`start_concurrency`*	Number of instances to be started simultaneously, or 0 to disable this option.
`-stopconcurrency` *`stop_concurrency`*	Number of instances to be stopped simultaneously, or 0 to disable this option.
`-dbname` *`db_name`*	The name of the database, if it is different from the unique name given by the `-db` parameter.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-5 (Cont.) srvctl add database Command Parameters

Parameter	Description
`-policy {AUTOMATIC | MANUAL | NORESTART | USERONLY | RANK}`	The management policy for the database. • `AUTOMATIC` (default): The database is automatically restored to its previous running condition (started or stopped) upon restart of the database host computer. • `MANUAL`: The database is never automatically restarted upon restart of the database host computer. A `MANUAL` setting does not prevent Oracle Clusterware from monitoring the database while it is running and restarting it if a failure occurs. • `NORESTART`: Similar to the `MANUAL` setting, the database is not automatically restarted upon restart of the database host computer. A `NORESTART` setting, however, does not restart the database, even if a failure occurs, unless it must be started for dependencies, such as services or PDBs. • `USERONLY`: The database can only be restarted by user command, not as a result of any other reason (auto-start, start by dependency, node failure, and so on.) • `RANK`: The database won't be restarted when the Oracle Clusterware stack is restarted unless it is restarted by start dependencies of its PDBs that are started according to RANK. For example, 2 CDBs have `policy` set to `RANK` and their PDBs have `policy` set to `RESTART`. If a PDB of CDB1 has a rank of 3 and a PDB in CDB2 has a rank of 2, and if there are only enough resources to start one CDB, then CDB1 will be started by dependency when its PDB is started. CDB2 is not started because its PDB has a lower rank number.
`-node` *`node_name`*	The node name on which you want to register a noncluster, or single instance, Oracle AI Database. **Note:** This parameter can be used only with Oracle Clusterware.
`-diskgroup "`*`disk_group_list`*`"`	A comma-delimited list of Oracle Automatic Storage Management (Oracle ASM) disk groups if database uses Oracle ASM storage.
`-acfspath "`*`acfs_path | acfs_path_list`*`"`	You can add (/+) or remove (/-) a single Oracle ACFS path with `acfs_path` or multiple Oracle ACFS paths using a comma-delimited list with `acfs_path_list` enclosed in double quotation marks ("") where the database's dependency is set. Use this parameter to create dependencies on Oracle ACFS file systems other than `ORACLE_HOME`, such as for when the database uses `ORACLE_BASE` on a file system that is different from the `ORACLE_HOME` file system. **Note** This parameter is available only on the platforms on which Oracle ACFS is supported.
`-css_critical {YES | NO}`	You can add weight to a service by setting this parameter to YES. In the event of a node failure within the cluster, Oracle Clusterware will evict the node with the least amount of weight, ensuring that critical services remain available.
`-memorytarget` *`memory_target`*	The target memory size, in MB, to be allocated for the database. The default is 0.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-5 (Cont.) srvctl add database Command Parameters

Parameter	Description
`-maxmemory` *max_memory*	The maximum memory size, in MB, to be allocated for the resource. If you specify `-memorytarget` but not `-maxmemory`, then `-maxmemory` will default to 0. Both `-maxmemory` and `-memorytarget` are validated as long as `-memorytarget` is less than or equal to `-maxmemory`.
`-defaultnetnum` *network_number*	Specify a network number (an integer) to which services will default in the event you do not specify a network number when you add the service. The number must match the value of the `-netnum` parameter you specified when you added the network.

Examples

This example shows how to add a database named `crm.example.com` in a specific Oracle Home directory.

```
$ srvctl add database -db crm -oraclehome /u01/oracle/product/23.0.0/mydb
-domain example.com
```

srvctl config database

Displays the configuration for an Oracle RAC database or lists all configured databases that are registered with Oracle Clusterware.

Syntax

```
srvctl config database [-db db_unique_name] [-all] [-verbose]
```

Parameters

Table A-6 srvctl config database Command Parameters

Parameter	Description
`-db` *db_unique_name*	Unique name for the database. If you do not specify this parameter, then the utility displays the configuration of all database resources.
`-all`	Print detailed configuration information.
`-verbose`	Display verbose output.

Example

This command returns output similar to the following:

```
$ srvctl config database -d main4
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

```
Database unique name: main
Database name:
Oracle home: /ade/mjkeenan_main4/oracle
Oracle user: mjkeenan
Spfile:
Password file:
Domain:
Start options: open
Stop options: immediate
Database role: PRIMARY
Management policy: AUTOMATIC
Disk Groups:
Mount point paths:
Services: test
Type: RAC
Start concurrency:
Stop concurrency:
OSDBA group: dba
OSOPER group: oper
Database instances: main41,main42
Configured nodes: mjkeenan_main4_0,mjkeenan_main4_1
CSS critical: no
CPU count: 0
Memory target : 0
Maximum memory: 0
CPU cap: 0
Database is administrator managed
```

srvctl convert database

Converts a database either to or from an Oracle RAC One Node database.

Syntax

Use this command with one of the following syntax models:

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

srvctl convert database -db db_unique_name -dbtype RAC [-node node_name]
```

Parameters

Table A-7 srvctl convert database Command Parameters

Parameter	Description
`-db db_unique_name`	Specify the unique name for the database. **Note:** If you specify a noncluster database, then command returns an error instructing you to first convert the noncluster database to Oracle RAC or Oracle RAC One Node.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-7 (Cont.) srvctl convert database Command Parameters

Parameter	Description
`-dbtype RACONENODE | RAC`	Specify the type of database to which you are converting, either Oracle RAC One Node or Oracle RAC. **Note:** If there is an ongoing or failed online database relocation, then the command returns an error instructing you to first complete or stop the online database relocation and then rerun the command.
`-instance` *`instance_name`*	Optionally, you can specify an instance name prefix for Oracle RAC One Node databases. The default value for this parameter is the first 12 characters of the global unique name of the database. **Notes:** • You can use this parameter only when converting from an Oracle RAC database to an Oracle RAC One Node database. • In order for the converted instance to come online, you must restart the database using the `srvctl stop/start database` commands.
`-timeout` *`timeout`*	Optionally, you can specify online database relocation timeout, in minutes, for Oracle RAC One Node databases. The default is `30`.
`-node` *`node_name`*	Optionally, you can specify the name of the node for an administrator-managed Oracle RAC database. The default is the first candidate. **Note:** If you do not specify a node name or you specify a node name where the database is not running, then the command returns an error instructing you specify the correct node.

Example

An example of this command is:

```
$ srvctl convert database -db myDB -dbtype RACONENODE -instance myDB3
```

srvctl disable database

Disables a running database.
If the database is a cluster database, then its instances are also disabled.

Syntax

```
srvctl disable database -db db_unique_name [-node node_name]
```

Parameters

Table A-8 srvctl disable database Command Parameters

Parameter	Description
`-db` *`db_unique_name`*	Specify the name of the database you want to disable.
`-node` *`node_name`*	Optionally, you can specify a node on which you want to disable the database. **Note:** You can only use this parameter only with Oracle Clusterware.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example

The following example disables the database `mydb1`:

```
$ srvctl disable database -db mydb1
```

srvctl downgrade database

Downgrades the configuration of a database and its services from its current version to a specific lower version.

Syntax

```
srvctl downgrade database -db db_unique_name -oraclehome Oracle_home
  -targetversion to_version
```

Parameters

Table A-9 srvctl downgrade database Command Parameters

Parameter	Description
`-db db_unique_name`	Specify the unique name of the database you want to downgrade.
`-oraclehome Oracle_home`	Specify the path to the Oracle home.
`-targetversion to_version`	Specify the database version to which you want to downgrade.

srvctl enable database

Enables a cluster database and its instances.

Syntax

```
srvctl enable database -db db_unique_name [-node node_name]
```

Parameters

Table A-10 srvctl enable database Command Parameters

Parameter	Description
`-db db_unique_name`	Specify the unique name of the database you want to enable.
`-node node_name`	Optionally, you can specify the name of the node on which the database resource resides that you want to enable. **Note:** You can only use this parameter with Oracle Clusterware.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example

The following example enables a database named `mydb1`:

```
$ srvctl enable database -db mydb1
```

srvctl getenv database

Displays the values for environment variables associated with a database.

Syntax

```
srvctl getenv database -db db_unique_name [-envs "name_list"]
```

Parameters

Table A-11 srvctl getenv database Command Parameters

Parameter	Description
`-db db_unique_name`	Specify the unique name of the database for which you want to display the environment variable values.
`-envs "name_list"`	Optionally, you can specify a comma-delimited list of the names of specific environment variables enclosed in double quotation marks ("") for which you want to display the values. If you do not use this parameter, then SRVCTL displays the values of all environment variables associated with the database.

Example

The following example displays the environment configuration for a database named `crm`:

```
$ srvctl getenv database -db crm
```

srvctl modify database

Modifies the configuration for a database.

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

Syntax

```
srvctl modify database -db db_unique_name [-dbname db_name] [-instance
instance_name]
     [-oraclehome oracle_home_path] [-localrolling] | [-localrolling_revert]
[-user user_name]
     [-server "server_list"] [-timeout timeout] [-domain db_domain]
     [-spfile spfile] [-pwfile password_file_path]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

```
     [-role {PRIMARY|PHYSICAL_STANDBY|LOGICAL_STANDBY|SNAPSHOT_STANDBY}]
     [-startoption start_options] [-stopoption stop_options]
     [-startconcurrency start_concurrency] [-stopconcurrency stop_concurrency]
     [-policy {AUTOMATIC | MANUAL | NORESTART | USERONLY | RANK}]
     [{-diskgroup "diskgroup_list" | -nodiskgroup}] [-acfspath "acfs_path |
acfs_path_list"]
     [-css_critical {YES | NO}] [-memorytarget memory_target] [-maxmemory
max_memory]
     [-defaultnetnum network_number] [-disabledreason {DECOMMISSIONED}] [-
force]
     [-verbose]
```

Parameters

Table A-12 srvctl modify database Command Parameters

Parameter	Description
`-db db_unique_name`	Unique name for the database.
`-dbname db_name`	The name of the database, if it is different from the unique name given by the `-db` parameter.
`-instance instance_name`	Instance name prefix; this parameter is required for administrator-managed Oracle RAC One Node databases.
`-oraclehome oracle_home`	The path for the Oracle AI Database home directory.
`-localrolling`	Creates a new Oracle RAC database instance to enable the local rolling feature.
`-localrolling_revert`	Use this parameter to revert the Oracle RAC local rolling configuration changes.
`-user user_name`	The name of the user who owns the Oracle home directory. **Note:** If you specify the `-user`parameter, then you must run this command in privileged mode.
`-server server_list`	List of candidate servers for Oracle RAC One Node databases. Oracle Clusterware attempts to start Oracle RAC One instance in the order in which the servers are listed. First, it attempts to start the instance on the first server in the list. If that does not succeed for any reason, then Oracle Clusterware attempts to start the instance on the second server defined in the list, and so on. For example, for a 3-node cluster: • Use "N3,N2,N1" to start a particular instance preferably on the third node. • Use "N3,N1,N2" to make the first node the designated failover node. • Use "N3,N1" to prevent the second node from being used. Oracle Clusterware refers to this list every time an instance starts, which can be either a regular start or a start that is triggered by a failure event.
`-timeout timeout`	Online database relocation timeout, in minutes, for Oracle RAC One Node databases. The default is `30`.
`-domain db_domain`	The domain for the database. **Note:** You must use this parameter if you set the `DB_DOMAIN` initialization parameter for the database.
`-spfile spfile`	The path name of the database server parameter file.
`-pwfile password_file_path`	Enter the full path to the location of the password file.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-12 (Cont.) srvctl modify database Command Parameters

Parameter	Description
`-role {PRIMARY | PHYSICAL_STANDBY | LOGICAL_STANDBY | SNAPSHOT_STANDBY}`	The role of the database in an Oracle Data Guard configuration. The default is `PRIMARY`.
`-startoption` *`start_options`*	Startup options for the database, such as `OPEN`, `MOUNT`, and `NOMOUNT`. The default value is `OPEN`. **Notes:** • For multi-word startup options, such as `read only` and `read write`, separate the words with a space and enclose in double quotation marks (`""`). For example, `"read only"`. • When performing a switch-over in an Oracle Data Guard configuration, the `-startoption` for a standby database that becomes a primary database is always set to `OPEN` after the switchover.
`-stopoption` *`stop_options`*	Stop options for the database, such as `NORMAL`, `TRANSACTIONAL`, `IMMEDIATE`, and `ABORT`.
`-startconcurrency` *`start_concurrency`*	Number of instances to be started simultaneously, or 0 to disable this option.
`-stopconcurrency` *`stop_concurrency`*	Number of instances to be stopped simultaneously, or 0 to disable this option.
`-policy {AUTOMATIC | MANUAL | NORESTART | USERONLY | RANK}`	The management policy for the database. • `AUTOMATIC` (default): The database is automatically restored to its previous running condition (started or stopped) upon restart of the database host computer. • `MANUAL`: The database is never automatically restarted upon restart of the database host computer. A `MANUAL` setting does not prevent Oracle Clusterware from monitoring the database while it is running and restarting it if a failure occurs. • `NORESTART`: Similar to the `MANUAL` setting, the database is not automatically restarted upon restart of the database host computer. A `NORESTART` setting, however, does not restart the database, even if a failure occurs, unless it must be started for dependencies, such as services or PDBs. • `USERONLY`: The database can only be restarted by user command, not as a result of any other reason (auto-start, start by dependency, node failure, and so on.) • `RANK`: The database won't be restarted when the Oracle Clusterware stack is restarted unless it is restarted by start dependencies of its PDBs that are started according to RANK. For example, 2 CDBs have `policy` set to `RANK` and their PDBs have `policy` set to `RESTART`. If a PDB of CDB1 has a rank of 3 and a PDB in CDB2 has a rank of 2, and if there are only enough resources to start one CDB, then CDB1 will be started by dependency when its PDB is started. CDB2 is not started because its PDB has a lower rank number.
`-diskgroup "`*`disk_group_list`*`"`	Comma-delimited list of Oracle ASM disk groups if database uses Oracle ASM storage.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-12 (Cont.) srvctl modify database Command Parameters

Parameter	Description
`-acfspath "acfs_path | acfs_path_list"`	You can add (/+) or remove (/-) a single Oracle ACFS path with `acfs_path` or multiple Oracle ACFS paths using a comma-delimited list with `acfs_path_list` enclosed in double quotation marks ("") where the database's dependency is set. Use this parameter to create dependencies on Oracle ACFS file systems other than `ORACLE_HOME`, such as for when the database uses `ORACLE_BASE` on a file system that is different from the `ORACLE_HOME` file system. **Note** This parameter is available only on the platforms on which Oracle ACFS is supported.
`-css_critical {YES | NO}`	You can add weight to a service by setting this parameter to YES. In the event of a node failure within the cluster, Oracle Clusterware will evict the node with the least amount of weight, ensuring that critical services remain available.
`-memorytarget memory_target`	Specify the target memory, in MB, to be allocated for the database. The default is 0.
`-maxmemory max_memory`	Specify the maximum memory, in MB, to be allocated for the resource. If you specify `-memorytarget` but not `-maxmemory`, then `-maxmemory` will be the default value of 0. Both `-maxmemory` and `-memorytarget` are validated as long as `-memorytarget` is less than or equal to `-maxmemory`.
`-defaultnetnum network_number`	Specify a network number to which services will default in the event you do not specify a network number when you add a service.
`-disabledreason {DECOMMISSIONED}`	Marks the database as being decommissioned, which means it cannot be started again and is not being used. This is intended for databases that will be deleted at a future date.

Usage Notes

- When using the `srvctl modify database` command, for a running database, if the server list is supplied, then the node where the database is running must be on that list.
- The instance name prefix cannot be modified after running the `srvctl add database` command.
- You cannot change the management policy from `AUTOMATIC` (using the `-policy` parameter) for Oracle RAC One Node databases. Any attempt to do so results in an error message.

Examples

The following example changes the role of a database to a logical standby:

```
$ srvctl modify database -db crm -role logical_standby
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

The following example directs the `racTest` database to use the `SYSFILES`, `LOGS`, and `OLTP` Oracle ASM disk groups:

```
$ srvctl modify database -db racTest -diskgroup "SYSFILES,LOGS,OLTP"
```

Related Topics

- Oracle Data Guard Configurations
- Database Startup
- Database Shutdown

srvctl predict database

Predicts the consequences of the failure of a specific database.

Syntax

```
srvctl predict database -db db_unique_name [-verbose]
```

Usage Notes

- Specify the unique name of the database you want to check.
- Optionally, you can use the `–verbose` parameter to display detailed output.

srvctl relocate database

Initiates the relocation of an Oracle RAC One Node database from one node to another node.

This command also cleans up after a failed relocation, and you can only use it for relocating Oracle RAC One Node databases.

Syntax

Use this command with one of the following syntax models:

To initiate the online relocation of an Oracle RAC One Node database:

```
srvctl relocate database -db db_unique_name {[-node target_node] [-timeout
timeout]
   [-stopoption NORMAL] | -abort [-revert]} [-drain_timeout drain_timeout] [-
eval] [-verbose]
```

To stop the failed online relocation of an Oracle RAC One Node database:

```
srvctl relocate database -db db_unique_name -abort [-revert]
   [-drain_timeout drain_timeout] [-eval] [-verbose]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

Table A-13 srvctl relocate database Command Parameters

Parameter	Description
`-db` *`db_unique_name`*	Specify the unique name of the database you want to relocate.
`-node` *`target_node`*	Optionally, you can specify a target node to which to relocate the Oracle RAC One Node database. **Note:** You must use this parameter if you are relocating an administrator-managed Oracle RAC One Node database.
`-timeout` *`timeout`*	Optionally, you can specify an online database relocation timeout, in minutes, for Oracle RAC One Node databases. The default is `30`.
`-stopoption NORMAL`	Use this parameter to override the default shutdown option for a running instance, such as the default of `SHUTDOWN TRANSACTIONAL LOCAL` for a primary database or `SHUTDOWN IMMEDIATE` for a standby database. The only value accepted for `-stopoption` is `NORMAL`.
`–abort`	Use this parameter to stop a failed online database relocation.
`–revert`	Use this parameter to remove the target node of a failed online relocation request from the candidate server list of an administrator-managed Oracle RAC One Node database.
`-drain_timeout` *`timeout`*	Specify the time, in seconds, allowed for resource draining to be completed. Accepted values are an empty string (""), 0, or any positive integer. The default value is an empty string, which means that this parameter is not set. If it is set to 0, then draining occurs, immediately. The draining period is intended for planned maintenance operations. During the draining period, all current client requests are processed, but new requests are not accepted. When set on the service this value is used when the command line value is not set.
`–eval`	Evaluates the effects of the operation without making any changes to the system.
`–verbose`	Use this parameter to display verbose output.

Usage Notes

- If the Oracle RAC One Node database you want to relocate is not running, then the command returns an error.
- If another online database relocation is active for this Oracle RAC One Node database, then the command returns an error.
- If the `-drain_timeout` value is higher than the `-timeout` value, then SRVCTL relocates the services but does not explicitly start or stop the services on the database instances.
- If an online database relocation for this Oracle RAC One Node database has failed and the target nodes are not the same for either relocation, then the command returns an error instructing you to stop the failed online database relocation and then initiate a new one.
- If an online database relocation for this Oracle RAC One Node database has failed and the target nodes are the same (or you do not specify the target), then the command attempts to relocate the database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example

The following example relocates an Oracle RAC One Node database named `rac1` to a server called `node7`.

```
$ srvctl relocate database -db rac1 -node node7
```

srvctl remove database

Removes database configurations.

After running this command, ensure that the password file is in the default location if you want to connect to the database as the SYS user with the SYS user's password.

Syntax

```
srvctl remove database -db db_unique_name [-force] [-noprompt] [-verbose]
```

Parameters

Table A-14 srvctl remove database Command Parameters

Parameter	Description
`-database` *`db_unique_name`*	Unique name for the database.
`-force`	Forcibly remove the database and ignore any dependencies.
`-noprompt`	Suppress prompts.
`-verbose`	Display verbose output.

Example

To remove a database named `crm`:

```
$ srvctl remove database -db crm
```

srvctl setenv database

Administers cluster database environment configurations.

Syntax

Use this command with one of the following syntax models:

```
srvctl setenv database -db db_unique_name -envs "name=val[,...]"
```

```
srvctl setenv database -db db_unique_name -env "name=val"
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

Table A-15 srvctl setenv database Command Parameters

Parameter	Description
`-db db_unique_name`	Specify a unique name for the database for which you want to set environment variables.
`-envs "name=val[,...]"`	Specify a comma-delimited list of name-value pairs of environment variables enclosed in double quotation marks ("") that you want to set.
`-env "name=val"`	Specify a single environment variable that you want to set to a value that contains commas or other special characters enclosed in double quotation marks ("").

Usage Notes

Add additional information about the command here.

Example

The following example sets the language environment variable for a cluster database:

```
$ srvctl setenv database -db crm -env LANG=en
```

srvctl start database

Starts a database and its enabled instances, and all listeners on nodes with database instances.
You can disable listeners that you do not want to start.

Syntax

```
srvctl start database -db db_unique_name [-eval] [-startoption start_options]
  [-startconcurrency number_of_instances] [-node node_name]
```

Parameters

Table A-16 srvctl start database Command Parameters

Parameter	Description
`-db db_unique_name`	Specify the unique name of the database you want to start.
`–eval`	Optionally, use this parameter to hypothetically evaluate the impact of the command on the system.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-16 (Cont.) srvctl start database Command Parameters

Parameter	Description
`-startoption` *`start_options`*	Optionally, you can set options for the startup command (for example: `OPEN`, `MOUNT`, or `NOMOUNT`). **Notes:** • This command parameter supports all database startup options. • For multi-word startup options, such as `read only` and `read write`, separate the words with a space and enclose in double quotation marks (`""`). For example, `"read only"`. **See Also:** *SQL*Plus User's Guide and Reference* for more information about startup options
`-startconcurrency` *`number_of_instances`*	Optionally, you can specify a number of database instances to start simultaneously, or specify `0` for an empty start concurrency value. When you use this parameter with the `srvctl start database` command, it overrides any `-startconcurrency` value configured using the `srvctl add | modify database` commands. **Note** If the value for the `-startconcurrency` parameter is greater than the number of total instances, then this parameter does not have any effect, same as `0`.
`-node` *`node_name`*	Optionally, you can specify the name of a node on which you want to start the database. **Notes:** • This command only applies to Oracle RAC One Node and Standard Edition High Availability databases. • The node you specify must be in the candidate list for an administrator-managed Oracle RAC One Node or Standard Edition High Availability database. • If the database is already running on a node than the one you specify, then the command returns an error. • If you do not specify a node, then Oracle Clusterware chooses which node on which to start the Oracle RAC One Node or Standard Edition High Availability database according to its policies, such as dispersion, number of resources, and order of candidate nodes. • If there is an active online database relocation for the Oracle RAC One Node database you are attempting to start, then both instances will already be running and the command returns an error message. Only during an online database relocation are two instances of an Oracle RAC One Node database in existence. If the online database relocation failed for the Oracle RAC One Node database and you do not specify a node, then the command attempts to start both database instances. If the online database relocation failed for the Oracle RAC One Node database and you specify a node, then the command attempts to stop the failed relocation and start the instance on that node.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Examples

The following example starts the `crm` database and sets the startup option to read only:

```
$ srvctl start database -db crm -startoption "read only"
```

srvctl status database

This command displays the current state of the of the database.

Syntax

```
srvctl status database {-db db_unique_name {[-sid] [-home]} | -thisversion | -
thishome}
   [-force] [-detail] [-verbose]
```

Parameters

Table A-17 srvctl status database Parameters

Parameter	Description
`-db db_unique_name`	Specify a unique name for the database.
`–sid`	Use this parameter to display the SID of the Oracle instance running on this node.
`–home`	Use this parameter to display the Oracle home of the specified database.
`-thisversion`	Use this parameter to display the status of databases that are of the same Oracle product version as SRVCTL.
`-thishome`	Use this parameter to display the status of databases that are configured in this Oracle home.
`-force`	Include disabled applications
`–detail`	Use this parameter to display detailed database status information.

`-verbose`	Displays `STATE_DETAILS` and `INTERNAL_STATE` attributes, which include `STABLE`, `STARTING`, `STOPPING`, and `CLEANING`.
	If the `INTERNAL_STATE` is `STABLE`, then SRVCTL displays no additional information. If the `INTERNAL_STATE` is `STARTING`, then SRVCTL displays:
	`Instance instance_name is being started`
	If the `INTERNAL_STATE` is `CLEANING`, then SRVCTL displays:
	`Instance instance_name is being cleaned up`
	If the `INTERNAL_STATE` is `STOPPING`, then SRVCTL displays:
	`Instance instance_name is being stopped`

Copyright © 1999, 2025, Oracle and/or its affiliates.

Usage Notes

The output of this command includes information on the Oracle ASM or Oracle ASM IOServer instance for each running instance of the database.

Examples

This command displays output similar to the following:

```
$ srvctl status database -db db00 -detail

Instance db00_1 is connected to ASM instance +ASM3
Instance db00_2 is connected to ASM I/O server instance +IOS1
```

srvctl stop database

Stops a database, its instances, and its services.

Syntax

```
srvctl stop database -db db_unique_name [-stopoption stop_options]
  [-stopconcurrency number_of_instances] [-drain_timeout timeout] [-eval]
  [-force] [-verbose]
```

Parameters

Table A-18 srvctl stop database Command Parameters

Parameter	Description
`-db db_unique_name`	Specify the unique name for the database that you want to stop.
`-stopoption stop_options`	Optionally, you can specify options for the shutdown command, such as `NORMAL`, `TRANSACTIONAL LOCAL`, `IMMEDIATE`, or `ABORT`.
`-stopconcurrency number_of_instances`	Optionally, you can specify a number of database instances to stop simultaneously, or specify `0` for an empty stop concurrency value. When you use this parameter with the `srvctl stop database` command, it overrides any `-stopconcurrency` value configured using the `srvctl add | modify database` commands. **Note** If the value for the `-stopconcurrency` parameter is greater than the number of total instances, then this parameter does not have any effect, same as `0`.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-18 (Cont.) srvctl stop database Command Parameters

Parameter	Description
`-drain_timeout` *timeout*	Optionally, you can specify the time, in seconds, allowed to complete the resource draining action. By default, this parameter is not set. You can specify 0 or any positive integer. An empty string unsets the parameter. If you specify zero, then the agent will perform the actions related to service draining, immediately. Drain timeout is the maximum time the service waits before exiting (in case of `srvctl stop service` or `srvctl stop instance`) or proceeding to stop database (`srvctl stop database`), until the draining of sessions is completed. If session draining completes in 10 seconds and the drain timeout value (on CLI or resource attribute) is 100 seconds, then SRVCTL moves on after 10 seconds. It does not wait for the remaining 90 seconds. **Note:** If the `-drain_timeout` value is not specified, then the maximum configured drain timeout of all the running services of the database on the specified nodes is used.
`-eval`	Optionally, you can use this parameter to hypothetically evaluate the impact of the command on the system.
`-force`	Optionally, you can use this parameter to stop the database, its instances, its services, and any resources that depend on those services.
`–verbose`	Optionally, you can use this parameter to display detailed output.

Example

The following command example stops a database and includes detailed output:

```
$ srvctl stop database -db db1 -drain_timeout 50 -verbose
Draining in progress on services svc1,svc2.
Drain complete on services svc1.
Draining in progress on services svc2.
Draining in progress on services svc2.
Drain complete on services svc2.
```

srvctl unsetenv database

Unsets the cluster database environment configurations.

Syntax

```
srvctl unsetenv database -db db_unique_name -envs "name_list"
```

Parameters

Table A-19 srvctl unsetenv database Command Parameters

Parameter	Description
`-db` *db_unique_name*	Specify the unique name of the database for which you want to unset environment variables.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-19 (Cont.) srvctl unsetenv database Command Parameters

Parameter	Description
`-envs "name_list"`	Specify a comma-delimited list of environment variable names enclosed in double quotation marks ("").

Example

The following example unsets two cluster database environment variables:

```
$ srvctl unsetenv database -db crm -envs "CLASSPATH,LANG"
```

srvctl update database

Updates the specified database to use the new listener endpoints.

Syntax

```
srvctl update database -db db_unique_name [-startoption start_options [-node node_name]]
```

Usage Notes

- You can only use this command with Oracle Clusterware.
- Specify the unique name of the database you want to update.

srvctl upgrade database

Upgrades the configuration of a database and all of its services to the version of the database home from where this command is run.

Syntax

```
srvctl upgrade database -db db_unique_name -oraclehome Oracle_home
```

Parameters

Table A-20 srvctl upgrade database Command Parameters

Parameter	Description
`-db db_unique_name`	Specify the unique name of the database you want to upgrade.
`-oraclehome Oracle_home`	Specify the path to the upgraded `ORACLE_HOME`.

diskgroup Commands

Use commands with the `diskgroup` keyword to manage Oracle ASM disk groups.

You can add, modify, list the configuration of, enable, disable, start, stop, obtain the status of, and remove Oracle ASM disk groups.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- srvctl disable diskgroup
- srvctl enable diskgroup
- srvctl predict diskgroup
- srvctl remove diskgroup
- srvctl start diskgroup
- srvctl status diskgroup
- srvctl stop diskgroup

srvctl disable diskgroup

Disables a specific disk group on a number of specified nodes.

Syntax

```
srvctl disable diskgroup -diskgroup diskgroup_name [-node "node_list"]
```

Parameters

Table A-21 srvctl disable diskgroup Command Parameters

Parameter	Description
`-diskgroup diskgroup_name`	Specify the name of the Oracle ASM disk group you want to disable.
`-node "node_list"`	Optionally, you can specify a comma-delimited list of node names enclosed in double quotation marks ("") on which to disable the disk group. **Note:** You can only use this parameter with Oracle Clusterware.

Example

The following example disables the Oracle ASM disk group, `dgroup1`, on two nodes in a cluster, `mynode1` and `mynode2`:

```
$ srvctl disable diskgroup -diskgroup dgroup1 -node "mynode1,mynode2"
```

srvctl enable diskgroup

Enables a specific disk group on a number of specified nodes.

Syntax

```
srvctl enable diskgroup -diskgroup diskgroup_name [-node "node_list"]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

Table A-22 srvctl enable diskgroup Command Parameters

Parameter	Description
`-diskgroup` *`diskgroup_name`*	Specify the name of the Oracle ASM disk group you want to enable.
`-node "`*`node_list`*`"`	Optionally, you can specify a comma-delimited list of node names enclosed in double quotation marks ("") on which to enable the disk group. **Note:** You can only use this parameter with Oracle Clusterware.

Example

The following example enables the `diskgroup1` Oracle ASM disk group on nodes `mynode1` and `mynode2`:

```
$ srvctl enable diskgroup -diskgroup diskgroup1 -node "mynode1,mynode2"
```

srvctl predict diskgroup

Predicts the consequences of an Oracle ASM disk group failure.

Syntax

```
srvctl predict diskgroup -diskgroup diskgroup_name [-verbose]
```

Usage Notes

Specify the name of the Oracle ASM disk group for which you want to evaluate a failure. Optionally, you can use the `–verbose` parameter top print detailed output.

srvctl remove diskgroup

Removes a specific Oracle ASM disk group resource from Oracle Clusterware or Oracle Restart.

Syntax

```
srvctl remove diskgroup -diskgroup diskgroup_name [-force]
```

Usage Notes

Specify the name of the Oracle ASM disk group you want to remove. Optionally, you can use the `–force` parameter to ignore any dependencies and forcibly remove the disk group.

Example

The following example forcibly removes the `DG1` Oracle ASM disk group:

```
$ srvctl remove diskgroup -diskgroup DG1 -force
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl start diskgroup

Starts a specific Oracle ASM disk group resource on a number of specified nodes.

Syntax

```
srvctl start diskgroup -diskgroup diskgroup_name [-node "node_list"]
```

Parameters

Table A-23 srvctl start diskgroup Command Parameters

Parameter	Description
`-diskgroup diskgroup_name`	Specify the name of the Oracle ASM disk group you want to start.
`-node "node_list"`	Optionally, you can specify a comma-delimited list of node names enclosed in double quotation marks ("") on which to start the disk group resource. **Note:** You can only use this parameter with Oracle Clusterware.

Example

The following example starts the `diskgroup1` Oracle ASM disk group on the nodes `mynode1` and `mynode2`:

```
$ srvctl start diskgroup -diskgroup diskgroup1 -node "mynode1,mynode2"
```

srvctl status diskgroup

Displays the status of a specific disk group on a number of specified nodes.

Syntax

```
srvctl status diskgroup -diskgroup diskgroup_name [-node "node_list"]
  [-detail] [-verbose]
```

Parameters

Table A-24 srvctl status diskgroup Command Parameters

Parameter	Description
`-diskgroup diskgroup_name`	Specify the name of the Oracle ASM disk group for which you want to display the status.
`-node "node_list"`	Optionally, you can specify a comma-delimited list of node names on which to check status of an Oracle ASM disk group. **Note:** You can only use this parameter with Oracle Clusterware.
`-detail`	Optionally, you can use this parameter to display detailed status information for the Oracle ASM disk group.
`-verbose`	Optionally, you can use this parameter to display verbose output.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Examples

The following example displays the status of the `dgrp1` Oracle ASM disk group:

```
$ srvctl status diskgroup -diskgroup dgrp1 -node "mynode1,mynode2" -detail
```

srvctl stop diskgroup

Stops a specific Oracle ASM disk group resource on a number of specified nodes.

Syntax

```
srvctl stop diskgroup -diskgroup diskgroup_name [-node "node_list"] [-force]
```

Parameters

Table A-25 srvctl stop diskgroup Command Parameters

Parameter	Description
`-diskgroup` *`diskgroup_name`*	Specify the name of the Oracle ASM disk group you want to stop.
`-node "`*`node_list`*`"`	Optionally, you can specify a comma-delimited list of node names enclosed in double quotation marks ("") on which to stop the Oracle ASM disk group resource. **Note:** You can only use this parameter with Oracle Clusterware.
`-force`	Optionally, you can use this parameter to perform a forceful dismount. While this parameter does not stop the databases that depend on the disk group you are stopping, it still may cause those databases to fail.

Example

The following command stops the `diskgroup1` Oracle ASM disk group on the two nodes `mynode1` and `mynode2`:

```
$ srvctl stop diskgroup -diskgroup diskgroup1 -node "mynode1,mynode2" -force
```

home Commands

Use commands with the `home` keyword to start, stop, and obtain the status of all clusterware resources related to a Home directory.

- srvctl start home
- srvctl status home
  Displays the status of all the Oracle Restart-managed or Oracle Clusterware-managed resources for the specified Oracle home.
- srvctl stop home
  Stops all the Oracle Restart-managed or Oracle Clusterware-managed resources that run from the specified Oracle home.

Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl start home

Starts all the Oracle Restart-managed or Oracle Clusterware-managed resources on the specified Oracle home.

Syntax

```
srvctl start home -oraclehome Oracle_home -statefile state_file -node
node_name
```

Parameters

Table A-26 srvctl start home Command Parameters

Parameter	Description
`-oraclehome` *`Oracle_home`*	Specify the path to the Oracle home for which you want to start the Oracle Restart or Oracle Clusterware-managed resources.
`-statefile` *`state_file`*	Specify the path to the directory where you want SRVCTL to write the state file.
`-node` *`node_name`*	Specify the name of the node on which the Oracle home resides. **Note:** You can only use this parameter with Oracle Clusterware.

Example

The following command starts an Oracle home:

```
$ srvctl start home -oraclehome /u01/app/oracle/product/12.2.0/db_1
  -statefile ~/state.txt -node node1
```

srvctl status home

Displays the status of all the Oracle Restart-managed or Oracle Clusterware-managed resources for the specified Oracle home.

Syntax

```
srvctl status home -oraclehome Oracle_home -statefile state_file -node
node_name
```

Parameters

Table A-27 srvctl status home Command Parameters

Parameter	Description
`-oraclehome` *`Oracle_home`*	Specify the path to the Oracle home for which you want to start the Oracle Restart or Oracle Clusterware-managed resources.
`-statefile` *`state_file`*	Specify the path to the directory that contains the text file that holds the state information generated by this command.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-27 (Cont.) srvctl status home Command Parameters

Parameter	Description
`-node node_name`	Specify the name of the node on which the Oracle home resides. **Note:** This parameter is required and you can only use it with Oracle Clusterware.

Example

The following example obtains the status of a particular Oracle home:

```
$ srvctl status home -oraclehome /u01/app/oracle/product/23.4.0/dbhome_1 -
statefile
 ~/state.txt -node stvm12
```

The preceding command returns output similar to the following:

```
Database cdb1 is running on node stvm12
```

srvctl stop home

Stops all the Oracle Restart-managed or Oracle Clusterware-managed resources that run from the specified Oracle home.

Syntax

```
srvctl stop home -oraclehome Oracle_home -statefile state_file -node node_name
  [-stopoption stop_options] [-force]
```

Parameters

Table A-28 srvctl stop home Command Parameters

Parameter	Description
`-oraclehome Oracle_home`	Specify the directory path to the Oracle home for which you want to start the Oracle Restart or Oracle Clusterware-managed resources. **Note:** The path to the Oracle home you specify must be the same version as the Oracle home from which you invoke SRVCTL.
`-statefile state_file`	Specify the path to the directory where you want SRVCTL to write the state file.
`-node node_name`	Specify the name of the node on which the Oracle home resides. **Note:** You can only use this parameter with Oracle Clusterware.
`-stopoption stop_options`	Optionally, you can specify shutdown options for the database, such as `NORMAL`, `TRANSACTIONAL`, `IMMEDIATE`, or `ABORT` **See Also:** *SQL*Plus User's Guide and Reference* for more information about shutdown options
`-force`	Optionally, you can use this parameter to stop the resources even if errors are reported.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example

The following example stops the Oracle home:

```
$ srvctl stop home -oraclehome /u01/app/oracle/product/23.4.0/db_1 -statefile
 ~/state.txt
```

instance Commands

Use commands with the `instance` keyword to add, modify, enable, disable, start, stop, obtain the status of, and remove database instances.

- srvctl add instance
  Adds a configuration for an instance to your cluster database configuration.
- srvctl disable instance
- srvctl enable instance
- srvctl modify instance
- srvctl remove instance
- srvctl start instance
  Starts instances and their dependencies in the cluster database.
- srvctl status instance
  Displays the status of instances.
- srvctl stop instance
  The `srvctl stop instance` command stops instances and stops any services running on specified instances.
- srvctl transfer instance
  Transfers Oracle RAC and Oracle Oracle RAC One Node databases, PDBs, and services from the old Oracle home to the new Oracle home in single-server rolling database maintenance.
- srvctl update instance
  The `srvctl update instance` command changes the open mode or the target Oracle ASM instance of the database instances.

srvctl add instance

Adds a configuration for an instance to your cluster database configuration.

Syntax

```
srvctl add instance -db db_unique_name -instance instance_name
     -node node_name [-force]
```

Parameters

Table A-29 srvctl add instance Command Parameters

Parameter	Description
`-db db_unique_name`	The unique name of the database you are adding the instance to

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-29 (Cont.) srvctl add instance Command Parameters

Parameter	Description
`-instance` *`instance_name`*	The name of the instance you are adding
`-node` *`node_name`*	The name of the node on which you are creating the instance
`-force`	Optionally, you can force the add operation, even though some resources will be stopped.

Usage Notes

- You can only use this command with Oracle Clusterware and Oracle RAC.
- This command increments the `CARDINALITY` resource attribute.
- If you attempt to use this command on an Oracle RAC One Node database, then the command returns an error stating you must convert the database to Oracle RAC.

Examples

Examples of this command are:

```
$ srvctl add instance -db crm -instance crm01 -node gm01
$ srvctl add instance -db crm -instance crm02 -node gm02
$ srvctl add instance -db crm -instance crm03 -node gm03
```

srvctl disable instance

Disables a database instance.
If the database instance that you disable with this command is the last enabled database instance, then this operation also disables the database.

> **Note**
>
> - This command is only available with Oracle Clusterware and Oracle RAC.
> - If you run this command on an Oracle RAC One Node database, then the command returns an error instructing you to use the `database` noun, instead.

Syntax

```
srvctl disable instance -db db_unique_name -instance "instance_name_list"
```

Parameters

Table A-30 srvctl disable instance Command Parameters

Parameter	Description
`-db` *`db_unique_name`*	Specify the unique name for the database for which you want to disable the instance.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-30 (Cont.) srvctl disable instance Command Parameters

Parameter	Description
`-instance "instance_name_list"`	Specify an instance name or a comma-delimited list of instance names enclosed in double quotation marks ("") you want to disable.

Example

The following example disables two instances of the `crm` database, named `crm1` and `crm2`:

```
$ srvctl disable instance -db crm -instance "crm1,crm3"
```

srvctl enable instance

Enables an instance of an Oracle RAC database.
If you use this command to enable all instances, then the database is also enabled.

> **Note**
>
> - You can only use this command with Oracle Clusterware and Oracle RAC.
> - If you run this command on an Oracle RAC One Node database, then the command returns an error instructing you to use the `database` noun, instead.

Syntax

```
srvctl enable instance -db db_unique_name -instance "instance_name_list"
```

Parameters

Table A-31 srvctl enable instance Command Parameters

Parameter	Description
`-db db_unique_name`	Specify the unique name of the database for which you want to enable instances.
`-instance "instance_name_list"`	Specify a comma-delimited list of instance names enclosed in double quotation marks ("") that you want to enable.

Example

The following example enables two instances of the `crm` database:

```
$ srvctl enable instance -db crm -instance "crm1,crm2"
```

srvctl modify instance

This command modifies the configuration for a database instance from its current node to another node.

Copyright © 1999, 2025, Oracle and/or its affiliates.

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

Syntax

```
srvctl modify instance -db db_unique_name -instance instance_name
     -node node_name
```

Parameters

Table A-32 srvctl modify instance Command Parameters

Parameter	Description
`-database db_unique_name`	Specify the unique name for the database.
`-instance instance_name`	Specify the database instance name. **Note** If you specify an instance name that has never been started before, then you may have to assign an instance number, undo, and redo in the SPFILE.
`-node node_name`	Name of the node on which to run the instance.

Usage Notes

You *cannot* use this command to rename or relocate a running instance.

Examples

The following example to changes the configuration of a database, `amdb`, so that the database instance, `amdb1`, runs on the specified node, `mynode`:

```
$ srvctl modify instance -db amdb -instance amdb1 -node mynode
```

The following example removes the directive established by the previous example:

```
$ srvctl modify instance -db pmdb -instance pmdb1_1 -node ""
```

srvctl remove instance

Removes the configurations for an instance of a database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Syntax

```
srvctl remove instance -db db_unique_name -instance instance_name
  [-noprompt] [-force]
```

Parameters

Table A-33 srvctl remove instance Command Parameters

Parameter	Description
`-db db_unique_name`	Specify the unique name of the database.
`-instance instance_name`	Specify the name of the instance that you want to remove.
`-noprompt`	Use this parameter to suppress prompts.
`–force`	Use this parameter to skip checking that the instance is not running, and remove it even though it is running. This parameter also skips checking that the instance has no running services using it, and causes those services to stop before the instance is removed.

Usage Notes

- You can use this command only with Oracle Clusterware and Oracle RAC.
- If you use the `-force` parameter, then any services running on the instance stop. Oracle recommends that you reconfigure services to not use the instance you want to removed as a preferred or available instance before removing the instance.
- If you attempt to use this command on an Oracle RAC One Node database, then the command returns an error stating that cannot remove the instance except by removing the database.

Example

The following example removes the `crm01` database instance from the `crm` database.

```
$ srvctl remove instance -db crm -instance crm01
```

srvctl start instance

Starts instances and their dependencies in the cluster database.

Use the `srvctl start instance` command to start database instances, and all listeners on nodes with database instances.

Syntax

Use the `srvctl start instance` command with one of these syntax models:

To start all Oracle Clusterware managed database instances on one or more nodes:

```
srvctl start instance -node "node_list" [-startoption start_options]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

To start an instance of a database on a specific node:

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

To start an instance of a database on one or more nodes:

```
srvctl start instance -db db_unique_name -node "node_list" [-startoption
start_options]
```

To start specific instances of a database on available nodes:

```
srvctl start instance -db db_unique_name -instance "inst_name_list"
    [-startoption start_options]
```

Parameters

Table A-34 srvctl start instance Parameters

Parameter	Description
`-db db_unique_name`	Unique name for the database
`-node node_name` or `-node "node_list"`	The name of a single node or a comma-delimited list of node names
`-instance "instance_name"` or `-instance "inst_name_list"`	The name of a single instance or a comma-delimited list of instance names
`-startoption start_options`	Options for startup command, such as `OPEN`, `MOUNT`, or `NOMOUNT`) **Note:** For multi-word startup options, such as `read only` and `read write`, separate the words with a space and enclose in double quotation marks (`""`). For example, `"read only"`.

Usage Notes

- This command is only available with Oracle Clusterware and Oracle RAC.
- If you run this command on an Oracle RAC One Node database, then the command returns an error instructing you to use the `database` noun, instead.

Related Topics

- STARTUP

srvctl status instance

Displays the status of instances.

Syntax

```
srvctl status instance -db db_unique_name {-node node_list | -instance
instance_name_list}
    [-force] [-detail] [-verbose]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

Table A-35 srvctl status instance Command Parameters

Parameter	Description
`-db db_unique_name`	The unique name of the database.
`-node node_list`	A comma-separated list of node names.
`-instance instance_name_list`	A comma-separated list of instance names.
`-force`	Optionally, you can include a list of the disabled applications.
`-detail`	Optionally, print detailed status information of the instance.
`-verbose`	Display verbose output.

Usage Notes

You can only use this command with Oracle Clusterware and Oracle RAC.

Examples

Example of this command is:

```
$ srvctl status instance -db crm -node gm01,gm02
```

srvctl stop instance

The `srvctl stop instance` command stops instances and stops any services running on specified instances.

Syntax

Use this command with one of the following syntax models.

To stop all instances on one or more nodes:

```
srvctl stop instance -node "node_list" [-stopoption stop_options]
    [-drain_timeout timeout] [-force] [-failover] [-verbose]
```

To stop instances for a database that are running on specific nodes:

```
srvctl stop instance -db db_unique_name -node "node_list"
    [-stopoption stop_options] [-drain_timeout timeout] [-force] [-failover]
[-verbose]
```

To stop one or more instances by name for a database:

```
srvctl stop instance -db db_unique_name -instance "instance_name_list"
    [-stopoption stop_options] [-drain_timeout timeout] [-force] [-failover]
[-verbose]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

Table A-36 srvctl stop instance Command Parameters

Parameter	Description
`-db db_unique_name`	
`-node "node_list"`	
`-instance "instance_name_list"`	
`-stopoption stop_options`	
`-drain_timeout timeout`	
`-force`	
`–failover`	
`-verbose`	

Usage Notes

If you run this command on an Oracle RAC One Node database, then the command returns an error instructing you to use the `srvctl stop database` command instead.

Example

The following command example stops the instance of the `db1` database running on the node `server1`, and includes verbose output:

```
$ srvctl stop instance -db db1 -node server1 -drain_timeout 50 -verbose
Draining in progress on services svc1
Draining in progress on services svc1
Drain complete on services svc1
```

Related Topics

- Database Shutdown

srvctl transfer instance

Transfers Oracle RAC and Oracle Oracle RAC One Node databases, PDBs, and services from the old Oracle home to the new Oracle home in single-server rolling database maintenance.

Syntax

```
srvctl transfer instance -db db_unique_name [-node node_list]
   [-stopoption stop_options] [-drain_timeout timeout] [-rollback] [-verbose]
```

Parameters

Table A-37 srvctl transfer instance Command Parameters

Parameter	Description
`-db db_unique_name`	Specify the unique name of the database you want to relocate.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-37 (Cont.) srvctl transfer instance Command Parameters

Parameter	Description
`-node target_node`	Comma-separated list of nodes on which you want to transfer the instances from the old Oracle homes to the new Oracle homes.
`-stopoption NORMAL`	Use this parameter to override the default shutdown option for the instance running from the old Oracle home, such as `NORMAL`, `TRANSACTIONAL`, `IMMEDIATE`, and `ABORT`.
`-drain_timeout timeout`	Specify the time, in seconds, allowed for resource draining to be completed. Accepted values are an empty string (""), 0, or any positive integer. The default value is an empty string, which means that this parameter is not set. If it is set to 0, then draining occurs, immediately. The draining period is intended for planned maintenance operations. During the draining period, all current client requests are processed, but new requests are not accepted. When set on the service this value is used when the command line value is not set. **Note:** If the `-drain_timeout` value is not specified, then the maximum configured drain timeout of all the running services of the database on the specified nodes is used.
`-rollback`	Use this parameter to rollback an online instance transfer.
`–verbose`	Use this parameter to display verbose output.

Example

The following example transfers an Oracle RAC One Node database instance of database named `rac1` on the node called `node1`.

```
$ srvctl transfer instance -db rac1 -node node1
```

srvctl update instance

The `srvctl update instance` command changes the open mode or the target Oracle ASM instance of the database instances.

Syntax

```
srvctl update instance -db db_unique_name [-instance "instance_name_list"
    | -node "node_list"] [-startoption start_options] [-targetinstance
instance_name]
```

Parameters

Parameter	Description
`-db db_unique_name`	The unique name of the database
`-instance "instance_name_list" | -node "node_list"`	A comma-delimited list of instance names or node names that you want to update. If you specify a list of node names, then SRVCTL udpates the instances running on those specific nodes.
`-startoption start_options`	The specify startup options for the database, such as `OPEN`, `MOUNT`, or `"READ ONLY"`

Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameter	Description
`-targetinstance` *`instance_name`*	The target Oracle ASM or Oracle ASM IOServer instance. Use double quotation marks ("") with no space in-between to specify the default target instance.

Examples

An example of this command is:

```
$ srvctl update instance -db db00 -instance db00_3 -targetinstance +ASM2
```

listener Commands

Use commands with the `listener` keyword to add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, and remove listeners.

- srvctl add listener
  Adds a listener to every node in a cluster.
- srvctl config listener
  Displays configuration information of a specific listener that is registered with Oracle Clusterware.
- srvctl disable listener
- srvctl enable listener
- srvctl getenv listener
- srvctl modify listener
- srvctl predict listener
- srvctl remove listener
- srvctl setenv listener
- srvctl start listener
- srvctl status listener
  Displays the status of listener resources.
- srvctl stop listener
- srvctl unsetenv listener
- srvctl update listener
  Updates the listener to listen on the new endpoints.

srvctl add listener

Adds a listener to every node in a cluster.

Syntax

Use this command with one of the following syntax models.

Copyright © 1999, 2025, Oracle and/or its affiliates.

To create an Oracle AI Database listener:

```
srvctl add listener [-listener listener_name] [-netnum network_number] [-
oraclehome Oracle_home]
  [-user user_name] [-endpoints "[TCP:]port_list[:FIREWALL={ON|OFF}][/IPC:key]
[/NMP:pipe_name]
  [/{TCPS|SDP|EXADIRECT}port_list[:FIREWALL={ON|OFF}]]" [-group group_name]]
[-skip]
```

To create an Oracle ASM listener:

```
srvctl add listener [-listener listener_name] -asmlistener [-subnet subnet]
  [-endpoints "[TCP:]port_list[:FIREWALL={ON|OFF}][/IPC:key][/NMP:pipe_name]
  [/{TCPS|SDP|EXADIRECT}port_list[:FIREWALL={ON|OFF}]]" [-group group_name]]
[-skip]
```

To create a SCAN listener, use the srvctl add scan_listener command.

Parameters

Table A-38 srvctl add listener Command Parameters

Parameter	Description
`-listener` *listener_name*	Specify a listener name. This parameter is optional. If you do not specify this parameter, then the name of the listener defaults to `LISTENER` for a database listener or `LISTENER_ASM` for an Oracle ASM listener.
`-netnum` *network_number*	The optional network number from which VIPs are obtained. If not specified, the VIPs are obtained from the same default network from which the `nodeapps` VIP is obtained. **Note:** Use this parameter when you add an Oracle AI Database listener.
`-oraclehome` *oracle_home*	Specify an Oracle home for the cluster database. If you do not include this parameter, then SRVCTL uses the Grid home by default. **Note:** Use this parameter when you add an Oracle AI Database listener.
`-user` *user_name*	Use this parameter to set the user who will run the listener to a less privileged user. Oracle recommends using this parameter to increase security. **Notes:** • You must be logged in as `root` to run this command and specify the `-user` parameter. • Use this parameter when you add an Oracle AI Database listener. • When you use the `-user` parameter, ensure the following: The listener log directory in the Oracle Base directory and the *Grid_home*`/network/admin/`*user_name* directory must both exist on each node before you can use this parameter. Additionally, *user_name* must have read, write, and execute permission in the directory. The *Oracle_Base*`/diag/tnslsnr/`*host_name*`/`*lower_case_listener_name* directory exists and *user_name* has read, write, and execute permission on it. • Before you can use LSNRCTL to manage a listener, you must set `TNS_ADMIN` to *Grid_home*`/network/admin/`*user_name*.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-38 (Cont.) srvctl add listener Command Parameters

Parameter	Description
`-endpoints "[TCP:]port_list[:FIREWALL={ON|OFF}][/IPC:key] [/NMP:pipe_name][/{TCPS|SDP|EXADIRECT}port_list[:FIREWALL={ON|OFF}]]"`	Protocol specifications for the listener. Use *port_list* to specify a comma-delimited list of TCP ports or listener endpoints. If you do not specify the `-endpoints` parameter for an Oracle AI Database listener, then SRVCTL searches for a free port between 1521 and 1540. You can also specify endpoints for TCPS, SDP, and EXADIRECT ports. **Note:** You can modify this attribute using Online Resource Attribute Modification.
`-group group_name`	Optionally, you can use the `-group` parameter with `-endpoints` to specify a group for the secure endpoint. This parameter is used for the EXADIRECT protocol on Exadata and Exalogic systems.
`-skip`	Indicates you want to skip the checking of ports.
`-asmlistener`	Specifies the listener type as an Oracle ASM listener. If you do not specify the `-listener` parameter, then the name of the Oracle ASM listener defaults to `LISTENER_ASM`. **Note:** You can only use this parameter with Oracle Clusterware.
`-subnet subnet`	Specifies the subnet to use for an Oracle ASM listener. **Note:** You can only use this parameter with Oracle Clusterware.

Usage Notes

You must run this command as `root` user on Linux and UNIX platforms when you specify the `-user` parameter.

Example

The following command adds a listener named `listener112` that is listening on ports 1341, 1342, and 1345 and runs from the Oracle home directory on every node in the cluster.

```
$ srvctl add listener -listener listener112 -endpoints "1341,1342,1345"
-oraclehome /u01/app/oracle/product/23.4.0/db1
```

When a listener is configured in the Oracle RAC home instead of the Grid home, then the `listener.ora` file is created under the location returned by the `$ORACLE_HOME/bin/orabasehome` utility, in the subdirectory `network/admin`, for example, `/u02/racbase/homes/OraDB23Home1/network/admin`.

srvctl config listener

Displays configuration information of a specific listener that is registered with Oracle Clusterware.

Syntax

```
srvctl config listener [-listener listener_name | -asmlistener] [-all]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

Table A-39 srvctl config listener Command Parameters

Parameter	Description
`-listener` *`listener_name`* `| -asmlistener`	The name of a specific listener name or the type of listener (Oracle ASM). If you do not specify this parameter, then SRVCTL displays the configuration for the default database listener.
`-all`	Print detailed configuration information.

Example

This command returns output similar to the following:

```
Name: LISTENER
Subnet: 10.100.200.195
Type: type
Owner: scott
Home: Grid_home
End points: TCP:1521
```

srvctl disable listener

Disables a listener resource.

Syntax

```
srvctl disable listener [-listener listener_name] [-node node_name]
```

Parameters

Table A-40 srvctl disable listener Command Parameters

Parameter	Description
`-listener` *`listener_name`*	Optionally, you can specify the name of a particular listener resource. If you do not specify this parameter, then the name of the listener defaults to `LISTENER`.
`-node` *`node_name`*	Optionally, you can specify the name of a cluster node on which the listener resource you want to disable is running. **Note:** This parameter is only available with Oracle Clusterware.

Example

The following example disables a listener resource named `listener_crm` on the node `node5`:

```
$ srvctl disable listener -listener listener_crm -node node5
```

srvctl enable listener

Enables a listener resource.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Syntax

```
srvctl enable listener [-listener listener_name] [-node node_name]
```

Parameters

Table A-41 srvctl enable listener Command Parameters

Parameter	Description
`-listener listener_name`	Optionally, you can specify the name of a listener resource. If you do not use this parameter, then the name of the listener defaults to `LISTENER`.
`-node node_name`	Optionally, you can specify the name of a cluster node on which to enable the listener. **Note:** You can only use this parameter with Oracle Clusterware.

Examples

The following example enables the listener named `listener_crm` on the node named `node5`:

```
$ srvctl enable listener -listener listener_crm -node node5
```

srvctl getenv listener

Displays the environment variables for the specified listener.

Syntax

```
srvctl getenv listener [-listener listener_name] [-envs "name_list"]
```

Parameters

Table A-42 srvctl getenv listener Command Parameters

Parameter	Description
`-listener listener_name`	Optionally, you can specify a listener name for which you want to obtain the environment variables. If you do not use this parameter, then the name of the listener defaults to `LISTENER`.
`-envs "name_list"`	Optionally, you can specify a comma-delimited list of the names of environment variables enclosed in double quotation marks ("""). If you do not use this parameter, then SRVCTL displays the values of all environment variables associated with the listener.

Example

The following example lists all environment variables specified for the default listener:

```
$ srvctl getenv listener
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl modify listener

Changes several aspects of the listener
Changes the Oracle home directory from which the listener runs, the name of the operating system user who owns Oracle home directory from which the listener runs, the listener endpoints, or the public subnet on which the listener listens, either for the default listener, or a specific listener, that is registered with Oracle Restart or with Oracle Clusterware.

If you want to change the name of a listener, then use the srvctl remove listener and srvctl add listener commands.

Syntax

```
srvctl modify listener [-listener listener_name] [-oraclehome oracle_home]
 [-endpoints "[TCP:]port_list[:FIREWALL={ON|OFF}][/IPC:key][/NMP:pipe_name]
 [/{TCPS|SDP|EXADIRECT}port_list[:FIREWALL={ON|OFF}]]"] [-group <group>]
 [-user user_name] [-netnum network_number]
```

Parameters

Table A-43 srvctl modify listener Command Parameters

Parameter	Description
`-listener listener_name`	Optionally, you can enter the name of the listener you want to modify. If you do not use this parameter, then SRVCTL uses the default name, `LISTENER`.
`-oraclehome oracle_home`	If you choose to use this parameter, then SRVCTL moves the listener to run from the Oracle home you specify. **Note:** When you use this parameter, run the command as a privileged user to enable SRVCTL to update resource ownership corresponding to the new `ORACLE_HOME` owner.
`-endpoints "[TCP:]port_list[:FIREWALL={ON|OFF}][/IPC:key][/NMP:pipe_name][/{TCPS|SDP|EXADIRECT}port_list[:FIREWALL={ON|OFF}]]"`	Optionally, you can use this parameter to modify protocol specifications for the listener. You must enclose the string of protocols in double quotation marks (`""`). `port_list` is comma-delimited list of port numbers. You can also modify endpoints for TCPS, SDP, and EXADIRECT ports. **Note:** You can modify this attribute using Online Resource Attribute Modification.
`-group group_name`	Optionally, you can use the `-group` parameter with `-endpoints` to specify a group for the secure endpoint. This parameter is used for the EXADIRECT protocol on Exadata and Exalogic systems.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-43 (Cont.) srvctl modify listener Command Parameters

Parameter	Description
`-user` *`user_name`*	Optionally, you can specify the name of the operating system user who will own the specified Oracle listener **Notes:** • You can only use this parameter with Oracle Clusterware. • You must be logged in as `root` to run this command and specify the `-user` parameter. • When you use the `-user` parameter, ensure the following: The listener log directory in `ORACLE_BASE` and the *`Grid_home`*`/network/admin/`*`user_name`* directory must both exist on each node before you can use this parameter. Additionally, *`user_name`* must have read, write, and execute permission in the directory. The `$ORACLE_BASE/diag/tnslsnr/`*`host_name`*`/`*`lower_case_listener_name`* directory exists and *`user_name`* has read, write, and execute permission on it. • Before you can use LSNRCTL to manage a listener, you must set `TNS_ADMIN` to *`Grid_home`*`/network/admin/`*`user_name`*.
`-netnum` *`network_number`*	Optionally, you can use this parameter to change the public subnet on which the listener listens. **Note:** Oracle recommends that you always have at least one listener on the default network. Do not use this parameter to change the network of the only listener that listens on the default network.

Example

The following example changes the TCP ports for the default listener:

```
$ srvctl modify listener -endpoints "TCP:1521,1522"
```

srvctl predict listener

Predicts the consequences of a listener failure.

Syntax

```
srvctl predict listener listener_name [-verbose]
```

Usage Notes

Specify the name of the listener for which you want to predict the consequences of a failure. Optionally, you can use the `–verbose parameter` for detailed output.

srvctl remove listener

Removes the configuration of a specific listener, or all listeners, from Oracle Clusterware or Oracle Restart.

Syntax

```
srvctl remove listener [-listener listener_name | -all] [-force]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Usage Notes

- Optionally, you can specify the name of a listener that you want to remove or use the `-all` parameter to remove all listeners. If you do not specify a listener name, then the listener name defaults to `LISTENER` for a database listener or `LISTENER_ASM` for an Oracle ASM listener.
- Optionally, you can use the `-force` parameter to skip checking whether there are other resources that depend on this listener, such as databases, and remove the listener anyway.

Example

The following example removes the configuration for the listener named `lsnr01`:

```
$ srvctl remove listener -listener lsnr01
```

srvctl setenv listener

Administers listener environment configurations.

Syntax

Use this command with one of the following syntax models:

```
srvctl setenv listener [-listener listener_name] -envs "name=val[,...]"

srvctl setenv listener [-listener listener_name] -env "name=val"
```

Parameters

Table A-44 srvctl setenv listener Command Parameters

Parameter	Description
`-listener` *`listener_name`*	Optionally, you can specify the name of a listener. If you do not use this parameter, then the listener name defaults to `LISTENER`.
`-envs` "*`name=val`*`[,...]`"	Specify a comma-delimited list of name-value pairs of environment variables enclosed in double quotation marks ("").
`-env "`*`name=val`*`"`	Use this parameter to enable single environment variable to be set to a value that contains commas or other special characters enclosed in double quotation marks ("").

Examples

The following example sets the language environment configuration for the default listener:

```
$ srvctl setenv listener -env "LANG=en"
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl start listener

Starts the default listener on specific node, or starts the specified listener on all nodes that are registered with Oracle Clusterware or on the given node.

Syntax

```
srvctl start listener [-node node_name] [-listener listener_name]
```

Parameters

Table A-45 srvctl start listener Command Parameters

Parameter	Description
`-node node_name`	Specify a particular node name to start the listener on that node. **Note:** You can only use this parameter with Oracle Clusterware.
`-listener listener_name`	Specify a particular listener name. Use the srvctl config listener command to obtain the name of a listener. If you do not assign a value to this parameter, then SRVCTL starts all known listeners in the cluster.

Examples

The following command starts all listeners managed by Oracle Clusterware on the node named `server3`.

```
$ srvctl start listener -node server3
```

srvctl status listener

Displays the status of listener resources.

Syntax

```
srvctl status listener [-listener listener_name] [-node node_name] [-verbose]
```

Parameters

Table A-46 srvctl status listener Command Parameters

Parameter	Description
`-listener listener_name`	Optionally, you can specify the name of a listener. If you do not use this parameter, then the listener name defaults to `LISTENER`.
`-node node_name`	Optionally, you can specify the name of a cluster node. **Note:** You can only use this parameter with Oracle Clusterware.
`–verbose`	Optionally, you can use this parameter to display verbose output.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Examples

The following example displays the status of the default listener on the node `node2`:

```
$ srvctl status listener -listener listener -node node2
Listener LISTENER is running on node(s): node2
```

srvctl stop listener

Stops the default listener or a specific listener on all nodes or the specified node.
You can also use this command to stop a listener on a non-cluster database from the non-cluster database home. However, SRVCTL does not accept the `-node` parameter when run from a non-cluster database home.

Syntax

```
srvctl stop listener [-listener listener_name] [-node node_name] [-force]
```

Parameters

Table A-47 srvctl stop listener Command Parameters

Parameter	Description
`-listener` *`listener_name`*	Specify the name of the listener you want to stop. If you do not assign a value to this parameter, then SRVCTL stops all known listeners in the cluster.
`-node` *`node_name`*	Optionally, you can specify the name of a single node on which a particular listener runs. **Note:** You can only use this parameter with Oracle Clusterware.
`–force`	Forcibly stop the listener.

Examples

The following command stops all listeners on the node `mynode1`:

```
$ srvctl stop listener -node mynode1
```

srvctl unsetenv listener

Unsets the environment configuration for a listener.

Syntax

```
srvctl unsetenv listener [-listener listener_name] -envs "name_list"
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

Table A-48 srvctl unsetenv listener Command Parameters

Parameter	Description
`-listener` *`listener_name`*	Optionally, you can specify the name of a listener for which you want to unset the environment configuration. If you do not use this parameter, then the name of the listener defaults to `LISTENER`.
`-envs "`*`name_list`*`"`	Specify a comma-delimited list of environment variable names enclosed in double quotation marks ("") that you want to unset.

Examples

The following example unsets the environment variable `TNS_ADMIN` for the default listener:

```
$ srvctl unsetenv listener -envs "TNS_ADMIN"
```

srvctl update listener

Updates the listener to listen on the new endpoints.

Syntax

```
srvctl update listener [-listener listener_name -asm -remove [-force]]
```

Parameters

Table A-49 srvctl update listener Command Parameters

Parameter	Description
`-listener` *`listener_name`*	Optionally, you can specify the name of a listener that you want to update. If you do not use this parameter, then the name of the listener defaults to `LISTENER`.
`-asm`	Oracle ASM listener type.
`-remove`	Remove Oracle ASM listener.
`-force`	Forcefully remove Oracle ASM listener.

Usage Notes

You can only use this command with Oracle Clusterware.

network Commands

Use commands with the `network` keyword to add, modify, list the configuration of, and remove a non-default Network.

- srvctl add network
  Adds a static or dynamic network.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- srvctl config network
  Displays the network configuration for the cluster.
- srvctl modify network
- srvctl predict network
- srvctl remove network

srvctl add network

Adds a static or dynamic network.

If your server connects to more than one network, then you can use this command to configure an additional network interface for Oracle RAC, allowing you to create VIPs on multiple public networks.

Syntax

```
srvctl add network [-netnum net_number] -subnet subnet/netmask[/if1[|if2...]]
   [-nettype {STATIC | DHCP | AUTOCONFIG | MIXED}] [-pingtarget
"ping_target_list"]
   [-skip] [-verbose]
```

Parameters

Table A-50 srvctl add network Command Parameters

Parameter	Description
`-netnum net_number`	The network number. The default is 1.
`-subnet subnet/netmask [/if1[|if2|...]]`	Defines a subnet. If you do not specify any interface names, then the network uses any interface on the given subnet. For IPv6, *netmask* is a prefix length, such as 64.
`-nettype {STATIC|DHCP|AUTOCONFIG|MIXED}`	Specify the network type: `STATIC`, `DHCP`, `AUTOCONFIG`, or `MIXED`. If you specify `STATIC` for the network type, then you must provide the virtual IP address using the srvctl add vip command. If you specify `DHCP` for the network type, then the VIP agent obtains the IP address from a DHCP server. If you specify `AUTOCONFIG` for the network type, then the VIP agent generates a stateless IPv6 address for the network. You can only use `AUTOCONFIG` for IPv6 networks. If the subnet/netmask specification is not for an IPv6 address, then SRVCTL returns an error. If you specify `MIXED` for the network type, then the VIP resource uses both a static IP address and an IP address obtained dynamically, either from a DHCP server for IPv4 or using stateless auto-configuration for IPv6.
`-pingtarget "ping_target_list"`	A comma-delimited list of IP addresses or host names to ping.
`-skip`	Use this parameter to skip the checking of subnet.
`-verbose`	Verbose output.

Usage Notes

- On Linux and UNIX systems, you must be logged in as the `root` user and on Windows, you must be logged in as a user with Administrator privileges to run this command.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- This command is only available with Oracle Clusterware.
- Oracle only supports DHCP-assigned networks for the default network, not for subsequent networks.
- You can also use the `LISTENER_NETWORKS` database initialization parameter to control client redirects to the appropriate network.

Example

An example of this command is:

```
# srvctl add network -netnum 3 -subnet 192.168.3.0/255.255.255.0
```

srvctl config network

Displays the network configuration for the cluster.

Syntax

```
srvctl config network [-netnum network_number]
```

Usage Notes

- Specify the network for which you want to display configuration information.
- This command is only available with Oracle Clusterware.

Example

An example of this command is:

```
$ srvctl config network -netnum 2
```

srvctl modify network

Modifies the subnet, network type, or IP address type for a network.

Syntax

```
srvctl modify network [-netnum network_number] [-subnet subnet/netmask
  [/if1[|if2|...]]] [-nettype network_type | -iptype {ipv4 | ipv6 | both}]
  [-pingtarget "ping_target_list"] [-verbose]
```

Parameters

Table A-51 srvctl modify network Command Parameters

Parameter	Description
`-netnum network_number`	Optionally, you can specify a network number that you want to modify. The default is 1.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-51 (Cont.) srvctl modify network Command Parameters

Parameter	Description
`-subnet` *`subnet/netmask`* `[/`*`if1`*`[|`*`if2`*`|...]]`	Optionally, you can specify a subnet number for the public network. The netmask and interfaces you specify, if any, change those of the network you are modifying. If you specify an IPv6 subnet, then enter a prefix length, such as 64, in place of *`netmask`*. If you do not specify any interface names, then the VIPs use any interface on the given subnet. If you are changing the network type using the `-nettype` parameter, then you must specify either an existing IPv4 or IPv6 network using the `-subnet` parameter. Additionally, the subnet and netmask you specify in the `-subnet` parameter do not change those of the network you are modifying.
`-nettype` *`network_type`*	Optionally, you can modify the network type using this parameter, to `static`, `dhcp`, `autoconfig`, or `mixed`.
`-iptype {ipv4 | ipv6 | both}`	Alternative to modifying the network type, you can modify the type of IP address to `ipv4`, `ipv6`, or `both`.
`-pingtarget "`*`ping_target_list`*`"`	Optionally, you can specify a comma-delimited list of IP addresses or host names to ping enclosed in double quotation marks ("").
`-verbose`	Optionally, you can use this parameter to display detailed output.

Usage Notes

- You can only use this command with Oracle Clusterware.
- On Linux and UNIX systems, you must be logged in as `root` and on Windows, you must be logged in as a user with Administrator privileges to run this command.
- You can modify the IP address type for a network from IPv4 to IPv6, or from IPv6 to IPv4.
- If you specify `static` for the network type, then you must provide the virtual IP address using the srvctl add vip command.
- If you specify `dhcp` for the network type, then the VIP agent obtains the IP address from a DHCP server.
- If you specify `autoconfig` for the network type, then the VIP agent generates a stateless IPv6 address for the network. You can only use this parameter for IPv6 networks. If the subnet/netmask specification is not for an IPv6 address, then SRVCTL returns an error.
- If you change a network from `static` to `mixed`, then you must first configure GNS, so that the dynamic addresses obtained can have names registered for them.
- If you specify `mixed` for the network type, then the VIP resource uses both a static IP address and an IP address obtained dynamically, either DHCP or autoconfig.
- If you specify `mixed_autoconfig` for the network type, then the VIP resource retains the static IP configuration and either obtains an IP address from a DHCP server for an IPv4 network specification or generates a stateless auto-configured IP address for an IPv6 network specification.

Examples

The following example changes the subnet number, netmask, and interface list:

```
# srvctl modify network -subnet 192.168.2.0/255.255.255.0/eth0
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

The following example changes the second network to DHCP:

```
# srvctl modify network -netnum 2 -nettype dhcp
```

The following example adds an IPv6 subnet and netmask to the default network:

```
# srvctl modify network -subnet 2606:b400:400:18c0::/64
```

The following example removes the IPv4 configuration from a network:

```
# srvctl modify network -iptype ipv6
```

Related Topics

- *Oracle Clusterware Administration and Deployment Guide*

srvctl predict network

Predicts the consequences of network failure.

Syntax

```
srvctl predict network [-netnum network_number] [-verbose]
```

Usage Notes

Optionally, you can specify a network for which you want to evaluate a failure. The default value is 1. You can also use the `–verbose` parameter to print detailed output.

Example

The following example predicts the consequences of a failure on network number 2:

```
$ srvctl predict network -netnum 2
```

srvctl remove network

Removes the network configuration.

Syntax

```
srvctl remove network {-netnum network_number | -all} [-force] [-verbose]
```

Parameters

Table A-52 srvctl remove network Command Parameters

Parameter	Description
`-netnum network_number | -all`	Specify which network number you want to remove. Alternatively, you can use the `–all` parameter to indicate that you want to remove all networks.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-52 (Cont.) srvctl remove network Command Parameters

Parameter	Description
`-force`	Optionally, you can use this parameter to remove the specified network regardless of any dependencies.
`-verbose`	Optionally, you can use this parameter to display detailed output.

Usage Notes

- You can only use the command with Oracle Clusterware.
- You must have full administrative privileges to run this command. On Linux and UNIX systems, you must be logged in as `root` and on Windows systems, you must be logged in as a user with Administrator privileges.

Example

The following example removes a network:

```
# srvctl remove network -netnum 3
```

nodeapps Commands

Use commands with the `nodeapps` keyword to add, modify, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, and remove node applications.

- srvctl add nodeapps
  Adds a node application configuration to the specified node.
- srvctl config nodeapps
  Displays the VIP configuration for each node in the cluster.
- srvctl disable nodeapps
  Disables node applications on all nodes in the cluster.
- srvctl enable nodeapps
  Enables the node applications on all nodes in the cluster.
- srvctl getenv nodeapps
  Displays the environment variables for the node application configurations.
- srvctl modify nodeapps
  Modifies the configuration for a node application.
- srvctl remove nodeapps
  Removes the node application configuration.
- srvctl setenv nodeapps
  Sets the environment variables for the node application configurations.
- srvctl start nodeapps
  Starts node-level applications on a node or all nodes in the cluster.
- srvctl status nodeapps
  Displays the status of node applications.
- srvctl stop nodeapps
  Stops node-level applications on a node in the cluster.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- srvctl unsetenv nodeapps
  Unsets the environment configuration for the node applications.

srvctl add nodeapps

Adds a node application configuration to the specified node.

Syntax

Use this command with one the following syntax models, specifying either a specific node and VIP or a specific subnet and netmask:

```
srvctl add nodeapps {-node node_name -address {vip_name |
    ip_address}/netmask[/if1[|if2|..]] [-skip]}
    [-emport em_port] [-onslocalport ons_local_port] [-onsremoteport
ons_remote_port]
    [-remoteservers hostname[:port][,hostname[:port]]
    [-clientdata client_data_file [-scanclient]] [-pingtarget
"pingtarget_list"] [-vipless] [-verbose]


srvctl add nodeapps -subnet subnet/netmask[/if1[|if2|...]] [-emport em_port]
    [-onslocalport ons_local_port] [-onsremoteport ons_remote_port]
    [-remoteservers hostname[:port][,hostname[:port]]
    [-clientdata client_data_file [-scanclient]] [-pingtarget
"pingtarget_list"] [-vipless] [-verbose]
```

Parameters

Table A-53 srvctl add nodeapps Command Parameters

Parameter	Description
`-node node_name`	The name of the node on which you want to create the node application. Node name is optional and unnecessary if you run the command on the local node.
`-address {vip_name | ip_address}/netmask[/if1[|if2|..]]}`	This specification creates a traditional VIP node application on the specified node. **Note:** You must use this parameter for upgrade configurations and new, non-DHCP configurations.
`-skip`	Specify this parameter to skip checking the reachability of the VIP address.
`-subnet subnet/netmask [/if1[|if2 |...]]`	Creates a DHCP subnet. If you do not specify any interface names, then the VIPs use any interface on the given subnet.
`-emport em_port`	Local port on which Oracle Enterprise Manager listens. The default port is 2016.
`-onslocalport ons_local_port`	The Oracle Notification Service daemon listener port on its node. If you do not specify this value, the Oracle Notification Service daemon listener port defaults to 6100. **Note:** The local port and remote port must each be unique.
`-onsremoteport ons_remote_port`	The port number for remote Oracle Notification Service daemon connections. If you do not specify a port number, the default value of 6200 is used for the Oracle Notification Service remote port. **Note:** The local port and remote port must each be unique.
`-remoteservers hostname[:port]`	A list of `hostname[:port]` pairs for Oracle Notification Service daemons on servers that are not in the cluster.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-53 (Cont.) srvctl add nodeapps Command Parameters

Parameter	Description
`-clientdata` *client_data_file*	Optionally, you can specify the file with a wallet to import, or an empty string to delete a wallet used for SSL to secure Oracle Notification Service communication. If you want to use a custom certificate, then add your custom certificate to an auto-login Oracle wallet using the `orapki` commands, and provide the complete certificate path in the following format: */wallet_directory/Oracle_wallet_name/custom_certificate_name*.sso
`-scanclient`	Optionally, you can specify SCAN client cluster name.
`-pingtarget` "*pingtarget_list*"	Optionally, you can specify a comma-delimited list of IPs or host names enclosed in double quotation marks ("") to ping.
`-vipless`	Specify this option to add a network without node VIP addresses.
`-verbose`	Verbose output.

Usage Notes

- On Linux and UNIX systems, you must be logged in as `root` and on Windows, you must be logged in as a user with Administrator privileges to run this command.
- This command is only available with Oracle Clusterware.

Example

An example of this command is:

```
# srvctl add nodeapps -node crmnode1 -address 1.2.3.4/255.255.255.0
```

Related Topics

- *Oracle AI Database Security Guide*

srvctl config nodeapps

Displays the VIP configuration for each node in the cluster.

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

Syntax

```
srvctl config nodeapps [-viponly] [-onsonly]
```

Usage Notes

Use `-viponly` to display the VIP address configuration. Use `-onsonly` to display the Oracle Notification Service configuration.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example

An example of this command is:

```
$ srvctl config nodeapps -viponly -onsonly
```

srvctl disable nodeapps

Disables node applications on all nodes in the cluster.

Syntax

```
srvctl disable nodeapps [-onsonly] [-adminhelper] [-verbose]
```

Parameters

Table A-54 srvctl disable nodeapps Command Parameters

Parameter	Description
`-onsonly`	Optionally, you can use this parameter to disable only the Oracle Notification Service (ONS).
`-adminhelper`	Optionally, you can use this parameter to disable the Administrator helper only.
`–verbose`	Optionally, you can use this parameter to display detailed output.

Usage Notes

You can only use this parameter with Oracle Clusterware.

Example

The following example disables ONS:

```
$ srvctl disable nodeapps -onsonly -verbose
```

srvctl enable nodeapps

Enables the node applications on all nodes in the cluster.

Syntax

```
srvctl enable nodeapps [-onsonly] [-adminhelper] [-verbose]
```

Parameters

Table A-55 srvctl enable nodeapps Command Parameters

Parameter	Description
`-onsonly`	Optionally, you can use this parameter to disable only the Oracle Notification Service (ONS).

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-55 (Cont.) srvctl enable nodeapps Command Parameters

Parameter	Description
`-adminhelper`	Optionally, you can use this parameter to enable the Administrator helper only.
`–verbose`	Optionally, you can use this parameter to display detailed output.

Usage Notes

You can only use this command with Oracle Clusterware.

Example

The following example enables ONS:

```
$ srvctl enable nodeapps -onsonly -verbose
```

srvctl getenv nodeapps

Displays the environment variables for the node application configurations.

Syntax

```
srvctl getenv nodeapps [-viponly] [-onsonly] [-envs "name_list"]
```

Parameters

Table A-56 srvctl getenv nodeapps Command Parameters

Parameter	Description
`-viponly`	Optionally, you can use this parameter to display the VIP address configuration.
`-onsonly`	Optionally, you can use this parameter to isplay the Oracle Notification Service configuration.
`-envs "name_list"`	Optionally, you can specify a comma-delimited list of the names of environment variables enclosed in double quotation marks (""). If you do not use this parameter, then SRVCTL displays the values of all environment variables associated with the node applications.

Usage Notes

You can only use this command with Oracle Clusterware.

Example

The following example lists all environment variables for the node applications:

```
$ srvctl getenv nodeapps -viponly
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl modify nodeapps

Modifies the configuration for a node application.

Syntax

Use this command with one of the following syntax models, specifying either a specific node and VIP or a specific subnet and netmask:

```
srvctl modify nodeapps {[-node node_name -address {vip_name|vip_address}/
  netmask[/if1[|if2|...]] [-skip]} [-nettype network_type] [-emport em_port]
    [-onslocalport ons_local_port] [-onsremoteport ons_remote_port]
    [-remoteservers host:[port][,...]] [-verbose]
    [-clientdata file] [-pingtarget "ping_target_list"]


srvctl modify nodeapps [-subnet subnet/netmask[/if1[|if2|...]]]
    [-nettype network_type] [-emport em_port]
    [-onslocalport ons_local_port] [-onsremoteport ons_remote_port]
    [-remoteservers host:[port][,host:port,...]] [-verbose]
    [-clientdata file] [-pingtarget "ping_target_list"]
```

Parameters

Table A-57 srvctl modify nodeapps Command Parameters

Parameter	Description
`-node node_name`	Specify the name of the node on which the node application you want to modify resides.
`-address {vip_name|vip_address}/netmask[/if1[|if2|...]]`	Specify a node-level virtual IP name or address. The address specified by name or IP must match the subnet number of the default network. **Note:** You must use this parameter for upgrade configurations and new non-DHCP configurations
`–skip`	Optionally, you can use this parameter to skip checking the reachability of the VIP address.
`-subnet subnet/netmask[/if1[|if2|...]]`	Alternative to specifying a node name and address, you can specify a subnet number for the public network. The netmask and interfaces you specify, if any, change those of the default network. Additionally, if you specify a value for the *netmask* option, then you need only specify it for the first node on each network.
`-nettype network_type`	Optionally, you can change the network server type to `static`, `dhcp`, or `mixed`.
`-emport em_port`	Optionally, you can change the local port on which Oracle Enterprise Manager listens. **Note:** You can also modify this attribute using Online Resource Attribute Modification.
`-onslocalport ons_local_port`	Optionally, you can change the port on which the Oracle Notification Service daemon listens for local client connections. **Notes:** • The local port and remote port must each be unique. • You can modify the local port while the resource remains online, without restarting the resource.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-57 (Cont.) srvctl modify nodeapps Command Parameters

Parameter	Description
`-onsremoteport` *`ons_remote_port`*	Optionally, you can change the port on which the Oracle Notification Service daemon listens for connections from remote hosts. **Notes:** • The local port and remote port must each be unique. • You can modify the remote port while the resource remains online, without restarting the resource.
`-remoteservers` *`host`*`:[`*`port`*`][,...]`	Optionally, you can modify the comma-delimited list of *`host`*`:[`*`port`* pairs of remote hosts that are part of the Oracle Notification Service network but are not part of the cluster. If you do not specify a port for a remote host, then the utility uses the value you specified for *`ons_remote_port`*.
`-clientdata` *`file`*	Optionally, you can specify the file with a wallet to import, or an empty string to delete a wallet used for SSL to secure Oracle Notification Service communication. **Note** You must manually restart the ONS server after updating the ONS security certificate using this parameter.
`-pingtarget` *`"ping_target_list"`*	Optionally, you can specify a comma-delimited list of IPs or host names enclosed in double quotation marks (`""`) to ping.
`-verbose`	Optionally, you can use this parameter to display detailed output.

Usage Notes

You can only use this command with Oracle Clusterware.

Example

The following example changes the nodeapps resource on `mynode1` to use the application VIP of 100.200.300.40 with a subnet mask of 255.255.255.0 on the network interface `eth0`:

```
$ srvctl modify nodeapps -node mynode1 -addr 100.200.300.40/255.255.255.0/eth0
```

Note

You must manually restart the ONS server after updating the ONS security certificate using the `-clientdata` parameter.

srvctl remove nodeapps

Removes the node application configuration.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Syntax

```
srvctl remove nodeapps [-force] [-noprompt] [-verbose]
```

Parameters

Table A-58 srvctl remove nodeapps Command Parameters

Parameter	Description
`-force`	Optionally, you can use this parameter to forcibly remove node application configurations, regardless of any dependencies.
`-noprompt`	Optionally, you can use this parameter to suppress prompts.
`-verbose`	Optionally, you can use this parameter to display detailed output.

Usage Notes

- You can only use this command with Oracle Clusterware.
- You must have full administrative privileges to run this command. On Linux and UNIX systems, you must be logged in as `root` and on Windows systems, you must be logged in as a user with Administrator privileges.

srvctl setenv nodeapps

Sets the environment variables for the node application configurations.

Syntax

```
srvctl setenv nodeapps {-envs "name=val[,...]" | -env "name=val"}
   [-viponly] [-onsonly] [-verbose]
```

Parameters

Table A-59 srvctl setenv nodeapps Command Parameters

Parameter	Description
`-envs "name=val[,...]"`	Use this parameter to specify a comma-delimited list of name-value pairs of environment variables enclosed in double quotation marks ("").
`-env "name=val"`	Alternatively, you can use this parameter to enable a single environment variable that is set to a value which contains commas or other special characters, enclosed in double quotation marks ("").
`-viponly`	Optionally, you can use this parameter to modify only the VIP configuration.
`-onsonly`	Optionally, you can use this parameter to modify only the ONS daemon configuration.
`-verbose`	Optionally, you can use this parameter to display detailed output.

Usage Notes

You can only use this command with Oracle Clusterware.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example

The following example sets the `CLASSPATH` environment variable for all node applications:

```
$ srvctl setenv nodeapps -env "CLASSPATH=/usr/local/jdk/jre/rt.jar" -verbose
```

srvctl start nodeapps

Starts node-level applications on a node or all nodes in the cluster.

Syntax

```
srvctl start nodeapps [-node node_name] [-onsonly] [-adminhelper] [-verbose]
```

Parameters

Table A-60 srvctl start nodeapps Command Parameters

Parameter	Description
`-node` *node_name*	Optionally, you can specify a node on which to start node-level applications. If you do not use this parameter, then SRVCTL starts the node applications on all active nodes in the cluster.
`-onsonly`	Optionally, you can use this parameter to disable only the Oracle Notification Service (ONS).
`-adminhelper`	Optionally, you can use this parameter to start only an Administrator helper instead of all node applications.
`-verbose`	Optionally, you can use this parameter to display detailed output.

Usage Notes

You can only use this command with Oracle Clusterware.

srvctl status nodeapps

Displays the status of node applications.

Syntax

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

Usage Notes

- You can only use this command with Oracle Clusterware.
- Optionally, you can specify a node for which to display the status of the node applications.

srvctl stop nodeapps

Stops node-level applications on a node in the cluster.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Syntax

```
srvctl stop nodeapps [-node node_name] [-onsonly] [-adminhelper] [-force]
   [-relocate] [-verbose]
```

Parameters

Table A-61 srvctl stop nodeapps Command Parameters

Parameter	Description
`-node node_name`	Optionally, you can use this parameter to specify a node on which you want to stop node applications. If you do not use this parameter, then the utility stops the node applications on all active nodes in the cluster.
`-onsonly`	Optionally, you can use this parameter to disable only the Oracle Notification Service (ONS).
`-adminhelper`	Optionally, you can use this parameter to stop only the Administrator helper instead of all node applications.
`-force`	Optionally, you can use this parameter to stop node applications regardless of any dependencies.
`-relocate`	Optionally, you can use this parameter to relocate the VIP and possibly-dependent services. **Note:** If you use this parameter, then you must also specify the `-node node_name` parameter.
`-verbose`	Optionally, you can use this parameter to display detailed output.

Usage Notes

You can only use this command with Oracle Clusterware.

srvctl unsetenv nodeapps

Unsets the environment configuration for the node applications.

Syntax

```
srvctl unsetenv nodeapps -envs "name_list" [-viponly] [-onsonly]
   [-verbose]
```

Parameters

Table A-62 srvctl unsetenv nodeapps Command Parameters

Parameter	Description
`-envs "name_list"`	Specify a comma-delimited list of the names of environment variables enclosed in double quotation marks ("") that you want to unset.
`-viponly`	Optionally, you can use this parameter to unset only the VIP configuration.
`-onsonly`	Optionally, you can use this parameter to unset only the ONS daemon configuration.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-62 (Cont.) srvctl unsetenv nodeapps Command Parameters

Parameter	Description
`-verbose`	Optionally, you can use this parameter to display detailed output.

Example

The following example unsets the environment configuration for the specified node applications:

```
$ srvctl unsetenv nodeapps -envs "test_var1,test_var2"
```

ons Commands

Use commands with the `ons` keyword to manage only Oracle Notification Service instances for Oracle Restart.

You can add, configure, enable, start, obtain the status of, stop, disable, and remove Oracle Notification Service instances for Oracle Restart.

- srvctl add ons
  Adds an Oracle Notification Service daemon to an Oracle Restart configuration.
- srvctl config ons
  Displays configuration information for the Oracle Notification Service daemon.
- srvctl disable ons
  Disables the Oracle Notification Service (ONS) daemon for Oracle Restart installations.
- srvctl enable ons
- srvctl export ons
  Exports ONS server information into a file.
- srvctl modify ons
- srvctl remove ons
- srvctl start ons
- srvctl status ons
- srvctl stop ons

srvctl add ons

Adds an Oracle Notification Service daemon to an Oracle Restart configuration.

Syntax

```
srvctl add ons [-emport em_port] [-onslocalport ons_local_port] [-
onsremoteport ons_remote_port]
   [-remoteservers host[:port][,host[:port]...]]
   [-clientcluster cluster_name] [-clientdata filename]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

Table A-63 srvctl add ons Command Parameters

Parameter	Description
`-emport em_port`	Local listen port for Oracle Enterprise Manager. The default port number is 2016.
`-onslocalport ons_local_port`	Optionally, you can specify the Oracle Notification Service daemon listening port for local client connections. **Note:** The local port and remote port must each be unique.
`-onsremoteport ons_remote_port`	Optionally, you can specify the Oracle Notification Service daemon listening port for connections from remote hosts. **Note:** The local port and remote port must each be unique.
`-remoteservers host[:port] [host[:port]...]`	Optionally, you can specify a comma-delimited list of `host:port` pairs of remote hosts that are part of the Oracle Notification Service network but are not part of the Oracle Clusterware cluster. **Note:** If `port` is not specified for a remote host, then `ons_remote_port` is used.
`-clientcluster cluster_name`	The name of the cluster that is running the shared SCAN listener.
`-clientdata filename`	Specify the path to the file to which credentials data will be written.

Usage Notes

You can only use this command with Oracle Restart.

Example

An example of this command is:

```
$ srvctl add ons -onslocalprt 6200
```

srvctl config ons

Displays configuration information for the Oracle Notification Service daemon.

Syntax

```
srvctl config ons [-all] [-clientcluster cluster_name]
```

Usage Notes

- You can only use this command with Oracle Restart.
- You can display the configuration for all ONS daemons, or those for a specific client cluster.

Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl disable ons

Disables the Oracle Notification Service (ONS) daemon for Oracle Restart installations.

Syntax

```
srvctl disable ons [-clientcluster cluster_name] [-verbose]
```

Usage Notes

- You can only use this command with Oracle Restart.
- You can disable all ONS daemons, or those for a specific client cluster.
- Optionally, you can use the `-verbose` parameter to display detailed output.

srvctl enable ons

Enables the Oracle Notification Service daemon.

Syntax

```
srvctl enable ons [-clientcluster cluster_name] [-verbose]
```

Usage Notes

- You can only use this command with Oracle Restart.
- You can enable all ONS daemons, or those for a specific client cluster.
- Optionally, you can use the `-verbose` parameter to display detailed output.

srvctl export ons

Exports ONS server information into a file.

Syntax

```
srvctl export ons -clientcluster cluster_name -clientdata filename
```

Parameters

Table A-64 srvctl export ons Command Parameters

Parameter	Description
`-clientcluster cluster_name`	Specify the cluster name.
`-clientdata filename`	Specify the path to the file to which credentials data will be written.

srvctl modify ons

Modifies the ports used by the Oracle Notification Service daemon that is registered with Oracle Restart.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Syntax

```
srvctl modify ons [-emport em_port] [-onslocalprt ons_local_port] [-
onsremoteport ons_remote_port]
  [-remoteservers host[:port][,host[:port],...]]
  [-clientcluster cluster_name] [-verbose]
```

Parameters

Table A-65 srvctl modify ons Command Parameters

Parameter	Description
`-emport em_port`	Optionally, you can specify the local port on which Oracle Enterprise Manager listens. The default port is 2016.
`-onslocalprt ons_local_port`	Optionally, you can modify the Oracle Notification Service daemon listening port for local client connections. **Note:** The local port and remote port must each be unique.
`-onsremoteport ons_remote_port`	Optionally, you can modify the Oracle Notification Service daemon listening port for connections from remote hosts. **Note:** The local port and remote port must each be unique.
`-remoteservers host[:port] [,host[:port],...]`	Optionally, you can specify a list of `host:port` pairs of remote hosts that are part of the Oracle Notification Service network but are not part of the Oracle Clusterware cluster. **Note:** If you do not specify `port` for a remote host, then SRVCTL uses the value for `ons_remote_port`.
`-clientcluster cluster_name`	The name of the cluster that is running the shared SCAN listener.
`-verbose`	Optionally, you can use this parameter to display detailed output.

Usage Notes

You can only use this command with Oracle Restart.

Example

An example of this command is:

```
$ srvctl modify ons -onslocalprt 6203
```

srvctl remove ons

Removes Oracle Notification Service from the Oracle Grid Infrastructure home.

Syntax

```
srvctl remove ons [-clientcluster cluster_name] [-force] [-verbose]
```

Usage Notes

- You can only use this command with Oracle Restart.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- If using the shared SCAN feature, then use the `-clientcluster` parameter to specify the name of the cluster that is running the shared SCAN listener.
- Optionally, you can use the `–force` parameter to remove Oracle Notification Service regardless of dependencies.
- Optionally, you can use the `–verbose` parameter to display detailed output.

srvctl start ons

Starts the Oracle Notification Service daemon.

Syntax

```
srvctl start ons [-clientcluster cluster_name] [-verbose]
```

Usage Notes

- You can only use this command with Oracle Restart.
- You can enable all ONS daemons, or those for a specific client cluster.
- Optionally, you can use the `-verbose` parameter to display detailed output.

srvctl status ons

Displays the current state of the Oracle Notification Service daemon.

Syntax

```
srvctl status ons [-clientcluster cluster_name]
```

Usage Notes

- You can only use this command with Oracle Restart.
- You can display the status for all ONS daemons, or those for a specific client cluster.

srvctl stop ons

Stops the Oracle Notification Service daemon.

Syntax

```
srvctl stop ons [-clientcluster cluster_name] [-force]
```

Usage Notes

- You can only use this command with Oracle Restart.
- You can stop all ONS daemons, or those for a specific client cluster.
- Optionally, you can use the `–force` parameter to stop the ONS daemons regardless of any dependencies.

Copyright © 1999, 2025, Oracle and/or its affiliates.

pdb Commands

Use commands with the `pdb` keyword to manage the Pluggable Databases (PDBs) in your cluster database.

You can add, modify, remove, list the configuration of, enable, disable, start, stop, and obtain the status of PDBs.

- srvctl add pdb
  Adds a Pluggable Database (PDB) configuration to Oracle Clusterware.
- srvctl config pdb
  Displays the configuration information for a Pluggable Database (PDB).
- srvctl disable pdb
  Disables a running Pluggable Database (PDB) from Oracle Clusterware management.
- srvctl enable pdb
  Enables the Pluggable Database (PDB) for Oracle Clusterware management.
- srvctl modify pdb
  Modifies the configuration for a Pluggable Database (PDB).
- srvctl remove pdb
  Removes the Pluggable Database (PDB) configuration from Oracle Clusterware management.
- srvctl start pdb
  Starts a Pluggable Database (PDB) and the associated services based on the management policy of the service. All the services with the `AUTOMATIC` management policy will start automatically.
- srvctl status pdb
  This command displays the current state of the Pluggable Database (PDB).
- srvctl stop pdb
  Stops a Pluggable Database (PDB) and its services.

Related Topics

- Pluggable Database Rank
  The PDB `-rank` parameter defines relative importance of the PDBs, which are created specifying cardinality, in a database with the `RANK` management policy.
- Pluggable Database Placement
  Configure PDBs to either run explicitly in the specified CDB instances or run dynamically in any CDB or a subset of CDBs in the cluster.
- Starting and Stopping PDBs in Oracle RAC
  You can use SRVCTL commands to manage PDBs.

srvctl add pdb

Adds a Pluggable Database (PDB) configuration to Oracle Clusterware.

Syntax

```
srvctl add pdb -db db_unique_name -pdb pdb_name
     [-cardinality {num_of_instances | ALL}]
     [-maxcpu max_cpu_usage] [-mincpuunit min_cpu_usage]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

```
[-approot approot_database] [-startoption start_options]
[-stopoption stop_options] [-policy policy]
```

Parameters

Table A-66 srvctl add pdb Command Parameters

Parameter	Description
`-db db_unique_name`	The unique name of the Container Database (CDB).
`-pdb pdb_name`	The name of the PDB.
`-cardinality {num_of_instances | ALL}`	The number of instances of the PDB to be open at any time. If you specify ALL, then the PDB can open in any or all instances of the cluster database. **Note** If this parameter is not set, then the PDB can run on any node on which the cluster database can run. Where the PDB runs is determined by the preferred and available lists of the PDB services. If this parameter is set, then the PDB services must be `UNIFORM`, `SINGLETON`, or `DUPLEX`.
`-maxcpu max_cpu_usage`	The maximum CPU usage limit for the PDB.
`-mincpuunit min_cpu_usage`	The minimum CPU usage limit for the PDB.
`-approot approot_database`	The application root PDB.
`-startoption start_options`	Startup options for the PDB, such as `OPEN` or `OPEN READ ONLY`. The default value is an empty string, which means the PDB uses the same open mode as the CDB. **Note** For multi-word startup options, such as `read only` and `read write`, separate the words with a space and enclose in double quotation marks (`""`). For example, `"read only"`.
`-stoption stop_options`	Stop options for the PDB, such as `NORMAL`. The default stop option is `IMMEDIATE`.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-66 (Cont.) srvctl add pdb Command Parameters

Parameter	Description
`-policy policy`	Management policy for the Pluggable Database, where *policy* can be one of the following values: • `AUTOMATIC`: The PDBs are started automatically when the database is started with the `srvctl start database` command. With this management policy, a PDB always starts with its CDB starts, whether it was running before or not. **Note** When you add a PDB using SRVCTL, then the PDB's default management policy depends on the CDB's management policy. For example, if CDB's management policy is `AUTOMATIC`, then PDB's management policy is also set to `AUTOMATIC`. • `MANUAL`: The PDB is restarted when the PDB services require it. The PDB restart is driven by the previous running state and management policy of the PDB services. A `MANUAL` PDB with `MANUAL` services is restarted after the CRS restart. • `RESTART`: The PDB is restarted upon restart of the database host computer.

Usage Notes

If a PDB's management policy is `RESTART`, then the management policy for the CDB must be `RANK`.

Related Topics

- Pluggable Database Rank
  The PDB `-rank` parameter defines relative importance of the PDBs, which are created specifying cardinality, in a database with the `RANK` management policy.

srvctl config pdb

Displays the configuration information for a Pluggable Database (PDB).

Syntax

```
srvctl config pdb -db db_unique_name [-pdb pdb_name]
```

Parameters

Table A-67 srvctl config pdb Command Parameters

Parameter	Description
`-db db_unique_name`	Unique name for the Container Database (CDB).

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-67 (Cont.) srvctl config pdb Command Parameters

Parameter	Description
`-pdb pdb_name`	The name of the PDB.

Example

This examples shows the configuration information for the `crmeast` PDB.

```
srvctl config pdb -db crm -pdb crmeast

Pluggable database name: crmeast
Application Root PDB:
Maximum CPU count (whole CPUs): 0
Minimum CPU count unit (1/100 CPU count): 0
Start Option: open
Stop Option: immediate
```

srvctl disable pdb

Disables a running Pluggable Database (PDB) from Oracle Clusterware management.

Syntax

```
srvctl disable pdb -db db_unique_name -pdb pdb_name [-node node_name]
```

Parameters

Table A-68 srvctl disable pdb Command Parameters

Parameter	Description
`-db db_unique_name`	The unique name of the Container Database (CDB).
`-pdb pdb_name`	The name of the PDB.
`-node node_name`	The node on which you want to disable the PDB. **Note:** You can only use this parameter only with Oracle Clusterware.

Example

The following example disables the PDB `crmeast`:

```
srvctl disable pdb -db crm -pdb crmeast
```

srvctl enable pdb

Enables the Pluggable Database (PDB) for Oracle Clusterware management.

Syntax

```
srvctl enable pdb -db db_unique_name -pdb pdb_name [-node node_name]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

Table A-69 srvctl enable pdb Command Parameters

Parameter	Description
`-db db_unique_name`	The unique name of the Container Database (CDB).
`-pdb pdb_name`	The name of the PDB that you want to enable.
`-node node_name`	The name of the node on which the PDB resource resides that you want to enable. **Note:** You can only use this parameter with Oracle Clusterware.

Example

The following example enables a PDB named `crmeast` for Oracle Clusterware management:

```
srvctl enable pdb -db crm -pdb crmeast
```

srvctl modify pdb

Modifies the configuration for a Pluggable Database (PDB).

Syntax

```
srvctl modify pdb -db db_unique_name -pdb pdb_name
     [-cardinality {num_of_instances | ALL}]
     [-maxcpu max_cpu_usage] [-mincpuunit min_cpu_usage]
     [-rank rank] [-startoption start_options]
     [-stopoption stop_options] [-policy policy]
```

Parameters

Table A-70 srvctl modify pdb Command Parameters

Parameter	Description
`-db db_unique_name`	Unique name for the Container Database (CDB).
`-pdb pdb_name`	The name of the PDB.
`-cardinality {num_of_instances | ALL}`	The number of instances of the PDB to be open at any time. If you specify ALL, then a PDB is open in every available Container Database (CDB) in the cluster database.

> **Note**
>
> You can modify the `-cardinality` parameter only if you had set the `-cardinality` parameter when creating the PDB.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-70 (Cont.) srvctl modify pdb Command Parameters

Parameter	Description
`-maxcpu max_cpu_usage`	The maximum CPU usage limit for the PDB in whole CPUs. Specify a positive integer value that is equal to or greater than 1. You must be logged in as either the `grid` or the `root` user to modify this parameter.
`-mincpuunit min_cpu_usage`	The minimum CPU usage limit for the PDB. Specify a positive integer value that is equal to or greater than 10. The value must be in hundredths of the total CPU count (1/100 CPU count). You must be logged in as either the `grid` or the `root` user to modify this parameter.
`-rank rank`	The rank of the PDB. The range of values you can specify for this parameter is 0 to 5. The default value is 0. You must be logged in as either the `grid` or the `root` user to modify this parameter. **Note** You can modify the `-rank` parameter only if you had set the `-rank` parameter when creating the PDB.
`-startoption start_options`	Startup options for the PDB, such as `OPEN` or `OPEN READ ONLY`. The default value is an empty string, which means the PDB uses the same open mode as the CDB. **Note:** For multi-word startup options, such as `read only` and `read write`, separate the words with a space and enclose in double quotation marks (`""`). For example, `"read only"`.
`-stoption stop_options`	Stop options for the PDB, such as `NORMAL`. The default stop option is `IMMEDIATE`.
`-policy policy`	Management policy for the Pluggable Database, where policy can be one of the following values: `AUTOMATIC`, `MANUAL`, or `RESTART`.

Related Topics

- Database Startup
- Database Shutdown
- Pluggable Database Rank
  The PDB `-rank` parameter defines relative importance of the PDBs, which are created specifying cardinality, in a database with the `RANK` management policy.

srvctl remove pdb

Removes the Pluggable Database (PDB) configuration from Oracle Clusterware management.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Syntax

```
srvctl remove pdb -db db_unique_name -pdb pdb_name
    [-force] [-noprompt] [-verbose]
```

Parameters

Table A-71 srvctl remove pdb Command Parameters

Parameter	Description
`-db db_unique_name`	Unique name of the Container Database (CDB).
`-pdb pdb_name`	The name of the PDB.
`-force`	Forcibly remove the PDB and ignore any dependencies.
`-noprompt`	Suppress prompts.
`-verbose`	Display verbose output.

Example

This example shows how to remove configuration from Oracle Clusterware management of the PDB named `crmeast` in the Container Database (CDB) named `crm`.

```
$ srvctl remove pdb -db crm -pdb crmeast
```

srvctl start pdb

Starts a Pluggable Database (PDB) and the associated services based on the management policy of the service. All the services with the `AUTOMATIC` management policy will start automatically.

Syntax

```
srvctl start pdb -db db_unique_name -pdb pdb_name
  [-startoption start_options] [-node node_list]
```

Parameters

Table A-72 srvctl start pdb Command Parameters

Parameter	Description
`-db db_unique_name`	The unique name of the database to start.
`-pdb pdb_name`	The name of the PDB to start.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-72 (Cont.) srvctl start pdb Command Parameters

Parameter	Description
`-startoption start_options`	Options for the startup command, such as `READ ONLY` or `OPEN`. **Notes:** • This command parameter supports all PDB startup options. • For multi-word startup options, such as `READ ONLY` and `READ WRITE`, separate the words with a space and enclose in double quotation marks (`""`). For example, `"READ ONLY"`. **See Also:** STARTUP command in *SQL*Plus User's Guide and Reference*
`-node node_list`	Comma-separated list of nodes on which to start the PDB.

Examples

The following example starts the `crmeast` PDB in `READ ONLY` mode:

```
srvctl start pdb -db crm -pdb crmeast -startoption "read only"
```

srvctl status pdb

This command displays the current state of the Pluggable Database (PDB).

Syntax

```
srvctl status pdb -db db_unique_name [-pdb pdb_name]
    [-detail]
```

Parameters

Table A-73 srvctl status pdb Parameters

Parameter	Description
`-db db_unique_name`	The unique name of the Container Database (CDB).
`-pdb pdb_name`	The name of the PDB. If you do not specify a PDB name, then information on all the PDBs in that particular database are displayed.
`–detail`	Display detailed status information.

Examples

This example shows sample output for the status information for the `crmeast` and `crmnorth` PDBs.

```
$ srvctl status pdb -db crm -pdb crmeast

Pluggable database crmeast of crm is running on nodes site1, site3.

$ srvctl status pdb -db crm -pdb crmnorth

Pluggable database crmnorth of crm is not running.
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl stop pdb

Stops a Pluggable Database (PDB) and its services.

Syntax

```
srvctl stop pdb -db db_unique_name -pdb pdb_name [-node node_name]
  [-stopoption stop_options] [-drain_timeout timeout]
  [-stopsvcoption stop_service_options] [-force]
```

Parameters

Table A-74 srvctl stop pdb Command Parameters

Parameter	Description
`-db db_unique_name`	The unique name for the Container Database (CDB).
`-pdb pdb_name`	The name of the PDB to stop.
`-node node_name`	The name of the node on which to stop the PDB. If you do not specify any nodes, then the specified PDB is stopped on all the nodes where the PDB is running.
`-stopoption stop_options`	Options for the shutdown command, such as `NORMAL` or `IMMEDIATE`. The default value is `IMMEDIATE`.
`-drain_timeout timeout`	The time, in seconds, for the resource draining action to complete. By default, this parameter is not set. You can specify 0 or any positive integer. An empty string unsets the parameter. If you specify zero, then the agent will perform the actions related to service draining, immediately. Drain timeout is the maximum time the service waits before exiting (in case of `srvctl stop service` or `srvctl stop instance`) or proceeding to stop database (`srvctl stop database` or `srvctl stop pdb`), until the draining of sessions is completed. If session draining completes in 10 seconds and the drain timeout value is 100 seconds, then SRVCTL continues after 10 seconds. It does not wait for the remaining 90 seconds.
`-stopsvcoption stop_service_options`	Options for stopping services, such as `TRANSACTIONAL` or `IMMEDIATE`. If you do not specify this open, then the stop option set in the service resource attribute `USR_ORA_STOP_MODE` is used.
`-force`	Use this parameter to force stop the PDB and its services, and any dependent resources.

Example

The following command stops a PDB named `crmeast` that is open in the `crm` database, allowing 50 seconds for all sessions to drain from the PDB:

```
srvctl stop pdb -db crm -pdb crmeast -drain_timeout 50
```

scan Commands

Use commands with the `scan` keyword to add, list the configuration of, modify, enable, disable, start, stop, relocate, obtain the status of, and remove SCAN VIPs.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- srvctl add scan
  Adds Oracle Clusterware resources for the given SCAN.
- srvctl config scan
  Displays the configuration information for all SCAN VIPs, by default, or a specific SCAN VIP identified by *ordinal_number*.
- srvctl disable scan
- srvctl enable scan
- srvctl modify scan
- srvctl predict scan
- srvctl relocate scan
  Relocates a specific SCAN VIP from its current hosting node to another node within the cluster.
- srvctl remove scan
- srvctl start scan
- srvctl status scan
- srvctl stop scan

srvctl add scan

Adds Oracle Clusterware resources for the given SCAN.

Syntax

```
srvctl add scan -scanname scan_name [-netnum network_number]
```

Parameters

Table A-75 srvctl add scan Command Parameters

Parameter	Description
`-scanname` *scan_name*	A fully-qualified host name, which includes the domain name. If the network is dynamic, then you do not have to use fully-qualified host name but, if you choose to do so, then the domain must be the GNS subdomain. **Note:** You can modify this attribute using Online Resource Attribute Modification.
`-netnum` *network_number*	The optional network number from which SCAN VIPs are obtained. If you do not specify this parameter, then the SCAN VIPs are obtained from the same default network from which the `nodeapps` VIP is obtained.

Usage Notes

- This command creates the same number of SCAN VIP resources as the number of IP addresses that SCAN resolves to, or 3 when *network_number* identifies a dynamic network and Oracle GNS configuration.
- For static networks, the addresses to which the SCAN resolves in DNS must match the address type of the subnet.
- For an IPv4 network, the SCAN must resolve to IPv4 addresses.
- This command is only available with Oracle Clusterware.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example

An example of this command is:

```
# srvctl add scan -scanname scan.mycluster.example.com
```

srvctl config scan

Displays the configuration information for all SCAN VIPs, by default, or a specific SCAN VIP identified by *ordinal_number*.

Syntax

```
srvctl config scan [[-netnum network_number] [-scannumber ordinal_number] | -
all]
```

Parameters

Table A-76 srvctl config scan Command Parameters

Parameter	Description
`-netnum network_number`	Use this parameter to view the configuration of a specific SCAN VIP.
`-scannumber ordinal_number`	Use this parameter to specify any one of the three SCAN VIPs, using values from 1 to 3, for which you want to view the configuration.
`-all`	Alternative to specifying network or ordinal numbers, you can use this parameter to view the configuration for all of the SCAN VIPs.

Usage Notes

This command is only available with Oracle Clusterware.

Example

This command returns output similar to the following:

```
$ srvctl config scan -scannumber 1

SCAN name: mjk12700890090-r, Network: 1
Subnet IPv4: 198.51.100.1/203.0.113.46/eth0, static
Subnet IPv6:
SCAN 1 IPv4 VIP: 198.51.100.195
SCAN VIP is enabled.
SCAN VIP is individually enabled on nodes:
SCAN VIP is individually disabled on nodes:
```

srvctl disable scan

Disables all SCAN VIPs, by default, or a specific SCAN VIP identified by *ordinal_number*.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Syntax

```
srvctl disable scan [-scannumber ordinal_number]
```

Usage Notes

- You can only use this command with Oracle Clusterware.
- Optionally, you can use the `-scannumber` parameter to specify any one of the three SCAN VIPs you want to disable. The parameter takes a range of values from 1 to 3.

Example

The following example disables the first SCAN VIP:

```
$ srvctl disable scan -scannumber 1
```

srvctl enable scan

Enables all SCAN VIPs, by default, or a specific SCAN VIP identified by its ordinal number.

Syntax

```
srvctl enable scan [-scannumber ordinal_number]
```

Usage Notes

- You can only use this command with Oracle Clusterware.
- Optionally, you can use the `-scannumber` parameter to specify any one of the three SCAN VIPs you want to enable. The parameter takes a range of values from 1 to 3.

Example

The following example enables the first SCAN VIP:

```
$ srvctl enable scan -scannumber 1
```

srvctl modify scan

Modifies the number of SCAN VIPs to match the number of IP addresses returned by looking up the *scan_name* you specify in DNS.
You use this command when DNS was modified to add, change, or remove IP addresses, and now you must adjust the Oracle Clusterware resource configuration to match.

Syntax

```
srvctl modify scan -scanname scan_name [-netnum network_number]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

Table A-77 srvctl modify scan Command Parameters

Parameter	Description
`-scanname` *scan_name*	Identifies the SCAN name that resolves to the SCAN VIPs that you want to modify. **Note:** You can modify this attribute using Online Resource Attribute Modification.
`-netnum` *network_number*	The optional network number from which VIPs are obtained. If not specified, the VIPs are obtained from the same default network from which the `nodeapps` VIP is obtained.

Example

Assume your system currently has a SCAN named `scan_name1`, and it resolves to a single IP address in DNS. If you modify the SCAN `scan_name1` in DNS to resolve to three IP addresses, then use the following command to create the additional SCAN VIP resources:

```
$ srvctl modify scan -scanname scan_name1
```

srvctl predict scan

Predicts the consequences of SCAN failure.

Syntax

```
srvctl predict scan -scannumber ordinal_number [-verbose]
```

Usage Notes

- Specify an ordinal number that identifies the SCAN VIP for which you want to simulate failure. The range of values you can specify for this parameter is 1 to 3.
- Optionally, you can use the `–verbose` parameter to display detailed output.

Add additional information about the command here.

srvctl relocate scan

Relocates a specific SCAN VIP from its current hosting node to another node within the cluster.

Syntax

```
srvctl relocate scan -scannumber ordinal_number [-netnum network_number] [-
node node_name]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

Table A-78 srvctl relocate scan Command Parameters

Parameter	Description
`-scannumber` *`ordinal_number`*	Specify an ordinal number that identifies which SCAN VIP you want to relocate. The range of values you can specify for this parameter is 1 to 3.
`-netnum` *`network_number`*	The optional network number from which VIPs are obtained. If not specified, the VIPs are obtained from the same default network from which the `nodeapps` VIP is obtained. The default network number is 1.
`-node` *`node_name`*	Optionally, you can specify the name of a single node to which SRVCTL relocates the SCAN VIP. If you do not use this parameter, then SRVCTL chooses the node to which the SCAN VIP is relocated.

Usage Notes

You can only use this command with Oracle Clusterware.

Example

The following example relocates the first SCAN VIP to `node1`:

```
$ srvctl relocate scan -scannumber 1 -node node1
```

srvctl remove scan

Removes Oracle Clusterware resources from all SCAN VIPs.

Syntax

```
srvctl remove scan [-netnum network_number] [-force] [-noprompt]
```

Parameters

Table A-79 srvctl remove scan Command Parameters

Parameter	Description
`-netnum` *`network_number`*	The optional network number from which VIPs are obtained. If not specified, the VIPs are obtained from the same default network from which the `nodeapps` VIP is obtained.
`–force`	Removes the SCAN VIPs even though there are SCAN listeners running that are dependent on the SCAN VIPs.
`–noprompt`	Use this parameter to suppress all prompts.

Usage Notes

If you use the `-force` option, then SCAN VIPs that are running are not stopped before the dependent resources are removed, which may require manual cleanup.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Examples

An example of this command is:

```
$ srvctl remove scan -force
```

srvctl start scan

Starts all SCAN VIPs, by default, or a specific SCAN VIP, on all nodes or a specific node in the cluster.

Syntax

```
srvctl start scan [-scannumber ordinal_number] [-node node_name]
```

Parameters

Table A-80 srvctl start scan Command Parameters

Parameter	Description
`-scannumber ordinal_number`	Optionally, you can specify an ordinal number that identifies which SCAN VIP you want to start. The range of values you can specify for this parameter is 1 to 3. If you do not use this parameter, then SRVCTL starts all the SCAN VIPs.
`-node node_name`	Optionally, you can specify the name of a single node on which the SCAN VIP resides that you want to start. If you do not specify this parameter, then SRVCTL starts the SCAN VIPs on all nodes in the cluster.

Usage Notes

You can only use this command with Oracle Clusterware.

Example

The following example starts the SCAN VIP identified by the ordinal number 1 on the `crm1` node:

```
$ srvctl start scan -scannumber 1 -node crm1
```

srvctl status scan

Displays the status for all SCAN VIPs, by default, or a specific SCAN VIP.

Syntax

```
srvctl status scan [-scannumber ordinal_number] [-verbose]
```

Usage Notes

- You can only use this command with Oracle Clusterware.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Optionally, you can specify an ordinal number that identifies a specific SCAN VIP for which you want to display the status. The range of values you can specify for this parameter is 1 to 3. If you do not use this parameter, then SRVCTL displays the status of all SCAN VIPs in the cluster.
- Optionally, you can use the `-verbose` parameter to display detailed output.

srvctl stop scan

Stops all SCAN VIPs, by default, that are running or in starting state, or stops a specific SCAN VIP identified by an ordinal number.

Syntax

```
srvctl stop scan [-scannumber ordinal_number] [-force]
```

Usage Notes

- You can only use this command with Oracle Clusterware.
- Optionally, you can specify an ordinal number that identifies which SCAN VIP you want to stop. The range of values you can specify for this parameter is 1 to 3. If you do not use this parameter, then SRVCTL stops all the SCAN VIPs.
- Optionally, you can use the `-force` parameter to stop the SCAN VIPs regardless of any dependencies.

Example

The following example stops the SCAN VIP identified by the ordinal number 1:

```
$ srvctl stop scan -scannumber 1
```

scan_listener Commands

Use commands with the `scan_listener` keyword to add, list the configuration of, modify, enable, disable, start, stop, relocate, obtain the status of, and remove SCAN listeners.

- srvctl add scan_listener
  Adds Oracle Clusterware resources for the SCAN listeners.
- srvctl config scan_listener
  Displays the configuration information for all SCAN listeners, by default, or a specific listener identified by network number or ordinal_number.
- srvctl disable scan_listener
- srvctl enable scan_listener
- srvctl export scan_listener
  Saves the SCAN listener configuration information to a file.
- srvctl modify scan_listener
  Modifies the SCAN listener to match that of the SCAN VIP, or modifies the SCAN listener endpoints or service registration restrictions.
- srvctl predict scan_listener
- srvctl relocate scan_listener

Copyright © 1999, 2025, Oracle and/or its affiliates.

- srvctl remove scan_listener
- srvctl start scan_listener
- srvctl status scan_listener
  Displays the status for all SCAN listeners, by default, or a specific listener.
- srvctl stop scan_listener
- srvctl update scan_listener

srvctl add scan_listener

Adds Oracle Clusterware resources for the SCAN listeners.

Syntax

```
srvctl add scan_listener [-netnum network_number] [-listener
lsnr_name_prefix] [-skip]
  [-endpoints "[TCP:]port_list[/IPC:key][/NMP:pipe_name]
  [/{TCPS|SDP|EXADIRECT}port_list]"]
  [-invitednodes "node_list"] [-invitedsubnets "subnet_list"]
  [-clientcluster cluster_name] [-clientdata <filename>]
```

Parameters

Table A-81 srvctl add scan_listener Command Parameters

Parameter	Description
`-netnum network_number`	The optional network number from which SCAN VIPs are obtained. If you do not specify this parameter, then the SCAN VIPs are obtained from the same default network from which the `nodeapps` VIP is obtained.
`-listener lsnr_name_prefix`	The SCAN listener name prefix.
`-skip`	Skip checking of the ports.
`-endpoints "[TCP:]port_list[/IPC:key][/NMP:pipe_name][/{TCPS|SDP|EXADIRECT}port_list]"`	Protocol specifications for the SCAN listener. Use `port_list` to specify a comma-delimited list of TCP ports or SCAN listener endpoints. You can also specify endpoints for TCPS, SDP, and EXADIRECT ports. **Note:** You can modify this attribute using Online Resource Attribute Modification.
`-invitednodes "node_list"`	A comma-delimited list of host names from outside the cluster that are allowed to register with the SCAN listener.
`-invitedsubnets "subnet_list"`	A comma-delimited list of subnets from outside the cluster that are allowed to register with the SCAN listener. You can specify the subnets using either CIDR notation or wildcards (such as 192.168.*).
`-clientcluster cluster_name`	The name of the cluster that is running the SCAN listener you want to share.
`-clientdata file_name`	The name of the cluster that is running the shared SCAN listener.

Usage Notes

- The number of SCAN listener resources created is the same as the number of SCAN VIP resources.
- This command is only available with Oracle Clusterware.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example

An example of this command is:

```
# srvctl add scan_listener -listener myscanlistener
```

srvctl config scan_listener

Displays the configuration information for all SCAN listeners, by default, or a specific listener identified by network number or ordinal_number.

Syntax

```
srvctl config scan_listener [[-netnum network_number] [-scannumber
ordinal_number]
 [-clientcluster cluster_name] | -all]
```

Parameters

Table A-82 srvctl config scan_listener Command Parameters

Parameter	Description
`-netnum network_number`	Use this parameter to view the configuration of the listener for a specific SCAN VIP.
`-scannumber ordinal_number`	Use this parameter to specify any one of the three SCAN VIPs, using values from 1 to 3, for which you want to view the configuration of the listener.
`-clientcluster cluster_name`	The name of the cluster that is running the shared SCAN listener.
`–all`	Alternative to specifying network or ordinal numbers, you can use this parameter to view the configuration of the listeners for all of the SCAN VIPs.

Usage Notes

This command is only available with Oracle Clusterware.

Example

This command returns output similar to the following:

```
$ srvctl config scan_listener -scannumber 1

SCAN Listener LISTENER_SCAN1 exists. Port: TCP:1529
Registration invited nodes:
Registration invited subnets:
SCAN Listener is enabled.
SCAN Listener is individually enabled on nodes:
SCAN Listener is individually disabled on nodes:
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl disable scan_listener

Disables all SCAN listeners, by default, or a specific listener identified by an ordinal number or client cluster.

Syntax

```
srvctl disable scan_listener [-netnum network_number] [-scannumber
ordinal_number]
  [-clientcluster cluster_name]
```

Parameters

Table A-83 srvctl disable scan_listener Command Parameters

Parameter	Description
`-netnum network_number`	Use this parameter to disable SCAN listeners for a specific network number.
`-scannumber ordinal_number`	Use this parameter to disable any one of the three SCAN VIPs, using values from 1 to 3. If you do not use this parameter, then SRVCTL disables all SCAN listeners.
`-clientcluster cluster_name`	The name of the cluster that is running the shared SCAN listener.

Usage Notes

This command is only available with Oracle Clusterware.

Example

The following example disables the SCAN listener identified as 1:

```
$ srvctl disable scan_listener -scannumber 1
```

srvctl enable scan_listener

Enables all SCAN listeners, by default, or a specific listener identified by its ordinal number.

Syntax

```
srvctl enable scan_listener [-netnum network_number] [-scannumber
ordinal_number]
  [-clientcluster <cluster_name>]
```

Parameters

Table A-84 srvctl enable scan_listener Command Parameters

Parameter	Description
`-netnum network_number`	Use this parameter to enable the listener for a specific SCAN VIP.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-84 (Cont.) srvctl enable scan_listener Command Parameters

Parameter	Description
`-scannumber` *`ordinal_number`*	Use this parameter to enable any one of the three SCAN VIPs, using values from 1 to 3. If you do not use this parameter, then SRVCTL enables all SCAN listeners.
`-clientcluster` *`cluster_name`*	The name of the cluster that is running the shared SCAN listener.

Usage Notes

This command is only available with Oracle Clusterware.

Example

The following example enables the SCAN listener identified as 1:

```
$ srvctl enable scan_listener -scannumber 1
```

srvctl export scan_listener

Saves the SCAN listener configuration information to a file.

Syntax

```
srvctl export scan_listener -clientcluster cluster_name -clientdata filename
```

Parameters

Table A-85 srvctl export scan_listener Command Parameters

Parameter	Description
`-clientcluster` *`cluster_name`*	Specify the cluster name.
`-clientdata` *`filename`*	Specify the path to the file to which credentials data will be written.

srvctl modify scan_listener

Modifies the SCAN listener to match that of the SCAN VIP, or modifies the SCAN listener endpoints or service registration restrictions.

Syntax

```
srvctl modify scan_listener {-update | -endpoints "[TCP:]port_list[/IPC:key]
     [/NMP:pipe_name][/{TCPS|SDP|EXADIRECT}port_list]"} [-invitednodes
"node_list"]
     [-invitedsubnets "subnet_list"] [-clientcluster cluster_name]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

Table A-86 srvctl modify scan_listener Command Parameters

Parameter	Description
`-update`	Use this parameter to update SCAN listener configuration to match the current SCAN VIP configuration. This parameter adds new resources or removes existing SCAN listener resources to match the number of SCAN VIP resources.
`-endpoints "[TCP:]port_list[/IPC:key] [/NMP:pipe_name][/{TCPS|SDP|EXADIRECT}port_list]"`	Protocol specifications for the SCAN listener. Use *`port_list`* to specify a comma-delimited list of TCP ports or listener endpoints. You can also specify endpoints for TCPS, SDP, and EXADIRECT ports.
`-invitednodes "node_list"`	Use this parameter to specify a comma-delimited list of host names from outside the cluster that are allowed to register with the SCAN listener.
`-invitedsubnets "subnet_list"`	Use this parameter to specify a comma-delimited list of subnets from outside the cluster that are allowed to register with the SCAN listener. You can specify the subnets using either CIDR notation or wildcards (such as 192.168.*).
`-clientcluster cluster_name`	The name of the cluster that is running the shared SCAN listener.

Example

Assume your system currently has a SCAN named `scan_name1`, and you recently modified the DNS entry to resolve to three IP addresses instead of one. After running the `srvctl modify scan` command to create additional SCAN VIP resources, use the following command to create Oracle Clusterware resources for the additional two SCAN listeners to go with the two additional SCAN VIPs:

```
$ srvctl modify scan_listener -update
```

srvctl predict scan_listener

Predicts the consequences of SCAN listener failure.

Syntax

```
srvctl predict scan_listener -scannumber ordinal_number [-verbose]
```

Usage Notes

- Use the `-scannumber` parameter to specify any one of the three SCAN listeners for which you want to predict the consequences of a failure. The range of values you can specify for this parameter is 1 to 3.
- Optionally, you can use the `–verbose` parameter to display detailed output.

Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl relocate scan_listener

Relocates a specific SCAN listener from its current hosting node to another node within the cluster.

Syntax

```
srvctl relocate scan_listener -scannumber ordinal_number [-node node_name]
```

Usage Notes

- You can only use this command with Oracle Clusterware.
- Specify an ordinal number that identifies which SCAN listener you want to relocate. The range of values you can specify for this parameter is 1 to 3.
- Optionally, you can specify the name of a single node to which you want to relocate the SCAN listener. If you do not specify this parameter, then SRVCTL chooses the node to which the SCAN listener is relocated.

Example

The following example relocates the SCAN listener identified as 3 to `node2` of the cluster:

```
$ srvctl relocate scan_listener -scannumber 3 -node node2
```

srvctl remove scan_listener

Removes Oracle Clusterware resources from all SCAN listeners.

Syntax

```
srvctl remove scan_listener [-netnum network_number] [-clientcluster
cluster_name]
  [-force] [-noprompt]
```

Parameters

Table A-87 srvctl remove scan_listener Command Parameters

Parameter	Description
`-netnum network_number`	The optional network number from which SCAN VIPs are obtained. If you do not specify this parameter, then the SCAN VIPs are obtained from the same default network from which the `nodeapps` VIP is obtained.
`-clientcluster cluster_name`	The name of the cluster that is running the shared SCAN listener.
`-force`	Removes the SCAN listener without stopping the SCAN listener if it is running.
`–noprompt`	Use this parameter to suppress all prompts.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example

An example of this command is:

```
$ srvctl remove scan_listener -force
```

srvctl start scan_listener

Starts all SCAN listeners, by default, or a specific listener on all nodes or a specific node in the cluster.

Syntax

```
srvctl start scan_listener [-netnum network_number] [-scannumber
ordinal_number]
  [-node node_name] [-clientcluster cluster_name]
```

Parameters

Table A-88 srvctl start scan_listener Command Parameters

Parameter	Description
`-netnum network_number`	Use this parameter to start SCAN listeners for a specific network number.
`-scannumber ordinal_number`	Use this parameter to start one of the three SCAN VIPs, using values from 1 to 3. If you do not use this parameter, then SRVCTL starts all SCAN listeners.
`-node node_name`	Specify the name of a single node on which you want to start a SCAN listener. If you do not use this parameter, then SRVCTL starts the SCAN listeners on all nodes in the cluster.
`-clientcluster cluster_name`	The name of the cluster that is running the shared SCAN listener.

Usage Notes

This command is only available with Oracle Clusterware.

Example

The following example starts the SCAN listener identified as 1:

```
$ srvctl start scan_listener -scannumber 1
```

srvctl status scan_listener

Displays the status for all SCAN listeners, by default, or a specific listener.

Syntax

```
srvctl status scan_listener [[-netnum network_number] [-scannumber
ordinal_number]
   | [-clientcluster cluster_name] | -all] [-verbose]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

Table A-89 srvctl status scan_listener Command Parameters

Parameter	Description
`-netnum` *network_number*	The network number. The default network number is 1.
`-scannumber` *ordinal_number*	An ordinal number that identifies a specific SCAN listener. The range of values you can specify for this parameter is 1 to 3. If you do not use this parameter, then the utility displays the status of all SCAN listeners in the cluster.
`-clientcluster` *cluster_name*	The name of the cluster that is running the shared SCAN listener.
`-all`	Display the status for SCAN listeners for all networks.
`-verbose`	Display detailed information.

Usage Notes

This command is only available with Oracle Clusterware.

srvctl stop scan_listener

Stops all SCAN listeners, by default, that are in a running or starting state, or a specific listener identified by an ordinal number.

Syntax

```
srvctl stop scan_listener [-netnum network_number] [-scannumber
ordinal_number]
  [-clientcluster cluster_name] [-force]
```

Parameters

Table A-90 srvctl stop scan_listener Command Parameters

Parameter	Description
`-netnum` *network_number*	Use this parameter to stop SCAN listeners for a specific network number.
`-scannumber` *ordinal_number*	Use this parameter to stop any one of the three SCAN VIPs, using values from 1 to 3. If you do not use this parameter, then SRVCTL stops all SCAN listeners.
`-clientcluster` *cluster_name*	The name of the cluster that is running the shared SCAN listener.
`-force`	Stops the SCAN listener regardless of any dependencies.

Usage Notes

This command is only available with Oracle Clusterware.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example

The following example stops the SCAN listener identified as 1:

```
$ srvctl stop scan_listener -scannumber 1
```

srvctl update scan_listener

Updates the SCAN listeners to listen on the new endpoints.

Syntax

```
srvctl update scan_listener
```

Usage Notes

- You can only use this command with Oracle Clusterware.
- This command does not accept any additional parameters, except for `-help`.

service Commands

Use commands with the `service` keyword to add, modify, list the configuration of, enable, disable, start, stop, obtain the status of, relocate, and remove services.

- srvctl add service
  Adds services to a database and assigns them to instances.
- srvctl config service
  Displays the configuration for a service.
- srvctl disable service
- srvctl enable service
- srvctl modify service
  Modifies a service configuration.
- srvctl predict service
- srvctl relocate service
  Temporarily relocates the specified service names from one specified instance to another specified instance.
- srvctl remove service
  Removes the service from Oracle Clusterware management.
- srvctl start service
  Starts a service or multiple services on a database, Pluggable Database (PDB) , or instance.
- srvctl status service
- srvctl stop service
  Stops one or more services globally across the cluster database, or on the specified instance.

Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl add service

Adds services to a database and assigns them to instances.

Syntax

Use this command with one of the following syntax models.

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

To add a service to a database:

```
srvctl add service -db db_unique_name -service service_name_list [-
true_cache_service true_cache_service_list]
   [-cardinality {UNIFORM | SINGLETON | DUPLEX} | -preferred "preferred_list"
    [-available "available_list"] [-tafpolicy {BASIC | NONE | PRECONNECT}] |
[-cardinality {UNIFORM | SINGLETON}] ]
   [-netnum network_number] [-role "[PRIMARY][,PHYSICAL_STANDBY]
[,LOGICAL_STANDBY][,SNAPSHOT_STANDBY]"]
   [-policy {AUTOMATIC | MANUAL}] [-notification {TRUE | FALSE}] [-dtp {TRUE
| FALSE}]
   [-clbgoal {SHORT | LONG}] [-rlbgoal {NONE | SMART_CONN | SERVICE_TIME |
THROUGHPUT}] [-resetstate {NONE | LEVEL1}]
   [-failovertype {NONE|SESSION|SELECT|TRANSACTION|AUTO}] [-failovermethod
{NONE | BASIC}] [-failoverretry failover_retries]
   [-failoverdelay failover_delay] [-failover_restore {NONE|LEVEL2|LEVEL1|
AUTO}] -template_timeout template_timeout [-failback {YES | NO}]
   [-edition edition_name] [-pdb pdb_name] [-global {TRUE | FALSE}] [-maxlag
max_lag_time]
   [-sql_translation_profile sql_translation_profile] [-commit_outcome {TRUE|
FALSE}] [-retention retention_time]
   [-replay_init_time replay_initiation_time] [-session_state {DYNAMIC |
AUTO}] [-importfile file_name]
   [-gsmflags gsm_flags] [-tablefamilyid table_family_id]
   [-drain_timeout timeout] [-stopoption {NONE|IMMEDIATE|TRANSACTIONAL}] [-
css_critical {yes | no}]
   [-force] [-verbose]
```

To update the preferred and available lists of an existing service:

```
srvctl add service -db db_unique_name -service "service_name_list"
   -update {-preferred "preferred_list" | -available "available_list"} [-
force]
   [-verbose]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

The following table lists and describes all the `srvctl add service` parameters and whether they can be used when adding a service to either an Oracle RAC database or non-cluster database.

Table A-91 srvctl add service Command Parameters

Parameter	Description
`-db` *db_unique_name*	Unique name for the database.
`-service` *service_name_list*	The *service_name*`.`*service_domain* should be unique within the cluster unless you want to spread connections across multiple databases that offer the same service. If you do not specify the service domain as part of the service name (such as `sales.example.com`), then the `DB_DOMAIN` database attribute is appended to the service name. You can specify one service or several services in a comma-delimited list. **Note:** The `-service` parameter has a 64 characters limit for its value. Therefore, the total length of the names of all services assigned to an instance cannot exceed 64 characters.
`-true_cache_service` *true_cache_service_list*	Specify this parameter to associate a primary service with the True Cache service.
`-cardinality {UNIFORM | SINGLETON | DUPLEX}`	The cardinality of the service, which can be one of the following: • `UNIFORM` – offered on all instances or PDBs in the database • `SINGLETON` – runs on only one instance or PDB at a time • `DUPLEX` – runs on two instances or PDBs at a time **Note** This parameter can be used only with Oracle RAC.
`-preferred "`*preferred_list*`"`	A comma-separated list of preferred instances on which the service runs. The list of preferred instances must be mutually exclusive with the list of available instances. **Note** This parameter can be used only with Oracle RAC.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-91 (Cont.) srvctl add service Command Parameters

Parameter	Description
`-available "available_list"`	A comma-separated list of available instances, in priority order, to which the service fails over. The list of available instances must be mutually exclusive with the list of preferred instances. **Note** This parameter can be used only with Oracle RAC.
`-tafpolicy {BASIC|NONE}`	Transparent Application Failover (TAF) policy specification.
`-netnum network_number`	Use this parameter to determine on which network this service is offered. The service is configured to depend on VIPs from the specified network. **Notes:** • If you omit this parameter, then the default is taken from the database configuration, which you specify using `srvctl add database` or `srvctl modify database`, with the `-defaultnetwork` parameter to specify the default network for that database's services. • This parameter can be used only with Oracle RAC and Oracle RAC One Node database configurations.
`-role "[PRIMARY][,PHYSICAL_STANDBY][,LOGICAL_STANDBY][,SNAPSHOT_STANDBY]"`	The service role. You can specify one or more roles in a comma-delimited list. Use this option to indicate that the service should only be automatically started upon database open when the Oracle Data Guard database role matches one of the specified service roles. Using SRVCTL to manually start a service is not affected by the service role. **Note:** The `-role` parameter is only used at database startup and by the Oracle Data Guard Broker. All manual service startup must specify the name of the service to be started by the user.
`-policy {AUTOMATIC | MANUAL}`	Service management policy. If `AUTOMATIC` (the default), then the service is automatically started upon restart of the database, either by a planned restart (with SRVCTL) or after a failure. Automatic restart is also subject to the service role, however (the `-role` parameter). If `MANUAL`, then the service is never automatically restarted upon planned restart of the database (with SRVCTL). A `MANUAL` setting does not prevent Oracle Clusterware from monitoring the service when it is running and restarting it if a failure occurs. **Note:** Using CRSCTL to stop and start the Oracle Clusterware restarts the service in the same way that a failure does.
`-notification {TRUE | FALSE}`	Enable Fast Application Notification (FAN) for OCI connections.
`-dtp {TRUE | FALSE}`	Indicates whether Distributed Transaction Processing should be enabled for this service. This service will be a preferred service on a single node in an administrator-managed database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-91 (Cont.) srvctl add service Command Parameters

Parameter	Description
`-clbgoal {SHORT | LONG}`	Connection Load Balancing Goal. Use a value of `SHORT` for this parameter for run-time load balancing, or if using an integrated connection pool. Use a value of `LONG` for this parameter for long running connections, such as batch jobs, that you want balanced by the number of sessions per node for the service.
`-rlbgoal {NONE | SMART_CONN | SERVICE_TIME | THROUGHPUT}`	Runtime Load Balancing Goal (for the Load Balancing Advisory). Set this parameter to `SMART_CONN` to enable Smart Connection Rebalance. Set this parameter to `SERVICE_TIME` to balance connections by response time. Set this parameter to `THROUGHPUT` to balance connections by throughput.
`-resetstate {NONE | LEVEL1}`	Reset state in a session to clean values. If set to `NONE`, then session state is not cleaned. If set to `LEVEL1`, then session states that cannot be restored are reset. The session state reset excludes SYS CONTEXT and secure roles.
`-failovertype {NONE | SESSION | SELECT | TRANSACTION | AUTO}`	Set the failover type. To enable Application Continuity for Java, set this parameter to `TRANSACTION`. To enable Transparent Application Continuity, set this parameter to `AUTO`. To enable TAF for OCI, set this parameter to `SELECT` or `SESSION`. **Note:** If you set `-failovertype` to `TRANSACTION`, then you must set `-commit_outcome` to `TRUE`.
`-failovermethod {NONE | BASIC}`	TAF failover method (for backward compatibility only). If the failover type (`-failovertype`) is set to a value other than `NONE`, then you should choose `BASIC` for this parameter. **Note:** This parameter can be used only with Oracle RAC.
`-failoverretry` *`failover_retries`*	For Application Continuity and TAF, this parameter determines the number of attempts to connect after an incident.
`-failoverdelay` *`failover_delay`*	For Application Continuity and TAF, this parameter specifies the time delay (in seconds) between reconnect attempts per incident at failover.
`-failover_restore {NONE|LEVEL2|LEVEL1|AUTO}`	For Application Continuity, when you set the `-failover_restore` parameter, session states are restored before replaying. Use `LEVEL2` for Application Continuity and `AUTO` for Transparent Application Continuity to restore session states using database templates. Use `LEVEL1` for Application Continuity or Transparent Application Continuity to restore session states using the earlier Oracle Database 19c modifiable parameter transport. Use wallets to allow restoring server-side session states transparently. • `AUTO`: Use this value to enable database templates for Transparent Application Continuity. • `LEVEL1`: Use this value to enable behavior of `FAILOVER_RESTORE=AUTO` with client based restore for Oracle Database 21c and earlier releases. • `LEVEL2`: Use this value to enable database templates for Transparent Application Continuity. For OCI applications using TAF or Application Continuity, setting `-failover_restore` to `LEVEL1` restores the current state. If the current state differs from the initial state, then a TAF callback is required. This restriction applies only to OCI.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-91 (Cont.) srvctl add service Command Parameters

Parameter	Description
`-template_timeout` *`template_timeout`*	Specify the time, in seconds, for template timeout. For Application Continuity, `TEMPLATE_TIMEOUT` parameter is used with `FAILOVER_RESTORE` values `LEVEL2` and `AUTO` in Oracle AI Database 26ai. When a template expires, the session is not restored when `FAILOVER_RESTORE` is set to `AUTO` or `LEVEL2`. You are recommended to set higher values so your pools are able to failover following the overnight maintenance. The default value is `RETENTION` (1 day). If `TEMPLATE_TIMEOUT` is set to 0, then the template timeout is turned off.
`-failback {YES | NO}`	If a service fails over to an available instance after the list of preferred instances was exhausted, then, if this parameter is set to YES, the service automatically fails back to a preferred instance when one becomes available.
`-edition` *`edition_name`*	The initial session edition of the service. When an edition is specified for a service, all subsequent connections that specify the service use this edition as the initial session edition. However, if a session connection specifies a different edition, then the edition specified in the session connection is used for the initial session edition. SRVCTL does not validate the specified edition name. During connection, the connect user must have USE privilege on the specified edition. If the edition does not exist or if the connect user does not have USE privilege on the specified edition, then an error is raised.
`-pdb` *`pluggable_database`*	The name of the Pluggable Database (PDB). You can specify a PDB property when you create or modify a service. The PDB property associates the service with the specified PDB. You can view the PDB property for a service by querying the `ALL_SERVICES` data dictionary view or, when using the SRVCTL utility, by running the `srvctl config service` command. **Note:** Starting with Oracle Database 21c, before using the `-pdb` option with the `srvctl add service` command, you must have previously added the PDB resource to Oracle Clusterware using the `srvctl add pdb` command.
`-global {TRUE | FALSE}`	Indicates whether this is a Global Data Services service. **Note:** This parameter can only be used with Global Data Services.
`-maxlag` *`maximum_lag_time`*	Maximum replication lag time in seconds for a global service. Must be a non-negative integer. The default value is `ANY`. You must also specify the `-global` option.
`-sql_translation_profile` *`profile_name`*	Use this parameter to specify a SQL translation profile for a service that you are adding after you have migrated applications from a non-Oracle database to an Oracle AI Database. This parameter corresponds to the SQL translation profile parameter in the `DBMS_SERVICE` service attribute. **Notes:** • Before using the SQL translation feature, you must migrate all server-side application objects and data to the Oracle AI Database. • Use the `srvctl config service` command to display the SQL translation profile.
`-commit_outcome {TRUE | FALSE}`	Enable Transaction Guard; when set to `TRUE`, the commit outcome for a transaction is accessible after the transaction's session fails due to a recoverable outage.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-91 (Cont.) srvctl add service Command Parameters

Parameter	Description
`-retention` *`retention_time`*	If `-commit_outcome` is set to `TRUE`, then this parameter determines the amount of time (in seconds) that the commit outcome is retained in the database.
`-replay_init_time` *`replay_initialization_time`*	For Application Continuity, this parameter specifies the difference between the time, in seconds, of original processing of the first operation of a request and the time that the replay is ready to start after a successful reconnect. Application Continuity will not replay after the specified amount of time has passed. This parameter is intended to avoid the unintentional processing of a transaction when a system is recovered after a long period. The default is 5 minutes (`300`). The maximum value is 24 hours (`86400`). If the `-failover_type` parameter is not set to `TRANSACTION`, then you cannot use this parameter.
`-session_state { DYNAMIC | AUTO}`	For Application Continuity; this parameter describes how the non-transactional session state is changed by the application within a request. Examples of session state are NLS settings, optimizer preferences, event settings, PL/SQL global variables, and temporary tables. For Transparent Application Continuity, you can set `session_state` to `AUTO`. Session state is tracked automatically. This parameter is considered only if `-failovertype` is set to `AUTO` or `TRANSACTION` for Application Continuity or `AUTO` for Transparent Application Continuity. • If `failover_type` is set to `TRANSACTION`, then Oracle recommends a value of `DYNAMIC` for `session_state`. • If `failover_type` is set to `AUTO`, then you can set `session_state` to `AUTO`. **Note** Use `AUTO` to enable Transparent Application Failover (TAF) style cursors with TAC. • If `failover_type` is set to any value other than `TRANSACTION` or `AUTO`, then the value of `session_state` is not set. If non-transactional values change after the request starts, then set this parameter to either `DYNAMIC` or `AUTO`. Most applications should use `DYNAMIC` or `AUTO` mode.
`-importfile` *`file_name`*	Re-create the resources using the resource attributes specified in the file.
`-gsmflags` *`gsm_flags`*	Set locality and region failover values for a global service. You must also specify the `-global` option.
`-tablefamilyid` *`table_family_id`*	Set table family ID for a service. See "Shared Table Family" for more information.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-91 (Cont.) srvctl add service Command Parameters

Parameter	Description
`-drain_timeout` *timeout*	Specify the time, in seconds, allowed for resource draining to be completed. Accepted values are an empty string (""), 0, or any positive integer. The default value is an empty string, which means that this parameter is not set. If it is set to 0, then draining occurs, immediately. The draining period is intended for planned maintenance operations. During the draining period, all current client requests are processed, but new requests are not accepted. When set on the service this value is used when the command line value is not set. **Note:** If the `-drain_timeout` value is not specified, then the maximum configured drain timeout of all the running services of the database on the specified nodes is used.
`-stopoption {NONE| IMMEDIATE| TRANSACTIONAL}`	Specify the default method of stopping the service. When set on the service, this value is used if you do not include the `-stopoption` parameter in other SRVCTL commands. If you do not provide a value, then the default option `NONE` is used. • `IMMEDIATE` permits sessions to drain before the service is stopped. Sessions that do not drain are terminated when the time limit specified by `-drain_timeout` is reached. • If you specify `TRANSACTIONAL`, then sessions are terminated as soon as they commit. The service is stopped when the time limit specified by `-drain_timeout` is reached and any remaining sessions are terminated. • If you specify `NONE`, then no sessions are terminated.
`-css_critical {yes | no}`	You can add weight to a service by setting this parameter to YES. In the event of a node failure within the cluster, Oracle Clusterware will evict the node with the least amount of weight, ensuring that critical services remain available.
`-update {-preferred` *new_preferred_instance* `| -available` *new_available_instance* `}`	Add a new preferred or available instance to the service configuration. `-preferred` specifies the name of the instance to add to the list of preferred instances for the service. `-available` specifies the name of the instance to add to the list of available instances for the service.
`-force`	Force the add operation even though a listener is not configured for a network.
`-verbose`	Display verbose output.

Usage Notes

This command does not accept placement parameters for Oracle RAC One Node databases.

Examples

Use the following examples to create your services:

Basic Service

```
$ srvctl add service -db mydb -service MYSERVICE –preferred inst1 -available
inst2
-pdb mypdb -notification TRUE -drain_timeout 300 -stopoption IMMEDIATE -role
PRIMARY
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Transparent Application Continuity Service

```
$ srvctl add service -db mydb -service TACSERVICE -pdb mypdb –preferred inst1
-available inst2
-failover_restore AUTO -commit_outcome TRUE -failovertype AUTO -
replay_init_time 600 -retention 86400
-notification TRUE -drain_timeout 300 -stopoption IMMEDIATE -role PRIMARY
```

Application Continuity Service

```
$ srvctl add service -db mydb -service ACSERVICE -pdb mypdb -preferred inst1 -
available inst2
-failover_restore LEVEL1 -commit_outcome TRUE -failovertype TRANSACTION -
session_state dynamic
-replay_init_time 600 -retention 86400 -notification TRUE -drain_timeout 300 -
stopoption IMMEDIATE -role PRIMARY
```

TAF Select Plus Service

```
$ srvctl add service -db mydb -service TAFSERVICE -pdb mypdb -preferred inst1
-available inst2
-failover_restore LEVEL1 -commit_outcome TRUE -failovertype SELECT -
notification TRUE -drain_timeout 300
-stopoption TRANSACTIONAL -role PRIMARY
```

Service Attribute Validation Matrix

The following table describes service attribute dependency for `FAILOVER_TYPE` and `COMMIT_OUTCOME` driver attributes.

Table A-92 Service Attribute Dependency Validation

Parent Attribute	Value	Dependent Attribute	Valid Value
failover_type	AUTO	failover_restore	AUTO | LEVEL1
failover_type	AUTO	session_state_consistency	AUTO
failover_type	AUTO	commit_outcome	TRUE
failover_type	TRANSACTION	failover_restore	LEVEL1 | NONE | LEVEL2
failover_type	TRANSACTION	session_state_consistency	DYNAMIC
failover_type	TRANSACTION	commit_outcome	TRUE
failover_type	SELECT	failover_restore	LEVEL1 | NONE
failover_type	SELECT	failover_method	BASIC | NONE
failover_type	SELECT	commit_outcome	TRUE | FALSE
failover_type	SESSION	failover_restore	NONE
failover_type	SESSION	failover_method	BASIC | NONE
failover_type	SESSION	commit_outcome	FALSE
failover_type	NONE	failover_restore	NONE
failover_type	NONE	failover_method	NONE
failover_type	NONE	commit_outcome	FALSE | TRUE
commit_outcome	FALSE	commit_outcome_fastpath	FALSE

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-92 (Cont.) Service Attribute Dependency Validation

Parent Attribute	Value	Dependent Attribute	Valid Value
commit_outcome	TRUE	commit_outcome_fastpath	TRUE | FALSE

Related Topics

- Sharded Table Family
- CREATE_SERVICE Procedure

srvctl config service

Displays the configuration for a service.

Syntax

```
srvctl config service {-db db_unique_name [-service service_name | -pdb
pdb_name [-brief]]
   | [-db db_unique_name]} [-verbose]
```

Parameters

Table A-93 srvctl config service Command Parameters

Parameter	Description
-db *db_unique_name*	Unique name for the database.
-service *service_name*	Optionally, you can specify the name of a service. If you do not use this parameter, then SRVCTL displays the configuration information for all services configured for the database.
-pdb *pdb_name*	Name of the PDB for which you want to show configured services.
-verbose	Displays verbose output.

Usage Notes

The `srvctl config service` command shows exactly the string value you specified for the edition using the `srvctl add | modify service` commands. If you specified the edition in upper case, then `srvctl config service` displays upper case. If it is surrounded by double quotation marks (""), then the command displays the double quotation marks. Otherwise, the command displays an empty string.

Examples

This command returns information similar to the following for a database:

```
$ srvctl config service -db crm -service webapps
Service name: webapps
Service is enabled
Cardinality: 1
Disconnect: false
Service role: PRIMARY
Management policy: AUTOMATIC
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

```
DTP transaction: false
AQ HA notifications: false
Failover type: NONE
Failover method: NONE
TAF failover retries: 0
TAF failover delay: 0
Connection Load Balancing Goal: LONG
Runtime Load Balancing Goal: NONE
TAF policy specification: NONE
Preferred instances: crm_1
Available instances:
Edition: "my Edition"
```

Service configuration for administrator-managed Oracle RAC One Node databases displays the one instance as preferred.

srvctl disable service

Disables a service.
Disabling an entire service affects all of the instances, disabling each one. If the entire service is already disabled, then running this command on the entire service 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

```
srvctl disable service -db db_unique_name -services "service_name_list"
   [-instance instance_name]
```

Parameters

Table A-94 srvctl disable service Command Parameters

Parameter	Description
`-db db_unique_name`	Specify a unique name for the database for which you want to disable the service.
`-services "service_name_list"`	Specify a comma-delimited list of service names enclosed in double quotation marks (""), or a single service name, that you want to disable.
`-instance instance_name`	Optionally, you can specify the name of the instance for which you want to disable the service. **Notes:** • Use this parameter with administrator-managed databases. • You can only use this parameter with Oracle Clusterware and Oracle RAC.

Examples

The following example globally disables two services for the `CRM` database:

```
$ srvctl disable service -db crm -service "crm,marketing"
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

The following example disables a service for the CRM database that is running on the CRM1 instance, resulting in the service still being available for the database, but on one less instance:

```
$ srvctl disable service -db crm -service crm -instance crm1
```

srvctl enable service

Enables a service for Oracle Clusterware.
Enabling an entire service also affects the enabling of the service over all of the instances by enabling the service on each one. If the entire service is already enabled, then running this command does not affect all of the instances and enable them but, instead, returns an error. Therefore, you cannot always use the entire set of service operations to manipulate the service indicators for each instance.

Syntax

```
srvctl enable service -db db_unique_name -service "service_name_list"
   [-instance instance_name | -node node_name]
```

Parameters

Table A-95 srvctl enable service Command Parameters

Parameter	Description
`-db db_unique_name`	Specify a unique name for the database for which you want to enable the service.
`-service "service_name_list"`	Specify a single service name or a comma-delimited list of service names enclosed in double quotation marks ("") that you want to enable.
`-instance instance_name`	Optionally, you can use this parameter to specify the name of the database instance where you want the service to run. **Notes:** • Use this parameter for administrator-managed databases. • You can only use this parameter with Oracle Clusterware and Oracle RAC.

Examples

The following example globally enables a service:

```
$ srvctl enable service -db crm -service crm
```

The following example enables a service to use a preferred instance:

```
$srvctl enable service -db crm -service crm -instance crm1
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl modify service

Modifies a service configuration.

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

This command supports some online modifications to the service, such as:

- Moving a service member from one instance to another
- Performing online changes to service attributes from `DBMS_SERVICE` (for example, failover delay, runtime load balancing goal, and so on)
- Adding a new preferred or available instance
- Removing preferred or available instances for a service

> **Caution**
>
> 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 Parameters

Use one of the following forms of the `srvctl modify service` command, depending on the task you want to perform, with the specified syntax:

To move a service from one instance to another:

```
srvctl modify service -db db_unique_name -service service_name
      -oldinst old_instance_name -newinst new_instance_name [-force]
```

> **Note**
>
> This form of the command is only available with Oracle Clusterware.

Table A-96 srvctl modify service Parameters for Moving a Service

Parameter	Description
`-db db_unique_name`	Specify the unique name for the database.
`-service service_name`	Specify a service name. If you do not specify a service name, then SRVCTL moves all services.
`-oldinst old_instance_name`	Specify the name of the instance from which you want to move the service.
`-newinst new_instance_name`	Specify the name of the instance to which you want to move the service.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-96 (Cont.) srvctl modify service Parameters for Moving a Service

Parameter	Description
`-force`	Force the modify operation, stopping the service on some nodes, as necessary.

To change an available instance to a preferred instance for a service:

```
srvctl modify service -db db_unique_name -service service_name
      -available avail_inst_name [-failback {YES|NO}] -toprefer [-force]
```

> **Note**
>
> This form of the command is only available with Oracle Clusterware and does not accept placement parameters for Oracle RAC One Node databases. This command also does not move or otherwise disconnect the service but only modifies the service attributes.

Table A-97 srvctl modify service Parameters for Changing to a Preferred Instance

Parameter	Description
`-db db_unique_name`	Specify the unique name for the database.
`-service service_name`	Specify the name of the service you want to modify.
`-available available_inst_name`	Specify the name of the available instance you want to change.
`-failback {YES|NO}`	If a service fails over to an available instance after the list of preferred instances was exhausted, then, if this parameter is set to YES, then the service automatically fails back to a preferred instance when one becomes available.
`-toprefer`	Specify this parameter to change the instance status to preferred.
`-force`	Force the modify operation. For planned operations, the user experience is best if using an Oracle Connection Pool with FAN. The FAN planned event causes the Oracle pool to drain the requests with no interruption to the users.

To change the available and preferred status for multiple instances:

```
srvctl modify service -db db_unique_name -service service_name
     -modifyconfig -preferred "preferred_list" [-available "available_list"]
     [-force]
```

> **Note**
>
> This form of the command is only available with Oracle Clusterware and does not accept placement parameters for Oracle RAC One Node databases. This command also does not move or otherwise disconnect the service but only modifies the service attributes.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-98 srvctl modify service Parameters for Changing Status of Multiple Instances

Parameter	Description
`-db db_unique_name`	Specify the unique name for the database.
`-service service_name`	Specify the name of the service you want to modify.
`-modifyconfig`	This parameter directs SRVCTL to use only the instances named for this service (unnamed instances already assigned to the service are removed).
`-preferred "preferred_instance_list"`	Specify a comma-delimited list of preferred instances enclosed within double quotation marks ("").
`-available "available_instance_list"`	Specify a comma-delimited list of available instances enclosed within double quotation marks ("").
`-force`	Force the modify operation. For planned operations, the user experience is best if using an Oracle Connection Pool with FAN. The FAN planned event causes the connection pool to drain the requests with no interruption to the users.

To modify other service attributes or to modify a service for Oracle Clusterware:

```
srvctl modify service -db db_unique_name -service service_name
 [-true_cache_service true_cache_service_list]
 [-cardinality {UNIFORM|SINGLETON|DUPLEX}] [-tafpolicy {BASIC|NONE}]
 [-role "[PRIMARY][,PHYSICAL_STANDBY][,LOGICAL_STANDBY][,SNAPSHOT_STANDBY]"]
 [-policy {AUTOMATIC|MANUAL}] [-notification {TRUE|FALSE}] [-dtp {TRUE|FALSE}]
 [-clbgoal {SHORT|LONG}] [-rlbgoal {NONE | SMART_CONN | SERVICE_TIME |
THROUGHPUT}]
 [-resetstate {NONE|LEVEL1}] [-failovertype {NONE|SESSION|SELECT|TRANSACTION|
AUTO}]
 [-failoverretry failover_retries] [-failoverdelay failover_delay]
 [-failover_restore [NONE|LEVEL2|LEVEL1|AUTO]] [-template_timeout
template_timeout] [-failback {YES|NO}]
 [-edition edition_name] [-pdb pluggable_database]
 [-sql_translation_profile profile_name] [-commit_outcome {TRUE|FALSE}]
 [-retention retention_time] [-replay_init_time replay_initiation_time]
 [-session_state {DYNAMIC|AUTO}] [-maxlag max_lag_time]
 [-gsmflags gsm_flags] [-tablefamilyid table_family_id]
 [-drain_timeout timeout] [-stopoption {NONE|IMMEDIATE|TRANSACTIONAL}]
 [-global_override] [-css_critical {YES | NO}] -hubsvc <hub_service>]} [-
verbose] [-force]
```

Table A-99 srvctl modify service Parameters

Parameter	Description
`-db db_unique_name`	The unique name for the database.
`-service service_name`	The name of the service you want to modify.
`-true_cache_service true_cache_service_list`	Specify this parameter to associate a primary service with the True Cache service.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-99 (Cont.) srvctl modify service Parameters

Parameter	Description
`-cardinality {UNIFORM|SINGLETON|DUPLEX}`	The cardinality of the service, which can be one of the following: • `UNIFORM` – offered on all instances or PDBs in the database • `SINGLETON` – runs on only one instance or PDB at a time • `DUPLEX` – runs on two instances or PDBs at a time **Note** This parameter can be used only with Oracle RAC.
`-tafpolicy {BASIC|NONE}`	Transparent Application Failover (TAF) policy specification.
`-role "[PRIMARY][,PHYSICAL_STANDBY][,LOGICAL_STANDBY][,SNAPSHOT_STANDBY]"`	The database modes for which the service should be started automatically. You can specify one or more roles in a comma-delimited list. **Note:** The `-role` parameter is only used at database startup and by the Oracle Data Guard Broker. All manual service startup must specify the name of the service to be started by the user.
`-policy {AUTOMATIC|MANUAL}`	The service management policy. If `AUTOMATIC` (the default), then the service is automatically started upon restart of the database, either by a planned restart (with SRVCTL) or after a failure. Automatic restart is also subject to the service role, however (the `-role` parameter). If `MANUAL`, then the service is never automatically restarted upon planned restart of the database (with SRVCTL). A `MANUAL` setting does not prevent Oracle Clusterware from monitoring the service when it is running and restarting it if a failure occurs.
`-notification {TRUE|FALSE}`	Use `TRUE` to enable Fast Application Notification (FAN) for Oracle Call Interface (OCI) connections.
`-dtp {TRUE|FALSE}`	Use `TRUE` to enable Distributed Transaction Processing for this service. This ensures that the service is offered at exactly one instance at a time for XA affinity. **Note:** This parameter can be used only with Oracle RAC.
`-clbgoal {SHORT|LONG}`	Connection Load Balancing Goal. Set to `SHORT` if using runtime load balancing, or set to `LONG` for long running connections, such as batch jobs, that you want balanced by the number of sessions per node for the service.
`-rlbgoal {NONE | SMART_CONN | SERVICE_TIME | THROUGHPUT}`	Runtime Load Balancing Goal (for the Load Balancing Advisory). Set this parameter to `SMART_CONN` to enable Smart Connection Rebalance. Set this parameter to `SERVICE_TIME` to balance connections by response time. Set this parameter to `THROUGHPUT` to balance connections by throughput.
`-resetstate {NONE | LEVEL1}`	Reset state in a session to clean values. If set to `NONE`, then session state is not cleaned. If set to `LEVEL1`, then session states for PL/SQL, temporary tables, session duration jobs, and cursors in `FETCH` are cleaned at the end of the request. Use `RESET_STATE` for pooled applications to prevent session state leaking to later re-usages.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-99 (Cont.) srvctl modify service Parameters

Parameter	Description
`-failovertype {NONE|SESSION|SELECT|TRANSACTION|AUTO}`	Use this parameter to set the failover type. To enable Application Continuity, set this parameter to `TRANSACTION`. To enable Transparent Application Continuity, set this parameter to `AUTO`. To enable TAF, set this parameter to `SELECT` or `SESSION`.
`-template_timeout` *`template_timeout`*	Specify the time, in seconds, for template timeout. For Application Continuity, `TEMPLATE_TIMEOUT` parameter is used with `FAILOVER_RESTORE` values `LEVEL2` and `AUTO` in Oracle AI Database 26ai. When a template expires, the session is not restored when `FAILOVER_RESTORE` is set to `AUTO` or `LEVEL2`. If `TEMPLATE_TIMEOUT` is set to 0, then the template timeout is turned off.
`-failoverretry` *`failover_retries`*	For Application Continuity and TAF, specify the number of attempts to connect after an incident.
`-failoverdelay` *`failover_delay`*	For Application Continuity and TAF, specify the time delay (in seconds) between reconnect attempts per incident at failover.
`-failover_restore {NONE|LEVEL2|LEVEL1|AUTO}`	For Application Continuity, when you set the `-failover_restore` parameter, session states are restored before replaying. Use `LEVEL2` for ODP.NET and Java with Application Continuity to restore the initial state. Set this parameter to `LEVEL1` or `AUTO` to enable Transparent Application Continuity to restore session states. • `AUTO`: Use this value to enable database templates for Application Continuity. • `LEVEL1`: Use this value to enable behavior of `FAILOVER_RESTORE=AUTO` with client based restore for Oracle Database 21c and earlier releases. For OCI applications using TAF or Application Continuity, setting `-failover_restore` to `LEVEL1` restores the current state. If the current state differs from the initial state, then a TAF callback is required. This restriction applies only to OCI.
`-failback {YES | NO}`	If a service fails over to an available instance after the list of preferred instances was exhausted, then, if this parameter is set to YES, the service automatically fails back to a preferred instance when one becomes available.
`-edition` *`edition_name`*	The initial session edition of the service. When an edition is specified for a service, all subsequent connections that specify the service use this edition as the initial session edition. However, if a session connection specifies a different edition, then the edition specified in the session connection is used for the initial session edition. SRVCTL does not validate the specified edition name. During connection, the connect user must have USE privilege on the specified edition. If the edition does not exist or if the connect user does not have USE privilege on the specified edition, then an error is raised.
`-pdb` *`pluggable_database`*	The name of a Pluggable Database (PDB). **Note:** You can specify a PDB property when you create or modify a service. The PDB property associates the service with the specified PDB. You can view the PDB property for a service by querying the ALL_SERVICES data dictionary view or, when using the SRVCTL utility, by running the `srvctl config service` command. When create or modify a service with the specified PDB, SRVCTL does not check if the PDB exists. Before running this command, you must ensure that the PDB exists.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-99 (Cont.) srvctl modify service Parameters

Parameter	Description
`-sql_translation_profile` *profile_name*	Use this parameter to specify a SQL translation profile for a service that you are modifying after you have migrated applications from a non-Oracle database to an Oracle AI Database. If you want to set the SQL translation profile to a NULL value, then you must enter an empty string after the `-p` flag. **Note:** Before using the SQL translation feature, you must migrate all server-side application objects and data to the Oracle AI Database.
`-commit_outcome {TRUE|FALSE}`	Enable Transaction Guard; when set to `TRUE`, the commit outcome for a transaction is accessible after the transaction's session fails due to a recoverable outage.
`-retention` *retention_time*	For Transaction Guard (with the `-commit_outcome` parameter set to `TRUE`); this parameter determines the amount of time (in seconds) that the commit outcome is retained in the database.
`-replay_init_time` *replay_initiation_time*	For Application Continuity; this parameter specifies the time, in seconds, from when the original request started. Application Continuity will not replay after the specified amount of time has passed. This attribute avoids the unintentional replay of a request when a system is recovered after a long period. The default value is 300 (5 minutes).
`-session_state {DYNAMIC|AUTO}`	For Application Continuity; this parameter describes how the non-transactional session state is changed by the application within a request. Examples of session state are NLS settings, optimizer preferences, event settings, PL/SQL global variables, and temporary tables. For Transparent Application Continuity, you can set `session_state` to `AUTO`. Session state is tracked automatically. This parameter is considered only if `-failovertype` is set to `AUTO` or `TRANSACTION` for Application Continuity or `AUTO` for Transparent Application Continuity. • If `failover_type` is set to `TRANSACTION`, then Oracle recommends a value of `DYNAMIC` for `session_state`. • If `failover_type` is set to `AUTO`, then you can set `session_state` to `AUTO`. **Note** Use `AUTO` to enable Transparent Application Failover (TAF) style cursors with TAC. • If `failover_type` is set to any value other than `TRANSACTION` or `AUTO`, then the value of `session_state` is not set. Oracle recommends a value of `AUTO` or `DYNAMIC` for most applications. If you are unsure which value to use, or if you can customize the application, then use `DYNAMIC`.
`-maxlag` *maximum_lag_time*	Maximum replication lag time in seconds for a global service. Must be a non-negative integer. The default value is `ANY`. You must also specify the `-global` option.
`-gsmflags` *gsm_flags*	Set locality and region failover values for a global service. You must also specify the `-global` option.
`-tablefamilyid` *table_family_id*	Set table family ID for a service. See Sharded Table Family for more information.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-99 (Cont.) srvctl modify service Parameters

Parameter	Description
`-drain_timeout` *timeout*	Specify the time, in seconds, allowed for resource draining to be completed. Accepted values are an empty string (""), 0, or any positive integer. The default value is an empty string, which means that this parameter is not set. If it is set to 0, then draining occurs, immediately. The draining period is intended for planned maintenance operations. During the draining period, all current client requests are processed, but new requests are not accepted. When set on the service this value is used when the command line value is not set. **Note:** If the `-drain_timeout` value is not specified, then the maximum configured drain timeout of all the running services of the database on the specified nodes is used.
`-stopoption {NONE| IMMEDIATE| TRANSACTIONAL}`	Specify the method of stopping the service. If this attribute was previously set for the service, then that value is used as the default value if you do not include the `-stopoption` parameter in your command. Otherwise, the default is `NONE`. • `IMMEDIATE` permits sessions to drain before the service is stopped. Sessions that do not drain are terminated when the time limit specified by `-drain_timeout` is reached. • If you specify `TRANSACTIONAL`, then sessions are terminated as soon as they commit. The service is stopped when the time limit specified by `-drain_timeout` is reached and any remaining sessions are terminated. • If you specify `NONE`, then no sessions are terminated. **Note:** You must use the `-stopoption` parameter with the `-force` parameter.
`-global_override`	Override value to modify the global service attributes. Use this parameter with the `-role`, `-policy`, `-notification`, `-failovertype`, `-failovermethod`, `-failoverdelay`, `-failoverretry` and `-edition` parameters.
`-css_critical {yes | no}`	You can add weight to a service by setting this parameter to YES. In the event of a node failure within the cluster, Oracle Clusterware will evict the node with the least amount of weight, ensuring that critical services remain available.
`–verbose`	Display verbose output.
`-force`	Force the modify operation, stopping the service on some nodes as necessary.

Usage Notes

- When performing online changes to service attributes (for example, failover delay, Runtime Load Balancing Goal, and so on), the changes take effect only when the service is next (re)started.
- When a service configuration is modified so that a new preferred or available instance is added, the running state of the existing service is not affected. However, the newly added instances will not automatically provide the service, until a `srvctl start service` command is issued.
- When there are available instances for the service, and the service configuration is modified so that a preferred or available instance is removed, the running state of the service may change unpredictably:

Copyright © 1999, 2025, Oracle and/or its affiliates.

- The service is stopped and then removed on some instances according to the new service configuration.
- The service may be running on some instances that are being removed from the service configuration.
- These services will be relocated to the next *free* instance in the new service configuration.

Because of these considerations, when the online service is being modified, users may experience a brief service outage on some instances even if the instances are not being removed. Or users may experience a brief service outage on instances that are being removed from the service.

Examples

An example of moving a service member from one instance to another is:

```
$ srvctl modify service -db crm -service crm -oldinst crm1 -newinst crm2
```

An example of changing an available instance to a preferred instance is:

```
$ srvctl modify service -db crm -service crm -available crm1 -toprefer
```

The following command exchanges a preferred and available instance:

```
$ srvctl modify service -db crm -service crm -modifyconfig -preferred "crm1" \
-available "crm2"
```

Related Topics

- *Oracle Data Guard Broker*
- *Oracle AI Database SQL Translation and Migration Guide*

srvctl predict service

Predicts the consequences of service failure.

Syntax

```
srvctl predict service -db db_unique_name -service service_name [-verbose]
```

Parameters

Table A-100 srvctl predict service Command Parameters

Parameter	Description
`-db db_unique_name`	Specify the unique name for the database on which the service operates that you want to check.
`-service service_name`	Specify a single service name or a comma-delimited list of service names enclosed in double quotation marks ("") that you want to check.
`–verbose`	Optionally, you can use this parameter to display detailed output.

Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl relocate service

Temporarily relocates the specified service names from one specified instance to another specified instance.

This command works on only one source instance and one target instance at a time, relocating a service or all services from a single source instance to a single target instance.

Syntax

To relocate a service from one instance to another instance or from one node to another node:

```
srvctl relocate service -db db_unique_name [-service service_name | -pdb
pluggable_database]
  {-oldinst old_inst_name [-newinst new_inst_name] | -currentnode
current_node [-targetnode target_node]}
  [-drain_timeout timeout] [-wait {YES | NO}] [-pq] [-force [-noreplay]] [-
stopoption stop_option]] [-eval] [-verbose]
```

Parameters

Table A-101 srvctl relocate service Command Parameters

Parameter	Description
`-db db_unique_name`	The unique name for the database on which the service currently runs.
`-service service_name`	The name of the service you want to relocate. If you do not specify any services, then all services that can be relocated, are relocated. Those that cannot be relocated remain in place.
`-pdb pluggable_database`	The name of the Pluggable Database in which the service you want to relocate is currently running.
`-oldinst old_inst_name`	The name of the instance from which you are relocating the service.
`-newinst new_inst_name`	The name of the instance to which you are relocating the service. This parameter is optional. If you do not specify an instance, then Oracle Clusterware chooses a new one. The policy-managed database deployment option is desupported in Oracle AI Database.
`-currentnode source_node`	Name of the node where the service is currently running.
`-targetnode target_node`	Name of node where the service is to be relocated. If you do not specify a target node, then Oracle Clusterware chooses a new location. The policy-managed database deployment option is desupported in Oracle AI Database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-101 (Cont.) srvctl relocate service Command Parameters

Parameter	Description
`-drain_timeout` *timeout*	Specify the time, in seconds, allowed for resource draining to be completed. Accepted values are an empty string (""), 0, or any positive integer. The default value is an empty string, which means that this parameter is not set. If it is set to 0, then draining occurs, immediately. The draining period is intended for planned maintenance operations. During the draining period, all current client requests are processed, but new requests are not accepted. When set on the service this value is used when the command line value is not set. **Note:** If the `-drain_timeout` value is not specified, then the maximum configured drain timeout of all the running services of the database on the specified nodes is used.
`-wait YES | NO`	Choose YES to wait until service draining is completed on the node from which you are relocating the service.
`-stopoption` *option*	Specify the method of stopping the service. If this attribute was previously set for the service, then that value is used as the default value if you do not include the `-stopoption` parameter in your command. Otherwise, the default is `NONE`. • `IMMEDIATE` permits sessions to drain before the service is stopped. Sessions that do not drain are terminated when the time limit specified by `-drain_timeout` is reached. • If you specify `TRANSACTIONAL`, then sessions are terminated as soon as they commit. The service is stopped when the time limit specified by `-drain_timeout` is reached and any remaining sessions are terminated. • If you specify `NONE`, then no sessions are terminated. **Note:** You must use the `-stopoption` parameter with the `-force` parameter.
`–pq`	Performs the action on a parallel query service.
`–force [-noreplay]`	Disconnect all sessions during stop or relocate service operations. Optionally, you can specify the `-noreplay` parameter if you do not want Application Continuity to replay in-flight transactions after a session is terminated during `relocate service` operations.
`–eval`	Use this parameter to hypothetically evaluate the impact of the command on the system.
`–verbose`	Verbose output.

Example

To temporarily relocate a named service member for the `crm` service from the database instance `crm1` to the database instance `crm3`:

```
$ srvctl relocate service -db crm -service crm -oldinst crm1 -newinst crm3
```

Related Topics

- Database Shutdown

Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl remove service

Removes the service from Oracle Clusterware management.

Syntax

```
srvctl remove service -db db_unique_name {-service service_name |
   -pdb pdb_name}[-global_override] [-force]
```

Parameters

Table A-102 srvctl remove service Command Parameters

Parameter	Description
`-db db_unique_name`	The unique name of the database or Container Database (CDB).
`-service service_name`	The name of the service you want to remove. You must specify either the service name or the Pluggable Database (PDB) name.
`-pdb pdb_name`	The name of the PDB that offers the service. You must specify either the service name or the PDB name.
`-global_override`	Indicates you are modifying global services. SRVCTL ignores this parameter for a non-global service.
`-force`	Ignore any dependencies when removing the service.

Usage Notes

If you use the `-pdb` option with this command, then SRVCTL removes all service resources for the specified PDB, but does not remove the PDB resource. If you specify the PDB service name using the `-service` option, then you do not have to also include the `-pdb` option because the PDB service name is unique within the Container Database (CDB).

Examples

This following example removes the `sales` service from all instances of the clustered database named `crm`:

```
$ srvctl remove service -db crm -service sales
```

The following example removes the services from the `crmeast` PDB:

```
$ srvctl remove service -db crm -pdb crmeast
```

srvctl start service

Starts a service or multiple services on a database, Pluggable Database (PDB) , or instance.

Syntax

```
srvctl start service [-db db_unique_name] [-service "service_name_list" [-
role] [-rf] | -pdb pluggable_database]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

```
  [-instance instance_name] [-global_override] [-role] [-startoption
start_options] [-eval] [-verbose]
```

Parameters

Parameter	Description
`-db db_unique_name`	Specify a unique name for the database.
`-service "service_list"`	Specify a service name or a comma-delimited list of service names enclosed in double quotation marks (""). If you do not include this parameter, then SRVCTL starts all of the services for the specified database. **Note:** All manual service startup must specify the name of the service to be started by the user.
`-role`	Start services for the local database role.
`-rf`	Specify this parameter to perform the start action on reader farm service.
`-pdb pluggable_database`	Specify the name of a Pluggable Database. Optionally, you can specify either the name of a node or the name of an instance to restrict the starting of services to that particular object on the Pluggable Database.
`-instance instance_name`	Specify the name of an instance where the services reside that you want to start. Use this parameter for administrator-managed databases.
`-global_override`	Override value to operate on a global service. Use this parameter only with global services; this parameter is ignored if specified for a non-global service.
`-startoption start_options`	Specify startup options used when service startup requires starting a database instance. Options include `OPEN`, `MOUNT`, and `NOMOUNT`. **Note:** For multi-word startup options, such as `read only` and `read write`, separate the words with a space and enclose in double quotation marks (""). For example, `"read only"`.
`-role`	Start services for the local database role.
`–verbose`	Display verbose output.

Usage Notes

- The `srvctl start service` command will fail if you attempt to start a service that is already running.
- 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.
- The `srvctl start service` without the `-role` parameter or a list of services, will start all the services configured for the database, regardless of the actual database role.
- You can move a service or change the status of a service on an instance with the `srvctl modify service` and `srvctl relocate service` commands.

Examples

The following example starts all services on a specific database:

```
$ srvctl start service -db myDB
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

The following examples start a list of services (optionally restricted to a parallel query services in the latter example) regardless of the Pluggable Database on which they may reside:

```
$ srvctl start service -db myDB -service "myServ01,myServ02"
$ srvctl start service -db myDB -service "myServ01,myServ02" -pq
```

The following examples start all services on a given Pluggable Database, optionally restricted to a single node or a single instance in the latter two examples, repectively:

```
$ srvctl start service -db myDB -pdb myPDB1
$ srvctl start service -db myDB -pdb myPDB1 -node myRACNode01
$ srvctl start service -db myDB -pdb myPDB1 -instance myDB01
```

The following example starts all services, for a given database, on a given instance (for all Pluggable Databases):

```
$ srvctl start service -db myDB -instance myDB01
```

The following example start all services for a given database on a given node (for all Pluggable Databases):

```
$ srvctl start service -db myDB -node myRACNode01
```

The following examples start a list of services on a given node or given instance:

```
$ srvctl start service -db myDB -service "myService01,myService02" -node
myRACNode01
$ srvctl start service -db myDB -service "myService01,myService02" -instance
myDB01
```

srvctl status service

Displays the status of a service.
For Oracle RAC One Node databases, if there is an online database relocation in process, then this command displays the source and destination nodes and the status of the relocation, whether it is active or failed.

Syntax

```
srvctl status service {-db db_unique_name [-service "service_name_list" | -
pdb pdb_name]
  | [-db db_unique_name]} [-force] [-verbose]
```

Parameters

Optionally, you can use this parameter to include disabled applications.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-103 srvctl status service Command Parameters

Parameter	Description
`-db db_unique_name`	Specify the unique name of the database on which the service operates for which you want to check the status.
`-pdb pdb_name`	Specify the name of the PDB on which the service operates for which you want to check the status.
`-service "service_name_list"`	Optionally, you can specify a comma-delimited list of service names for which you want to check status. If you do not use this parameter, then SRVCTL lists the status of all services for the specified database.
`–force`	Optionally, you can use this parameter to include disabled applications.
`–verbose`	Optionally, you can use this parameter to display detailed output.

srvctl stop service

Stops one or more services globally across the cluster database, or on the specified instance.

Syntax

To stop services for a particular node in the cluster:

```
srvctl stop service -node node_name [-stopoption IMMEDIATE|TRANSACTIONAL|
NONE]
   [-drain_timeout timeout] [-wait {YES | NO}] [-force] [-noreplay] [-
global_override] [-verbose]
```

To stop services for a database:

```
srvctl stop service -db db_unique_name [-pdb pluggable_database |
   -service "service_list" [-eval]] [-node node_name | -instance
instance_name |
   [-stopoption IMMEDIATE|TRANSACTIONAL|NONE] [-drain_timeout timeout] [-wait
{YES | NO}] [-force [-noreplay]
   [-global_override] [-verbose]
```

Parameters

> **Note**
>
> The policy-managed database deployment option is desupported in Oracle AI Database.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-104 srvctl stop service Command Parameters

Parameter	Description
`-node` *node_name*	Optionally, you can specify the name of the node on which you want to stop services. Use this parameter without the `-db` parameter to stop all services on a specific node. If you use the `-db` parameter, then only the services on the specified node for that database are stopped.
`-db` *db_unique_name*	Specify a unique name for the database.
`-pdb` *pluggable_database*	Alternatively, use this parameter to stop services running on a specific Pluggable Database.
`-service` "*service_list*"	Specify a particular service or a comma-delimited list of service names enclosed in double quotation marks ("") you want to stop. If you do not provide a service name list, then SRVCTL stops all services on the database or on a specific instance.
`-instance` *instance_name*	Optionally, you can specify the name of the instance for which you want to stop services.
`-eval`	Use this parameter to hypothetically evaluate the impact of the command on the system.
`-stopoption IMMEDIATE|TRANSACTIONAL|NONE`	Specify the method of stopping the service. If this attribute was previously set for the service, then that value is used as the default value if you do not include the `-stopoption` parameter in your command. Otherwise, the default is `NONE`. • `IMMEDIATE` permits sessions to drain before the service is stopped. Sessions that do not drain are terminated when the time limit specified by `-drain_timeout` is reached. • If you specify `TRANSACTIONAL`, then sessions are terminated as soon as they commit. The service is stopped when the time limit specified by `-drain_timeout` is reached and any remaining sessions are terminated. • If you specify `NONE`, then no sessions are terminated. **Note:** You must use the `-stopoption` parameter with the `-force` parameter.
`-drain_timeout` *timeout*	Specify the time, in seconds, allowed for resource draining to be completed. Accepted values are an empty string (""), 0, or any positive integer. The default value is an empty string, which means that this parameter is not set. If it is set to 0, then draining occurs, immediately. The draining period is intended for planned maintenance operations. During the draining period, all current client requests are processed, but new requests are not accepted. When set on the service this value is used when the command line value is not set. **Note:** If the `-drain_timeout` value is not specified, then the maximum configured drain timeout of all the running services of the database on the specified nodes is used.
`-wait {YES | NO}`	Choose YES to wait until service draining is completed on the node to stop the service.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-104 (Cont.) srvctl stop service Command Parameters

Parameter	Description
`-force [-noreplay]`	Force SRVCTL to stop the service; this causes SRVCTL to disconnect all of the sessions using the stop option you specify (`IMMEDIATE` or `TRANSACTIONAL`), requiring the sessions using the service to reconnect and then connect to another instance. **Notes:** • If you do not specify the `-force` parameter, then sessions already connected to this service stay connected, but new sessions cannot be established to the service. • Optionally, you can specify the `-noreplay` parameter if you do not want Application Continuity to replay in-flight transactions after a session is terminated. The `-noreplay` parameter is not limited to use with `-force`. However, if you do not want to replay in-flight transactions after you force the service to stop, then `-force` requires `-noreplay`.
`-global_override`	Override value to operate on a global service. SRVCTL ignores this parameter if the service is not a global service.
`-verbose`	Use this parameter to display verbose output.

Examples

The following example command stops services running on the `crmeast` PDB in the `crm` database on instance `crm1` using the `IMMEDIATE` method, allowing 60 seconds for services to transfer to another node:

```
$ srvctl stop service -db crm -pdb crmeast -instance crm1 -drain_timeout 60 -
force
- stopoption immediate -verbose
```

The following example command stops all services running on the node `node1` that are managed by Oracle Clusterware using the default stop option specified for each service and waiting until all sessions have drained from that node.

```
$ srvctl stop service -node node1 -wait yes
```

vip Commands

Use commands with the `vip` keyword to add, manage environment variables for, list the configuration of, enable, disable, start, stop, obtain the status of, and remove a VIP.

- srvctl add vip
  Adds a virtual IP address (VIP) to a node.
- srvctl config vip
  Displays all VIPs on all networks in the cluster except for user VIPs.
- srvctl disable vip
- srvctl enable vip
- srvctl getenv vip

Copyright © 1999, 2025, Oracle and/or its affiliates.

- srvctl modify vip
  Modifies IP address type but you can also use it to modify only the IP address.
- srvctl predict vip
- srvctl relocate vip
- srvctl remove vip
  Removes the specified VIPs.
- srvctl setenv vip
- srvctl start vip
- srvctl status vip
- srvctl stop vip
- srvctl unsetenv vip

srvctl add vip

Adds a virtual IP address (VIP) to a node.

Syntax

```
srvctl add vip -node node_name -address {VIP_name|ip}/netmask/if1
      -netnum network_number [-skip] [-verbose]
```

Parameters

Table A-105 srvctl add vip Command Parameters

Parameter	Description
`-node node_name`	The name of the node on which you are adding the VIP.
`-address {VIP_name|ip}/ netmask/if1`	This specification creates a traditional VIP node application on the specified node. You can specify one *VIP_name* or address, along with an IPv4 netmask or IPv6 prefix length.
`-netnum network_number`	The network number from which VIPs are obtained. The default network number is 1.
`-skip`	Specify this parameter to skip checking the reachability of the VIP address.
`-verbose`	Verbose output

> **Note**

Usage Notes

- You cannot have multiple VIPs on the same net number (subnet or interface pair) on the same node.
- This command is only available with Oracle Clusterware.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example

An example of this command is:

```
# srvctl add network -netnum 2 -subnet 192.168.16.0/255.255.255.0
# srvctl add vip -node node7 -address 192.168.16.17/255.255.255.0 -netnum 2
```

The first command creates a network number, `2`, and the second command adds a VIP to this network. You can specify the network number after the `-netnum` parameter in other SRVCTL commands.

srvctl config vip

Displays all VIPs on all networks in the cluster except for user VIPs.

Syntax

```
srvctl config vip {-node node_name | -vip vip_name}
```

Parameters

Table A-106 srvctl config vip Command Parameters

Parameter	Description
`-node node_name`	Specify the node name.
`-vip vip_name`	Alternatively, you can specify the VIP name.

Usage Notes

This command is only available with Oracle Clusterware.

Example

This command returns output similar to the following:

```
$ srvctl config vip -node crmnode1

VIP exists: ipv4, ipv6, network number 1, hosting node adc2100252
```

srvctl disable vip

Disables a specific VIP.

Syntax

```
srvctl disable vip -vip vip_name [-verbose]
```

Usage Notes

- This command is only available with Oracle Clusterware.
- Specify the name of the VIP you want to disable.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- Optionally, you can use the `-verbose` parameter to display detailed output.

Example

The following command disables a VIP:

```
$ srvctl disable vip -vip vip1 -verbose
```

srvctl enable vip

Enables a specific VIP.

Syntax

```
srvctl enable vip -vip vip_name [-verbose]
```

Usage Notes

- You can only use this command with Oracle Clusterware.
- Specify the name of the VIP you want to enable.
- Optionally, you can use the `-verbose` parameter to display detailed output.

Example

The following example enables a VIP named `crm1-vip`:

```
$ srvctl enable vip -vip crm1-vip -verbose
```

srvctl getenv vip

Obtains the values of environment variables for a specific VIP.

Syntax

```
srvctl getenv vip -vip vip_name [-envs "name_list"] [-verbose]
```

Parameters

Table A-107 srvctl getenv vip Command Parameters

Parameter	Description
`-vip vip_name`	Specify the name of the VIP for which you want to obtain the values of the environment variables.
`-envs "name_list"`	Optionally, you can specify a comma-delimited list of the names of specific environment variables. If you do not use this parameter, then SRVCTL displays the values of all environment variables associated with the VIP.

Usage Notes

You can only use this command with Oracle Clusterware.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example

The following example lists all environment variables for the specified VIP:

```
$ srvctl getenv vip -vip node1-vip
```

srvctl modify vip

Modifies IP address type but you can also use it to modify only the IP address.

Syntax

```
srvctl modify vip -node node_name -address {VIP_name|ip}/netmask/if1
   [-netnum network_number] [-verbose]
```

Parameters

Table A-108 srvctl modify vip Command Parameters

Parameter	Description
`-node node_name`	Specify the name of the node on which you are changing the VIP.
`-address {VIP_name|ip}/netmask/if1`	Use this parameter to change the configuration of an existing VIP. If the VIP has an IPv4 address and the address you specify is IPv6, and the IP address type is set to `both` and the network type is set to `static`, then SRVCTL adds the IPv6 address to the existing IPv4 address of that resource. You can specify one *VIP_name* or IP address, along with an IPv4 netmask or IPv6 prefix length.
`-netnum network_number`	Optionally, you can specify the network number from which VIPs are obtained. If you do not use this parameter, then SRVCTL obtains the VIPs from the same default network from which the `nodeapps` VIP is obtained.
`–verbose`	Optionally, you can use this parameter to display detailed output.

Usage Notes

- You can only use this command with Oracle Clusterware.
- You cannot have multiple VIPs on the same net number (subnet or interface pair) on the same node.

Example

The following example adds an IPv4 address to a VIP, if one does not already exist. If the VIP has an IPv4 address, then it is replaced with the new network specification.

```
# srvctl modify vip -node node7 -address 192.168.16.17/255.255.255.0 -netnum 2
```

srvctl predict vip

Predicts the consequences of VIP failure.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Syntax

```
srvctl predict vip [-vip vip_name] [-verbose]
```

Usage Notes

- Optionally, you can specify the name of a VIP for which you want to evaluate the consequences of failure.
- Optionally, you can use the `–verbose` parameter to display detailed output.

srvctl relocate vip

Relocates a specific VIP from its current hosting node to another node within the cluster.

Syntax

```
srvctl relocate vip -vip vip_name [-node node_name] [-force] [-verbose]
```

Parameters

Table A-109 srvctl relocate vip Command Parameters

Parameter	Description
`-vip vip_name`	Specify the name of the VIP you want to relocate.
`-node node_name`	Optionally, you can specify the name of the target node where you want to relocate the VIP.
`–force`	Optionally, you can use this parameter to force the relocation of the VIP regardless of any dependencies.
`–verbose`	Optionally, you can use this parameter to display detailed output.

Example

The following example relocates a VIP to a different node in the cluster:

```
$ srvctl relocate vip -vip vip1 -node node3
```

srvctl remove vip

Removes the specified VIPs.

Syntax

```
srvctl remove vip -vip "vip_name_list" [-force] [-noprompt] [-verbose]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

Table A-110 srvctl remove vip Command Parameters

Parameter	Description
`-vip "vip_name_list"`	Specify a comma-delimited list of VIP names that you want to remove surrounded by double quotation marks ("").
`–force`	Optionally, you can use this parameter to remove a VIP regardless of any dependencies.
`-noprompt`	Optionally, you can use this parameter to suppress prompts.
`-verbose`	Optionally, you can use this parameter to display detailed output.

Usage Notes

You can only use this command with Oracle Clusterware.

Example

The following example removes several VIPs from the system:

```
# srvctl remove vip -vip "vip1,vip2,vip3" -force -noprompt -verbose
```

srvctl setenv vip

Administers cluster VIP environment configurations.

Syntax

```
srvctl setenv vip -vip vip_name {-envs "name=val[,...]" | -env "name=val"}
   [-verbose]
```

Parameters

Table A-111 srvctl setenv vip Command Parameters

Parameter	Description
`-vip vip_name`	Specify the name of the VIP for which you want to set environment variables.
`-envs "name=val[,...]"`	Specify a comma-delimited list of name-value pairs of environment variables enclosed in double quotation marks ("") that you want to set.
`-env "name=val"`	Alternative to a list of environment variables, you can use this parameter to set a single environment variable to a value that contains commas or other special characters enclosed in double quotation marks ("").
`–verbose`	Optionally, you can use this parameter to display detailed output.

Usage Notes

You can only use this command with Oracle Clusterware.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Example

The following example sets the language environment configuration for a cluster VIP:

```
$ srvctl setenv vip -vip crm1-vip -env "LANG=en"
```

srvctl start vip

Starts a specific VIP or a VIP on a specific node.

Syntax

```
srvctl start vip {-node node_name | -vip vip_name} [-verbose]
```

Parameters

Table A-112 srvctl start vip Command Parameters

Parameter	Description
`-node node_name`	Specify the name of the node on which the VIP resides that you want to start.
`-vip vip_name`	Alternative to specifying a node, you can specify a VIP that you want to start.
`-verbose`	Optionally, you can use this parameter to display detailed ouptut.

Usage Notes

You can only use this command with Oracle Clusterware.

Example

The following example starts a specific VIP:

```
$ srvctl start vip -vip crm1-vip -verbose
```

srvctl status vip

Displays status for a specific VIP or a VIP on a specific node.

Syntax

```
srvctl status vip {-node node_name | -vip vip_name} [-verbose]
```

Parameters

Table A-113 srvctl status vip Command Parameters

Parameter	Description
`-node node_name`	Specify the name of the node on which the VIP resides that you want to check the status.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-113 (Cont.) srvctl status vip Command Parameters

Parameter	Description
`-vip vip_name`	Alternative to specifying a node, you can specify a VIP that you want to check the status.
`-verbose`	Optionally, you can use this parameter to display detailed output.

Usage Notes

You can only use this command with Oracle Clusterware.

srvctl stop vip

Stops a specific VIP or all VIPs on a specific node, including any VIPs that were relocated due to a failover.

Syntax

```
srvctl stop vip {-node node_name | -vip vip_name} [-force] [-relocate] [-
verbose]
```

Parameters

Table A-114 srvctl stop vip Command Parameters

Parameter	Description
`-node node_name`	Specify the name of a node on which a VIP resides that you want to stop. If you use this parameter, then SRVCTL stops all VIPs on the specific node, including failed-over VIPs.
`-vip vip_name`	Alternative to specifying a node, you can specify a VIP that you want to stop.
`-force`	Optionally, you can use this parameter to stop the VIP regardless of any dependencies.
`-relocate`	Optionally, you can use this parameter to relocate the VIP. **Note:** You must use the `-node node_name` parameter with the `-relocate` parameter.
`–verbose`	Optionally, you can use this parameter to display detailed output.

Usage Notes

You can only use this command with Oracle Clusterware.

Example

The following example stops all the VIPs on `mynode1`, including any failed-over VIPs:

```
$ srvctl stop vip -node mynode1 -verbose
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl unsetenv vip

Unsets the environment configuration for the specified cluster VIP.

Syntax

```
srvctl unsetenv vip -vip "vip_name_list" -envs "name_list" [-verbose]
```

Parameters

Table A-115 srvctl unsetenv vip Command Parameters

Parameter	Description
`-vip "vip_name_list"`	Specify a comma-delimited list of VIP names enclosed in double quotation marks ("").
`-envs "name_list"`	Specify a comma-delimited list of environment variable names enclosed in double quotation marks ("") that you want to unset.
`–verbose`	Optionally, you can use this parameter to display detailed output.

Example

The following example unsets the `CLASSPATH` environment variable for a cluster VIP:

```
$ srvctl unsetenv vip -vip "crm2-vip" -envs "CLASSPATH"
```

volume Commands

Use commands with the `volume` keyword to list the configuration of, enable, disable, start, stop, obtain the status of, and remove an Oracle ACFS volume.

> **Note**
>
> The `volume` commands work only on the platforms on which Oracle ACFS is supported.

- srvctl config volume
  Displays the configuration for a specific volume or all volumes.
- srvctl disable volume
- srvctl enable volume
- srvctl remove volume
- srvctl start volume
- srvctl status volume
- srvctl stop volume

Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl config volume

Displays the configuration for a specific volume or all volumes.

Syntax

```
srvctl config volume [-volume volume_name] [-diskgroup disk_group_name]
   [-device volume_device]
```

Parameters

Table A-116 srvctl config volume Command Parameters

Parameter	Description
`-volume volume_name`	Specify the name of the volume for which you want to view the configuration.
`-diskgroup disk_group_name`	Specify the name of the disk group in which the volume resides for which you want to display the configuration.
`-device volume_device`	Specify the path to the volume device for which you want to display the configuration.

Usage Notes

- If you do not specify any of the optional parameters, then SRVCTL displays the configuration information for all volumes.
- If you specify only the `-volume` parameter, then SRVCTL displays the configuration for all volumes with that name, regardless of the diskgroup.
- If you specify only the `-diskgroup` parameter, then SRVCTL displays the configuration information for the volumes that reside in the disk group that you specify.
- If you specify only the `-device` parameter, then SRVCTL displays the configuration information for the volume matching that device specifier.
- If you specify the `-diskgroup` and `-device` parameters, then SRVCTL displays the configuration information for the volume device that resides in the disk group that you specify.
- This command is only available with Oracle Clusterware.

Examples

This command returns information similar to the following:

```
$ srvctl config volume -device /dev/asm/volume1-123

Diskgroup Name: DG1
Volume Name   : VOL1
Volume Device : /dev/asm/volume1-123
Volume is enabled.
Volume is enabled on nodes:
Volume is disabled on nodes:
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

If you do not specify any parameters, then SRVCTL returns configuration information for all volumes, similar to the following:

```
$ srvctl config volume

Diskgroup name: DG1
Volume name: VOL1
Volume device: /dev/asm/volume1-123
Volume is enabled.
Volume is enabled on nodes:
Volume is disabled on nodes:
Diskgroup name: DG1
Volume name: VOL2
Volume device: /dev/asm/volume2-456
Volume is enabled.
Volume is enabled on nodes:
Volume is disabled on nodes:
```

srvctl disable volume

Disables Oracle Clusterware management for a specific volume or all volumes.
This command allows a volume device to be stopped by operating on the Oracle Clusterware resource for the volume. This command does not stop volume device.

Syntax

```
srvctl disable volume {-volume volume_name -diskgroup disk_group_name |
    -device volume_device}
```

Parameters

Table A-117 srvctl disable volume Command Parameters

Parameter	Description
`-volume volume_name`	Specify the name of the volume that you want to disable.
`-diskgroup disk_group_name`	Specify the name of the disk group in which the volume that you want to disable resides.
`-device volume_device`	Alternative to using the `-diskgroup` parameter, you can specify the path to the volume device that you want to disable.

Usage Notes

- You can only use this command with Oracle Clusterware.
- You must specify a particular volume that you want to disable. You can specify a volume that resides in either a particular disk group or on a particular volume device.

Example

The following example disables a volume named `VOLUME1` that resides in a disk group named `DATA`:

```
$ srvctl disable volume -volume VOLUME1 -diskgroup DATA
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

srvctl enable volume

Enables Oracle Clusterware management for a specific volume or all volumes.
This command allows a volume device to be started by operating on the Oracle Clusterware resource for the volume. This command does not start the volume device, and is different from the SQL command `ALTER DISKGROUP ENABLE VOLUME` or the ASMCMD command `volenable`, because these two commands bring the volume device online, in a running state, making the volume device accessible.

Syntax

```
srvctl enable volume {-volume volume_name -diskgroup disk_group_name |
   -device volume_device}
```

Parameters

Table A-118 srvctl enable volume Command Parameters

Parameter	Description
`-volume volume_name`	Specify the name of the volume that you want to enable.
`-diskgroup disk_group_name`	Specify the name of the disk group in which the volume that you want to enable resides.
`-device volume_device`	Alternative to using the `–diskgroup` parameter, you can specify the path to the volume device that you want to enable.

Usage Notes

- You can only use this command with Oracle Clusterware.
- You must specify a particular volume that you want to enable. You can specify a volume that resides in either a particular disk group or on a particular volume device.

Example

The following example enables a volume named `VOLUME1` that resides in a disk group named `DATA`:

```
$ srvctl enable volume -volume VOLUME1 -diskgroup DATA
```

srvctl remove volume

Removes a specific volume.

Syntax

Use this command with one of the following syntax models:

```
srvctl remove volume -volume volume_name -diskgroup disk_group_name [-force]

srvctl remove volume -device volume_device [-force]
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Parameters

Table A-119 srvctl remove volume Command Parameters

Parameter	Description
`-volume volume_name`	Specify the name of the volume that you want to remove.
`-diskgroup disk_group_name`	Specify the name of the disk group in which the volume that you want to remove resides.
`-device volume_device`	Specify the path to the file system resource in which the volume that you want to remove resides.
`–force`	You can use this parameter to remove the volume even if it is running.

Usage Notes

- You can only use this command with Oracle Clusterware.
- The volume gets created when you create volumes in Oracle ASM.
- You must specify a particular volume that you want to remove. You can specify a volume that resides in either a particular disk group or on a particular volume device.

Example

The following example removes a volume named `VOLUME1` that resides in a disk group named `DATA`:

```
$ srvctl remove volume -volume VOLUME1 -diskgroup DATA
```

Related Topics

- *Oracle Automatic Storage Management Administrator's Guide*

srvctl start volume

Starts a specific, enabled volume.

Syntax

```
srvctl start volume {-volume volume_name -diskgroup disk_group_name |
    -device volume_device} [-node node_list]
```

Parameters

Table A-120 srvctl start volume Command Parameters

Parameter	Description
`-volume volume_name`	Specify the name of the volume that you want to start.
`-diskgroup disk_group_name`	Specify the name of the disk group in which the volume that you want to start resides.
`-device volume_device`	Alternative to using the `–diskgroup` parameter, you can specify the path to the volume device that you want to start.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-120 (Cont.) srvctl start volume Command Parameters

Parameter	Description
`-node node_list`	Optionally, you can specify a comma-delimited list of node names enclosed in double quotation marks ("") where volumes that you want to start reside.

Usage Notes

- You can only use this command with Oracle Clusterware.
- The `srvctl start volume` command does not create a volume service. Provided that the volume already exists and the volume resource is enabled, SRVCTL attempts to start it. If the volume exists but the resource is disabled, then `srvctl start volume` returns an error.

Example

The following example starts a volume named `VOLUME1` that resides in a disk group named `DATA`:

```
$ srvctl start volume -volume VOLUME1 -diskgroup DATA
```

srvctl status volume

Displays the status of a specific volume or all volumes.

Syntax

```
srvctl status volume [-device volume_device] [-volume volume_name]
   [-diskgroup disk_group_name] [-node "node_list"]
```

Parameters

Table A-121 srvctl status volume Command Parameters

Parameter	Description
`-device volume_device`	Optionally, you can specify the path to the volume device for which you want to display the status.
`-volume volume_name`	Optionally, you can specify the name of the volume for which you want to view the status.
`-diskgroup disk_group_name`	Optionally, you can specify the name of the disk group in which the volume resides for which you want to display the status.
`-node "node_list"`	Optionally, you can specify a comma-delimited list of node names enclosed in double quotation marks ("") where volumes for which you want to view the status reside.

Usage Notes

- You can only use this command with Oracle Clusterware.
- If you do not specify any of the optional parameters, then SRVCTL displays the status for all volumes.

Copyright © 1999, 2025, Oracle and/or its affiliates.

- If you specify only the `-volume` parameter, then SRVCTL displays the status for the volume that you specify.
- If you specify only the `-diskgroup` parameter, then SRVCTL displays the status for the volumes that reside in the disk group that you specify.
- If you specify only the `-device` parameter, then SRVCTL displays the status for the volume device that you specify.
- If you specify the `-diskgroup` and `-device` parameters, then SRVCTL displays the status for the volume device in the disk group that you specify.
- If you specify the `-node` parameter, then SRVCTL displays the status of the volumes that reside on the nodes you list.

Examples

This command displays information similar to the following:

```
$ srvctl status volume –volume vol1
Volume vol1 of diskgroup diskgrp1 for device volume_device_path1 is enabled
Volume vol1 of diskgroup diskgrp1 for device volume_device_path1 is running
```

In the preceding example, SRVCTL performs a status query on all nodes because the `-node` parameter is not specified.

```
$ srvctl status volume
Volume vol1 of diskgroup diskgrp for device volume_device_path1 is enabled
Volume vol1 of diskgroup diskgrp for device volume_device_path1 is running
Volume vol2 of diskgroup diskgrp for device volume_device_path2 is enabled
Volume vol2 of diskgroup diskgrp for device volume_device_path2 is running
```

In the preceding example, SRVCTL displays the status of all registered volumes because the no parameter is specified.

srvctl stop volume

Stops a specific, running volume.

Syntax

```
srvctl stop volume {-volume volume_name -diskgroup disk_group_name |
    -device volume_device} [-node "node_list"]
```

Parameters

Table A-122 srvctl stop volume Command Parameters

Parameter	Description
`-volume volume_name`	Specify the name of the volume you want to stop.
`-diskgroup disk_group_name`	Specify the name of the disk group in which the volume you want to stop resides.
`-device volume_device`	Alternative to using the `–diskgroup` parameter, you can specify the path to the volume device that you want to stop.

Copyright © 1999, 2025, Oracle and/or its affiliates.

Table A-122 (Cont.) srvctl stop volume Command Parameters

Parameter	Description
`-node "node_list"`	Optionally, you can specify a comma-delimited list of node names enclosed in double quotation marks ("") where volumes that you want to stop reside.

Usage Notes

- You can only use this command with Oracle Clusterware.
- The `srvctl stop volume` command attempts to stop (disable) the volume but it does not disable the resource or remove the volume from Oracle ASM.

Example

The following example stops a volume named `VOLUME1` that resides in a disk group named `DATA`:

```
$ srvctl stop volume -volume VOLUME1 -diskgroup DATA
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

B

Troubleshooting Oracle RAC

This appendix explains how diagnose problems for Oracle Real Application Clusters (Oracle RAC) components using trace and log files.

> **Note**
>
> Trace and log files, similar to those generated for Oracle AI Database with Oracle RAC, are also available for the Oracle Clusterware components. For Oracle Clusterware, Oracle AI Database stores these under a unified directory log structure.

- Where to Find Files for Analyzing Errors
  Oracle AI Database records information about important events that occur in your Oracle RAC environment in trace files.
- Managing Diagnostic Data in Oracle RAC
  Problems that span Oracle RAC instances can be the most difficult types of problems to diagnose.
- Using Instance-Specific Alert Files in Oracle RAC
  Each instance in an Oracle RAC database has one alert file.
- Enabling Tracing for Java-Based Tools and Utilities in Oracle RAC
  All Java-based tools and utilities that are available in Oracle RAC are called by processing scripts of the same name as the tool or utility.
- Resolving Pending Shutdown Issues
  In some situations a `SHUTDOWN IMMEDIATE` may be pending and Oracle AI Database will not quickly respond to repeated shutdown requests.
- How to Determine If Oracle RAC Instances Are Using the Private Network
  This topic describes how to manually determine if Oracle RAC instances are using the private network.

Where to Find Files for Analyzing Errors

Oracle AI Database records information about important events that occur in your Oracle RAC environment in trace files.

The trace files for Oracle RAC are the same as those in noncluster Oracle AI Databases. As a best practice, monitor and back up trace files regularly for all instances to preserve their content for future troubleshooting.

Information about `ORA-600` errors appear in the `alert_`*`SID`*`.log` file for each instance where *`SID`* is the instance identifier.

The alert log and all trace files for background and server processes are written to the Automatic Diagnostic Repository, the location of which you can specify with the `DIAGNOSTIC_DEST` initialization parameter. For example:

```
$ORACLE_BASE/diag/rdbms/$DBNAME/$SID_NAME/trace
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

Oracle AI Database creates a different trace file for each background thread. Oracle RAC background threads use trace files to record database operations and database errors. These trace logs help troubleshoot and also enable Oracle Support to more efficiently debug cluster database configuration problems. The names of trace files are operating system specific, but each file usually includes the name of the process writing the file (such as LGWR and RECO). For Linux, UNIX, and Windows systems, trace files for the background processes are named `SID_process_name_process_identifier.trc`.

Trace files are also created for user processes if you set the `DIAGNOSTIC_DEST` initialization parameter. User process trace file names have the format `SID_ora_process_identifier/thread_identifier.trc`, where *process_identifier* is a 5-digit number indicating the process identifier (PID) on Linux and UNIX systems, and *thread_identifier* is the thread identifier on Windows systems.

Related Topics

- Troubleshooting Oracle Clusterware
- Monitoring the Database

Managing Diagnostic Data in Oracle RAC

Problems that span Oracle RAC instances can be the most difficult types of problems to diagnose.

For example, you may need to correlate the trace files from across multiple instances, and merge the trace files. Oracle AI Database includes an advanced fault diagnosability infrastructure for collecting and managing diagnostic data, and uses the Automatic Diagnostic Repository (ADR) file-based repository for storing the database diagnostic data. When you create the ADR base on a shared disk, you can place ADR homes for all instances of the same Oracle RAC database under the same ADR Base. With shared storage:

- You can use the ADRCI command-line tool to correlate diagnostics across all instances.

  ADRCI is a command-line tool that enables you to view diagnostic data in the ADR and package incident and problem information into a zip file for transmission to Oracle Support. The diagnostic data includes incident and problem descriptions, trace files, dumps, health monitor reports, alert log entries, and so on.
- You can use the Data Recovery Advisor to help diagnose and repair corrupted data blocks, corrupted or missing files, and other data failures.

  The Data Recovery Advisor is an Oracle AI Database infrastructure that automatically diagnoses persistent data failures, presents repair options, and repairs problems at your request.

Related Topics

- ADRCI: ADR Command Interpreter
- Diagnosing and Resolving Problems

Using Instance-Specific Alert Files in Oracle RAC

Each instance in an Oracle RAC database has one alert file.

The alert file for each instance, `alert_SID.log`, contains important information about error messages and exceptions that occur during database operations. Information is appended to the alert file each time you start the instance. All process threads can write to the alert file for the instance.

Copyright © 1999, 2025, Oracle and/or its affiliates.

The `alert_SID.log` file is in the directory specified by the `DIAGNOSTIC_DEST` initialization parameter.

Enabling Tracing for Java-Based Tools and Utilities in Oracle RAC

All Java-based tools and utilities that are available in Oracle RAC are called by processing scripts of the same name as the tool or utility.

This includes the Configuration Verification Utility (CVU), Oracle DBCA, the Net Configuration Assistant (NETCA), and the Server Control Utility (SRVCTL). For example, to run Oracle DBCA, enter the command `dbca`.

By default, Oracle AI Database enables traces for Oracle DBCA and the Database Upgrade Assistant (DBUA). For the CVU, and SRVCTL, you can set the `SRVM_TRACE` environment variable to `TRUE` to make Oracle AI Database generate traces. Oracle AI Database writes traces to log files. For example, Oracle AI Database writes traces to log files in *`Oracle_base`*`/cfgtoollogs/dbca` and *`Oracle_base`*`/cfgtoollogs/dbua` for Oracle DBCA and Oracle DBUA, respectively.

Resolving Pending Shutdown Issues

In some situations a `SHUTDOWN IMMEDIATE` may be pending and Oracle AI Database will not quickly respond to repeated shutdown requests.

This is because Oracle Clusterware may be processing a current shutdown request. In such cases, issue a `SHUTDOWN ABORT` using SQL*Plus for subsequent shutdown requests.

How to Determine If Oracle RAC Instances Are Using the Private Network

This topic describes how to manually determine if Oracle RAC instances are using the private network.

However, the best practice for this task is to use the Oracle Enterprise Manager Cloud Control graphical user interface (GUI) to check the interconnect.

With most network protocols, you can issue the `oradebug ipc` command to see the interconnects that the database is using. For example:

```
oradebug setmypid
oradebug ipc
```

These commands dump a trace file to the location specified by the `DIAGNOSTIC_DEST` initialization parameter. The output may look similar to the following:

```
SSKGXPT 0x1a2932c flags SSKGXPT_READPENDING     info for network 0
        socket no 10    IP 172.16.193.1         UDP 43749
        sflags SSKGXPT_WRITESSKGXPT_UP  info for network 1
        socket no 0     IP 0.0.0.0      UDP 0...
```


Copyright © 1999, 2025, Oracle and/or its affiliates.

In the example, you can see the database is using IP 172.16.193.1 with a User Datagram Protocol (UDP) protocol. Also, you can issue the `oradebug tracefile_name` command to print the trace location where the output is written.

Additionally, you can query the `V$CLUSTER_INTERCONNECTS` view to see information about the private interconnect. For example:

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

NAME   IP_ADDRESS                IS_   SOURCE
-----  ------------------------- ---   -------------------------------
eth0   138.2.236.114              NO   Oracle Cluster Repository
```


Copyright © 1999, 2025, Oracle and/or its affiliates.