

Oracle® Data Guard

Broker

12c Release 1 (12.1)

E48241-07

November 2014

Oracle Data Guard Broker, 12c Release 1 (12.1)

E48241-07

Copyright © 1999, 2014, Oracle and/or its affiliates. All rights reserved.

Primary Author: Kathy Rich

Contributor: Larry Carpenter, Laurence Clarke, Jeff Detjen, Mahesh Girkar, Michael Harvey, Nitin Karkhanis, Sadhana Kyathappala, Chang Kyu Lee, Steve Lee, Jiangbin Luo, Bob McGuiirk, Joe Meeks, Darryl Presley, Ashish Ray, Lawrence To, Di Yang, Zaixian Xie

Contributor: The Oracle Database 12c documentation is dedicated to Mark Townsend, who was an inspiration to all who worked on this release.

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

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. 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 and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon 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, Opteron, the AMD logo, and the AMD Opteron 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 on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

Preface	xv
Audience	xv
Documentation Accessibility	xv
Related Documents	xv
Conventions	xv
Changes in This Release for Oracle Data Guard Broker	xvii
Changes in Oracle Database 12c Release 1 (12.1.0.2)	xvii
Changes in Oracle Database 12c Release 1 (12.1.0.1)	xviii
1 Oracle Data Guard Broker Concepts	
1.1 Overview of Oracle Data Guard and the Broker	1-1
1.1.1 Oracle Data Guard Configurations and Broker Configurations	1-1
1.1.2 Oracle Data Guard Broker	1-2
1.1.3 Oracle Data Guard Broker and CDBs	1-4
1.1.4 Oracle Data Guard Broker and Oracle Global Data Services	1-5
1.2 Benefits of Oracle Data Guard Broker	1-5
1.3 Oracle Data Guard Broker Components	1-8
1.4 Oracle Data Guard Broker User Interfaces	1-9
1.4.1 Oracle Enterprise Manager Cloud Control	1-9
1.4.2 Oracle Data Guard Command-Line Interface (DGMRGL)	1-10
1.5 Oracle Data Guard Monitor	1-10
1.5.1 Oracle Data Guard Monitor (DMON) Process	1-10
1.5.2 Configuration Management	1-11
1.5.3 Database Property Management	1-12
2 Oracle Data Guard Installation	
2.1 Oracle Data Guard Installation	2-1
2.2 Prerequisites	2-2
3 Managing Broker Configurations	
3.1 Configuration Support	3-1
3.2 Configuration Properties	3-3
3.3 Setting Up the Broker Configuration Files	3-3

3.3.1	Renaming the Broker Configuration Files	3-4
3.3.2	Managing Broker Configuration Files in an Oracle RAC Environment.....	3-5
3.3.2.1	Using Cluster File System (CFS) for Configuration Files	3-5
3.3.2.2	Using Oracle ASM Disk Groups for Configuration Files	3-6
3.4	Starting the Data Guard Broker	3-6
3.5	Management Cycle of a Broker Configuration.....	3-7
3.6	Enable and Disable Operations.....	3-9
3.7	Configuration Status.....	3-10

4 Managing the Members of a Broker Configuration

4.1	Managing Broker Configuration Members.....	4-1
4.2	Managing States of Broker Configuration Members	4-1
4.2.1	Database State Transitions.....	4-3
4.3	Managing Database Properties	4-4
4.3.1	Monitorable (Read-Only) Properties	4-6
4.3.2	Configurable (Changeable) Properties	4-6
4.3.2.1	Resetting Broker Configurable Properties to Default Values	4-7
4.4	Managing Redo Transport Services	4-7
4.4.1	Setting Up For Redo Transport.....	4-8
4.4.2	Managing Redo Transport Services for Data Protection Modes	4-8
4.4.3	Advanced Redo Transport Settings	4-9
4.4.4	Turning Redo Transport Services On and Off.....	4-12
4.4.5	Specifying Locations for Archived Redo Log Files.....	4-13
4.4.6	Other Redo Transport Settings	4-14
4.4.7	Redo Transport Services in an Oracle RAC Database Environment.....	4-14
4.4.8	Transport Lag	4-15
4.5	Managing Log Apply Services.....	4-15
4.5.1	Managing Delayed Apply	4-16
4.5.2	Managing Parallel Apply with Redo Apply	4-16
4.5.3	Allocating Resources to SQL Apply.....	4-17
4.5.4	Managing the DBA_LOGSTDBY_EVENTS Table	4-17
4.5.5	Apply Services in an Oracle RAC Database Environment	4-17
4.5.5.1	Selecting the Apply Instance.....	4-17
4.5.5.2	Apply Instance Failover.....	4-19
4.5.6	Apply Lag	4-20
4.6	Managing Data Protection Modes	4-20
4.6.1	Setting the Protection Mode for Your Configuration	4-20
4.6.2	How the Protection Modes Influence Broker Operations.....	4-24
4.6.2.1	Upgrading or Downgrading the Current Protection Mode.....	4-24
4.6.2.2	Switchover Operations	4-25
4.6.2.3	Failover Operations.....	4-25
4.6.2.4	Disable and Enable Operations	4-26
4.6.2.5	Requirements For Removing a Database from the Configuration.....	4-26
4.6.2.6	Requirements On Other Operations	4-26
4.7	Managing Far Sync Instances.....	4-27
4.8	Managing Fast-Start Failover	4-28
4.8.1	Configure Properties to Tune Fast-Start Failover	4-29

4.8.2	Configure Conditions for Fast-start Failover.....	4-30
4.8.3	Application Initiated Fast-Start Failover	4-31
4.9	Managing Database Conversions	4-31
4.10	Database Status	4-32
4.10.1	Querying Database Status	4-32
4.10.2	Validating a Database Before a Role Change	4-34

5 Switchover and Failover Operations

5.1	Overview of Switchover and Failover in a Broker Environment	5-1
5.2	Choosing a Target Standby Database	5-2
5.2.1	Choosing a Target Standby Database for Switchover	5-2
5.2.2	Choosing a Target Standby Database for Failover	5-3
5.3	Switchover.....	5-4
5.3.1	Before You Perform a Switchover Operation	5-4
5.3.2	Starting a Switchover	5-6
5.3.3	How the Broker Performs a Switchover.....	5-6
5.4	Manual Failover	5-7
5.4.1	Complete and Immediate Manual Failovers	5-7
5.4.2	Performing a Manual Failover Operation.....	5-8
5.4.2.1	How the Broker Performs a Complete Failover Operation.....	5-10
5.4.2.2	How the Broker Performs an Immediate Failover Operation	5-12
5.4.3	Reenabling Disabled Databases After a Role Change.....	5-12
5.4.3.1	How to Reinstall a Database	5-13
5.4.3.2	How to Re-create and Reenable a Disabled Database	5-14
5.5	Fast-Start Failover	5-14
5.5.1	Prerequisites for Enabling Fast-Start Failover	5-16
5.5.2	Enabling Fast-Start Failover	5-17
5.5.2.1	When Fast-Start Failover Is Enabled and the Observer Is Running	5-23
5.5.2.2	Restrictions When Fast-Start Failover is Enabled	5-25
5.5.2.3	Shutting Down the Primary Database When Fast-Start Failover Is Enabled ...	5-26
5.5.2.4	Performing Manual Role Changes When Fast-Start Failover Is Enabled	5-26
5.5.3	Directing a Fast-Start Failover From an Application.....	5-27
5.5.4	Viewing Fast-Start Failover Configuration Statistics and Status.....	5-27
5.5.4.1	V\$DATABASE View	5-28
5.5.4.2	V\$FS_FAILOVER_STATS View	5-30
5.5.5	Disabling Fast-Start Failover	5-30
5.5.6	Performance Considerations for Fast-Start Failover	5-33
5.5.7	Managing the Observer	5-34
5.5.7.1	Installing and Starting the Observer	5-35
5.5.7.2	Viewing Information About the Observer.....	5-35
5.5.7.3	What Happens if the Observer Fails?	5-36
5.5.7.4	Managing Observer's Connection to the Primary	5-37
5.5.7.5	Stopping the Observer	5-37
5.5.7.6	Moving the Observer to Another Computer	5-38
5.5.7.7	How the Observer Maintains Fast-Start Failover Configuration Information .	5-38
5.5.7.8	Patching an Environment When the Observer Is Running and Fast-start Failover Is Enabled	5-38

5.5.8	Reinstating the Former Primary Database in the Broker Configuration.....	5-39
5.5.8.1	Requirements	5-39
5.5.8.2	Restrictions on Reinstatement	5-39
5.5.8.3	How the Broker Handles a Failed Reinstatement.....	5-40
5.5.9	Shutting Down Databases In a Fast-Start Failover Environment.....	5-40
5.6	Database Client Considerations.....	5-40
5.6.1	Oracle Data Guard Specific FAN and FCF Configuration Requirements.....	5-41
5.6.1.1	Oracle Net Configuration Requirements	5-41
5.6.1.2	Database Service Configuration Requirements	5-42
5.6.1.3	ONS Configuration Requirements.....	5-43
5.6.1.4	Application Continuity	5-44

6 Scenarios Using the DGMGRL Command-Line Interface

6.1	Prerequisites for Getting Started	6-1
6.2	Scenario 1: Creating a Configuration.....	6-2
6.3	Scenario 2: Setting Database Properties.....	6-5
6.4	Scenario 3: Enabling the Configuration and Databases	6-6
6.5	Scenario 4: Setting the Configuration Protection Mode.....	6-7
6.6	Scenario 5: Setting up Maximum Availability Mode with a Far Sync Instance	6-9
6.7	Scenario 6: Enabling Fast-Start Failover and Starting the Observer.....	6-11
6.8	Scenario 7: Enabling Fast-Start Failover When a Far Sync Instance Is In Use	6-14
6.9	Scenario 8: Performing Routine Management Tasks.....	6-14
6.9.1	Changing Properties and States.....	6-14
6.9.1.1	Alter a Database Property	6-14
6.9.1.2	Reset a Property to Its Default Value.....	6-15
6.9.1.3	Alter the State of a Standby Database	6-15
6.9.1.4	Alter the State of a Primary Database.....	6-15
6.9.2	Disabling the Configuration and Databases.....	6-15
6.9.2.1	Disable a Configuration.....	6-16
6.9.2.2	Disable a Standby Database	6-16
6.9.2.3	Disabling a Far Sync Instance	6-17
6.9.3	Removing the Configuration, a Standby Database, or a Far Sync Instance.....	6-17
6.10	Scenario 9: Performing a Switchover Operation	6-19
6.11	Scenario 10: Performing a Manual Failover Operation.....	6-23
6.12	Scenario 11: Reinstating a Failed Primary Database.....	6-25
6.13	Scenario 12: Converting a Physical Standby to a Snapshot Standby	6-27
6.14	Scenario 13: Monitoring a Data Guard Configuration	6-27

7 Oracle Data Guard Command-Line Interface Reference

7.1	Starting the Data Guard Command-Line Interface	7-1
7.1.1	DGMGRL Optional Parameters.....	7-1
7.1.2	DGMGRL Command Format and Parameters.....	7-2
7.1.3	DGMGRL Command Usage Notes.....	7-5
7.2	Exiting the Data Guard Command-Line Interface.....	7-6
	ADD DATABASE	7-7
	ADD FAR_SYNC	7-8

CONNECT	7-9
CONVERT DATABASE.....	7-11
CREATE CONFIGURATION	7-13
DISABLE CONFIGURATION	7-15
DISABLE DATABASE.....	7-16
DISABLE FAR_SYNC.....	7-17
DISABLE FAST_START FAILOVER.....	7-18
DISABLE FAST_START FAILOVER CONDITION.....	7-20
EDIT CONFIGURATION (Property).....	7-21
EDIT CONFIGURATION (Protection Mode).....	7-22
EDIT CONFIGURATION (RENAME).....	7-24
EDIT CONFIGURATION RESET (Property).....	7-25
EDIT DATABASE (Property)	7-26
EDIT DATABASE (Rename)	7-28
EDIT DATABASE (State).....	7-29
EDIT DATABASE RESET (Property).....	7-30
EDIT FAR_SYNC	7-31
EDIT FAR_SYNC RESET (Property).....	7-32
EDIT INSTANCE (AUTO PFILE).....	7-33
EDIT INSTANCE (Property).....	7-34
EDIT INSTANCE RESET (Property).....	7-36
ENABLE CONFIGURATION	7-38
ENABLE DATABASE	7-39
ENABLE FAR_SYNC	7-40
ENABLE FAST_START FAILOVER	7-41
ENABLE FAST_START FAILOVER CONDITION	7-43
EXIT.....	7-44
FAILOVER	7-45
HELP	7-48
QUIT	7-49
REINSTATE DATABASE	7-50
REMOVE CONFIGURATION	7-51
REMOVE DATABASE	7-53
REMOVE FAR_SYNC	7-55
REMOVE INSTANCE	7-56
SHOW CONFIGURATION.....	7-57
SHOW CONFIGURATION WHEN PRIMARY IS.....	7-59
SHOW DATABASE.....	7-61
SHOW FAR_SYNC.....	7-64
SHOW FAST_START FAILOVER.....	7-66
SHOW INSTANCE.....	7-67

SHUTDOWN	7-69
SQL	7-71
START OBSERVER	7-72
STARTUP	7-75
STOP OBSERVER	7-77
SWITCHOVER	7-78
VALIDATE DATABASE	7-81
VALIDATE FAR_SYNC	7-87

8 Oracle Data Guard Broker Properties

8.1	Configuration Properties	8-2
8.1.1	BystandersFollowRoleChange	8-3
8.1.2	CommunicationTimeout	8-4
8.1.3	ExternalDestination1	8-4
8.1.4	ExternalDestination2	8-5
8.1.5	FastStartFailoverAutoReinstate	8-6
8.1.6	FastStartFailoverLagLimit	8-7
8.1.7	FastStartFailoverPmyShutdown	8-7
8.1.8	FastStartFailoverThreshold	8-8
8.1.9	ObserverOverride	8-9
8.1.10	ObserverReconnect	8-9
8.1.11	OperationTimeout	8-10
8.1.12	PrimaryLostWriteAction	8-10
8.1.13	TraceLevel	8-11
8.2	Monitorable (Read-Only) Properties	8-12
8.2.1	InconsistentLogXptProps (Inconsistent Redo Transport Properties)	8-12
8.2.2	InconsistentProperties (Inconsistent Properties)	8-13
8.2.3	LogXptStatus (Redo Transport Status)	8-13
8.2.4	LsbyFailedTxnInfo (Logical Standby Failed Transaction Information)	8-14
8.2.5	LsbyParameters (Logical Standby Parameters)	8-14
8.2.6	RecvQEntries (Receive Queue Entries)	8-14
8.2.7	SendQEntries (Send Queue Entries)	8-16
8.2.8	TopWaitEvents	8-17
8.3	Configurable Properties	8-17
8.3.1	AlternateLocation	8-20
8.3.2	ApplyInstanceTimeout	8-21
8.3.3	ApplyLagThreshold	8-21
8.3.4	ApplyParallel	8-22
8.3.5	ArchiveLagTarget	8-22
8.3.6	Binding	8-23
8.3.7	DbFileNameConvert	8-23
8.3.8	DelayMins	8-24
8.3.9	DGConnectIdentifier	8-25
8.3.10	FastStartFailoverTarget	8-26
8.3.11	LogArchiveFormat	8-27
8.3.12	LogArchiveMaxProcesses	8-27

8.3.13	LogArchiveMinSucceedDest.....	8-28
8.3.14	LogArchiveTrace.....	8-28
8.3.15	LogFileNameConvert.....	8-29
8.3.16	LogShipping	8-30
8.3.17	LogXptMode	8-31
8.3.18	LsbyMaxEventsRecorded.....	8-32
8.3.19	LsbyMaxServers.....	8-33
8.3.20	LsbyMaxSga.....	8-33
8.3.21	LsbyPreserveCommitOrder	8-34
8.3.22	LsbyRecordAppliedDdl.....	8-34
8.3.23	LsbyRecordSkipDdl.....	8-35
8.3.24	LsbyRecordSkipErrors	8-35
8.3.25	MaxConnections	8-36
8.3.26	MaxFailure	8-37
8.3.27	NetTimeout.....	8-37
8.3.28	ObserverConnectIdentifier.....	8-38
8.3.29	PreferredApplyInstance.....	8-38
8.3.30	RedoCompression.....	8-39
8.3.31	RedoRoutes.....	8-39
8.3.31.1	Redo Routing Rules.....	8-40
8.3.32	ReopenSecs	8-41
8.3.33	SidName.....	8-42
8.3.34	StandbyArchiveLocation	8-43
8.3.35	StandbyFileManagement.....	8-43
8.3.36	StaticConnectIdentifier	8-44
8.3.37	TransportDisconnectedThreshold.....	8-45
8.3.38	TransportLagThreshold	8-45

9 Troubleshooting Oracle Data Guard

9.1	Sources of Diagnostic Information	9-1
9.2	General Problems and Solutions.....	9-1
9.2.1	ORA-16596: database not part of the Oracle Data Guard broker configuration	9-2
9.2.2	Redo Accumulating on the Primary Is Not Sent to Some Standby Databases.....	9-2
9.2.3	Many Log Files Are Received on a Standby Database But Not Applied	9-3
9.2.4	The Request Timed Out or Cloud Control Performance Is Sluggish.....	9-3
9.2.5	The Primary Database is Flashed Back.....	9-4
9.2.6	Standby Fails to Automatically Start Up Due to Unknown Service (ORA-12514)	9-4
9.3	Troubleshooting Problems During a Switchover Operation.....	9-4
9.4	Troubleshooting Problems During a Failover Operation	9-5
9.4.1	Failed Failovers to Physical Standby Databases	9-5
9.4.1.1	Failed Broker Complete Physical Failovers	9-5
9.4.1.2	Failed Broker Immediate Physical Failovers	9-5
9.4.2	Failed Failovers to Logical Standby Databases	9-5
9.5	Troubleshooting Problems with the Observer	9-6
9.5.1	Problems Starting the Observer.....	9-6
9.5.2	Problems Because the Observer Has Stopped.....	9-7
9.5.3	Capturing Observer Actions in the Observer Log File	9-8

A Oracle Data Guard Broker Upgrading and Downgrading

- A.1 Upgrading from Oracle Database 9i Release 2 (9.2) to Oracle Database 12c Release 1 (12.1)..
A-1
- A.2 Upgrading from Oracle Database 10g and Oracle Database 11g to Oracle Database 12c
Release 1 (12.1) A-2
- A.3 Downgrading from Oracle Database 12c Release 1 (12.1)..... A-4
- A.4 Preserving Broker Configurations During Rolling Upgrades..... A-4

Glossary

Index

List of Examples

4-1	Using the SHOW DATABASE VERBOSE Command to Display Properties.....	4-5
4-2	Using the RedoRoutes Property for Real-Time Cascading.....	4-10
4-3	Using the RedoRoutes Property for Remote Alternate Destinations.....	4-11
4-4	Turn Off Redo Transport Services to All Standby Databases.....	4-13
4-5	Turn Off Redo Transport Services to a Specific Standby Database.....	4-13
6-1	Connecting to the Primary Database on the Local System.....	6-3
6-2	Connecting to the Primary Database on a Remote System.....	6-3
6-3	Altering a Database Property.....	6-14
6-4	Resetting LogXptMode to Its Default.....	6-15
6-5	Resetting TraceLevel to Its Default Value.....	6-15
6-6	Altering a Standby Database State.....	6-15
6-7	Disabling the Configuration and Primary Database.....	6-16
6-8	Disabling a Standby Database.....	6-16
6-9	Disabling a Far Sync Instance.....	6-17
6-10	Validate the Primary Database.....	6-22
6-11	Validate the Logical Standby Database.....	6-22
6-12	Validate the Physical Standby Database.....	6-22

List of Figures

1-1	Oracle Data Guard Broker	1-9
1-2	Databases With Broker (DMON) Processes.....	1-11
3-1	Oracle Data Guard Broker Configuration	3-3
3-2	Broker Configuration Setup in a CFS Area	3-5
3-3	Broker Configuration Setup with Oracle ASM.....	3-6
3-4	Life Cycle of a Broker Configuration and Its Databases	3-7
5-1	Relationship of Primary and Standby Databases and the Observer	5-16
5-2	The Observer in the Fast-Start Failover Environment.....	5-34

List of Tables

1-1	Configuration Management With and Without the Broker	1-3
4-1	Database States and Descriptions	4-2
4-2	Oracle Data Guard Protection Modes and Requirements	4-22
5-1	FS_FAILOVER_STATUS Column of the V\$DATABASE View	5-29
5-2	FS_FAILOVER_OBSERVER_PRESENT Column of the V\$DATABASE View	5-36
7-1	Summary of DGMGRL Commands	7-3
7-2	Examples of Health Conditions	7-43
8-1	Configurable Properties	8-18

Preface

This document provides information about Oracle Data Guard broker, a management and monitoring interface that helps you configure, monitor, and control an Oracle Data Guard broker configuration.

Audience

Oracle Data Guard Broker is intended for database administrators (DBAs) and system administrators who want to use the Oracle Data Guard broker to automate many of the tasks involved in configuring and monitoring an Oracle Data Guard configuration.

The information presented in this book assumes that you are already familiar with Oracle Data Guard, Oracle Enterprise Manager Cloud Control (Cloud Control), Oracle Real Application Clusters (Oracle RAC), and the network services provided by Oracle Net Services.

Documentation Accessibility

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

Access to Oracle Support

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

Related Documents

Refer to the following documentation for more information about Oracle Data Guard:

- *Oracle Data Guard Concepts and Administration*.
- Oracle release notes specific to your operating system.
- Oracle installation guide specific to your operating system.
- For more information about managing Oracle Data Guard through the Cloud Control graphical user interface (GUI), see the Cloud Control online Help.

Conventions

The following text conventions are used in this document:

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

Changes in This Release for Oracle Data Guard Broker

This preface lists changes in *Oracle Data Guard Broker*.

Changes in Oracle Database 12c Release 1 (12.1.0.2)

The following are changes in *Oracle Data Guard Broker* for Oracle Database 12c Release 1 (12.1.0.2).

New Features

The following features are new in this release:

- The `ExternalDestination1` and `ExternalDestination2` configuration properties let you set up a redo transport destination that can receive redo data from the current primary database. See "[ExternalDestination1](#)" on page 8-4 and "[ExternalDestination2](#)" on page 8-5. (These properties are available in Oracle Database 12c Release 1 (12.1.0.2) and later. They are also available in Oracle Database 11g Release 2 (11.2.0.4), but not in Oracle Database 12c Release 1 (12.1.0.1).)
- The new `PrimaryLostWriteAction` configuration property determines what action is taken if a standby database detects that a lost write has occurred at the primary database. See "[PrimaryLostWriteAction](#)" on page 8-10. (This property is available in Oracle Database 12c Release 1 (12.1.0.2) and later. It is also available in Oracle Database 11g Release 2 (11.2.0.4), but not in Oracle Database 12c Release 1 (12.1.0.1).)
- You can now use the information provided in the new `V$FS_OBSERVER_HISTOGRAM` view for help in determining an appropriate value for the `FastStartFailoverThreshold` configuration property. See "[FastStartFailoverThreshold](#)" on page 8-8. (This view is available in Oracle Database 12c Release 1 (12.1.0.2) and later. It is also available in Oracle Database 11g Release 2 (11.2.0.4), but not in Oracle Database 12c Release 1 (12.1.0.1).)
- You can now use the `RedoRoutes` property to specify an alternate destination to ship redo data to if the first destination fails. To see an example of how to do this, see "[Advanced Redo Transport Settings](#)" on page 4-9.
- The new `SHOW CONFIGURATION WHEN PRIMARY IS` command displays the redo transport configuration that would be in effect if the specified database were the primary database. See "[SHOW CONFIGURATION WHEN PRIMARY IS](#)" on page 7-59.

- The new `VALIDATE FAR_SYNC` command performs a comprehensive set of checks to validate a far sync instance. The optional `WHEN PRIMARY` clause allows you to validate a far sync instance based upon a different database being the primary database. See "[VALIDATE FAR_SYNC](#)" on page 7-87.
- As of Oracle Database 12c release 1 (12.1.0.2), Data Guard broker configurations support redo transport from the current primary database to a Zero Data Loss Recovery Appliance (Recovery Appliance) through the use of the `ExternalDestination1` and `ExternalDestination2` configuration properties. Recovery Appliance is an enterprise-level backup solution that provides a single repository for backups of all of your Oracle databases. See "[ExternalDestination1](#)" on page 8-4 and "[ExternalDestination2](#)" on page 8-5. (These properties are available in Oracle Database 12c Release 1 (12.1.0.2) and later. They are also available in Oracle Database 11g Release 2 (11.2.0.4), but not in Oracle Database 12c Release 1 (12.1.0.1).

Other Changes

The following are additional changes in the release:

- An Oracle Data Guard broker configuration can now contain up to 253 members, including one primary database and a combination of standby databases and far sync instances that receive redo directly from the primary database.
- You now need to define and register a static service only if Oracle Clusterware or Oracle Restart is not being used.
- Application Continuity is supported for Oracle Data Guard switchovers to physical standby databases. It is also supported for fast-start failover to physical standbys in maximum availability data protection mode. Note that primary and standby databases must be licensed for Oracle RAC or Oracle Active Data Guard in order to use Application Continuity. See "[Application Continuity](#)" on page 5-44.
- The following properties are no longer available:
 - `LsbyASkipCfgPr`
 - `LsbyASkipErrorCfgPr`
 - `LsbyASkipTxnCfgPr`
 - `LsbyDSkipCfgPr`
 - `LsbyDSkipErrorCfgPr`
 - `LsbyDSkipTxnCfgPr`
 - `LsbySkipTable`
 - `LsbySkipTxnTable`

Changes in Oracle Database 12c Release 1 (12.1.0.1)

The following are changes in *Oracle Data Guard Broker* for Oracle Database 12c Release 1 (12.1.0.1).

New Features

The following features are new in this release:

- Oracle Data Guard broker supports multitenant container databases (CDBs) within a broker configuration.

See ["Oracle Data Guard Broker and CDBs"](#) on page 1-4.

- Support for the new Oracle Data Guard `SYSDG` administrative privilege when using `DGMGRL` to connect to a database.

See ["CONNECT"](#) on page 7-9.

- A new `DGMGRL` command, `VALIDATE DATABASE`, enables you to quickly and easily determine whether a database is ready to perform a role change.

See ["VALIDATE DATABASE"](#) on page 7-81.

- New `DGMGRL` commands enable you to add far sync instances to an Oracle Data Guard configuration and to manage them.

See [Chapter 7](#) for information on commands related to management of far sync instances in a broker configuration.

- A new configurable property, `RedoRoutes`, supports redo cascading and far sync instances.

See ["RedoRoutes"](#) on page 8-39.

- You can use the new `EDIT CONFIGURATION (RENAME)` command to change the name of an Oracle Data Guard broker configuration. You no longer have to remove and re-create the configuration using a different name.

See ["EDIT CONFIGURATION \(RENAME\)"](#) on page 7-24.

- You can now easily reset an Oracle Data Guard broker property back to its default value. You can do this at the configuration, database, or instance level.

See ["Resetting Broker Configurable Properties to Default Values"](#) on page 4-7.

- A new high performance redo transport mode, `FASTSYNC`, supports the maximum availability data protection modes.

See ["LogXptMode"](#) on page 8-31 for information about setting redo transport modes.

- An Oracle Data Guard broker configuration can be preserved during a rolling upgrade so that it does not have to be rebuilt after the upgrade is complete.

See ["Preserving Broker Configurations During Rolling Upgrades"](#) on page A-4.

- You can now execute a SQL statement or a PL/SQL stored procedure from the Data Guard command-line interface (`DGMGRL`) so that you do not have to exit and restart command line interfaces multiple times for routine tasks.

See ["SQL"](#) on page 7-71.

- Oracle Data Guard broker supports the new Oracle Global Data Services (GDS) feature of Oracle Database 12c Release 1 (12.1).

See ["Oracle Data Guard Broker and Oracle Global Data Services"](#) on page 1-5.

See Also:

- *Oracle Database Global Data Services Concepts and Administration Guide* for more information about GDS.
- You can now call the `DBMS_DG` PL/SQL procedure from a fast-start failover target standby database.
See ["Directing a Fast-Start Failover From an Application"](#) on page 5-27.
- You can now resume a switchover after a failure.

See "[Troubleshooting Problems During a Switchover Operation](#)" on page 9-4.

- You can now control the amount of tracing performed by the broker for every database in the configuration.

See the "[TraceLevel](#)" configuration property on page 8-11.

- The new `ObserverReconnect` configuration property allows you to specify an interval at which the observer disconnects and reconnects to the primary and standby database.

See the "[ObserverReconnect](#)" property on page 8-9.

- The new `ObserverOverride` configuration property allows you to specify whether the observer should initiate an automatic failover when it loses connectivity to the primary, even if the standby has a healthy connection to the primary.

See the "[ObserverOverride](#)" configuration property on page 8-9.

- The new `ApplyLagThreshold` configurable property allows you to set a maximum value for the apply lag on physical and logical standbys. A warning is issued if the limit is exceeded.

See the "[ApplyLagThreshold](#)" property on page 8-21.

- The new `TransportDisconnectedThreshold` configurable property allows you to set a limit on the amount of time that can elapse without communication from the primary. A warning is issued if the limit is exceeded.

See the "[TransportDisconnectedThreshold](#)" property on page 8-45.

- The `TransportLagThreshold` configurable property allows you to set a limit for a database's transport lag. A warning is issued if the limit is exceeded.

See the "[TransportLagThreshold](#)" property on page 8-45.

Deprecated Features

The following features are deprecated in this release, and may be desupported in a future release:

- The following configurable properties have been deprecated:
 - `LsbyASkipCfgPr`
 - `LsbyASkipErrorCfgPr`
 - `LsbyASkipTxnCfgPr`
 - `LsbyDSkipCfgPr`
 - `LsbyDSkipErrorCfgPr`
 - `LsbyDSkipTxnCfgPr`
- The following monitorable properties have been deprecated:
 - `LsbySkipTable`
 - `LsbySkipTxnTable`
- Oracle Restart is deprecated. Oracle Restart is a feature provided as part of Oracle Grid infrastructure. It is currently restricted to managing single instance Oracle databases and Oracle ASM instances only, and is subject to desupport in future releases. For more information, see the My Oracle Support note 1584742.1 at <http://support.oracle.com>.

Desupported Features

Some features previously described in this document are desupported in Oracle Database 12c Release 1 (12.1). See *Oracle Database Upgrade Guide* for a list of desupported features.

Other Changes

The following are additional changes in the release:

- You cannot create a broker configuration if the primary database for that configuration has any remote redo transport destinations, unless those remote redo transport destinations have the `NOREGISTER` attribute.

If the primary database has any remote redo transport destinations that do not have the `NOREGISTER` attribute, then you must clear them before proceeding. You can then configure remote redo transport destinations after a broker configuration has been created.

Oracle Data Guard Broker Concepts

This chapter describes the Oracle Data Guard broker, its architecture and components, and how it automates the creation, control, and monitoring of an Oracle Data Guard configuration. The following topics are discussed:

- [Overview of Oracle Data Guard and the Broker](#)
- [Benefits of Oracle Data Guard Broker](#)
- [Oracle Data Guard Broker Components](#)
- [Oracle Data Guard Broker User Interfaces](#)
- [Oracle Data Guard Monitor](#)

See *Oracle Data Guard Concepts and Administration* for the definition of an Oracle Data Guard configuration and for complete information about Oracle Data Guard concepts and terminology.

1.1 Overview of Oracle Data Guard and the Broker

Oracle Data Guard ensures high availability, data protection, and disaster recovery for enterprise data. Oracle Data Guard provides a comprehensive set of services that create, maintain, manage, and monitor one or more standby databases to enable production Oracle databases to survive disasters and data corruptions. Oracle Data Guard maintains these standby databases as transactionally consistent copies of the primary database. If the primary database becomes unavailable because of a planned or an unplanned outage, Oracle Data Guard enables you to switch any standby database to the production role, thus minimizing the downtime associated with the outage. Oracle Data Guard can be used with traditional backup, recovery, and cluster techniques, as well as with the Flashback Database feature to provide a high level of data protection and data availability.

Oracle Data Guard and Oracle RAC One Node

As of Oracle Database 11g Release 2 (11.2.0.2), Oracle Data Guard and Oracle Data Guard broker are fully integrated with Oracle Real Application Clusters One Node (Oracle RAC One Node).

1.1.1 Oracle Data Guard Configurations and Broker Configurations

An *Oracle Data Guard configuration* consists of one primary database and a combination of standby databases and far sync instances that receive redo directly from the primary database. The databases in an Oracle Data Guard configuration are connected by Oracle Net and may be dispersed geographically. There are no restrictions on where the databases are located as long as they can communicate with each other. For

example, you can have a standby database on the same system as the primary database, along with two standby databases on another system.

The Oracle Data Guard broker logically groups these primary and standby databases into a *broker configuration* that allows the broker to manage and monitor them together as an integrated unit. You can manage a broker configuration using either Oracle Enterprise Manager Cloud Control (Cloud Control) or the Oracle Data Guard command-line interface.

1.1.2 Oracle Data Guard Broker

The Oracle Data Guard broker is a distributed management framework that automates and centralizes the creation, maintenance, and monitoring of Oracle Data Guard configurations. The following list describes some of the operations the broker automates and simplifies:

- Creating Oracle Data Guard configurations that include one primary database, new or existing standbys (physical, logical, snapshot, and far sync), and redo transport services and log apply services. The standbys can be Oracle Real Application Clusters (Oracle RAC) databases.
- Adding additional new or existing standbys (physical, logical, snapshot, far sync, Oracle RAC or non-Oracle RAC) to an existing Data Guard configuration.
- Managing the protection mode for the broker configuration.
- Invoking switchover or failover with a single command to initiate and control complex role changes across all databases in the configuration.
- Configuring failover to occur automatically upon loss of the primary database, increasing availability without manual intervention.
- Monitoring the status of the entire configuration, capturing diagnostic information, reporting statistics such as the Redo Apply rate and the redo generation rate, and detecting problems quickly with centralized monitoring, testing, and performance tools.
- Assessing whether a database is ready to become a primary. (See "[VALIDATE DATABASE](#)" on page 7-81).

You can perform all management operations locally or remotely through the broker's easy-to-use interfaces: the Oracle Data Guard management pages in Cloud Control, and the Oracle Data Guard command-line interface called DGMGRL.

These interfaces simplify the configuration and management of an Oracle Data Guard configuration. [Table 1-1](#) provides a comparison of configuration management using the broker's interfaces and using SQL*Plus.

Table 1–1 Configuration Management With and Without the Broker

	With the Broker	Without the Broker
General	Provides primary database, standby database, and far sync instance management as one unified configuration.	You must manage the primary database, standby databases, and far sync instances separately.
Standby Database Creation	Provides Cloud Control wizards that automate and simplify the steps required to create a configuration with an Oracle database on each site, including creating the standby control file, online redo log files, datafiles, and server parameter files.	<p>You must manually (using RMAN or other tools):</p> <ul style="list-style-type: none"> ■ Copy the database files to the standby database system. ■ Create a control file on the standby database system. ■ Create server parameter or initialization parameter files on the standby database system. ■ Copy the primary database password file to the standby database system. <p>Note: Even when you are using the broker, you must ensure that all physical and snapshot standby databases are using a current copy of the password file from the primary database. The copy must be refreshed whenever an administrative privilege (SYSDBA, SYSOPER, SYSDBA, and so on) is granted or revoked, and after the password of any user with administrative privileges is changed.</p>

Table 1–1 (Cont.) Configuration Management With and Without the Broker

	With the Broker	Without the Broker
Configuration and Management	Enables you to configure and manage multiple databases from a single location and automatically unifies all of the databases in the broker configuration.	You must manually connect to multiple locations in order to: <ul style="list-style-type: none"> Set up redo transport services and log apply services on each database in the configuration. Manage the primary database and standby databases individually.
Control	<ul style="list-style-type: none"> Automatically sets up redo transport services and log apply services. Simplifies management of these services, especially in an Oracle RAC environment. Simplifies switchovers, failovers, reinstatements, and conversions to and from a snapshot standby database, allowing you to invoke them through a single command. Automates failover by allowing the broker to determine if failover is necessary and to initiate failover to a specified target standby database, with no need for DBA intervention and with either no loss of data or with a configurable amount of data loss. Integrates role changes with Oracle Clusterware. Manageable and monitorable through Cloud Control Integrates with Oracle Global Data Services, providing support for role-specific global services. 	You must manually: <ul style="list-style-type: none"> Use multiple SQL statements to manage the database. Coordinate sequences of multiple commands across multiple database sites to execute switchover and failover operations. Coordinate sequences of multiple commands to manage services and instances during role transitions.
Monitoring	<ul style="list-style-type: none"> Provides continuous monitoring of the configuration health, database health, and other runtime parameters. Provides a unified updated status and detailed reports. Provides integration with Cloud Control events. 	You must manually: <ul style="list-style-type: none"> Monitor the status and runtime parameters using fixed views on each database—there is no unified view of status for all of the databases in the configuration. Provide a custom method for monitoring Cloud Control events.

1.1.3 Oracle Data Guard Broker and CDBs

Oracle Data Guard broker supports multitenant container databases (CDBs) within a broker configuration. Keep the following in mind:

- When the primary database is a CDB, all standby databases in the broker configuration must also be CDBs.
- When a broker configuration is made up of CDBs, all broker actions execute at the root level and not at the individual pluggable database (PDB) level.
- To administer a multitenant environment you must have the CDB_DBA role.
- After a role change, the broker opens all PDBs on the new primary. The broker also opens all PDBs on a logical standby and a physical standby (if the former physical standby was open for read).

See Also:

- *Oracle Database Concepts* for more information about CDBs
- *Oracle Database Security Guide* for more information about privileges and roles in CDBs and PDBs

1.1.4 Oracle Data Guard Broker and Oracle Global Data Services

To manage broker configurations that support Oracle Global Data Services (GDS), you use both the broker command-line interface, DGMGRL, and the GDS command line interface, GDSCTL. Broker configurations are added to GDS pools (logical groupings of GDS databases which span across GDS regions).

Managing Broker Configurations in a GDS Pool

To add a broker configuration to a GDS pool, use the GDS command-line interface, GDSCTL. A broker configuration can only be added to an empty pool. A broker configuration cannot span multiple pools.

To delete a broker configuration that is in a pool, you must first remove it from the pool by using the `GDSCTL REMOVE BROKERCONFIG . . .` command. Otherwise an error is returned.

Only entire broker configurations are added and removed using GDSCTL; to add and remove individual databases to/from a broker configuration, use the broker command-line interface, DGMGRL.

Managing Individual Configuration Databases When They Are Part of a GDS Pool

If a GDS pool already contains a broker configuration, then only databases that belong to that configuration can be added to the pool. The only way to add or remove individual databases to or from a broker configuration in a pool is to use the broker command-line interface, DGMGRL.

When a database is added to a broker configuration, that database is automatically also added to the GDS pool to which the configuration belongs.

When a database is removed from a broker configuration, it is automatically removed from the GDS pool to which the configuration belongs.

The broker interacts with GDS and notifies it of any role or configuration changes to ensure that the appropriate database services are active and that the appropriate FAN events are published after a role change.

See Also:

- *Oracle Database Global Data Services Concepts and Administration Guide* for more information about GDS.

1.2 Benefits of Oracle Data Guard Broker

The broker's interfaces improve usability and centralize management and monitoring of the Oracle Data Guard configuration. Available as a feature of the Enterprise Edition and Personal Edition of the Oracle database, the broker is also integrated with the Oracle database and Cloud Control. These broker attributes result in the following benefits:

Disaster protection: By automating many of the manual tasks required to configure and monitor an Oracle Data Guard configuration, the broker enhances the high availability, data protection, and disaster protection capabilities that are inherent in

Oracle Data Guard. Access is possible through a client to any system in the Oracle Data Guard configuration, eliminating any single point of failure. If the primary database fails, the broker automates the process for any one of the standby databases to replace the primary database and take over production processing. The database availability that Oracle Data Guard provides makes it easier to protect your data.

Higher availability and scalability with Oracle Real Application Clusters (Oracle RAC) Databases: While Oracle Data Guard broker enhances disaster protection by maintaining transactionally consistent copies of the primary database, Oracle Data Guard, configured with Oracle high availability solutions such as Oracle Real Application Clusters (Oracle RAC) databases, further enhances the availability and scalability of any given copy of that database. The *intrasite* high availability of an Oracle RAC database complements the *intersite* protection that is provided by Oracle Data Guard broker.

Consider that you have a cluster system hosting a primary Oracle RAC database comprised of multiple instances sharing access to that database. Further consider that an unplanned failure has occurred. From an Oracle Data Guard broker perspective, the primary database remains available as long as at least one instance of the clustered database continues to be available for transporting redo data to the standby databases. Oracle Clusterware manages the availability of instances of an Oracle RAC database. It works to rapidly recover failed instances to keep the primary database available. If Oracle Clusterware is unable to recover a failed instance, the broker continues to run automatically with one less instance. If the last instance of the primary database fails, the broker provides a way to fail over to a specified standby database. If the last instance of the primary database fails, and fast-start failover is enabled, the broker can continue to provide high availability by automatically failing over to a pre-determined standby database.

The broker is integrated with Oracle Clusterware so that database role changes occur smoothly and seamlessly. This is especially apparent in the case of a planned role switchover (for example, when a physical standby database is directed to take over the primary role while the former primary database assumes the role of standby). The broker and Oracle Clusterware work together to temporarily suspend service availability on the primary database, accomplish the actual role change for both databases during which Oracle Clusterware works with the broker to properly restart the instances as necessary on the old primary database, and then start services defined on the new primary database. The broker manages the underlying Oracle Data Guard configuration and its database roles while Oracle Clusterware manages service availability that depends upon those roles. Applications that rely on Oracle Clusterware for managing service availability will see only a temporary suspension of service as the role change occurs in the Oracle Data Guard configuration.

Note that while Oracle Clusterware helps to maintain the availability of the individual instances of an Oracle RAC database, the broker coordinates actions that maintain one or more physical or logical copies of the database across multiple geographically dispersed locations to provide disaster protection. Together, the broker and Oracle Clusterware provide a strong foundation for Oracle's high-availability architecture.

See Also: *Oracle Real Application Clusters Administration and Deployment Guide* for information about Oracle Clusterware

Support for Oracle RAC One Node: As of Oracle Database 11g Release 2 (11.2.0.2), Oracle Data Guard and Oracle Data Guard broker are fully integrated with Oracle Real Application Clusters One Node (Oracle RAC One Node). This support provides a higher level of availability at the local site by leveraging many of the Oracle Clusterware benefits just noted for Oracle RAC configurations.

Automated creation of an Oracle Data Guard configuration: The broker helps you to logically define and create an Oracle Data Guard configuration. The broker automatically communicates between the members of an Oracle Data Guard configuration using Oracle Net Services. The members can be local or remote, connected by a LAN or geographically dispersed over a WAN.

Cloud Control provides a wizard that automates the complex tasks involved in creating a broker configuration, including:

- Adding an existing standby database, or a new standby database created from existing backups taken through Cloud Control
- Configuring the standby control file, server parameter file, and datafiles
- Initializing communication with the standbys
- Creating standby redo log files
- Enabling Flashback Database if you plan to use fast-start failover

Although DGMGRL cannot automatically create a new standby, you can use DGMGRL commands to configure and monitor an existing standby, including those created using Cloud Control.

Easy configuration of additional standbys: After you create an Oracle Data Guard configuration, you can add new or existing standbys to each Oracle Data Guard configuration. Cloud Control provides an Add Standby Database wizard to guide you through the process of adding more databases.

Simplified, centralized, and extended management: You can issue commands to manage many aspects of the broker configuration. These include:

- Simplify the management of all components of the configuration, including the primary database, standbys, redo transport services, and log apply services.
- Coordinate database state transitions and update database properties dynamically, with the broker recording the changes in a broker configuration file that includes information about all the databases in the configuration. The broker propagates the changes to all databases in the configuration and their server parameter files.
- Simplify the control of the configuration protection modes (to maximize protection, to maximize availability, or to maximize performance).
- Invoke the Cloud Control verify operation to ensure that redo transport services and log apply services are configured and functioning properly.

Simplified switchover and failover operations: The broker simplifies switchovers and failovers by allowing you to invoke them using a single key click in Cloud Control or a single command at the DGMGRL command-line interface (referred to in this documentation as *manual* failover). For *lights-out administration*, you can enable fast-start failover to allow the broker to determine if a failover is necessary and to initiate the failover to a pre-specified target standby automatically, with no need for DBA intervention. Fast-start failover can be configured to occur with no data loss or with a configurable amount of data loss.

Fast-start failover allows you to increase availability with less need for manual intervention, thereby reducing management costs. Manual failover gives you control over exactly when a failover occurs and to which target standby. Regardless of the method you choose, the broker coordinates the role transition on all databases in the configuration. Once failover is complete, the broker publishes a Fast Application Notification (FAN) event to notify applications that the new primary is available.

Note that you can use the `DBMS_DG` PL/SQL package to enable an application to initiate a fast-start failover when it encounters specific conditions. See [Section 4.8.3](#) for more information.

Only one command is required to initiate complex role changes for switchover or failover operations across all databases in the configuration. The broker automates switchover and failover to a specified standby database in the broker configuration. Cloud Control enables you to select a new primary database from a set of viable standby databases (enabled and running, with normal status). The DGMGRL `SWITCHOVER` and `FAILOVER` commands only require you to specify the target standby database before automatically initiating and completing the many steps in switchover or failover operations across the multiple databases in the configuration.

Built-in monitoring and alert and control mechanisms: The broker provides built-in validation that monitors the *health* of all of the databases in the configuration. While connected to any database in the configuration, you can capture diagnostic information and detect obvious and subtle problems quickly with centralized monitoring, testing, and performance tools. Both Cloud Control and DGMGRL retrieve a complete configuration view of the progress of redo transport services on the primary database and the progress of Redo Apply or SQL Apply on the standby database.

The ability to monitor local and remote databases and respond to events is significantly enhanced by the broker's health check mechanism and tight integration with the Cloud Control event management system.

Transparent to application: Use of the broker is possible for any database because the broker works transparently with applications; no application code changes are required to accommodate a configuration that you manage with the broker.

1.3 Oracle Data Guard Broker Components

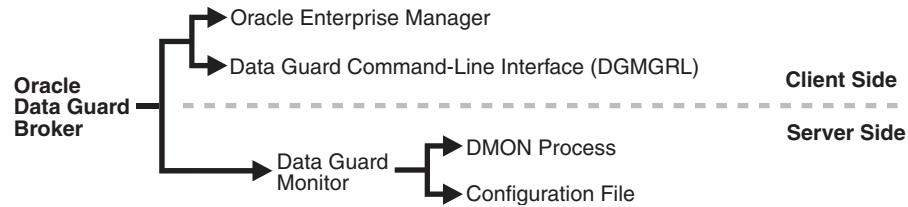
The Oracle Data Guard broker consists of the following components:

- [Oracle Enterprise Manager Cloud Control](#)
- [Oracle Data Guard Command-Line Interface \(DGMGRL\)](#)
- [Oracle Data Guard Monitor](#)

Cloud Control and the Oracle Data Guard command-line interface (DGMGRL) are the broker client interfaces that help you define and manage a configuration consisting of a collection of primary and standby databases. DGMGRL also includes commands to create an **observer**, a process that facilitates fast-start failover. [Section 1.4](#) describes these interfaces in more detail.

The Oracle Data Guard monitor is the broker server-side component that is integrated with the Oracle database. The Oracle Data Guard monitor is composed of several processes, including the DMON process, and broker configuration files that allow you to control the databases of that configuration, modify their behavior at runtime, monitor the overall health of the configuration, and provide notification of other operational characteristics. [Section 1.5](#) describes the Oracle Data Guard monitor in more detail.

[Figure 1–1](#) shows these components of the broker.

Figure 1–1 Oracle Data Guard Broker

1.4 Oracle Data Guard Broker User Interfaces

You can use either of the broker's user interfaces to create a broker configuration and to control and monitor the configuration. The following sections describe the broker's user interfaces:

- [Oracle Enterprise Manager Cloud Control](#)
- [Oracle Data Guard Command-Line Interface \(DGMGRL\)](#)

1.4.1 Oracle Enterprise Manager Cloud Control

Oracle Enterprise Manager Cloud Control (Cloud Control) works with the Oracle Data Guard monitor to automate and simplify the management of an Oracle Data Guard configuration.

With Cloud Control, the complex operations of creating and managing standby databases are simplified through Oracle Data Guard management pages and wizards, including:

- An Add Standby Database wizard that helps you to create a broker configuration, if one does not already exist, having a primary database and a local or remote standby database. The wizard can create a physical, snapshot, or logical standby database or import an existing physical, snapshot, or logical (Oracle RAC or non-Oracle RAC) standby database. If the wizard creates a physical, snapshot, or logical standby database, the wizard also automates the creation of the standby control file, server parameter file, online and standby redo log files, and the standby datafiles.
- A switchover operation that helps you switch roles between the primary database and a standby database.
- A failover operation that changes one of the standby databases to the role of a primary database.
- Performance tools and graphs that help you monitor and tune redo transport services and log apply services.
- Property pages that allow you to set database properties on any database and, if applicable, the settings are immediately propagated to all other databases and server parameter files in the configuration.
- Event reporting through e-mail.

In addition, it makes all Oracle Net Services configuration changes necessary to support redo transport services and log apply services.

See Also: My Oracle Support note 787461.1 at <http://support.oracle.com> for information about which versions of Cloud Control are required to manage the full set of Oracle Data Guard features in various Oracle Database releases

1.4.2 Oracle Data Guard Command-Line Interface (DGMGRL)

The Oracle Data Guard command-line interface (DGMGRL) enables you to control and monitor an Oracle Data Guard configuration from the DGMGRL prompt or within scripts. You can perform most of the activities required to manage and monitor the databases in the configuration using DGMGRL commands.

DGMGRL also includes commands to create an observer process that continuously monitors the primary and target standby databases and evaluates whether failover is necessary, and then initiates a fast-start failover when conditions warrant.

See Also: [Chapter 7](#) for complete reference information for the Oracle Data Guard command-line interface. [Table 7-1](#) provides a summary of DGMGRL commands.

1.5 Oracle Data Guard Monitor

The configuration, control, and monitoring functions of the broker are implemented by server-side software and configuration files that are maintained for each database that the broker manages. The software is called the Oracle Data Guard monitor.

The following sections describe how the Oracle Data Guard monitor interacts with the Oracle database and with remote Oracle Data Guard monitors to manage the broker configuration.

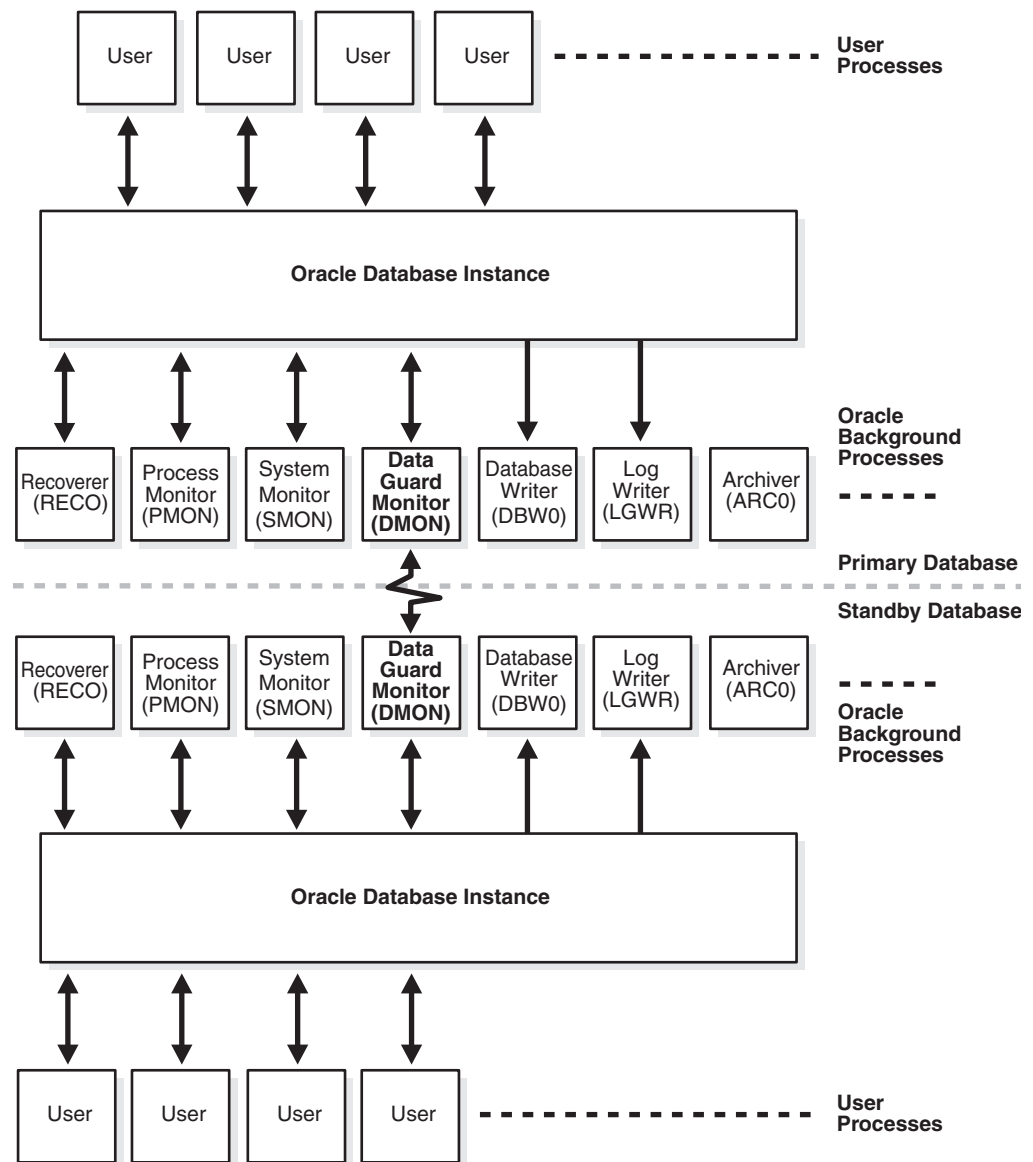
1.5.1 Oracle Data Guard Monitor (DMON) Process

The Oracle Data Guard monitor process (DMON) is an Oracle background process that runs on every database instance that is managed by the broker. When you start the Oracle Data Guard broker, a DMON process is created.

See Also: [Section 3.4](#) on page 3-6 for information on starting the broker

Whether you use Cloud Control or DGMGRL to manage a database, the DMON process is the server-side component that interacts with the local database and the DMON processes of the other databases to perform the requested function. The DMON process is also responsible for monitoring the health of the broker configuration and for ensuring that every database has a consistent description of the configuration.

[Figure 1-2](#) shows the broker's DMON process as one of several background processes that constitute an instance of the Oracle database. Each database instance shown in the figure has its own DMON process.

Figure 1–2 Databases With Broker (DMON) Processes

The zigzag arrow in the center of [Figure 1–2](#) represents the two-way Oracle Net Services communication channel that exists between the DMON processes of two databases in the same broker configuration.

This two-way communication channel is used to pass requests between databases and to monitor the health of all of the databases in the broker configuration.

1.5.2 Configuration Management

The broker's DMON process persistently maintains information about all members of the broker configuration in a binary configuration file. A copy of this file is maintained by the DMON process for each of the databases that belong to the broker configuration. If it is an Oracle RAC database, each database's copy of the file is shared by all instances of the database.

This configuration file contains information that describes the states and properties of the databases in the configuration. For example, the file records the databases that are part of the configuration, the roles and properties of each of the databases, and the state of each database in the configuration.

The configuration data is managed transparently by the DMON process to ensure that the configuration information is kept consistent across all of the databases. The broker uses the data in the configuration file to configure and start the databases, control each database's behavior, and provide information to DGMGRL and Cloud Control.

See Also: [Section 4.3](#) on page 4-4 for more information

Whenever you add databases to a broker configuration, or make a change to an existing database's properties, each DMON process records the new information in its copy of the configuration file.

1.5.3 Database Property Management

Associated with each database are various properties that the DMON process uses to control the database's behavior. The properties are recorded in the configuration file. Many database properties are used to control database initialization parameters related to the Oracle Data Guard environment.

To ensure that the broker can update the values of parameters in both the database itself and in the configuration file, you must use a server parameter file to control static and dynamic initialization parameters. The use of a server parameter file gives the broker a mechanism that allows it to reconcile property values selected by the database administrator (DBA) when using the broker with any related initialization parameter values recorded in the server parameter file.

When you set values for database properties in the broker configuration, the broker records the change in the configuration file and propagates the change to all of the databases in the Oracle Data Guard configuration.

Note: The broker supports both the default and nondefault server parameter file filenames. If you use a nondefault server parameter filename, the initialization parameter file must include the complete filename and location of the server parameter file. If this is an Oracle RAC database, there must be one nondefault server parameter file for all instances.

See Also: [Section 4.3.2](#) on page 4-6 for more information

Oracle Data Guard Installation

This chapter includes the following topics:

- [Oracle Data Guard Installation](#)
- [Prerequisites](#)

See Also: [Appendix A](#) for help with upgrading to a new release in an Oracle Data Guard broker configuration

2.1 Oracle Data Guard Installation

Oracle Data Guard is included with the Enterprise Edition and Personal Edition of the Oracle database software. You can manage an Oracle Data Guard configuration by using either SQL*Plus, the Oracle Data Guard broker's command-line interface (DGMGRL), or a compatible version of Oracle Enterprise Manager Cloud Control (Cloud Control).

Install the Oracle Enterprise Edition or Personal Edition database software on each location you expect to include in broker configurations. You must install a compatible version of Cloud Control if you want to use the graphical user interface for Oracle Data Guard.

In addition, to use fast-start failover you must install DGMGRL and run the observer software. Oracle recommends running the observer on a computer system that is separate from the primary and standby systems. To install DGMGRL on the observer computer, use one of the methods described in the following list:

- Install the complete Oracle Client Administrator by choosing the `Administrator` option from Oracle Universal Installer.

This installation includes DGMGRL but it does not include the Cloud Control agent. This allows you to manage the observer using DGMGRL commands but not Cloud Control. (The Cloud Control agent can be installed separately.)

- Install the full Oracle Database Enterprise Edition or Personal Edition software kit

This installation includes DGMGRL and the Cloud Control agent, allowing you to manage the observer using Cloud Control or DGMGRL commands.

Note: The observer can be run from any platform that supports it, and that platform can be different from the platform of the primary or target standby database.

2.2 Prerequisites

The following conditions must be met before you can use the broker:

- The primary and standby databases must be using the same version of Oracle Database and each can be installed in either a single-instance or multi-instance environment. The database must be licensed for Oracle Enterprise Edition or Personal Edition.
- You must use a server parameter file (SPFILE) to ensure the broker can persistently reconcile values between broker properties and any related initialization parameter values. See [Section 4.3.2](#) for more information.

If any of the databases in the configuration is an Oracle RAC database, you must configure the server parameter file (SPFILE) appropriately for use in an Oracle RAC environment.

See Also: *Oracle Real Application Clusters Administration and Deployment Guide* for information about initialization files in an Oracle RAC

- The value of the `DG_BROKER_START` initialization parameter must be set to `TRUE`. See [Section 3.4](#) for more information. (Cloud Control sets this parameter automatically.)
- If any of the databases in the configuration is an Oracle RAC database, you must set up the `DG_BROKER_CONFIG_FILEn` initialization parameters for that database such that they point to the same shared files for all instances of that database. The shared files could be files on a cluster file system, if available, or stored using Oracle Automatic Storage Management (Oracle ASM).

See Also: Configuration file information in [Section 1.5.2](#). Also, see [Section 3.3](#) for details about setting up the broker configuration file.

- Oracle Net Services network configuration files must be set up on the primary database and on the standby database if you configure an existing standby database into the broker configuration. Cloud Control can assist you in creating the configuration files when creating a standby database.

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

- In a broker configuration, you use the `DGConnectIdentifier` property to specify a connect identifier for each database. The connect identifier for a database must:
 - Allow all other databases in the configuration to reach it.
 - Allow all instances of an Oracle RAC database to be reached.
 - Specify a service that all instances dynamically register with the listeners so that connect-time failover on an Oracle RAC database is possible.
 - Have failover attributes set to allow the primary database's Redo Transport Services to continue shipping redo data to an Oracle RAC standby database, even if the receiving instance of that standby database has failed.

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

- To enable DGMGRL to restart instances during the course of broker operations, a static service must be registered with the local listener of each instance. A static service registration is also required to enable the observer to restart instances as part of automatic reinstatement of the old primary database after a fast-start failover has occurred. A static service needs to be defined and registered only if Oracle Clusterware or Oracle Restart is not being used.

By default, the broker assumes a static service name of `db_unique_name_DGMGRL.db_domain` and expects the listener has been started with the following content in the `listener.ora` file:

```
LISTENER = (DESCRIPTION =
  (ADDRESS_LIST = (ADDRESS = (PROTOCOL = tcp) (HOST = host_name)
    (PORT = port_num)))
  SID_LIST_LISTENER = (SID_LIST = (SID_DESC = (SID_NAME = sid_name)
    (GLOBAL_DBNAME = db_unique_name_DGMGRL.db_domain)
    (ORACLE_HOME = oracle_home)
    (ENVS = "TNS_ADMIN=oracle_home/network/admin"))))
```

Alternatively, you can use a different static service name. If you do, be sure to modify the `StaticConnectIdentifier` instance-specific property to reflect the static service that has been registered.

See Also:

- See *Oracle Database Net Services Administrator's Guide* for more information about configuring static services
- See the My Oracle Support Note 1387859.1 at <http://support.oracle.com> for additional information about using static services and the `StaticConnectIdentifier` configurable property
- The primary database must be opened in ARCHIVELOG mode.
- You must set the `COMPATIBLE` initialization parameter to 11.2 or higher for both the primary and standby databases. However, if you want to take advantage of new Oracle Database 12c Release 1 (12.1) features, then set the `COMPATIBLE` parameter to 12.1 on all databases within the Oracle Data Guard configuration.

For the broker to work, the `COMPATIBLE` initialization parameter must be set to the same value on both the primary and standby databases. If the values differ, redo transport services may be unable to transmit redo data from the primary database to the standby databases.

- Select and configure a redo transport authentication method, as described in *Oracle Data Guard Concepts and Administration*.
- As of Oracle Database 12c Release 1 (12.1), for all databases to be added to a broker configuration, any `LOG_ARCHIVE_DEST_n` parameters that have the `SERVICE` attribute set, but not the `NOREGISTER` attribute, must be cleared.

See Also:

- [Section 3.4](#) for more information about preparing and starting Oracle Data Guard broker

Managing Broker Configurations

This chapter discusses how to manage the broker configuration itself; for information about managing the components of a broker configuration, see [Chapter 4](#). This chapter contains the following topics:

- [Configuration Support](#)
- [Configuration Properties](#)
- [Setting Up the Broker Configuration Files](#)
- [Starting the Data Guard Broker](#)
- [Management Cycle of a Broker Configuration](#)
- [Enable and Disable Operations](#)
- [Configuration Status](#)

3.1 Configuration Support

Oracle Data Guard broker enables you to create a broker configuration consisting of up to 253 members, including one primary database and a combination of standby databases and far sync instances that receive redo directly from the primary database. The broker controls the member of the configuration, monitors the health of the configuration, and reports health and other operational characteristics through the Oracle Enterprise Management notification mechanisms if you are using Oracle Enterprise Manager Cloud Control, or through `SHOW` commands if you are using DGMGRL.

A supported Oracle Data Guard configuration contains the following components:

- A primary database (Oracle RAC or non-Oracle RAC)
- A combination of standby databases and far sync instances that receive redo directly from the primary database.
- Oracle Net Services network configuration that defines a connection between the databases
- Standby (archived redo log files) destination parameters and configuration properties
- Redo transport services that transmit the redo data from the primary database to the standbys and far sync instances
- Log apply services that apply redo data to the standby databases from the archived redo log files or standby redo log files

The Oracle Data Guard log apply services update standby databases with redo data that is transmitted automatically from the primary database by redo transport services. The archived redo log files and standby redo log files contain all of the database changes except for *unrecoverable* or *unlogged* changes.

- On physical standby databases, Redo Apply applies the redo data to keep the standby consistent with the primary database.
- On logical standby databases, SQL Apply applies the redo data to keep the standby consistent with the primary database.
- On snapshot standby databases, the redo data is received but not applied until the snapshot standby database is converted back to a physical standby database.
- On far sync instances, the redo data is received and then forwarded to a physical standby database. A far sync instance does not have data files and does not apply any of the redo data it has received.

The broker's Oracle Data Guard monitor (DMON) process configures and maintains the broker configuration as a group of objects that you can manage and monitor as a single unit. Thus, when you enter a command that affects multiple databases, the DMON process:

- Carries out your request on the primary database
- Coordinates with the DMON process for each of the other databases, as required for your request
- Updates the configuration file on the local system
- Communicates with the DMON process for each of the other databases to update their copies of the configuration file

Through the DMON process, you can configure, monitor, and control the databases and the configuration together as a unit. If you disable the configuration, broker management of all of the databases in the configuration is also disabled. If you later enable the configuration, broker management is enabled for each database in the configuration.

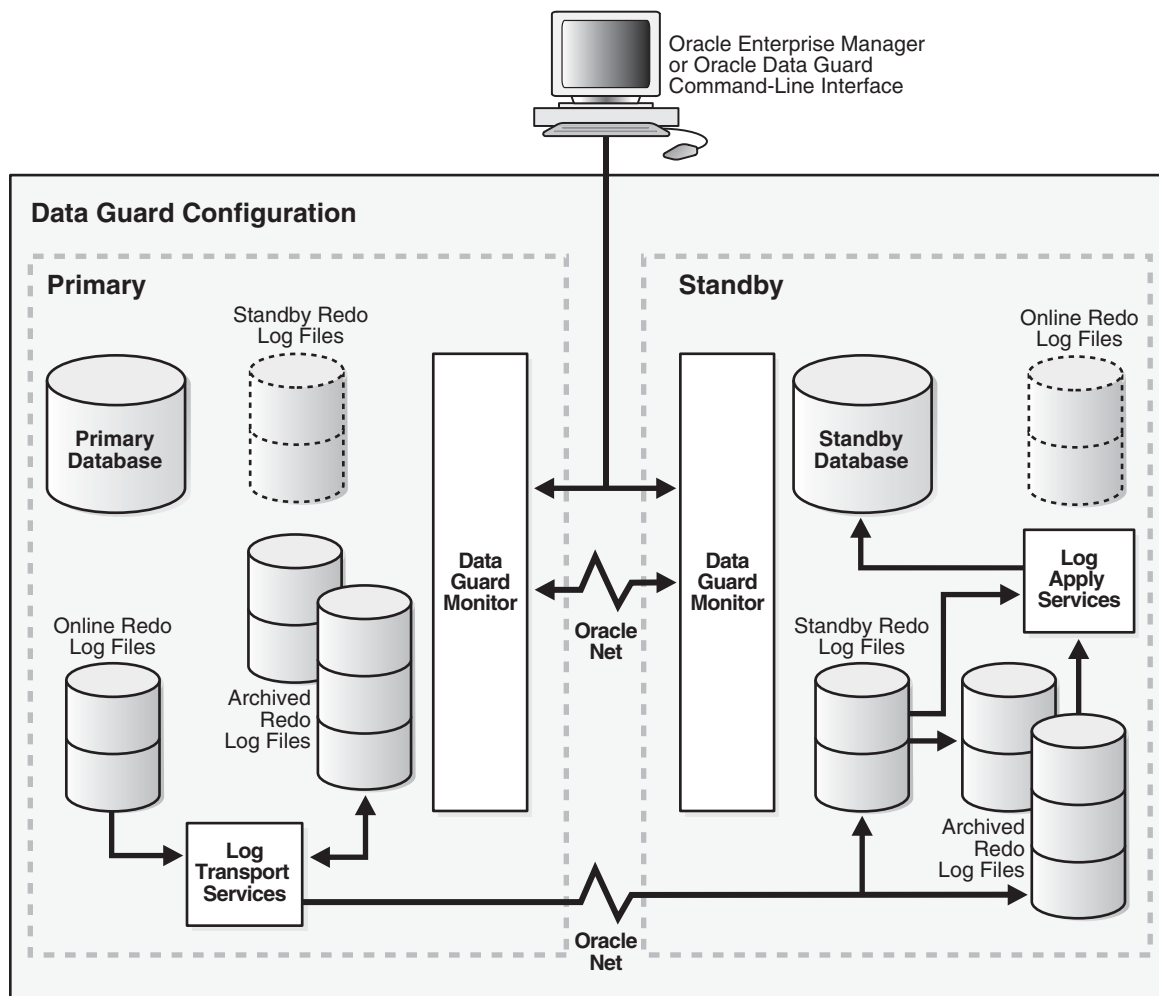
[Figure 3–1](#) shows a broker configuration with a primary database and physical standby database.

On the primary database, the figure shows the redo transport services in addition to the following main components: the primary database, DMON, the online redo log files, and the archived redo log files. The figure also shows standby redo log files in outline form on the primary side; the standby redo logs are outlined to indicate they are currently inactive but have been configured in preparation for a switchover or failover to the standby role.

The physical standby database includes the following components: a standby database, log apply services, DMON, archived redo log files, and standby redo log files. The online redo log files on the physical standby database are outlined to indicate they are currently inactive but have been configured in preparation for a switchover or failover to the primary role.

See Also: *Oracle Data Guard Concepts and Administration* for more information about standby databases

Figure 3–1 Oracle Data Guard Broker Configuration



3.2 Configuration Properties

Configuration properties control the behavior of the broker configuration. You can view and dynamically update the values of these properties using either DGMGRL or Cloud Control. However, some properties can only be updated through DGMGRL.

A configuration property has configuration-wide scope; meaning that the value you set for the property applies uniformly to each database in the configuration.

See Also: [Chapter 8](#) for complete descriptions of all broker configuration properties

3.3 Setting Up the Broker Configuration Files

Two copies of the configuration file are maintained for each database so as to always have a record of the last known *valid* state of the configuration. When the broker is started for the first time, the configuration files are automatically created and named using a default path name and filename that is operating-system specific. You can override this default path name and filename by setting the following initialization parameters for that database:

```
DG_BROKER_CONFIG_FILE1
DG_BROKER_CONFIG_FILE2
```

Note the following restrictions when setting the `DG_BROKER_CONFIG_FILE1` and `DG_BROKER_CONFIG_FILE2` initialization parameters:

- These parameters can only be set or changed when the Oracle Data Guard broker is not running (`DG_BROKER_START=FALSE`).
- These parameters must specify an Oracle ASM, Oracle OCFS, or NFS that is shared for all Oracle RAC instances.

The Oracle Data Guard broker works with databases that use either Oracle managed or user managed datafiles. These datafiles can reside on a file system or an Oracle ASM disk group. The following section contains these topics:

- [Renaming the Broker Configuration Files](#)
- [Managing Broker Configuration Files in an Oracle RAC Environment](#)

3.3.1 Renaming the Broker Configuration Files

You can change the configuration filenames dynamically by issuing the `ALTER SYSTEM SQL` statement. However, you cannot alter these parameters when the broker's `DMON` process is running. To change the names of these configuration files for a given database, perform the following steps:

1. Disable the broker configuration using the `DGMGRL DISABLE` command. See [Section 3.6](#) on page 3-9.
2. Stop the Oracle Data Guard broker `DMON` process using the following SQL statement:

```
SQL> ALTER SYSTEM SET DG_BROKER_START=FALSE;
```

3. Change the configuration filenames for the database:

```
SQL> ALTER SYSTEM SET DG_BROKER_CONFIG_FILE1=filespec1;  
SQL> ALTER SYSTEM SET DG_BROKER_CONFIG_FILE2=filespec2;
```

Note: If the broker is managing an Oracle RAC database, the value of `DG_BROKER_CONFIG_FILE1` and the value of `DG_BROKER_CONFIG_FILE2` for each of the instances must point to the same set of physical files.

4. The method of moving the files depends upon where they currently reside and where you want to move them to:
 - If the files reside on an operating system file system, use operating system commands to move the files to their new location.
 - If the old or new location is an Oracle ASM disk group, use the `DBMS_FILE_TRANSFER.COPY_FILE` function to transfer the files to their new location.
5. Restart the Oracle Data Guard broker `DMON` process, as follows:


```
SQL> ALTER SYSTEM SET DG_BROKER_START=TRUE;
```
6. Enable the broker configuration using the `DGMGRL ENABLE` command or the Enable operation in the Oracle Data Guard management pages of Cloud Control.

Note: As of Release 11.2, the files can reside on disks having any supported sector size (physical block size) up to and including 4KB sectors. Configuration files that were generated by a release prior to 11.2 are restricted to reside only on disks having the same sector size as was used at the time the files were first created. These files must first be upgraded to Release 11.2 before they can be moved to a location having a different sector size, or are otherwise expected to reside across a mixture of sector sizes in a given broker configuration. See [Section A.2](#) for more information.

3.3.2 Managing Broker Configuration Files in an Oracle RAC Environment

If the broker is managing an Oracle RAC database, the value of `DG_BROKER_CONFIG_FILE1` and the value of `DG_BROKER_CONFIG_FILE2` for each of the instances must point to the same set of physical files. In other words, all instances of the database must reference the same set of configuration files. The configuration files can be deployed using one of the following methods:

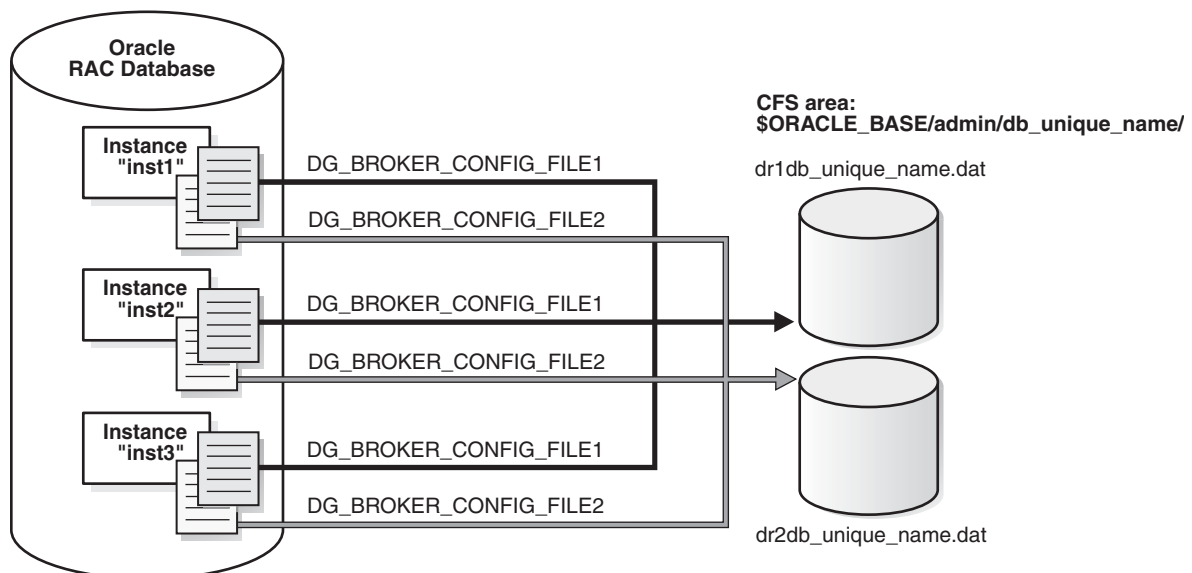
- [Using Cluster File System \(CFS\) for Configuration Files](#)
- [Using Oracle ASM Disk Groups for Configuration Files](#)

3.3.2.1 Using Cluster File System (CFS) for Configuration Files

If cluster file system (CFS) is available, and the configuration files reside there, the `DG_BROKER_CONFIG_FILEn` parameters on all of the instances must be set to these files including the path to the CFS area. [Figure 3–2](#) shows the set up for the broker configuration files on CFS. In this scenario, the parameters and value for all instances would be:

```
DG_BROKER_CONFIG_FILE1=$ORACLE_BASE/admin/db_unique_name/dr1db_unique_name.dat
DG_BROKER_CONFIG_FILE2=$ORACLE_BASE/admin/db_unique_name/dr2db_unique_name.dat
```

Figure 3–2 Broker Configuration Setup in a CFS Area

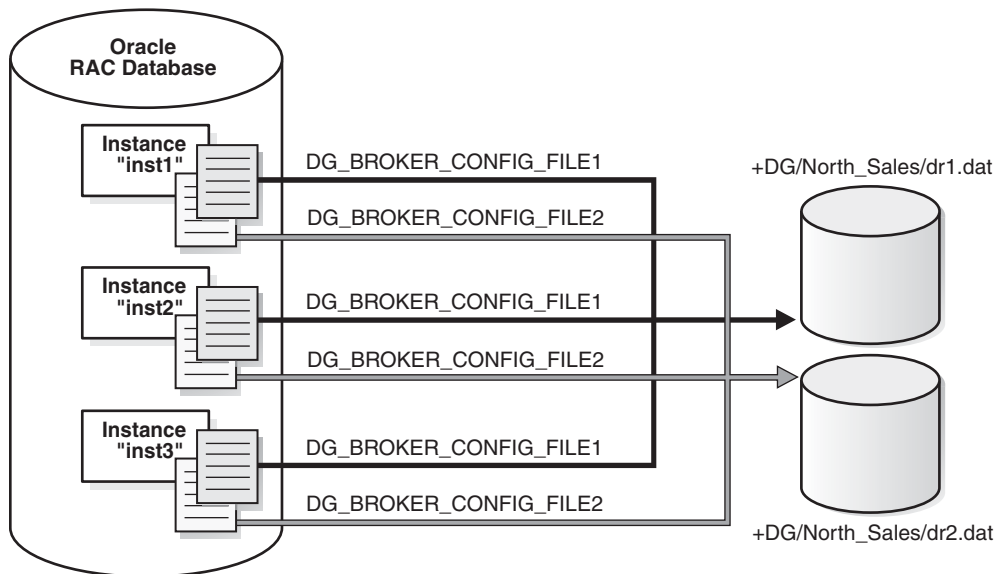


3.3.2.2 Using Oracle ASM Disk Groups for Configuration Files

The broker's configuration files can also reside on an Oracle ASM disk group. [Figure 3–3](#) shows the setup for the broker configuration files on Oracle ASM devices. In this scenario, the parameters and values would be specified, as follows:

```
ALTER SYSTEM SET DG_BROKER_CONFIG_FILE1 = '+DG/DIRECTORY/DR1.DAT' SCOPE=BOTH;
ALTER SYSTEM SET DG_BROKER_CONFIG_FILE2 = '+DG/DIRECTORY/DR2.DAT' SCOPE=BOTH;
```

Figure 3–3 Broker Configuration Setup with Oracle ASM



Because the configuration files must be explicitly named by the user, these configuration files are not Oracle Managed Files (OMF).

To create the broker's configuration files on an Oracle ASM disk group, set the `DG_BROKER_CONFIG_FILE1` and `DG_BROKER_CONFIG_FILE2` initialization parameters to a string value that includes the name of an existing Oracle ASM disk group, an existing directory in that disk group, and the name of the configuration file itself.

3.4 Starting the Data Guard Broker

After setting up the configuration files, the `DG_BROKER_START` initialization parameter must be set to `TRUE` for each database to start the `DMON` processes.

By default, the `DG_BROKER_START` initialization parameter is set to `FALSE`. However, you can set the value in the following ways:

- If you are using Cloud Control, it automatically sets the `DG_BROKER_START` initialization parameter to `TRUE` for new standby databases that it creates.
- If you are using DGMGRL, you must explicitly set the `DG_BROKER_START` initialization parameter to `TRUE`; otherwise, the broker will not start. You can set the `DG_BROKER_START` initialization parameter with the following SQL statement:

```
SQL> ALTER SYSTEM SET DG_BROKER_START=TRUE;
```

```
System altered.
```

```
SQL> SHOW PARAMETER DG_BROKER_START
```

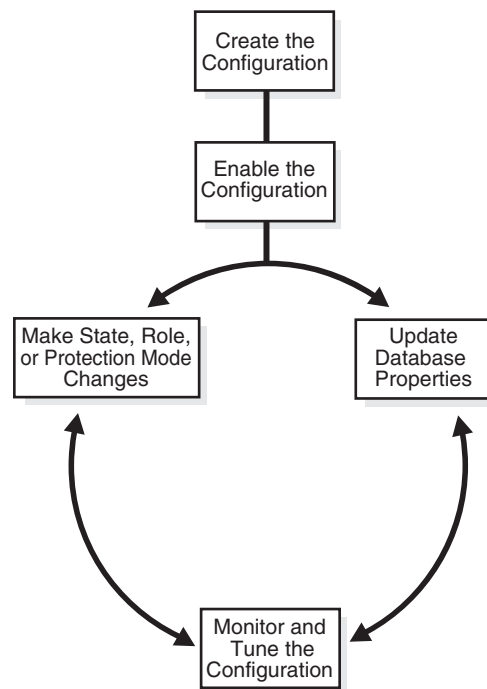
NAME	TYPE	VALUE
dg_broker_start	boolean	TRUE

Whether you use Cloud Control or DGMGRL, set the value of the `DG_BROKER_START` initialization parameter to `TRUE` in the server parameter file on each primary and standby database. Doing so ensures that Oracle Data Guard broker will start automatically the next time you start any instance of the database.

3.5 Management Cycle of a Broker Configuration

The broker helps you to create a new configuration or manage an existing configuration. [Figure 3-4](#) shows the life cycle of a broker configuration.

Figure 3-4 Life Cycle of a Broker Configuration and Its Databases



Create the Broker Configuration

When using Cloud Control, the Add Standby Database wizard can either add an existing (Oracle RAC or non-Oracle RAC) standby database into the configuration or create a new (Oracle RAC or non-Oracle RAC) standby database and add it to the configuration. The standby database can be a physical, logical, or snapshot database.

When using DGMGRL, the primary database and a standby database must already exist. You construct the standby database from backups of the primary database control files and datafiles, and then prepare it for recovery.

Enable the Broker Configuration

An Oracle Data Guard configuration must be enabled to be managed or monitored by the broker. Conversely, you disable a configuration if you no longer want to manage it with the broker. When you disable a configuration, broker management of all of its databases is also disabled.

Note: You can enable or disable the configuration using DGMGRL. You cannot disable the configuration using Cloud Control. You can enable the configuration using Cloud Control in the event that it was previously disabled using DGMGRL.

A broker configuration, when first created using Cloud Control, is automatically enabled as soon as the Add Standby Database wizard completes.

A broker configuration, when first created using DGMGRL, is in a disabled condition. This means its constituent databases are not yet under active control of the broker. When you finish configuring the databases into a broker configuration with DGMGRL, you must enable the configuration to allow the broker to manage the configuration.

You can enable:

- The entire configuration, including all of its databases
- An individual standby database

You can easily disable a database if a problem occurs such that it cannot function properly in a broker configuration. Note that you cannot disable the primary database. You must disable the entire configuration to disable the primary database.

You may also want to disable a configuration temporarily, and then change some properties in the broker configuration without affecting the actual database properties. The changed properties will take effect when the configuration is enabled again for management by the broker.

Make Role Changes Within the Broker Configuration, As Needed

At any time, you can issue a single command to change the roles of the databases in the configuration. If some event renders the primary database unusable, you can fail over one of the standby databases to become the new primary database.

If Flashback Database was enabled on the former primary database, then after failover has completed, you can reinstate the former primary database as a standby database for the new primary database.

In addition, planned downtime for maintenance can be reduced because you can quickly switch over production processing from the current primary database to a standby database, and then switch back again after the planned maintenance.

See Also: [Chapter 5](#) for more information about role changes

Convert to Snapshot Standby Database

At any time, you can issue a single command to convert a physical standby database to a snapshot standby database. The snapshot standby database is a fully updatable database that receives redo data generated from the primary database, but does not apply it.

Once you are done using the snapshot standby database, you can again issue a single command to convert it to a physical standby database. After the conversion to a physical standby has completed, Redo Apply services will start (assuming the state is `APPLY-ON`) and apply all of the accumulated redo data.

Make State Changes to the Databases, As Needed

When you enable a configuration for the first time, the broker, by default, starts redo transport services on the primary and starts log apply services on the standby (except for a snapshot standby).

At any time, you can issue a single command through Cloud Control or DGMGRL to change the state of the database. For example, you could bring the primary database into the `TRANSPORT-OFF` state to temporarily stop sending redo data to the standby databases. Then, to resume sending redo data to the standbys, you could bring the primary database into the `TRANSPORT-ON` state.

See Also: [Chapter 4](#) for more information about database state changes

Update Database Properties, As Needed

The broker enables you to set database properties, some of which correspond to database initialization parameters. You can change these properties to dynamically control such things as redo transport, standby file management, log apply, and to support the overall configuration protection mode. The broker records the changes in the broker configuration file for each database in the Oracle Data Guard configuration and propagates the changes to the related initialization parameters in the server parameter files, if needed.

See Also: [Chapter 4](#) and [Chapter 8](#) for complete information about database properties

Set Data Protection Modes, As Needed

The broker enables you to set the data protection mode for the configuration. You can configure the protection mode to maximize data protection, maximize availability, or maximize performance.

See Also: [Section 4.6](#) for information about managing data protection modes

Monitor the Configuration

You can check the health of the configuration, display and update the properties of the databases, and set Cloud Control events.

Cloud Control also provides a dynamic performance page that automatically and dynamically refreshes chart data and status at specified intervals. The performance chart shows a graphical summary of how far behind and how much redo data is being generated and applied.

3.6 Enable and Disable Operations

A key concept of management with the broker is the notion of enabling and disabling broker management of the databases in a broker configuration. The enable and disable operations are defined for databases that are in a broker configuration; you cannot perform these broker operations on databases that are not part of the broker configuration. This is because when you enable or disable a database in the broker configuration, you are effectively enabling or disabling the ability of the broker to manage and monitor the specified database.

However, disabling a broker configuration does not affect current services and operations in the actual Oracle Data Guard configuration. For example, when you disable a broker configuration, redo transport services and log apply services in the

Oracle Data Guard configuration continue to function unchanged, but you can no longer manage them through the broker interfaces.

In addition, disabling a database *does not* remove or delete it from the broker configuration file. You can still change the properties of a disabled database and later reenabling your ability to manage with the broker using the `DGMGRL ENABLE CONFIGURATION` or `ENABLE DATABASE` commands, or the Enable option in the Oracle Data Guard management pages of Cloud Control.

Caution: If you disable broker management of a standby database in the broker configuration, that standby database cannot be used by the broker as a failover target in the event of loss of the primary database.

Disabling broker management of the configuration may be useful to do even though you are removing the broker's ability to monitor and control the databases. For example, it may be advantageous to disable a configuration temporarily in order to change one or more properties in the broker configuration all at the same time. When you change properties in a disabled configuration, it does not affect the actual database properties underneath because the changes are not applied to the running database until you reenabling the configuration. For example, you might want to change the overall configuration protection mode and the redo transport services properties on a disabled configuration so that all changes are applied to the configuration at the same time upon the next enable operation.

See Also: [Section 4.6.2, "How the Protection Modes Influence Broker Operations"](#)

3.7 Configuration Status

A configuration status reveals the overall health of the configuration. Status of the configuration is acquired from the status of all of its databases.

The following list describes the possible status modes for a configuration:

- Success
The configuration, including all of the databases configured in it, is operating as specified by the user without any warnings or errors.
- Warning
One or more of the databases in the configuration are not operating as specified by the user. To obtain more information, use the `DGMGRL SHOW DATABASE <db-unique-name>` command or the Cloud Control display to locate each database and examine its status to reveal the source of the problem.
- Error
One or more of the databases in the configuration failed or may no longer be operating as specified by the user. To obtain more information, use the `DGMGRL SHOW DATABASE <db-unique-name>` command or the Cloud Control display to locate each database and examine its status to reveal the source of the problem.
- Unknown/Disabled
Broker management of the configuration is disabled and the broker is not monitoring the status of the databases in the configuration.

Managing the Members of a Broker Configuration

This chapter describes how to manage the various members of a broker configuration. For information about how to manage the broker configuration itself, see [Chapter 3](#). This chapter contains the following topics:

- [Managing Broker Configuration Members](#)
- [Managing States of Broker Configuration Members](#)
- [Managing Database Properties](#)
- [Managing Redo Transport Services](#)
- [Managing Log Apply Services](#)
- [Managing Data Protection Modes](#)
- [Managing Far Sync Instances](#)
- [Managing Fast-Start Failover](#)
- [Managing Database Conversions](#)
- [Database Status](#)

4.1 Managing Broker Configuration Members

The broker uses information in its configuration file to manage and monitor the state of each member of the configuration.

The broker distinguishes between different types of members in a broker configuration. Each type of member has a set of states and properties that are appropriate for that type of member.

4.2 Managing States of Broker Configuration Members

When a configuration is enabled, its members can be in one of several states that direct the behavior of Oracle Data Guard, for example transmitting redo data or applying redo data. [Table 4-1](#) describes the various states.

Snapshot standby databases are not listed in this table because they do not have states; they only receive redo data.

Far sync instances are also not included in this table because they do not have states; they only receive redo and forward it to a standby database.

Table 4–1 Database States and Descriptions

Database Role	State Name	Description
Primary	TRANSPORT-ON	<p>Redo transport services are set up to transmit redo data to the standby databases or far sync instances when the primary database is open for read/write access.</p> <p>If this is an Oracle RAC database, all instances open in read/write mode will have redo transport services running.</p> <p>This is the default state for a primary database when it is enabled for the first time.</p>
Primary	TRANSPORT-OFF	<p>Redo transport services are stopped on the primary database.</p> <p>If this is an Oracle RAC database, redo transport services are not running on any instances.</p>
Physical standby	APPLY-ON	<p>Redo Apply is started on a physical standby database.</p> <p>If the standby database is an Oracle RAC database, the broker starts Redo Apply on exactly one standby instance, called the apply instance. If this instance fails, the broker automatically chooses another instance that is either mounted or open read-only. This new instance then becomes the apply instance.</p> <p>This is the default state for a physical standby database when it is enabled for the first time.</p> <p>If a license for the Oracle Active Data Guard option has been purchased, a physical standby database can be open while Redo Apply is active. This capability is known as real-time query. See <i>Oracle Data Guard Concepts and Administration</i> for more details.</p>
Physical standby	APPLY-OFF	<p>Redo Apply is stopped.</p> <p>If this is an Oracle RAC database, there is no instance running Apply Services until you change the database state to APPLY-ON.</p>
Logical standby	APPLY-ON	<p>SQL Apply is started on the logical standby database when it is opened and the logical standby database guard is on.</p> <p>If this is an Oracle RAC database, SQL Apply is running on one instance, the apply instance. If this instance fails, the broker automatically chooses another open instance. This new instance becomes the apply instance.</p> <p>This is the default state for a logical standby database when it is enabled for the first time.</p>
Logical standby	APPLY-OFF	<p>SQL Apply is stopped. The logical standby database guard is on.</p> <p>If this is an Oracle RAC database, there is no instance running SQL Apply until you change the state to APPLY-ON.</p>

4.2.1 Database State Transitions

You can use the DGMGRL `EDIT DATABASE` command to explicitly change the state of a database. For example, the `EDIT DATABASE` command in the following example changes the state of the `North_Sales` database to `TRANSPORT-OFF`.

```
DGMGRL> EDIT DATABASE 'North_Sales' SET STATE='TRANSPORT-OFF';
Succeeded.
```

See Also: [Chapter 7](#) for complete information about the `EDIT DATABASE` command.

The following sections describe in more detail the possible state transitions for primary and standby databases.

Primary database state transitions

When transitioning the primary database to the `TRANSPORT-ON` state, the broker sets up redo transport services using the redo transport-related properties of the configuration members and the `RedoRoutes` property at the primary database. (See [Section 4.4](#) for the list of redo transport-related properties and [Section 4.4.3](#) for a description of the `RedoRoutes` property.) Redo transport services setup is done by setting the `LOG_ARCHIVE_DEST_n` and `LOG_ARCHIVE_DEST_STATE_n` initialization parameters on the primary database, and the `LOG_ARCHIVE_CONFIG` initialization parameter on all databases (primary or standby) and far sync instances. If necessary, the broker also sets up the data protection mode of the database to match the protection mode recorded in the broker configuration file. Finally, if the database is open, the broker switches a log for each thread to initiate redo transport services.

When transitioning the primary database to the `TRANSPORT-OFF` state, the broker turns off redo transport services to all broker-managed standbys by resetting the `LOG_ARCHIVE_DEST_STATE_n` initialization parameter. Transmission of redo data to all broker-managed standbys is stopped. Log files continue to be archived at the primary database.

If the primary database is an Oracle RAC database, the broker configures redo transport services on all primary instances with the exact same settings.

See Also: [Section 4.4](#) for more details on managing redo transport services

Physical standby database state transitions

When transitioning a physical standby database to the `APPLY-ON` state, the broker starts Redo Apply with options specified by the Redo Apply-related properties (see [Section 4.5](#) for the property list). If the standby is an Oracle RAC database, the broker starts Redo Apply on one standby instance, called the apply instance.

If a license for the Oracle Active Data Guard option has been purchased, a physical standby database can be open while Redo Apply is active. This capability is known as real-time query. See *Oracle Data Guard Concepts and Administration* for more details.

Note: The apply instance must be open before starting Redo Apply if any other instance is open.

When transitioning to the `APPLY-OFF` state, the broker stops Redo Apply.

Note: If you perform online database relocation with Oracle RAC One Node on a physical standby, then the new instance is started in the same mode as the currently running instance. Therefore, if the database is mounted on the original instance, then the database will be mounted on the new instance. Likewise, if the database is open on the original instance, then the database will be open on the new instance. This may result in the new instance starting in a mode that does not match the start option recorded with Oracle Clusterware for the database.

Logical standby database state transitions

When transitioning a logical standby database to the `APPLY-ON` state, the broker will wait until the database is open, and then enable the database guard to prevent modifications to tables in the logical standby database, and start SQL Apply with options specified by the log apply-related properties. If the logical standby database is an Oracle RAC database, the broker starts SQL Apply on one standby instance, the apply instance.

When transitioning to the `APPLY-OFF` state, the broker stops SQL Apply.

See Also:

- [Section 4.5](#) on page 4-15 for information about managing SQL Apply
- *Oracle Data Guard Concepts and Administration* for information about the database guard

4.3 Managing Database Properties

There are two types of database properties: monitorable and configurable. Both monitorable and configurable properties can be further divided into properties having database-wide scope or instance-specific scope.

- Monitorable property values can be viewed only when the associated database is enabled.

Monitorable properties allow you to view run-time information related to databases, but you cannot change the values of these properties.

- Configurable property values can be viewed and dynamically updated.

Configurable properties affect the operation or configuration of the broker. You can change the value of these properties using DGMGRL or Cloud Control. You can edit properties whether the configuration and its members are enabled or disabled. However, if a configuration member is disabled, then the new property value does not take effect until you enable the configuration or member, as appropriate.

See Also: [Chapter 8](#) for a detailed list of all database properties

To see these properties, you can use the DGMGRL `SHOW` command or Edit Properties page in Cloud Control. [Example 4-1](#) uses the `SHOW DATABASE VERBOSE` command to display information about the `North_Sales` database.

Example 4-1 Using the SHOW DATABASE VERBOSE Command to Display Properties

```
DGMGRL> SHOW DATABASE VERBOSE 'North_Sales';
```

```
Database - North_Sales
```

```
Role:          PRIMARY
Intended State: TRANSPORT-ON
Instance(s):
  north_sales1
```

Properties:

```
DGConnectIdentifier      = 'North_Sales.example.com'
ObserverConnectIdentifier = ''
LogXptMode              = 'ASYNCR'
RedoRoutes              = ''
DelayMins               = '0'
Binding                 = 'optional'
MaxFailure              = '0'
MaxConnections          = '1'
ReopenSecs              = '30'
NetTimeout              = '300'
RedoCompression        = 'DISABLE'
LogShipping              = 'ON'
PreferredApplyInstance  = ''
ApplyInstanceTimeout    = '0'
ApplyLagThreshold       = '0'
TransportLagThreshold   = '0'
TransportDisconnectedThreshold = '0'
ApplyParallel           = 'AUTO'
StandbyFileManagement   = 'AUTO'
ArchiveLagTarget        = '0'
LogArchiveMaxProcesses  = '5'
LogArchiveMinSucceedDest = '1'
DbFileNameConvert       = 'dbs/bt, dbs/t, dbs/ct, dbs/t'
LogFileNameConvert      = 'dbs/bt, dbs/t, dbs/ct, dbs/t'
FastStartFailoverTarget = ''
InconsistentProperties   = '(monitor)'
InconsistentLogXptProps = '(monitor)'
SendQEntries            = '(monitor)'
LogXptStatus            = '(monitor)'
RecvQEntries            = '(monitor)'
StaticConnectIdentifier  = '(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)
(HOST=North_Sales.example.com) (PORT=2840))
(CONNECT_DATA=(SERVICE_NAME=North_Sales_DGMGRL.example.com)
(INSTANCE_NAME=north_sales1) (SERVER=DEDICATED)))'
StandbyArchiveLocation   = 'USE_DB_RECOVERY_FILE_DEST'
AlternateLocation        = ''
LogArchiveTrace          = '255'
LogArchiveFormat         = 'dblr_%d_%t_%s_%R.arc'
TopWaitEvents            = '(monitor)'
```

```
Database Status:
SUCCESS
```

See Also: [Chapter 7](#) for complete information about the DGMGRL command-line interface

4.3.1 Monitorable (Read-Only) Properties

Monitorable properties allow you to view information related to a configuration member, but you cannot change the values of these properties. These properties can be very helpful when you are trying to diagnose problems in the broker configuration. For example, you can view the `InconsistentLogXptProps` monitorable property to determine where there is a discrepancy in redo transport services properties between the broker configuration file and the actual value currently used by the database.

You can list all monitorable properties using the `DGMGRL SHOW DATABASE VERBOSE` command. Use the `SHOW DATABASE` command to obtain more details about a particular property. For example, the following shows the `InconsistentLogXptProps` property:

```
DGMGRL> SHOW DATABASE 'North_Sales' 'InconsistentLogXptProps';

INCONSISTENT LOG TRANSPORT PROPERTIES

INSTANCE_NAME   STANDBY_NAME   PROPERTY_NAME   MEMORY_VALUE   BROKER_VALUE
-----
south_sales1    South_Sales    ReopenSecs     600            300
```

Cloud Control displays the information obtained from these properties on the Edit Properties page.

4.3.2 Configurable (Changeable) Properties

Configurable properties affect the operation or configuration of a database or far sync instance. When you use `DGMGRL` or Cloud Control to create a primary database and import existing standby databases and far sync instances into a new broker configuration, the property values are initially imported from the database or far sync instance settings.

You can update many property values when a configuration member is either disabled or enabled. When a new member is added into the configuration, the broker connects to that member and imports initial values for the member's properties from the current member settings. For example:

```
DGMGRL> SHOW DATABASE 'North_Sales' 'ArchiveLagTarget';
ArchiveLagTarget = '0'

DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY 'ArchiveLagTarget'=1200;
Property "ArchiveLagTarget" updated

DGMGRL> SHOW DATABASE 'North_Sales' 'ArchiveLagTarget';
ArchiveLagTarget = '1200'
```

When the configuration is enabled, the broker keeps member property values in the broker configuration file consistent with the values being used by the member. For those that are related to initialization parameter properties, the broker maintains consistency among the value in the broker configuration file, the current value, and the initialization parameter value in the server parameter file, as follows:

- For dynamic parameters, the broker keeps the value of the parameter consistent in the system global area (SGA) for the instance, in the broker configuration file, and in the server parameter file.
- For static parameters and properties, the parameter value in the system global area (SGA) for the instances may temporarily differ from what is in the broker configuration file and in the server parameter file. Typically, the in-memory value becomes the same as the server parameter file value and the broker configuration file value the next time the instance is stopped and restarted.

Even when the configuration is disabled, you can update property values through the broker. The broker retains the property settings (without validating the values) and updates the initialization parameters in the server parameter file and the settings in memory the next time you enable the broker configuration.

Note: Even though you can change a property value when the configuration is disabled, the change does not take effect on the configuration member unless the configuration is enabled. Also note that some property values can only be changed in the disabled state.

4.3.2.1 Resetting Broker Configurable Properties to Default Values

Most broker configurable properties have a default value. You can specify a different value to override the default. In releases prior to Oracle Database 12c Release 1 (12.1), once you changed a default value the broker considered it to be a user-supplied value even if you later set the property back to its former default value.

As of Oracle Database 12c Release 1 (12.1), the broker recognizes when a default value has been restored for a property and no longer considers it a user-supplied value. This is expedient in upgrade scenarios because if a default value for a property changes between releases, then the new default value is automatically put into effect after an upgrade is complete. Values that are considered user-supplied are not automatically upgraded.

You do not need to know the actual default value in order to reset it. You can reset a default value at the configuration, configuration member, or instance level.

See Also:

- ["EDIT CONFIGURATION RESET \(Property\)"](#) on page 7-25
- ["EDIT DATABASE RESET \(Property\)"](#) on page 7-30
- ["EDIT INSTANCE RESET \(Property\)"](#) on page 7-36
- ["EDIT FAR_SYNC RESET \(Property\)"](#) on page 7-32

4.4 Managing Redo Transport Services

To manage redo transport services, you specify the following set of configurable properties on each configuration member:

- `DGConnectIdentifier`
- `AlternateLocation`
- `Binding`
- `LogShipping`
- `LogXptMode`
- `MaxConnections`
- `MaxFailure`
- `NetTimeout`
- `RedoCompression`
- `RedoRoutes`

- ReopenSecs
- StandbyArchiveLocation

You can use these properties to specify how the broker configures redo transport services for the standby database. The actual redo transport setup, such as setting the `LOG_ARCHIVE_DEST_n` initialization parameter, is carried out by the broker on the primary database (except for the `StandbyArchiveLocation` property). If changing the property requires that you change the `LOG_ARCHIVE_DEST_n` initialization parameter attributes, the broker forces a log switch on each thread so that the new setting is adopted immediately by the primary database.

You should also preset these properties on the primary database in preparation for it to be switched over to a standby database.

4.4.1 Setting Up For Redo Transport

Redo data is transported to a standby database using Oracle Net. An Oracle Net service name is specified with the `SERVICE` attribute of the `LOG_ARCHIVE_DEST_n` initialization parameter and is used to transmit redo data to the standby database. The Oracle Net service name is translated into a connect descriptor that contains the information necessary for connecting to the standby database.

The `SERVICE` attribute can be set or changed by using the `DGConnectIdentifier` database property. The `DGConnectIdentifier` property is set when a database is first added to the configuration. Its initial value is the connect identifier that is specified in the optional `CONNECT IDENTIFIER IS` clause of the `ADD DATABASE` command.

The `DGConnectIdentifier` property value is also used to set up the `FAL_SERVER` initialization parameter. If the `DGConnectIdentifier` property for any database is changed, the `SERVICE` attribute of the corresponding `LOG_ARCHIVE_DEST_n` initialization parameter will also be changed. In addition, the `FAL_SERVER` initialization parameter will also be updated on every enabled standby database in the configuration.

4.4.2 Managing Redo Transport Services for Data Protection Modes

[Section 4.6](#) describes how the broker handles data protection modes. As a part of the overall configuration protection mode, you must ensure that redo transport services are also properly set up for the data protection mode that you choose.

You use the `LogXptMode` or `RedoRoutes` database properties to set the `SYNC`, `ASYNC`, or `FASTSYNC` mode for redo transport services. See [Table 4-2](#) for additional information about protection modes and redo transport services.

The following redo transport modes are supported:

SYNC

Configures redo transport services for this standby database using the `SYNC` and `AFFIRM` attributes of the `LOG_ARCHIVE_DEST_n` initialization parameter. This mode, along with standby redo log files, is required for configurations operating in either maximum protection mode or maximum availability mode. This redo transport service enables the highest grade of data protection to the primary database, but also can incur a higher performance impact.

ASYNC

Configures redo transport services for this standby database using the `ASYNC` and `NOAFFIRM` attributes of the `LOG_ARCHIVE_DEST_n` initialization parameter. This mode,

along with standby redo log files, enables a moderate grade of protection to the primary database, and lower performance impact.

FASTSYNC

Configures redo transport services for this standby database using the `SYNC` and `NOAFFIRM` attributes of the `LOG_ARCHIVE_DEST_n` initialization parameter. This mode is only available in maximum availability protection mode.

4.4.3 Advanced Redo Transport Settings

By default, a primary database sends its redo to every possible redo transport destination in the configuration. You can use the `RedoRoutes` property to create a more complex redo transport topology, such as one in which a physical standby or a far sync instance forwards redo received from the primary database to one or more destinations, or one in which the redo transport mode used for a given destination is dependent on which database is in the primary role.

The `RedoRoutes` property is set to a character string that contains one or more redo routing rules, each contained within a set of parentheses, as follows:

```
(redo_routing_rule_1) [(redo_routing_rule_n)]
```

A redo routing rule contains a redo source field and a redo destination field separated by a colon:

```
(redo source : redo destination)
```

The redo source field must contain the keyword `LOCAL` or `ANY`, or a comma-separated list of `DB_UNIQUE_NAME` values, as follows:

```
{LOCAL | ANY | db_unique_name_1 [,db_unique_name_n]}
```

- The `LOCAL` keyword is an alias for the local database name. This keyword cannot be used at a far sync instance.
- The `ANY` keyword is an alias for any database in the configuration.
- A database cannot be specified as a redo source in more than one redo routing rule defined at a given database, either explicitly or implicitly through use of the `LOCAL` keyword.

The redo destination field must contain the keyword `ALL` or a comma-separated list of database names, each of which can be followed by an optional redo transport attribute:

```
{ALL [attribute] | db_unique_name_1 [attribute] [,db_unique_name_n [attribute]]}
```

- The `ALL` keyword is an alias for all possible destinations in the configuration.

The optional redo transport attribute specifies the redo transport mode to be used to send redo to the associated destination. It can have one of three values:

```
{ASync | SYNC | FASTSYNC}
```

If the redo transport attribute is not specified, then the transport mode used will be the one specified by the `LogXptMode` property for the redo destination.

The optional `ALT` keyword is used to specify the `DB_UNIQUE_NAME` of an alternate redo destination that is to receive redo if redo transport to the non-alternate destination is not possible:

```
[ALT=(alternate db_unique_name [ASync | SYNC | FASTSYNC] [FALLBACK])]
```

The optional `FALLBACK` keyword is used to specify that redo transport to the non-alternate member should be resumed if it becomes available. Oracle recommends that you use the `FALLBACK` keyword. If you omit it and the `ALTERNATE` redo destination subsequently fails, no redo will be shipped to the terminal standby databases serviced by the alternate destination.

Note: The non-alternate destination must have a non-zero value for its `MaxFailures` property in order for an alternate to be specified.

The following usage notes apply regarding advanced redo transport settings:

- The `RedoRoutes` property has a default value of `NULL`, which is treated as (`LOCAL : ALL`) at a primary database.
- A redo routing rule is active if its redo source field specifies the current primary database. If a rule is active, primary database redo is sent by the database at which the rule is defined to each destination specified in the redo destination field of that rule.
- The `ASYNC` redo transport attribute must be explicitly specified for a cascaded destination to enable real-time cascading to that destination.
- The `RedoRoutes` property cannot be set on a logical or snapshot standby database.
- The `RedoRoutes` property can be set for a logical standby database only if the redo destination field is set to `LOCAL`.
- The non-alternate member must have a non-zero value for its `MaxFailure` configurable property in order for an alternate to be specified.

Example 4–2 Using the RedoRoutes Property for Real-Time Cascading

Consider a configuration that has a primary database (`North_Sales`) and two physical standby databases (`Local_Sales` and `Remote_Sales`). The `Local_Sales` database is located in the same data center as the primary for high availability purposes. The `Remote_Sales` database is located in a remote data center for disaster recovery purposes. Instead of the primary having to ship its redo to both databases, it is possible to use the `RedoRoutes` property to configure real-time cascading, in which the local physical standby database forwards redo from `North_Sales` to the remote physical standby database, `Remote_Sales`. To accomplish this, the `RedoRoutes` property must be set as follows:

- On the `North_Sales` database, the `RedoRoutes` property must specify that if `North_Sales` is in the primary role, then it should ship redo to the `Local_Sales` database using synchronous transport mode. This rule prevents the primary from shipping redo data directly to the `Remote_Sales` database.
- On the `Local_Sales` database, the `RedoRoutes` property must specify that if `North_Sales` is in the primary role, then `Local_Sales` should forward redo it receives from `North_Sales` on to `Remote_Sales`.

```
DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY 'RedoRoutes' = '(LOCAL : Local_Sales SYNC)';
DGMGRL> EDIT DATABASE 'Local_Sales' SET PROPERTY 'RedoRoutes' = '(North_Sales : Remote_Sales ASYNC)';
```

To see the runtime `RedoRoutes` configuration, use the `SHOW CONFIGURATION` command. For example:

```
DGMGRL> SHOW CONFIGURATION;
```

```

Configuration - Sales_Configuration

Protection Mode: MaxAvailability
Databases:
North_Sales - Primary database
Local_Sales - Physical standby database
Remote_Sales - Physical standby database (receiving current redo)

Fast-Start Failover: DISABLED

Configuration Status:
SUCCESS

```

Note that the ASYNC redo transport attribute was explicitly specified in the redo route rule for the Remote_Sales destination to enable real-time cascading of redo to that destination.

To disable real-time cascading of redo, do not specify the ASYNC redo transport attribute. For example:

```
DGMGRL> EDIT DATABASE 'Local_Sales' SET PROPERTY 'RedoRoutes' = '(North_Sales : Remote_Sales)';
```

See Also:

- ["RedoRoutes"](#) on page 8-39

Example 4-3 Using the RedoRoutes Property for Remote Alternate Destinations

The RedoRoutes property can also be used to set up a remote alternate destination (using the ALT keyword) so that a terminal member can still receive redo data even if the member from which it was receiving the redo data fails. Using the previous example, it would be possible to have the primary database, North_Sales, send redo data directly to Remote_Sales if the Local_Sales standby database failed. It is also possible, using the FALLBACK keyword, to set it up so that once the Local_Sales failure has been resolved it can resume shipping redo to Remote_Sales. Oracle recommends that you always use the FALLBACK keyword. If it is not set and the connection to Remote_Sales fails, then redo will not be sent to either Local_Sales or Remote_Sales even if Local_Sales becomes operational again.

To set up a remote alternate, the first step is to set a maximum failure count for the Local_Sales database using the MaxFailure configurable property. When the MaxFailure property for Local_Sales is set to a non-zero value, redo will be sent directly to Remote_Sales if a situation ever occurs in which the maximum failure count is reached for Local_Sales. In the following example, the maximum failure count for Local_Sales is set to 1.

```

DGMGRL> EDIT DATABASE 'Local_Sales' SET PROPERTY 'MaxFailure' = 1;
Property "MaxFailure" updated

DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY 'RedoRoutes' =
'(LOCAL : Local_Sales ASYNC ALT=(Remote_Sales ASYNC FALLBACK))';
Property "RedoRoutes" updated

DGMGRL> EDIT DATABASE 'Local_Sales' SET PROPERTY 'RedoRoutes' =
'(North_Sales : Remote_Sales ASYNC)';
Property "RedoRoutes" updated

DGMGRL> SHOW CONFIGURATION;

```

```

Configuration - Sales_Configuration

Protection Mode: MaxPerformance
Databases:
North_Sales - Primary database
Local_Sales - Physical standby database
Remote_Sales - Physical standby database
Fast-Start Failover: DISABLED
    
```

```

Configuration Status:
SUCCESS
    
```

To see the full RedoRoutes configuration, use the SHOW CONFIGURATION VERBOSE command. For example:

```
DGMGRL> SHOW CONFIGURATION VERBOSE;
```

```

Configuration - Sales_Configuration

Protection Mode: MaxPerformance
Members:
North_Sales - Primary database
Local_Sales - Physical standby database
Remote_Sales - Physical standby database
Remote_Sales - Physical standby database (alternate of Local_Sales)
    
```

```

Properties:
FastStartFailoverThreshold      = '30'
OperationTimeout                = '30'
TraceLevel                      = 'USER'
FastStartFailoverLagLimit       = '30'
CommunicationTimeout            = '180'
ObserverReconnect               = '0'
FastStartFailoverAutoReinstate = 'TRUE'
FastStartFailoverPmyShutdown   = 'TRUE'
BystandersFollowRoleChange      = 'ALL'
ObserverOverride                = 'FALSE'
ExternalDestination1            = ''
ExternalDestination2            = ''
PrimaryLostWriteAction          = 'CONTINUE'
    
```

```
Fast-Start Failover: DISABLED
```

```

Configuration Status:
SUCCESS
    
```

4.4.4 Turning Redo Transport Services On and Off

Turn redo transport services on and off by setting the state of the primary database. Setting the primary database state to TRANSPORT-ON starts redo transport services at the primary database, and setting the primary database state to TRANSPORT-OFF stops redo transport services at the primary database.

Note: Oracle does not recommend turning off redo transport services to all standby databases. This increases the risk of data loss if the primary database fails.

Turn redo transport services on and off to an individual standby database using the `LogShipping` database property on the standby database. The `LogShipping` property accepts values `ON` and `OFF`. If you set the `LogShipping` property to `OFF` for a standby database, redo transport services to this standby database are turned off, while redo transport services to other databases are not affected. You can set `LogShipping` to `ON` to turn back on redo transport services to the standby database.

The relationship between setting the primary database state and setting the `LogShipping` property is as follows:

- If the primary database state is set to `TRANSPORT-OFF`, redo transport services to all the standby databases are stopped regardless of the `LogShipping` property values of the individual standby databases.
- If the primary database state is set to `TRANSPORT-ON`, redo transport services to each standby database are determined by the `LogShipping` property of that database.

[Example 4-4](#) and [Example 4-5](#) show how to turn off redo transport services in two different scenarios.

Example 4-4 Turn Off Redo Transport Services to All Standby Databases

```
DGMGRL> EDIT DATABASE 'North_Sales' SET STATE='TRANSPORT-OFF';
DGMGRL> SHOW DATABASE 'North_Sales';
```

```
Database - North_Sales
```

```
Role:                PRIMARY
Intended State:     TRANSPORT-OFF
Instance(s):
  north_sales1
```

```
Database Status:
SUCCESS
```

Example 4-5 Turn Off Redo Transport Services to a Specific Standby Database

```
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY 'LogShipping'='OFF';
Property "LogShipping" updated
```

```
DGMGRL> SHOW DATABASE 'South_Sales' 'LogShipping';
LogShipping = 'OFF'
```

4.4.5 Specifying Locations for Archived Redo Log Files

You can set up locations on the standby database to store the archived redo log files to be used by log apply services on the standby database. This is done by setting the `StandbyArchiveLocation` and `AlternateLocation` database properties on the standby database.

`StandbyArchiveLocation` specifies a standby location where the archived redo log files will be stored. The broker only uses the location to store archived redo log files received from the primary database. For archived redo log files generated locally when the database is either the primary database or a logical standby database, you need to set up local destinations directly through the `LOG_ARCHIVE_DEST_n` initialization parameter. The broker allows the value of `StandbyArchiveLocation` to be the same as the location you set up for locally generated logs, in which case the broker sets up the `VALID_FOR` attribute of the destination appropriately so that it can be used for both the archived redo log files received from the primary database and archived redo log files generated locally.

Note: On a logical standby database, Oracle recommends that the `LOCATION` attribute of the `LOG_ARCHIVE_DEST_n` initialization parameter for the local destination be different from the value of either the `StandbyArchiveLocation` or `AlternateLocation` property unless you are using a database recovery area.

You can also set up an alternate location to store archived redo log files on the standby using the `AlternateLocation` property on the standby database. This is useful for avoiding disk capacity problems or disk errors when archiving primary redo at the standby database. `AlternateLocation` specifies a standby location where the archived redo log files will be stored if the location specified by the `StandbyArchiveLocation` fails. The broker sets up the alternate location properly using the `ALTERNATE` attribute of the `LOG_ARCHIVE_DEST_n` initialization parameter.

Note: You can use the database recovery area to store archived redo log files on the standby. In such a case, the value of the `StandbyArchiveLocation` or `AlternateLocation` properties can be set to `USE_DB_RECOVERY_FILE_DEST`.

4.4.6 Other Redo Transport Settings

You can use the `Binding`, `MaxFailure`, `MaxConnections`, `NetTimeout`, `RedoCompression`, and `ReopenSecs` database properties to tune the performance of redo transport services and to set up redo transport services failure policies. These properties correspond to attributes on the `LOG_ARCHIVE_DEST_n` initialization parameter.

See Also: [Chapter 8](#) for complete information about these database properties

4.4.7 Redo Transport Services in an Oracle RAC Database Environment

If the primary database is an Oracle RAC database, the broker ensures that redo transport services are set up identically on each of the primary database instances. Each instance has the same remote destinations, and for each remote destination, all instances are set up the same in terms of redo transport service, performance related settings, and so on. If an instance has different settings, the broker raises a health check warning on that particular instance

Settings relative to redo transport services are saved in the broker configuration file as properties. When you update a redo transport-related property on a standby database, the corresponding change is also made automatically by the broker to the `LOG_ARCHIVE_DEST_n` initialization parameter on all of the primary database instances. If a new instance comes up on the primary database, the broker sets up redo transport services for the new instance using the redo transport-related properties of all the standby databases currently being managed by the broker. After the new instance is opened for activity, all archived redo log files generated on this instance will begin to transmit to the standby databases.

See also: *Oracle Data Guard Concepts and Administration* for additional information about the `LOG_ARCHIVE_DEST_n` initialization parameter

4.4.8 Transport Lag

Transport lag is a measure of the degree to which the transport of redo to the standby database lags behind the generation of redo on the primary database.

If there are one or more redo gaps on the standby database, the transport lag is calculated as if no redo has been received after the beginning of the earliest redo gap.

Both Cloud Control and the DGMGRL client display the redo transport lag for each managed standby database. Cloud Control displays the transport lag on the Oracle Data Guard home page. The DGMGRL client displays the transport lag in the `SHOW DATABASE` output for a standby database. There is no transport lag displayed for a primary database. For example:

```
DGMGRL> show database 'South_Sales';

Database - South_Sales

Role:                PHYSICAL STANDBY
Intended State:      APPLY-ON
Transport Lag:      0 seconds (computed 0 seconds ago)
Apply Lag:           0 seconds (computed 0 seconds ago)
Apply Rate:          255.00 KByte/s
Real Time Query:    OFF
Instance(s):
    south_sales1

Database Status:
SUCCESS
```

The transport lag can help you identify any problems that may exist with the redo transport services.

You can set the `TransportLagThreshold` database configurable property to generate a health check warning when the transport of redo data to the standby database lags behind the generation of redo data on the primary database.

The following command sets the `TransportLagThreshold` property to 15 seconds:

```
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY 'TransportLagThreshold'=15;
Property TransportLagThreshold updated
```

Additionally, you can set the `TransportDisconnectedThreshold` database configurable property to generate a health check warning if the standby finds that it has not had any redo transport-related communication with the primary database. The property has a default value of 30 seconds.

The following command sets the `TransportDisconnectedThreshold` property to 15 seconds:

```
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY
'TransportDisconnectedThreshold'=15;
Property TransportDisconnectedThreshold updated
```

4.5 Managing Log Apply Services

You can manage Redo Apply and SQL Apply on physical and logical standby databases through the following log apply-related database properties:

- Properties common to Redo Apply and SQL Apply
 - `ApplyInstanceTimeout`

- DelayMins
- PreferredApplyInstance
- Properties specific to Redo Apply
 - ApplyParallel
- Properties specific to SQL Apply
 - LsbyMaxEventsRecorded
 - LsbyPreserveCommitOrder
 - LsbyRecordSkipErrors
 - LsbyRecordSkipDdl
 - LsbyRecordAppliedDdl
 - LsbyMaxSga
 - LsbyMaxServers

There are some properties related to SQL Apply that, if changed, may require a restart of SQL Apply if the current database state is `APPLY-ON`. See the information in [Chapter 8](#) about properties related to SQL Apply, to determine which ones require SQL Apply to be restarted.

If the current database state is `APPLY-OFF`, the property changes will take effect the next time the database state is changed to `APPLY-ON`.

4.5.1 Managing Delayed Apply

You can set up Apply Services so that the application of redo to the standby database is delayed. This allows the standby database to lag behind the primary database, and if a user error (for example, dropping a table) occurs during this window of time, the standby database will still contain the correct data that can be transmitted back to the primary database to repair the data.

By default, no delay is configured and the redo data is applied on a standby database as soon as possible. If the standby database has standby redo logs configured, the broker will enable real-time apply. When Redo Apply and SQL Apply apply redo in real time, the redo data is recovered directly from the standby redo log files as they are being filled. This means that the standby database does not have to wait for the log files to be archived before applying redo data from the archived redo log files. This minimizes the transactional lag between the primary and the standby.

Use the `DelayMins` database property to specify the number of minutes that log apply services must wait before applying redo data to the standby database. Note that only log apply services on the standby database are delayed. Redo transport services on the primary database are not delayed, thus the primary database data is still well protected by the standby database.

Caution: Because the broker automatically enables real-time apply on standby databases, Oracle recommends that you configure all databases to use Flashback Database.

4.5.2 Managing Parallel Apply with Redo Apply

For Redo Apply, you can configure whether multiple parallel processes are used to apply redo data received from the primary database by using the `ApplyParallel`

database property. Parallelism is enabled by default, which means Redo Apply automatically chooses the optimal number of parallel processes based on the number of CPUs in the system. (This is equivalent to setting the `ApplyParallel` property to `AUTO`.) You can disable parallelism by setting the `ApplyParallel` property to `NO`.

Note: The `ApplyParallel` database property is not displayed on the Edit Properties page of Cloud Control.

See Also: The `ApplyParallel` property in [Section 8.3.4, "ApplyParallel"](#)

4.5.3 Allocating Resources to SQL Apply

You can control how much SGA memory is available for SQL Apply. This can be set using the `LsbyMaxSga` database property.

To control the number of parallel query servers used by SQL Apply, you can use the `LsbyMaxServers` database property.

You can control the trade off between SQL Apply performance and the commit order of transactions. The `LsbyPreserveCommitOrder` database property controls whether transactions are committed on the logical standby database in the exact same order in which they were committed on the primary database. Preserving commit order may affect performance.

See Also: [Section 8.3.20, "LsbyMaxSga"](#) and [Section 8.3.19, "LsbyMaxServers"](#)

4.5.4 Managing the DBA_LOGSTDBY_EVENTS Table

The `DBA_LOGSTDBY_EVENTS` table records important events that affect SQL Apply. Because every logical standby database might have a different interest in the set of events to be recorded in this table, Oracle Data Guard provides a means to control the event recording. From the Oracle Data Guard broker, you can use the `LsbyRecord*` database properties (for example, `LsbyRecordSkipDdl` or `LsbyRecordSkipErrors`) to control recording of a particular set of events. The value of these properties are either `TRUE` or `FALSE`, indicating the turning on or off of the event recording.

4.5.5 Apply Services in an Oracle RAC Database Environment

If a standby database is an Oracle RAC database, only one instance of the Oracle RAC database can have log apply services running at any time. This instance is called the **apply instance**. If the apply instance fails, the broker automatically moves log apply services to a different instance; this is called **apply instance failover**.

4.5.5.1 Selecting the Apply Instance

Note: The information in this section is not applicable to snapshot standby databases or far sync instances.

If you have no preference which instance is to be the apply instance in an Oracle RAC standby database, the broker randomly picks an apply instance. If you want to select a particular instance as the apply instance, there are two methods to do this.

- The first method is to set the value of the PreferredApplyInstance database property to the name of the instance (see the SidName property) you want to be the apply instance. The broker starts log apply services on that instance if no apply instance is yet selected in the Oracle RAC standby database. This could be the case before you enable the standby database for the first time, or if the apply instance just failed and the broker is about to do an apply instance failover, or if the Oracle RAC database is currently the primary and you want to specify its apply instance in preparation for a switchover. Once the apply instance is selected and, as long as the apply instance is still running, the broker disregards the value of the PreferredApplyInstance property even if you change it.
- The second method is to change the apply instance when the apply instance is already selected and is running. To change the apply instance, issue the DGMGRL SET STATE command to set the standby database state to APPLY-ON, with a specific apply instance argument. The SET STATE command will update the PreferredApplyInstance property to the new apply instance value, and then move log apply services to the new instance. For example, use DGMGRL SHOW command to show the available instances for the standby database, then issue the EDIT DATABASE command to move log apply services to the new instance:

```
DGMGRL> SHOW DATABASE 'South_Sales'
```

```
Database - South_Sales
```

```

Role:                PHYSICAL STANDBY
Intended State:      APPLY-ON
Transport Lag:       0 seconds (computed 1 second ago)
Apply Lag:           0 seconds (computed 1 second ago)
Apply Rate:          1017.00 KByte/s
Real Time Query:    OFF
Instance(s):
  south_sales1      (apply instance)
  south_sales2

```

```
Database Status:
SUCCESS
```

```
DGMGRL> EDIT DATABASE 'South_Sales' SET STATE='APPLY-ON' WITH APPLY
INSTANCE='south_sales2';
Succeeded.
```

```
DGMGRL> SHOW DATABASE 'South_Sales' 'PreferredApplyInstance';
PreferredApplyInstance = 'south_sales2'
```

```
DGMGRL> SHOW DATABASE 'South_Sales'
```

```
Database - South_Sales
```

```

Role:                PHYSICAL STANDBY
Intended State:      APPLY-ON
Transport Lag:       0 seconds (computed 1 second ago)
Apply Lag:           0 seconds (computed 1 second ago)
Apply Rate:          1017.00 KByte/s
Real Time Query:    OFF
Instance(s):
  south_sales1
  south_sales2      (apply instance)

```

```
Database Status:
SUCCESS
```

Ensure that the new apply instance is running when the command is issued. Otherwise, the apply instance remains the same.

Once the apply instance is selected, the broker keeps apply instance information in the broker configuration file so that even if the standby database is shut down and restarted, the broker still selects the same instance to start log apply services. The apply instance remains unchanged until changed by the user or it fails for any reason and the broker decides to do an apply instance failover.

4.5.5.2 Apply Instance Failover

To tolerate a failure of the apply instance, the broker leverages the availability of the Oracle RAC standby database by automatically failing over log apply services to a different standby instance. The apply instance failover capability provided by the broker enhances data protection.

To set up apply instance failover, set the `ApplyInstanceTimeout` database property to specify the time period that the broker will wait after detecting an apply instance failure and before initiating an apply instance failover. To select an appropriate timeout value, you need to consider:

- If there is another mechanism in the cluster (such as Oracle Clusterware) that will try to recover the failed apply instance.
- How long your business can tolerate not applying redo data on the standby database.
- The overhead associated with moving the log apply services to a different instance. The overhead may include retransmitting, from the primary database, all log files accumulated on the failed apply instance that have not been applied if those log files are not saved in a shared file system that can be accessed from other standby instances.

The broker default value of the `ApplyInstanceTimeout` property is 0 seconds, indicating that apply instance failover should occur immediately upon detection of the failure of the current apply instance.

After the broker initiates an apply instance failover, the broker selects a new apply instance according to the following rule: if the `PreferredApplyInstance` property indicates an instance that is currently running, select it as the new apply instance; otherwise pick a random instance that is currently running to be the new apply instance.

In addition, if the physical standby database was operating in real-time query mode when the apply instance failed, then after Oracle recovery cleanup is completed, the broker opens any instances that had been automatically closed. If the failed apply instance was the only instance open, then the instance chosen as the new apply instance is opened before starting apply services so that real-time query is once again in effect.

See Also:

- *Oracle Data Guard Concepts and Administration* for more information about real-time query mode
- The My Oracle Support note 1357597.1 at <http://support.oracle.com> for additional information about apply instance failures in an Oracle Active Data Guard Oracle RAC standby

4.5.6 Apply Lag

Apply lag is a measure of the degree to which the data in a standby database lags behind the data in the primary database, due to delays in propagating and applying redo to the standby database.

Both Cloud Control and the DGMGRL client display the apply lag for each managed standby database. Cloud Control displays the apply lag on the Oracle Data Guard home page. The DGMGRL client displays the apply lag in the `SHOW DATABASE` output for a standby database. There is no apply lag displayed for a primary database. For example:

```
show database 'South_Sales';
Database - South_Sales

Role:                PHYSICAL STANDBY
Intended State:      APPLY-ON
Transport Lag:       0 seconds (computed 0 seconds ago)
Apply Lag:        0 seconds (computed 0 seconds ago)
Apply Rate:          255.00 KByte/s
Real Time Query:    OFF
Instance(s):
  south_sales1

Database Status:
SUCCESS
```

The apply lag can help you identify any problems that may exist with both the redo transport services and the log apply services.

You can set the `ApplyLagThreshold` database configurable property to generate a health check warning when a standby database lags behind the data in the primary database.

The following command sets the `ApplyLagThreshold` property to 15 seconds:

```
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY 'ApplyLagThreshold'=15;
Property ApplyLagThreshold updated
```

4.6 Managing Data Protection Modes

The broker can simplify the process of setting up your configuration for any of the different grades of data protection: maximum protection, maximum availability, or maximum performance.

This section contains the following topics to help you configure the proper protection for your configuration:

- [Setting the Protection Mode for Your Configuration](#)
- [How the Protection Modes Influence Broker Operations](#)

4.6.1 Setting the Protection Mode for Your Configuration

To set the protection mode, perform the following steps:

Step 1 Determine which data protection mode you want to use.

Each data protection mode provides a different balance of data protection, data availability, and database performance. To select the data protection mode that meets

the needs of your business, carefully consider your data protection requirements and the performance expectations of your users.

Note: Maximum protection mode cannot be used in the following situations:

- If the only standby database in a configuration is a snapshot standby
 - If a far sync instance is the only configuration member receiving redo in synchronous mode from the primary database
-

Maximum Availability This protection mode provides the highest level of data protection that is possible without compromising the availability of a primary database. Transactions do not commit until all redo data needed to recover those transactions has been written to the online redo log and to the standby redo log on at least one synchronized standby database. If the primary database cannot write its redo stream to at least one synchronized standby database, it operates as if it were in maximum performance mode to preserve primary database availability until it is again able to write its redo stream to a synchronized standby database.

This mode ensures that no data loss will occur if the primary database fails, but only if a second fault does not prevent a complete set of redo data from being sent from the primary database to at least one standby database.

You can enable fast-start failover if the protection mode is maximum availability.

Maximum Performance This protection mode provides the highest level of data protection that is possible without affecting the performance of a primary database. This is accomplished by allowing transactions to commit as soon as all redo data generated by those transactions has been written to the online log. Redo data is also written to one or more standby databases, but this is done asynchronously with respect to transaction commitment, so primary database performance is unaffected by delays in writing redo data to the standby database(s).

This protection mode offers slightly less data protection than maximum availability mode and has minimal impact on primary database performance.

This is the default protection mode.

You can enable fast-start failover if the protection mode is maximum performance.

Maximum Protection This protection mode ensures that no data loss will occur if the primary database fails. To provide this level of protection, the redo data needed to recover a transaction must be written to both the online redo log and to the standby redo log on at least one synchronized standby database before the transaction commits. To ensure that data loss cannot occur, the primary database will shut down, rather than continue processing transactions, if it cannot write its redo stream to at least one synchronized standby database.

Transactions on the primary are considered protected as soon as Oracle Data Guard has written the redo data to persistent storage in a standby redo log file. Once that is done, acknowledgment is quickly made back to the primary database so that it can proceed to the next transaction. This minimizes the impact of synchronous transport on primary database throughput and response time. To fully benefit from complete Oracle Data Guard validation at the standby database, be sure to operate in real-time apply mode so that redo changes are applied to the standby database as fast as they

are received. Oracle Data Guard signals any corruptions that are detected so that immediate corrective action can be taken.

Because this data protection mode prioritizes data protection over primary database availability, Oracle recommends that a minimum of two standby databases be used to protect a primary database that runs in maximum protection mode to prevent a single standby database failure from causing the primary database to shut down. If only one standby database is supporting maximum protection mode, Oracle Data Guard broker will disallow the shutdown of the apply instance. This prevents the primary database from shutting down.

You cannot enable fast-start failover if the protection mode is maximum protection.

See Also:

- [Section 5.5](#) for information on fast-start failover
- *Oracle Data Guard Concepts and Administration* for complete information about data protection modes

Step 2 Set up standby redo log files.

You must add standby redo log files on all standby databases, regardless of the protection mode you are using. Also, Oracle requires that you add standby redo log files on the primary database in preparation for a future switchover or failover. Standby redo log files are required on the primary database if you want to enable fast-start failover.

Cloud Control automatically prompts you to select one or more standby databases in the configuration and sets up standby redo log (SRL) files on them and on the primary database in preparation for a future role change.

See Also: If you are using the DGMGRL command-line interface, follow the instructions in *Oracle Data Guard Concepts and Administration* to configure standby redo log files.

Step 3 Set the redo transport mode.

If the data protection mode requires that you change the redo transport mode used by any of the standby databases, then either change the `LogXptMode` database property on each standby database, or set the `RedoRoutes` property on the primary database or on the far sync instance that is directly connected to the standby database. See [Section 4.4](#) for more information about setting the redo transport service. [Table 4-2](#) shows the protection modes and the corresponding redo transport service.

Cloud Control automatically specifies the correct redo transport service on the primary database in preparation for a future switchover.

Table 4-2 Oracle Data Guard Protection Modes and Requirements

Protection Mode	Redo Transport	Standby Redo Log Files Needed?	Usable with Fast-Start Failover?
MAXPROTECTION	SYNC	Yes	No
MAXAVAILABILITY	SYNC, FASTSYNC	Yes	Yes ¹
MAXPERFORMANCE	ASync	Yes	Yes

¹ Because FASTSYNC transport mode uses the NOAFFIRM attribute of the LOG_ARCHIVE_DEST_n parameter, data loss is possible. This means that a fast-start failover cannot be initiated when FASTSYNC is used and the standby is missing redo data.

Step 4 Set the protection mode.

Set the protection mode using DGMGRL commands or Cloud Control.

With DGMGRL:

1. Use the `EDIT DATABASE (property)` command and specify the standby database whose redo transport service should be changed to correspond to the protection mode you plan to set. For example, if you plan to set the overall Oracle Data Guard configuration to operate in maximum availability mode, you must use the `EDIT DATABASE` command to set the `SYNC` mode for redo transport services. For example:

```
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY LogXptMode='SYNC';
```

Do this also for the primary database or another standby database in the configuration to ensure that it can support the chosen protection mode after a switchover.

You could also use the `RedoRoutes` property, as follows:

```
EDIT DATABASE 'North_Sales' SET PROPERTY RedoRoutes = '(LOCAL : South_Sales SYNC)';
```

2. Use the `EDIT CONFIGURATION SET PROTECTION MODE AS protection-mode` command to set the overall configuration protection mode. For example:

```
DGMGRL> EDIT CONFIGURATION SET PROTECTION MODE AS MAXAVAILABILITY;
```

See [Section 6.5](#) for a DGMGRL scenario showing how to set the protection mode.

With Cloud Control:

1. On the Oracle Data Guard overview page, click the link to the right of the Protection Mode label.
2. Select Maximum Protection, Maximum Availability, or Maximum Performance and click Continue.
3. If prompted, log in to the database with `SYSDBG` or `SYSDBA` privileges and click Login.
4. Select one or more standby databases to support the protection mode that you selected. If standby redo log files are needed, verify the names of the log files. Click OK.
5. On the Confirmation page, click Yes.

After upgrading the protection mode from maximum performance mode to maximum protection mode using either DGMGRL or Cloud Control, the primary database will be restarted automatically if you are upgrading to maximum protection mode. The primary database need not be restarted following a downgrade of the protection mode.

No restart is required when upgrading the protection mode from maximum performance to maximum availability or from maximum availability to maximum protection.

Note: You can avoid a restart of the primary database when upgrading from maximum performance mode to maximum protection mode by first upgrading to maximum availability. Once in maximum availability mode, then upgrade to maximum protection.

Note: Online database relocation with Oracle RAC One Node cannot be performed on the only physical standby database that supports maximum protection mode.

If online database relocation must be performed, then you must first downgrade the protection mode to maximum availability. After the online database relocation of the physical standby database is performed, you can upgrade the protection mode back to maximum protection. A restart of the primary database is not required.

4.6.2 How the Protection Modes Influence Broker Operations

This section describes how operations such as switchover, failover, disabling, or enabling the Oracle Data Guard configuration can have an effect on the configuration's protection mode and redo transport services. This section contains the following sections:

- [Upgrading or Downgrading the Current Protection Mode](#)
- [Switchover Operations](#)
- [Failover Operations](#)
- [Disable and Enable Operations](#)
- [Requirements For Removing a Database from the Configuration](#)
- [Requirements On Other Operations](#)

4.6.2.1 Upgrading or Downgrading the Current Protection Mode

No restart is necessary when you upgrade the current Oracle Data Guard protection mode to maximum availability or when you downgrade the current Oracle Data Guard protection mode. Follow these recommendations when upgrading or downgrading the Oracle Data Guard protection mode:

- When upgrading the protection mode, upgrade the redo transport service before you upgrade the overall protection mode. At the time when you change the protection mode or reset the redo transport service of a standby database, the broker verifies that there is at least one standby database in the configuration that can support the desired grade of protection. If not, then the broker does not change the protection mode and returns an error.
- When downgrading the protection mode, downgrade the protection mode first and then change the redo transport service (if necessary). The broker will disallow a change of the redo transport service if doing so invalidates the current overall protection mode.

If you upgrade the protection mode from the maximum performance mode, the broker ensures that there is at least one standby database that receives redo via the *SYNC* transport, either directly or through a far sync instance. Additionally, for upgrades to maximum protection mode, the broker ensures there are no gaps in the redo data on the standby database. If there are no standby databases in the configuration that meet these requirements, the request to upgrade the protection mode is rejected with an error.

The protection mode cannot be changed if fast-start failover is enabled.

WARNING: The primary database will be shut down and restarted if you upgrade the protection mode from maximum performance to maximum protection. You can avoid this by first upgrading to maximum availability. Once in maximum availability mode, then upgrade to maximum protection.

4.6.2.2 Switchover Operations

A switchover does not change the overall Oracle Data Guard protection mode. The protection mode remains the same as it was before the switchover.

This requires that there be a standby database that is properly configured to support the current protection mode once the switchover completes. This can be either another standby database in the configuration or the current primary database that will become a standby database after the switchover completes.

Before you perform a switchover, if necessary you can add standby redo log files and set the redo transport service on the current primary database, or on another standby database in the configuration, to the transport mode that is required to support the Oracle Data Guard protection mode. Then, when the switchover begins:

- The broker verifies the presence of standby redo log files and the redo transport service setting on each standby database and on the current primary database.
- The broker verifies there are no gaps in the redo data present on the target standby database.

If the verification is successful, the switchover continues; otherwise, the switchover fails, and the database roles and the broker configuration files remain unchanged.

WARNING:

- If the target of the switchover is a physical standby database, then the broker shuts down and restarts the primary database.
 - If the configured protection mode is maximum protection and there is only one standby that supports that mode, then when the switchover is performed both the primary and standby are restarted.
-
-

See Also: [Section 5.3](#) for more information about switchovers

4.6.2.3 Failover Operations

After you perform a manual failover, the Oracle Data Guard protection mode is downgraded to maximum performance mode if the protection mode was at maximum protection. You can upgrade the protection mode later, if necessary. If the protection mode was at maximum availability or maximum performance, it remains unchanged. The redo transport services of the standby databases remain unchanged.

If fast-start failover occurs, the broker preserves the protection mode that was in effect just prior to the fast-start failover.

See Also: [Section 5.4](#) and [Section 5.5](#) for more information about manual failover and fast-start failover, respectively

4.6.2.4 Disable and Enable Operations

When you disable broker management of a standby database, the broker checks to see if the overall protection mode can still be satisfied by any of the remaining standby databases. If not, the broker rejects the disable operation. Otherwise, the broker allows the disable operation to proceed as long as fast-start failover is not enabled. If it is enabled, the broker allows the disable operation to proceed only if the standby database is not the target standby database for fast-start failovers.

Caution: If you disable broker management of a standby database in the broker configuration, that standby database cannot be used by the broker as a failover target in the event of loss of the primary database.

After a standby database is successfully disabled, you can change the redo transport service for that database and the broker will record the change in the broker configuration file. The change will not affect the overall protection mode because it is guaranteed that at least one of the enabled standby databases already satisfies the overall protection mode requirement.

As long as fast-start failover is not enabled, you can disable the entire configuration regardless of the protection mode. You cannot disable the configuration if fast-start failover is enabled. See [Section 5.5.2.2, "Restrictions When Fast-Start Failover is Enabled"](#) for more information.

If the entire configuration is disabled, you can change any broker settings, including the redo transport services of the standby databases and the protection mode of the configuration. The broker saves the changes in the broker configuration file, but the changes will not be made to the database itself.

When enabling broker management of the entire configuration, the broker first checks to see if the protection mode will be satisfied by the redo transport services of the standby databases that will be enabled. If not, the enable operation fails and the configuration remains disabled. Otherwise, the enable operation successfully enables the configuration, and the broker enables the database using the settings saved in the broker configuration file.

4.6.2.5 Requirements For Removing a Database from the Configuration

When removing a standby database from the broker configuration, the broker checks to see if the protection mode will still be satisfied. The operation fails if:

- Removing the database compromises the protection mode
- Fast-start failover is enabled and you try to remove the standby database that is the target of the fast-start failover
- The configuration member to be removed has its `RedoRoutes` configurable property set to a non-null value

You can remove the configuration at any time, unless fast-start failover is enabled.

4.6.2.6 Requirements On Other Operations

Some operations that take place in a broker configuration, especially operations related to redo transport services, can affect the overall protection mode. These operations include:

- Stopping redo transport services on the primary database

- Stopping redo transport services to individual standby databases
- Downgrading the redo transport mode from SYNC to ASYNC to the only standby database that supports a configuration operating in maximum availability mode or maximum protection mode

Before any of these operations can proceed, the broker checks to see if the protection mode will be supported by the redo transport service settings on the standby databases after the operation completes. If not, the broker fails the operation and returns an error.

4.7 Managing Far Sync Instances

An Oracle Data Guard far sync instance is a redo transport destination that accepts redo from a primary database and forwards that redo to one or more redo destinations in the configuration. It is similar to a physical standby database in that it has a control file, receives redo into Standby Redo Log files (SRLs), and archives those SRLs to local Archived Redo Logs (ARLs). But unlike a standby database, a far sync instance does not have data files, cannot be opened, and cannot apply received redo. These limitations yield the benefit of using fewer disk and processing resources. More importantly, a far sync instance provides the ability to failover to a terminal database with no data loss if it receives redo data using synchronous transport mode and the configuration protection mode is set to maximum availability.

The following example shows how to add a far sync instance to a broker configuration.

```
DGMGRL> ADD FAR_SYNC FS1 AS CONNECT IDENTIFIER IS FS1.example.com;
Far Sync FS1 added
DGMGRL> ENABLE FAR_SYNC FS1;
Enabled.
DGMGRL> SHOW CONFIGURATION;
```

Configuration - DRSolution

```
Protection Mode: MaxPerformance
Databases:
North_Sales - Primary database
FS1         - Far Sync
South_Sales - Physical standby database
```

```
Fast-Start Failover: DISABLED
Configuration Status:
SUCCESS
```

After a far sync instance has been added to the configuration, set up redo transport to support maximum availability and then upgrade the protection mode:

```
DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY 'RedoRoutes' = '(LOCAL : FS1
SYNC)';
DGMGRL> EDIT FAR_SYNC 'FS1' SET PROPERTY 'RedoRoutes' = '(North_Sales : South_
Sales ASYNC)';
DGMGRL> EDIT CONFIGURATION SET PROTECTION MODE AS MaxAvailability;
DGMGRL> SHOW CONFIGURATION;
```

Configuration - DRSolution

```
Protection Mode: MaxAvailability
Databases:
North_Sales - Primary database
```

```

FS1          - Far Sync
South_Sales - Physical standby database

```

```

Fast-Start Failover: DISABLED
Configuration Status:
SUCCESS

```

To ensure that maximum availability protection mode can be maintained when `South_Sales` is the primary database, after a switchover or a failover, add a second far sync instance to the configuration so that `South_Sales` can send redo in synchronous mode which in turn will send redo to the new terminal database, `North_Sales`, after the role transition.

The following example shows how to add a second far sync instance to the broker configuration:

```

DGMGRL> ADD FAR_SYNC FS2 AS CONNECT IDENTIFIER IS FS2.example.com;
Far Sync FS2 added
DGMGRL> EDIT FAR_SYNC 'FS2' SET PROPERTY 'RedoRoutes' = '(South_Sales : North_
Sales ASYNC)';
DGMGRL> ENABLE FAR_SYNC FS2;
Enabled.
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY 'RedoRoutes' = '(LOCAL : FS2
SYNC)';
DGMGRL> SHOW CONFIGURATION;

```

```

Configuration - DRSolution

```

```

Protection Mode: MaxAvailability
Databases:
North_Sales - Primary database
FS1          - Far Sync
South_Sales - Physical standby database
FS2          - Far Sync (inactive)

```

```

Fast-Start Failover: DISABLED
Configuration Status:
SUCCESS

```

If a far sync instance is monitored for availability by Oracle Clusterware (for example, in an Oracle Restart, Oracle Real Application Clusters (Oracle RAC), or Oracle RAC One Node installation), then use the `SRVCTL` utility to specify a default open mode of mount. You can use a command such as the following:

```

srvctl modify database -d <db_unique_name> -startoption MOUNT

```

See Also:

- *Oracle Data Guard Concepts and Administration* for more information about creating a far sync instance and redo transport

4.8 Managing Fast-Start Failover

For *lights out administration*, you can enable **fast-start failover** to allow the broker to determine if a failover is necessary and to initiate a failover to a pre-specified target standby database, with either no data loss or a configurable amount of data loss. In addition, you can specify under which conditions or errors you want a failover to be initiated. Oracle also provides the `DBMS_DG` PL/SQL package to allow an application to request a fast-start failover.

You use broker configuration properties to control the behavior of fast-start failover. You can also use Cloud Control or the `DGMGRL ENABLE FAST_START FAILOVER CONDITION` and `DISABLE FAST_START FAILOVER CONDITION` commands to specify conditions for which a fast-start failover should occur.

4.8.1 Configure Properties to Tune Fast-Start Failover

The configurable properties for fast-start failover include:

- `FastStartFailoverThreshold`

Set the `FastStartFailoverThreshold` configuration property to specify the number of seconds you want the observer and target standby database to wait (after detecting the primary database is unavailable) before initiating a failover. See [Section 5.5.2, "Enabling Fast-Start Failover"](#) on page 5-17 for more information and an example.

- `FastStartFailoverPmyShutdown`

The `FastStartFailoverPmyShutdown` configuration property controls whether the primary database will shut down if redo generation has been stalled (`FS_FAILOVER_STATUS` column of `V$DATABASE` contains a value of `STALLED`) and the primary database has lost connectivity with the observer and target standby database for longer than the number of seconds specified by the `FastStartFailoverThreshold` configuration property. The default value for `FastStartFailoverPmyShutdown` is `TRUE`.

Note: The primary database is always shut down if a user configurable fast-start failover condition is detected or if an application initiated a fast-start failover by calling the `DBMS_DG.INITIATE_FS_FAILOVER` function.

- `FastStartFailoverLagLimit`

The fast-start failover feature can be configured on databases operating in maximum performance mode. Destinations that receive redo in `ASYN` mode will be acceptable fast-start failover target standby databases, and these destinations can lag the primary in terms of redo received and applied. A configurable time-based limit can be specified through the `FastStartFailoverLagLimit` configuration property. If the standby database's applied redo point is within this many seconds of the primary's redo generation point, a fast-start failover will be allowed. If its applied point lags beyond this limit, a fast-start failover is not allowed. The `FastStartFailoverLagLimit` configuration property is not used if fast-start failover is enabled when the configuration is operating in maximum availability mode.

An `ASYN` destination is only a valid fast-start failover target in a configuration operating in maximum performance mode. If you want to change protection mode or change the destination to `sync`, you must first disable fast-start failover. Likewise, changing the protection mode from `MAXAVAILABILITY` to `MAXPERFORMANCE` will require first disabling fast-start failover and changing the redo transport mode used to send redo to the primary and target standby to `ASYN`.

Reinstatement of an old primary will be possible after a fast-start failover to an `ASYN` target standby. If the observer rediscovers the old primary, it will automatically reinstate the old primary and any redo generated within the specified lag will be lost.

See Also: [Chapter 8, "Oracle Data Guard Broker Properties"](#) for more information

- `FastStartFailoverAutoReinstate`

The `FastStartFailoverAutoReinstate` configuration property controls whether the former primary database is automatically reinstated if a fast-start failover occurred because the primary database crashed or was stalled for longer than `FastStartFailoverThreshold` seconds. The default value for `FastStartFailoverAutoReinstate` is `TRUE`.

If you want to perform diagnostic or repair work after failover has completed, you can avoid an automatic reinstatement by setting the `FastStartFailoverAutoReinstate` configuration property to `FALSE`.

Note: The former primary database is never automatically reinstated if a fast-start failover occurred because a user configurable fast-start failover condition was detected or because an application initiated a fast-start failover by calling the `DBMS_DG.INITIATE_FS_FAILOVER` function.

- `FastStartFailoverTarget`

The `FastStartFailoverTarget` configuration property specifies the `DB_UNIQUE_NAME` of the database that will be the target of a fast-start failover when this database is the primary database.

- `ObserverReconnect`

The `ObserverReconnect` configuration property specifies how often the observer establishes a new connection to the primary database. When this property is set to the default value of 0, it prevents the observer from periodically establishing a new connection with the primary database. While this eliminates the processing overhead associated with periodically establishing a new observer connection to the primary database, it also prevents the observer from detecting that it is not possible to create new connections to the primary database. Oracle recommends that this property be set to a value that is small enough to allow timely detection of faults at the primary database, but large enough to limit the overhead associated with periodic observer connections to an acceptable level.

- `ObserverOverride`

The `ObserverOverride` configuration property, when set to `TRUE`, allows an automatic failover to occur when the observer has lost connectivity to the primary, even if the standby has a healthy connection to the primary.

4.8.2 Configure Conditions for Fast-start Failover

By default, a fast-start failover is done when neither the observer nor the standby can reach the primary after the configured time threshold (`FastStartFailoverThreshold`) has passed. There are also other conditions under which you might want a fast-start failover to occur.

The configurable conditions fall into two classes: those detected through the database health-check mechanism and those detected through errors raised by the Oracle server (such as ORA errors). When a specified condition occurs, the observer will initiate a fast-start failover without waiting for `FastStartFailoverThreshold` to expire, assuming the standby is in a valid state to accept a failover.

Each condition may be enabled or disabled individually. The Oracle Data Guard configuration persists all user specified configurable fast-start failover conditions in the broker configuration file.

The observer will detect when the primary database has signaled any of the enabled health-check conditions and will immediately initiate a fast-start failover, assuming the standby is in a valid fast-start failover state (observed and either synchronized or within lag limits) to accept a failover.

For specified Oracle ORA-Error conditions, the primary database will notify the observer if the error is signaled and the observer will immediately initiate a fast-start failover, assuming the standby is in a valid fast-start failover state (observed and either synchronized or within lag limits) to accept a failover.

Note: The primary database will shut down and the observer will not attempt to automatically reinstate the former primary database.

See Also:

- Cloud Control online help
- [ENABLE FAST_START FAILOVER CONDITION](#) on page 7-43
- [DISABLE FAST_START FAILOVER CONDITION](#) on page 7-20

4.8.3 Application Initiated Fast-Start Failover

You can use the DBMS_DG PL/SQL package to allow an application to direct a fast-start failover when it encounters specific conditions. See "[Directing a Fast-Start Failover From an Application](#)" on page 5-27.

4.9 Managing Database Conversions

You can use the DGMGRL CONVERT DATABASE command to convert a physical standby database to a snapshot standby database. A snapshot standby database is a fully updatable standby database.

Like a physical or logical standby database, a snapshot standby database receives and archives redo data from a primary database. However, unlike a physical or logical standby database, a snapshot standby database does not apply the redo data that it receives. The redo data received by a snapshot standby database is not applied until the snapshot standby is converted back into a physical standby database, after first discarding any local updates made to the snapshot standby database.

To convert a physical standby database to a snapshot standby database you must have Flashback Database enabled. The following example shows how to convert a physical standby database to a snapshot standby database:

```
DGMGRL> CONVERT DATABASE 'South_Sales' TO SNAPSHOT STANDBY;
```

When you are ready to convert the snapshot back into a physical standby, use the DGMGRL CONVERT DATABASE command as follows:

```
DGMGRL> CONVERT DATABASE 'South_Sales' TO PHYSICAL STANDBY;
```

4.10 Database Status

Database status reveals the health of the database. In general, the broker checks the health of a database by verifying if the actual database state and settings match with those described in the broker configuration file. This is done by checking if any component of the Oracle Data Guard configuration is functioning incorrectly (for example, if redo transport services have an error), and by checking if other required database settings are correctly set (for example, if the server parameter files are available and if the `ARCHIVELOG` mode is turned on). The following is a detailed list of what is being checked by the broker on a primary database and a standby database.

On a primary database, the health check determines whether the following conditions are met:

- Database is in the state specified by the user, as recorded in the broker configuration file
- Database is in the correct data protection mode
- Database is using a server parameter file
- Database is in the `ARCHIVELOG` mode
- Database guard is turned off
- Supplemental logging is turned on when there is a logical standby database in the configuration
- Redo transport services do not have any errors
- Database settings match those specified by the broker configurable properties
- Redo transport settings match those specified by the redo transport-related properties of the standby databases
- Current data protection level is consistent with configured data protection mode
- Primary database is able to resolve all gaps for all standby databases

On a standby database, the health check determines whether the following conditions are met:

- Database is in the state specified by the user, as recorded in the broker configuration file
- Database is using a server parameter file
- Database settings match those specified by the broker configurable properties
- Database guard is turned on when the database is a logical standby database
- Primary and target standby databases are synchronized or within lag limits if fast-start failover is enabled

4.10.1 Querying Database Status

The following monitorable properties can be used to query the database status:

- `LogXptStatus`
- `InconsistentProperties`
- `InconsistentLogXptProps`

Note: These properties are directly accessed through the DGMGRL command-line interface. Cloud Control rearranges the values of these properties for presentation in the GUI.

You can use the `SHOW DATABASE <db_unique_name>` command to get a brief description of the database (name, role, and so on), database status, and information about any health check problems. For example, the output of the following `SHOW DATABASE` command shows two problems: some redo transport services errors and an inconsistent redo transport-related property

```
DGMGRL> SHOW DATABASE 'North_Sales';

Database - North_Sales
Role: PRIMARY
Intended State: TRANSPORT-OFF
Instance(s):
  north_sales1
    Error: ORA-16737: the redo transport service for standby
      database "South_Sales" has an error

  north_sales2
    Error: ORA-16737: the redo transport service for standby
      database "South_Sales" has an error
    Warning: ORA-16715: redo transport-related property
      ReopenSecs of standby "South_Sales" is inconsistent
Database Status:
ERROR
```

To further check the details about the database status, you can use the `LogXptStatus`, `InconsistentProperties`, and `InconsistentLogXptProps` monitorable properties. `LogXptStatus` lists all log transport errors detected on all instances of the primary database. `InconsistentProperties` lists all properties that have inconsistent values between the broker configuration file and the database settings. `InconsistentLogXptProps` lists all redo transport-related properties of standby databases that have inconsistent values between the broker configuration file and the redo transport settings.

Issue the following `SHOW DATABASE` commands to obtain further details about the problems.

```
DGMGRL> SHOW DATABASE 'North_Sales' 'LogXptStatus';
LOG TRANSPORT STATUS
PRIMARY_INSTANCE_NAME  STANDBY_DATABASE_NAME  STATUS
      north_sales1           South_Sales  ORA-12541: TNS:no listener
      north_sales2           South_Sales  ORA-12541: TNS:no listener

DGMGRL> SHOW DATABASE 'North_Sales' 'InconsistentLogXptProps';
INCONSISTENT LOG TRANSPORT PROPERTIES
INSTANCE_NAME  STANDBY_NAME  PROPERTY_NAME  MEMORY_VALUE  BROKER_VALUE
      north_sales2  South_Sales  ReopenSecs           600           300
```

See Also: [Chapter 8](#) for detailed information about database properties

4.10.2 Validating a Database Before a Role Change

You can use the `VALIDATE DATABASE` command to perform a comprehensive set of database checks prior to performing a role change. The command checks the following items:

- Whether there is missing redo data on a standby database
- Whether flashback is enabled
- The number of temporary tablespace files configured
- Whether an online data file move is in progress
- Whether online redo logs are cleared for a physical standby database
- Whether standby redo logs are cleared for a primary database
- The online log file configuration
- The standby log file configuration
- Apply-related property settings
- Transport-related property settings
- Whether there are any errors in the Automatic Diagnostic Repository (for example, control file corruptions, system data file problems, user data file problems)

See Also:

- ["VALIDATE DATABASE"](#) on page 7-81 for a description of the command and for examples that show command output for various scenarios

Switchover and Failover Operations

This chapter describes how the broker manages databases during switchover and failover. It contains the following topics:

- [Overview of Switchover and Failover in a Broker Environment](#)
- [Choosing a Target Standby Database](#)
- [Switchover](#)
- [Manual Failover](#)
- [Fast-Start Failover](#)
- [Database Client Considerations](#)

5.1 Overview of Switchover and Failover in a Broker Environment

An Oracle database operates in one of two roles: primary or standby. Oracle Data Guard helps you change the role of a database using either a switchover or a failover:

- A **switchover** is a role reversal between the primary database and one of its standby databases. A switchover guarantees no data loss and is typically done for planned maintenance of the primary system. During a switchover, the primary database transitions to a standby role, and the standby database transitions to the primary role.
- A failover is a role transition in which one of the standby databases is transitioned to the primary role after the primary database (all instances in the case of an Oracle RAC database) fails or has become unreachable. A failover may or may not result in data loss depending on the protection mode in effect at the time of the failover.

Without the broker, you perform role transitions by issuing a series of SQL statements (as described in *Oracle Data Guard Concepts and Administration*). The broker simplifies switchovers and failovers by allowing you to invoke them using a single key click in Oracle Enterprise Manager Cloud Control (Cloud Control) or a single command in the DGMGRL command-line interface (referred to in this documentation as *manual* failover). Moreover, you can enable fast-start failover to fail over automatically when the conditions for fast-start failover are met. When fast-start failover is enabled, the broker determines if a failover is necessary and initiates the failover to the specified target standby database automatically, with no need for DBA intervention.

Fast-start failover allows you to increase availability with less need for manual intervention, thereby reducing management costs. Manual failover gives you control over exactly when a failover occurs and to which target standby database. Regardless

of the method you choose, the broker coordinates the role transition on all databases in the configuration.

Note: Neither a switchover nor a failover is possible to a far sync instance.

5.2 Choosing a Target Standby Database

There are many factors to take into consideration when selecting a standby database to be the next primary database after a switchover or a failover. You need to consider all of the options at the time you are building your Oracle Data Guard configuration, including factors such as the characteristics of physical standbys versus logical standbys versus snapshot standbys, the network latency to your standby database sites, the computing capabilities at a future primary database site, and so on.

Note: A snapshot standby cannot be the target of a switchover or fast-start failover operation. You can, however, perform a manual failover to a snapshot standby.

A far sync instance is not a database and therefore cannot be the target of a role transition.

For switchovers, understanding all of the factors can simplify the choice of which standby database to consider as your new primary database. In disaster situations where a failover is necessary, you may be more limited as to which standby database is the best one to pick up the failed primary database's activities. [Section 5.2.1](#) and [Section 5.2.2](#) provide guidelines to help you choose a target standby database.

Note: For fast-start failover, you must pre-select the target standby database that will be used. [Section 5.5](#) provides more information about fast-start failover.

Determining a Database's Readiness to Change Roles

To help you select an appropriate switchover or failover target, use the DGMGRL `VALIDATE DATABASE` command. This command performs an exhaustive set of checks on the database to determine whether it is ready to complete a role change.

See Also:

- ["VALIDATE DATABASE"](#) on page 7-81
- ["Scenario 9: Performing a Switchover Operation"](#) on page 6-19 for an example of using the `VALIDATE DATABASE` command to show a database's readiness to complete a role switchover
- ["Scenario 10: Performing a Manual Failover Operation"](#) on page 6-23 for an example of using the `VALIDATE DATABASE` command to show a database's readiness to complete a role failover

5.2.1 Choosing a Target Standby Database for Switchover

When performing a switchover in a configuration whose standby databases are all of the same type (all physical or all logical standby databases), choose the standby

database that has the least amount of unapplied redo. By choosing the standby database with the least amount of unapplied redo, you can minimize the overall time it takes to complete the switchover operation. For example:

- Using DGMGRL, you can do this by examining the Apply Lag row of the `SHOW DATABASE` output for each standby database in the configuration.
- Using Cloud Control, you can view the value of the `ApplyLag` column for each standby database in the `Standby Databases` section of the Oracle Data Guard Overview page.

If the configuration contains both physical and logical standby databases, consider choosing a physical standby database (that has the least amount of unapplied redo) to be the target standby database. A switchover to a physical standby database is preferable because all databases in the configuration will be available as standby databases to the new primary database after the switchover operation completes. Whereas a switchover to a logical standby database will invalidate and disable all of the physical and snapshot standby databases in the configuration. You will then need to re-create the physical standby databases from a copy of the new primary database before you can reenab them.

You cannot perform a switchover to a snapshot standby database unless you first convert it back to a physical standby database.

Note: If the Oracle Data Guard configuration is operating in maximum protection mode, the broker does not allow a switchover to occur to a logical standby database. The configuration must be operating in either maximum availability mode or maximum performance mode in order to be able to switch over to a logical standby database.

5.2.2 Choosing a Target Standby Database for Failover

When performing a failover in a configuration whose standbys are all of the same type, choose the standby database that has the smallest transport lag. Doing so can minimize the amount of data loss and in some cases, incur no data loss at all.

If the configuration contains physical, snapshot, and logical standby databases, consider choosing a physical standby database as the target standby database. A failover to a physical standby database is preferable because it is likely that all standby databases in the configuration will still be available as standby databases to the new primary database after the failover operation completes.

You may failover to a snapshot standby database. However failing over to a snapshot standby database will require more time because the broker must first convert it back to a physical standby database. After the conversion, the broker will start Redo Apply to apply accumulated redo data, before failing the database over to the primary role. Because the broker performs the failover after converting the snapshot standby database to a physical standby database, it is likely that all standby databases in the configuration will still be available as standby databases to the new primary database after the failover operation completes.

A failover to a logical standby database requires that all physical and snapshot standby databases be re-created from a copy of the new primary database after the failover completes. In addition, a logical standby database may contain only a subset of the data present in the primary database. (For example, if the `DBMS_LOGSTDBY.SKIP` procedure was used to specify which database operations done on the primary database will not be applied to the logical standby database.)

However, there may be exceptions to the recommendation to choose a physical standby database as the target standby database. For example, if all your physical standbys are also unavailable, then failing over to a logical standby is your only choice.

5.3 Switchover

You can switch a database from the primary role to the standby role, as well as from standby to primary. This is known as a database switchover, because the standby database that you specify becomes the primary database, and the original primary database becomes a standby database, with no loss of data.

Whenever possible, you should switch over to a physical standby database:

- If the switchover transitions a physical standby database to the primary role, then:
 - The original primary database will be switched to a physical standby role.
 - Bystander standbys will receive redo from the new primary database.
 - The original primary database will be restarted as a part of the switchover operation. Note that the new primary database does not need to be restarted.

Standby databases not involved in the switchover (known as bystander standby databases) continue operating in the state they were in before the switchover occurred and will automatically begin applying redo data received from the new primary database.

- If the switchover transitions a logical standby database to the primary role, then:
 - The original primary database will be switched to a logical standby role.
 - Neither the primary database nor the logical standby database needs to be restarted after the switchover completes.

Other logical standby bystander databases in the broker configuration will remain viable after the switchover. There is no need to restart any databases. All physical and snapshot standby databases will be disabled and must be re-created from a copy of the new primary database after a switchover to a logical standby database.

Switchover to a logical standby database is disallowed when the configuration is operating in maximum protection mode.

WARNING: Switching over to a logical standby database results in the snapshot and physical standby databases in the broker configuration being disabled by the broker, making these databases no longer viable as standby databases. [Section 5.4.3](#) describes how to restore their viability as standby databases.

If you intend to switch back to the original primary database relatively soon, you may allow the physical and snapshot standbys to remain disabled. Once you have completed the switchover back to the original primary, you may then reenable the physical and snapshot standby databases since they are still viable standbys for the original primary database.

5.3.1 Before You Perform a Switchover Operation

Consider the following points before you begin a switchover:

- When you start a switchover, the broker verifies that at least one standby database, including the primary database that is about to be transitioned to the standby role, is configured to support the overall protection mode (maximum protection, maximum availability, or maximum performance) after the switchover is completed.
- Prepare the primary database in advance for its possible future role as a standby database in the context of the overall protection mode (see [Section 4.6](#)). Such preparation includes:
 - Ensuring that standby redo log files are configured on the primary database.
 - Presetting database properties related to redo transport services, such as `LogXptMode`, `NetTimeout`, `StandbyArchiveLocation`, `AlternateLocation`, and `RedoRoutes`. For more details about managing redo transport services using database properties, see [Section 4.4](#).
 - Presetting database properties related to Redo Apply services, such as `DelayMins`. For more details about managing Redo Apply services using properties, see [Section 4.5](#).
 - For each temporary table, verifying that temporary files associated with that table on the primary database also exist on the standby database.

Note that the broker does not use the properties to set up redo transport services and Redo Apply services until you actually switch over the primary database to the standby role. Thus, the validity of the values of these properties is not verified until after the switchover. Once you set these properties, their values persist through role changes during switchover and failover.

- Before performing a switchover to a physical standby database that is in real-time query mode, consider bringing all instances of that standby database to the mounted but not open state to achieve the fastest possible role transition and to cleanly terminate any user sessions connected to the physical standby database prior to the role transition.
- If fast-start failover is enabled, then a switchover can be performed only to the pre-specified target standby database and only if the standby database is synchronized with the primary database or is within the configured lag limit, for the max availability and max performance modes respectively. For information about enabling fast-start failover, see [Section 5.5.2](#).

After a switchover completes, the broker preserves the overall Oracle Data Guard protection mode as part of the switchover process by keeping the protection mode at the same protection level (maximum protection, maximum availability, or maximum performance) it was at before the switchover. Apply services on all other bystander standby databases automatically begin applying redo data received from the new primary database.

If there are physical or snapshot standby databases in the configuration and the switchover occurs to a logical standby database, you need to re-create those databases from a copy of the new primary database and then reenable those databases, as described in [Section 5.4.3](#).

Note: In an Oracle Data Guard configuration, the `SRVCTL -startoption` for a standby database is always set to `OPEN` after a switchover. See "[Database Service Configuration Requirements](#)" on page 5-42 for additional information about how the broker interacts with Oracle Restart.

5.3.2 Starting a Switchover

The act of switching roles should be a well-planned activity. The primary and standby databases involved in the switchover should have as small a redo lag as possible. *Oracle Data Guard Concepts and Administration* provides information about setting up the databases in preparation of a switchover.

To start a switchover using Cloud Control, select the standby database that you want to change to the primary role and click **Switchover**. When using DGMGRL, you need to issue only one `SWITCHOVER` command to specify the name of the standby database that you want to change into the primary role.

The broker controls the rest of the switchover, as described in [Section 5.3.3](#).

5.3.3 How the Broker Performs a Switchover

Once you start the switchover, the broker:

1. Verifies that the primary and the target standby databases are in the following states:

- a. The primary database is enabled and is in the `TRANSPORT-ON` state.
- b. The target standby database is enabled and is in the `APPLY-ON` state.

The broker allows the switchover to proceed as long as there are no errors for the primary database and the standby database that you selected to participate in the switchover operation. Errors occurring for any other bystander standby databases will not impede the switchover.

2. Shuts down all instances except one, if required.

If you are switching over to a physical standby database, the broker shuts down all but one instance on the current primary database. No instances will be shut down on the target physical standby database.

No instances will be shut down if switching over to a logical standby database. You cannot switch over to a snapshot standby database.

3. Switches roles between the primary and standby databases.

The broker first converts the original primary database to run in the standby role. Then, the broker transitions the target standby database to the primary role. If any errors occur during either conversion, the broker stops the switchover. See [Section 9.3, "Troubleshooting Problems During a Switchover Operation"](#) for more information.

4. Updates the broker configuration file to record the change in roles.

This ensures that each database will run in the correct role and state should it be restarted later for any reason.

5. Restarts the new standby (former primary) database if the switchover occurs to a physical standby database, and Redo Apply begins applying redo data from the new primary database. If this is an Oracle RAC physical standby database, the broker directs Oracle Clusterware to restart the instances that were shut down prior to the switchover. In a configuration operating in maximum protection mode, the new primary database will also be restarted.

6. The new primary database is opened in read/write mode and redo transport services are started.

If the former physical standby database was running with real-time query enabled, the new physical standby database will run with real-time query enabled.

The broker verifies the state and status of the databases to ensure that the switchover transitioned the databases to their new role correctly. Bystander standby databases that are not disabled by the broker after the switchover will continue operating in the state they were in before the switchover. Redo Apply and SQL Apply on all other bystander standby databases automatically begin applying redo data received from the new primary database.

In the rare event that a switchover operation fails and you are left with no primary database, retry the switchover command. You can switch back to the original primary and then either retry the switchover to the original target standby, or choose another standby in the configuration to switch over to.

5.4 Manual Failover

You can convert a standby database to a primary database when the original primary database fails and there is no possibility of recovering the primary database in a timely manner. This is known as a **manual failover**. There may or may not be data loss depending upon whether your primary and target standby databases were synchronized at the time of the primary database failure. The word *manual* is used to contrast this type of failover with a fast-start failover (described in [Section 5.5](#)).

Note: You can perform a manual failover even if fast-start failover is enabled. See [Section 5.5.2.4](#) for more information.

The following sections describe how to perform manual failovers:

- [Section 5.4.1, "Complete and Immediate Manual Failovers"](#)
- [Section 5.4.2, "Performing a Manual Failover Operation"](#)
- [Section 5.4.3, "Reenabling Disabled Databases After a Role Change"](#)

5.4.1 Complete and Immediate Manual Failovers

Using Cloud Control or DGMGRL, you can perform either a *complete* (recommended) or an *immediate* failover:

- A *complete* failover is the recommended and default failover option. It automatically recovers the maximum amount of redo data for the protection mode the configuration is operating in. A complete failover also attempts to avoid disabling any standby databases that were not the target of the failover, so that they may continue serving as standby databases to the new primary database.

Whether or not standby databases that were not the target of failover (bystander standby databases) are disabled depends upon how much redo data they have applied relative to the failover target and the standby type of the failover target:

- If the failover target is a physical or snapshot standby database, the original primary database must be reinstated or re-created in order to be a standby database for the new primary database. In addition, some standby databases may be disabled by the broker during the failover if the broker detects that they have applied redo beyond where the new primary database had applied. Any standby database that was disabled by the broker must be reinstated or re-created, as described in [Section 5.4.3](#), before it can be a standby database for the new primary database.

Note that if failover was performed on a snapshot standby database, the old primary must be either reinstated or re-created as a physical standby database.

- If the failover target is a logical standby database, the original primary database and all physical and snapshot standby databases in the configuration will be disabled. The primary database can be reinstated if it had flashback database enabled. The physical and snapshot standby databases will have to be re-created from a copy of the new primary database. See [Section 5.4.3](#) for more information.

If the primary database can be mounted, it may be possible to flush any unsent redo data from the primary database to the target standby database using the `ALTER SYSTEM FLUSH REDO` SQL statement. If this operation is successful, a zero data loss failover may be possible even if the primary database is not in a zero data loss protection mode. See *Oracle Data Guard Concepts and Administration* for more information on using the `ALTER SYSTEM FLUSH REDO` statement.

During a complete failover, the broker performs the failover steps described in [Section 5.4.2.1](#).

- An *immediate* failover is the fastest type of failover. However, no additional data is applied on the standby database once you invoke the failover. Another consequence of immediate failover is that all other databases in the configuration are disabled and must be reinstated or re-created before they can serve as standby databases for the new primary database. [Section 5.4.3](#) describes how to do this. During an immediate failover, the broker performs the failover steps described in [Section 5.4.2.2](#).

Caution: Always try to perform a complete failover first unless redo apply has stopped at the failover target due to an ORA-752 or ORA-600 [3020] error. If one of these errors has occurred, follow the guidelines in "Resolving ORA-752 or ORA-600 [3020] During Standby Recovery" in My Oracle Support Note 1265884.1 before proceeding. This support note is available at <http://support.oracle.com>.

An immediate failover should only be performed when a complete failover is unsuccessful or in the error cases just noted. A complete failover can occur without any data loss, depending on the destination attributes of redo transport services, but an immediate failover usually results in some data loss.

5.4.2 Performing a Manual Failover Operation

After determining that there is no possibility of recovering the primary database in a timely manner, ensure that the primary database is shut down and then begin the failover operation.

The steps in this section describe how to perform a manual failover. Depending on the failover and the types of standby databases involved, some of the databases may need to be reinstated or re-created. The instructions guide you through the appropriate steps for each type of situation.

Step 1 Determine which of the available standby databases is the best target for the failover.

Follow the guidelines described in [Section 5.2, "Choosing a Target Standby Database"](#).

Step 2 Start the failover.

Using Cloud Control or DGMGRL, perform either a *complete* (recommended) or an *immediate* failover.

Manual Failover Using Cloud Control:

On the Oracle Data Guard Overview page in Cloud Control, select the standby database that you want to change to the primary role and click **Failover**. Then, on the Failover Confirmation page, click **Yes** to invoke the default Complete failover option.

Manual Failover Using DGMGRL:

On the target standby database, issue the `FAILOVER` command to perform a failover, specifying the name of the standby database that you want to become the primary database:

```
DGMGRL> FAILOVER TO database-name;
```

Specify the optional `IMMEDIATE` clause to perform an immediate failover if any of the following conditions are true:

- An `ORA-752` error has occurred at the standby database
- An `ORA-600 [3020]` error has occurred at the standby database and Oracle support has determined that it was caused by a lost write at the primary database
- A complete failover is not possible

```
DGMGRL> FAILOVER TO database-name IMMEDIATE;
```

See Also:

- [FAILOVER](#) on page 7-45

If you are performing a complete failover, then all accumulated redo data is applied before the database role is changed to primary. If you are performing an immediate failover, then the database role is changed to primary without applying any accumulated redo data.

If the target is a snapshot standby database, the broker first converts the database to a physical standby database.

No instances will be shut down if failing over to a physical or logical standby database.

Step 3 Reset the protection mode.

After a manual failover (complete or immediate), the overall Oracle Data Guard protection mode is handled as follows:

- If the protection mode was at maximum protection, it is reset to maximum performance. You can upgrade the protection mode later, if necessary, as described in [Section 4.6.1](#).
- If the protection mode was at maximum availability or maximum performance, it remains unchanged.

Note: If you perform a manual failover when fast-start failover is enabled:

- The failover can only be performed to the pre-selected target standby database.
 - The broker preserves the protection mode that was in effect prior to the failover.
-

Step 4 Re-establish a disaster-recovery configuration.

To maintain a viable disaster-recovery solution in the event of another disaster, you may need to perform the additional steps described in [Section 5.4.3](#) to:

- Reinstall the original primary database to act as a standby database in the new configuration.

Caution: Do not attempt to reinstall the old primary database if an ORA-752 or ORA-600 [3020] error has occurred at the failover target, because doing so may lead to data loss. Instead, the old primary database must be re-created as a standby from a backup of the new primary using the procedure described in [Section 5.4.3.2](#).

- Reinstall or re-create standby databases in the configuration that were disabled by the broker.

After a complete failover finishes, any bystander standby database that is not viable as a standby for the new primary database will be disabled by the broker. This can happen for either of the following reasons:

- A bystander standby database has applied more redo data than the new primary database itself had applied when it was a standby database. The standby database must be re-created or reinstated before it can serve as a standby for the new primary database.
- The failover was to a logical standby database. The broker disables all of the physical and snapshot standby databases in the configuration. They must be re-created before they can serve as standby to the new primary database.

5.4.2.1 How the Broker Performs a Complete Failover Operation

Once you start a complete failover, the broker:

1. Verifies that the target standby database is enabled. If the database is not enabled, you will not be able to perform a failover to this database.
2. Waits for the target standby database to finish applying any unapplied redo data before stopping Redo Apply (if the target is a physical standby database) or SQL Apply (if the target is a logical standby database).

If the target is a snapshot standby database, the broker first converts the database back to a physical standby and then starts Redo Apply to apply all the accumulated redo before completing the failover and opening the database as a primary database.

3. Transitions the target standby database into the primary database role, as follows:
 - a. Changes the role of the database from standby to primary.
 - b. Opens the new primary database in read/write mode.
 - c. Determines whether or not any standby databases that did not participate in the failover operation have applied redo data beyond the new primary database, and thus need to be disabled.

If a bystander standby database is not disabled by the broker during this failover, it will remain in the state it was in before the failover. For example, if a physical standby database was in the `APPLY-OFF` state, it will remain in the `APPLY-OFF` state.

By default, the broker always determines whether bystander standby databases will be viable standby databases for the new primary when performing a complete failover. If you want the broker to skip this viability check of bystander standby databases during a complete failover, thus decreasing the overall failover time, set the `BystandersFollowRoleChange` configuration property to `NONE`.

When this property is set to `NONE`, the broker will disable all bystander standby databases without checking whether they have applied more redo data than the new primary database. You will have to reinstate or re-create (see [Section 5.4.3](#)) the standby databases after failover has completed. The `SHOW CONFIGURATION` command will show you which databases can be reinstated and which databases must be re-created. Use the `SHOW CONFIGURATION BystandersFollowRoleChange` command to see the value of this property. The default value is `ALL`.

This property also affects whether the broker skips viability checks of bystander standby databases when a fast-start failover occurs.

- d. Starts redo transport services to begin transmitting redo data to all bystander standby databases that were not disabled.

Note: Bystander standby databases may be disabled by the broker during the failover, and they must be reinstated or re-created before they can serve as standby databases to the new primary database. Oracle recommends configuring Flashback Database on every database so that if failover occurs to a physical standby database, you can more easily reinstate any disabled standby databases. If failover occurs to a logical standby database, all physical and snapshot standby databases will be disabled by the broker. In this case, Flashback Database cannot be used to reinstate databases. They must be re-created from a copy of the new primary database. Logical standby databases that are disabled during failover can be reinstated.

- 4. If the failover target database is an Oracle RAC physical or snapshot standby database, the broker directs Oracle Clusterware to restart all instances that may have been shut down prior to the failover.

The broker allows the failover to proceed as long as there are no errors for the standby database that you selected to participate in the failover. Errors occurring for any bystander standby databases will not stop the failover. If you initiated a complete failover and it fails, you might need to use immediate failover.

Complete Failovers in Configurations Using Far Sync Instances

It is possible to manually perform a complete failover to a standby database that receives redo data from a far sync instance. To failover, connect to the standby database and use the `DGMGRL FAILOVER TO db-unique-name` command. Any unsent redo data residing on the far sync instance is transmitted to the target physical standby prior to converting the physical standby into a primary database.

Complete Failovers in Configurations Using Cascaded Standbys

In a complete failover, it is also possible to failover to a standby database (terminal standby) that gets redo from another standby database (cascader). In such a case, no attempt is made to transmit any unsent redo from the cascader to the terminal standby.

5.4.2.2 How the Broker Performs an Immediate Failover Operation

An immediate failover is started with the following DGMGRL command:

```
DGMGRL> FAILOVER TO database-name IMMEDIATE;
```

Once an immediate failover is started, the broker:

1. Verifies that the target standby database is enabled. If the standby database is not enabled for management by the broker, then the failover cannot occur.
2. Stops Redo Apply or SQL Apply on the standby database immediately, without waiting until all available redo data has been applied. This may result in data loss.
3. Transitions the target standby database into the primary role, opens the new primary database in read/write mode, and starts redo transport services.

After an immediate failover completes, all the standby databases in the configuration, regardless of their type, are disabled. They may be reinstated if Flashback Database is enabled on those databases. Otherwise, they must be re-created from a copy of the new primary database.

The broker allows a complete failover to proceed as long as there are no errors present on the standby database that you selected to participate in the failover.

The broker allows an immediate failover to proceed even if there are errors present on the standby database that you selected to participate in the failover.

Immediate Failovers in Configurations Using Far Sync Instances

It is possible to manually perform an immediate failover to a standby database that receives redo data from a far sync instance. In this case, no attempt is made to transmit any unsent redo from the far sync instance to the target physical standby prior to converting the physical standby into a primary database.

Immediate Failovers in Configurations Using Cascaded Standbys

In an immediate failover, it is also possible to failover to a standby database (terminal standby) that gets redo from another standby database (cascader). In such a case, no attempt is made to transmit any unsent redo from the cascader to the terminal standby.

5.4.3 Reenabling Disabled Databases After a Role Change

To restore your original disaster-recovery solution after switchover to a logical standby database or after failover to any standby database, you may need to perform additional steps.

Databases that have been disabled after a role transition are not *removed* from the broker configuration, but they are no longer managed by the broker.

To reenabling broker management of these databases, you must reinstate or re-create the databases using one of the following procedures:

- If a database can be reinstated, the database will show the following status:

```
ORA-16661: the standby database needs to be reinstated
```

Reinstate the database using the DGMGRL `REINSTATE DATABASE` command or the `reinst` option in Cloud Control, as described in [Section 5.4.3.1, "How to Reinststate a Database"](#) on page 5-13. The broker automatically reenables the database as part of reinstating it.

- If a database must be re-created from a copy of the new primary database, it will have the following status:

ORA-16795: the standby database needs to be re-created

Re-create the standby database from a copy of the primary database and then reenable it. The procedures for creating a standby database are documented in *Oracle Data Guard Concepts and Administration*. See [Section 5.4.3.2, "How to Re-create and Reenable a Disabled Database"](#) on page 5-14 for more information.

Note: Any database that was disabled while multiple role changes were performed cannot be reinstated. You must re-create the database manually from a copy of the current primary database and then reenable the database in the broker configuration.

Whether you reinstate or re-create a database depends on if you performed a switchover or failover and on the type of standby database that was the target of the operation. Note that role changes to logical and snapshot standby databases will always result in physical standby database bystanders being disabled. They cannot be reinstated. They must be re-created from a copy of the new primary database.

The following sections describe how to reinstate or reenable a database.

5.4.3.1 How to Reinstate a Database

You can use the broker's reinstate capability to make the failed primary database a viable standby database for the new primary. This can be done regardless of whether the failover was done to a physical, logical, or snapshot standby database.

You can also reinstate bystander standby databases that were disabled during a failover operation.

Databases that can be reinstated will have the following status value:

ORA-16661: the standby database needs to be reinstated

For the `REINSTATE` command to succeed, Flashback Database must have been enabled on the database prior to the failover and there must be sufficient flashback logs on that database. In addition, the database to be reinstated and the new primary database must have network connectivity.

To reinstate a database:

1. Restart the database to the mounted state
2. Connect to the new primary database
3. Use Cloud Control or DGMGRL to reinstate the database

The broker reinstates a failed primary database as a standby database of the same type (physical or logical standby database) as the old standby database. The only exception to this is failovers to snapshot standby databases. In such cases, the failed primary database is reinstated as a physical standby database.

The broker reinstates bystander standby databases that were disabled during a failover as standby databases to the new primary database.

Reinstatement Using Cloud Control

On the Oracle Data Guard Overview page, click **Database must be reinstated**. This brings up the General Properties page that provides a **Reinstate** button. After you click the **Reinstate** button, Cloud Control begins reinstating the database.

When the process is complete, the database will be enabled as a standby database to the new primary database, and Cloud Control displays the Oracle Data Guard Overview page.

Reinstatement Using DGMGRL

Issue the following command while connected to any database in the broker configuration, except the database that is to be reinstated:

```
DGMGRL> REINSTATE DATABASE db_unique_name;
```

The newly reinstated standby database will begin serving as a standby database to the new primary database. If reinstatement of a database fails, its status changes to ORA-16795: the standby database needs to be re-created. You must then re-create it from a copy of the new primary database and reenable it as described in [Section 5.4.3.2](#).

5.4.3.2 How to Re-create and Reenable a Disabled Database

If you performed a failover or switchover that requires you to re-create the failed primary database or standby databases that were disabled during the role transition, then follow the procedures in the *Oracle Data Guard Concepts and Administration* chapter, "Creating a Physical Standby Database" and also the *Oracle Data Guard Concepts and Administration* chapter, "Creating a Logical Standby Database."

Note that if you are re-creating the old primary database, it must be created as the standby type of the old standby database. For example, if the old standby was a physical or snapshot standby, then the old primary must be re-created as a physical standby.

After the database has been re-created, enable broker management of the re-created standby database by using the DGMGRL `ENABLE DATABASE` command.

5.5 Fast-Start Failover

Fast-start failover allows the broker to automatically fail over to a previously chosen standby database in the event of loss of the primary database. Fast-start failover quickly and reliably fails over the target standby database to the primary database role, without requiring you to perform any manual steps to invoke the failover. Fast-start failover can be used only in a broker configuration and can be configured only through DGMGRL or Cloud Control.

Either maximum availability mode or maximum performance mode can be used with fast-start failover. Maximum availability mode provides an automatic failover environment guaranteed to lose no data. Maximum performance mode provides an automatic failover environment guaranteed to lose no more than the amount of data (in seconds) specified by the `FastStartFailoverLagLimit` configuration property. This property indicates the maximum amount of data loss that is permissible in order for an automatic failover to occur. It is only used when fast-start failover is enabled and the configuration is operating in maximum performance mode.

Once fast-start failover is enabled, the broker will ensure that fast-start failover is only possible when the configured data loss guarantee can be upheld. If the configured data loss guarantee cannot be upheld, redo generation on the primary database will be stalled. To avoid a prolonged stall, either the observer or target standby database may allow the primary database to continue redo generation after first recording that a fast-start failover cannot happen.

The broker restores the ability to automatically failover once the configured data loss guarantee can be satisfied. For a configuration that is operating in maximum availability mode, this occurs once the target standby database has received all missing redo data. For a configuration that is operating in maximum performance mode, this occurs once the target standby database's redo applied point is no longer lagging the primary database's redo generation point by more than the value specified by the `FastStartFailoverLagLimit` configuration property.

Fast-start failover can be enabled in maximum availability mode when the fast-start failover target is a logical or physical standby database that receives redo data from a far sync instance. This lets you take advantage of the broker's automatic failover feature in configurations set up for zero data loss protection at any distance. (Fast-start failover cannot be enabled in maximum performance mode when the target standby receives redo from a far sync instance.) See [Section 6.8](#) for an example of how to set this up.

This section describes how to enable fast-start failover and an **observer** site that monitors the fast-start failover environment. The observer is a separate OCI client-side component that runs on a different computer from the primary and standby databases and monitors the availability of the primary database. The observer is described in more detail in [Section 5.5.7](#).

Once the observer is started, no further user interaction is required. If both the observer and designated standby database lose connectivity with the primary database for longer than the number of seconds specified by the `FastStartFailoverThreshold` configuration property, the observer will initiate a fast-start failover to the standby database. In addition, the primary database will shut down if it perceives a loss of connectivity for a period longer than `FastStartFailoverThreshold` seconds, if the `FastStartFailoverPmyShutdown` configuration property is set to `TRUE`. After the failover completes, the former primary database is automatically reinstated as a standby database when a connection to it is reestablished, if the `FastStartFailoverAutoReinstate` configuration property is set to `TRUE`.

Note: When a fast-start failover occurs because either a user configurable fast-start failover condition is detected or an application initiates a fast-start failover by calling the `DBMS_DG.INITIATE_FS_FAILOVER` function, the former primary database is always shut down and never automatically reinstated. This is true regardless of the settings for the `FastStartFailoverPmyShutdown` and `FastStartFailoverAutoReinstate` configuration properties. See [Section 5.5.2](#) for more information.

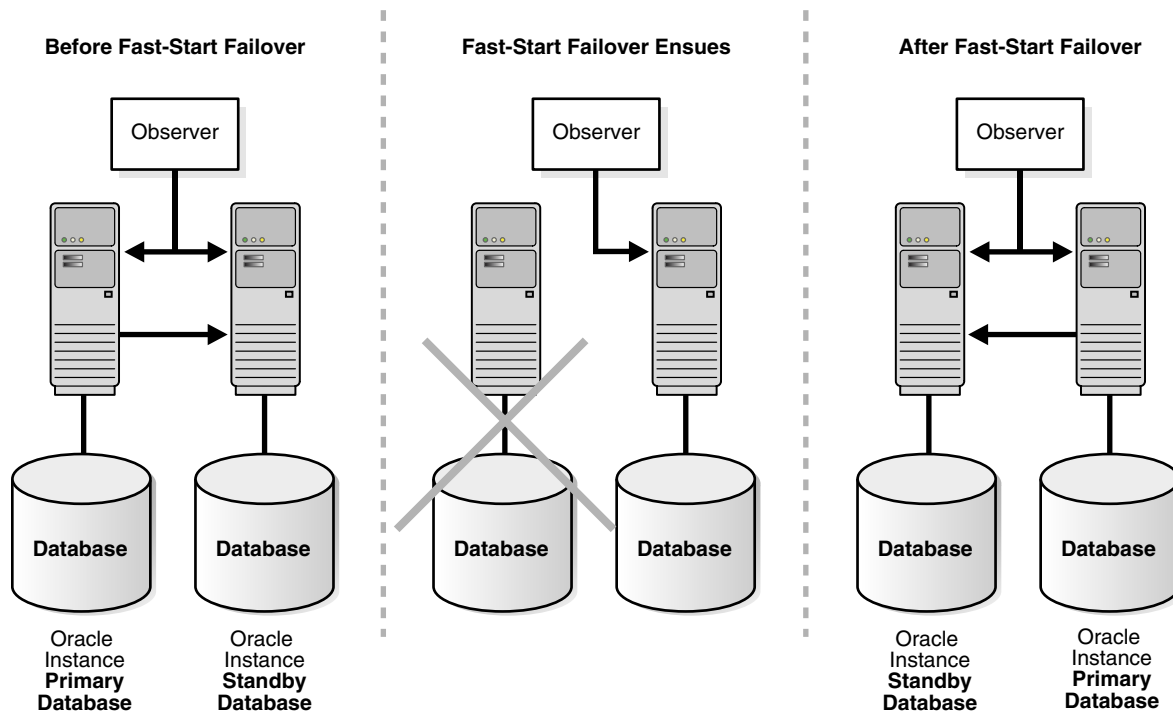
[Figure 5–1](#) shows the relationships between the primary database, target standby database, and the observer during fast-start failover:

- **Before Fast-Start Failover:** Oracle Data Guard is operating in a steady state, with the primary database transmitting redo data to the target standby database and the observer monitoring the state of the entire configuration.
- **FastStart Failover Ensues:** Disaster strikes the primary database and its network connections to both the observer and the target standby database are lost. Upon detecting the break in communication, the observer attempts to reestablish a connection with the primary database for the amount of time defined by the `FastStartFailoverThreshold` property before initiating a fast-start failover. If the observer is unable to regain a connection to the primary database within the

specified time, and the target standby database is ready for fast-start failover, then fast-start failover ensues.

- After Fast-Start Failover:** The fast-start failover has completed and the target standby database is running in the primary database role. After the former primary database has been repaired, the observer reestablishes its connection to that database and reinstates it as a new standby database. The new primary database starts transmitting redo data to the new standby database.

Figure 5–1 Relationship of Primary and Standby Databases and the Observer



The following sections describe these topics:

- [Prerequisites for Enabling Fast-Start Failover](#)
- [Enabling Fast-Start Failover](#)
- [Viewing Fast-Start Failover Configuration Statistics and Status](#)
- [Disabling Fast-Start Failover](#)
- [Performance Considerations for Fast-Start Failover](#)
- [Managing the Observer](#)
- [Reinstating the Former Primary Database in the Broker Configuration](#)
- [Shutting Down Databases In a Fast-Start Failover Environment](#)

5.5.1 Prerequisites for Enabling Fast-Start Failover

The following prerequisites must be met before the broker allows you to enable fast-start failover:

- Ensure the broker configuration is operating in either maximum availability mode or maximum performance mode.

See [Section 4.6.1](#) for information about configuring the protection mode, standby redo logs, and the redo transport mode.

- The selected standby database that will be the fast-start failover target must receive redo directly from the primary database or from a far sync instance.
- Ensure that the standby database you choose to be the target of fast-start failover has its `LogXptMode` property set to either `SYNC` or `FASTSYNC` if you wish to enable fast-start failover in maximum availability mode, or to `ASYNC` if you wish to enable fast-start failover in maximum performance mode. The current primary database must have its `LogXptMode` property set accordingly and must have standby redo logs configured. Alternatively, use the `RedoRoutes` property to configure the redo transport mode for the target standby and the database currently in the primary role.
- To use a far sync instance with fast-start failover, the far sync instance transport mode must be set to either `SYNC` or `FASTSYNC` and the target standby database transport mode must be set to `ASYNC`.
- Enable Flashback Database and set up a fast recovery area on both the primary database and the target standby database.

See *Oracle Database Backup and Recovery User's Guide*.

- Install the DGMGRL command-line interface on the observer computer as described in [Section 2.1](#).
- Configure the `TNSNAMES.ORA` file on the observer system so that the observer is able to connect to the primary database and to the pre-selected target standby database.
- If you are not using Oracle Clusterware or Oracle Restart, then you must create static service names so that the observer can automatically restart a database as part of reinstatement. See [Section 2.2, "Prerequisites"](#) for more information.

5.5.2 Enabling Fast-Start Failover

You can enable fast-start failover from any site while connected to any database in the broker configuration. Enabling fast-start failover does not trigger a failover. Instead, it allows the observer that is monitoring the configuration to initiate a fast-start failover should database conditions warrant a failover. (If there are other conditions, unique to an application, that would warrant a fast-start failover then the application can be set up to call the `DBMS_DG.INITIATE_FS_FAILOVER` function and start a fast-start failover immediately should any of those conditions occur. See [Section 5.5.3](#))

Perform the following steps to enable fast-start failover and start the observer. The steps assume that you are connected as `SYS` and that a primary and standby database are already set up in a broker configuration.

Step 1 Determine which of the available standby databases is the best target for the failover.

Follow the guidelines described in [Section 5.2, "Choosing a Target Standby Database"](#).

Step 2 Specify the target standby database with the `FastStartFailoverTarget` configuration property.

You can specify only one target standby database when setting the `FastStartFailoverTarget` configuration property on the current primary database:

- If there is only one standby database in the configuration, you can skip this step and continue with Step 3. When enabling fast-start failover, the broker automatically sets the `FastStartFailoverTarget` property on the primary and

standby databases to point to each other as their respective target during a failover.

- If there is more than one standby database in the configuration, you must explicitly set the `FastStartFailoverTarget` property on the primary database to select a target standby database. When enabling fast-start failover, the broker verifies that the property indicates an existing standby, and then reciprocally sets the standby database's `FastStartFailoverTarget` property to the primary database. (Note that the target standby cannot be a far-sync standby. However the target can receive redo from a far sync instance.)

Note: To change the `FastStartFailoverTarget` property to point to a different standby database, disable fast-start failover, set the `FastStartFailoverTarget` property, and reenables fast-start failover.

See [Section 8.3.10, "FastStartFailoverTarget"](#) for more information about this property.

Step 3 Determine the protection mode you want

Fast-start failover can be enabled for either maximum availability mode or maximum performance mode. If you cannot tolerate any loss of data, then ensure that the configuration protection mode is set to maximum availability. To do this, set the `LogXptMode` property for both the primary and target standby databases to `SYNC` or `FASTSYNC`. For example:

```
DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY LogXptMode=SYNC;
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY LogXptMode=SYNC;
DGMGRL> EDIT CONFIGURATION SET PROTECTION MODE AS MaxAvailability;
```

Alternatively, use the `RedoRoutes` property to set the redo transport mode for the target standby and database that is currently in the primary role. Then set the configuration protection mode to maximum availability.

If you are more concerned about the performance of the primary database than a minimal loss of data, consider enabling fast-start failover when the configuration protection mode is set to maximum performance. In this mode you will need to consider how much data loss is acceptable in terms of seconds and set the `FastStartFailoverLagLimit` configuration property accordingly. This property specifies the amount of data, in seconds, that the target standby database can lag behind the primary database in terms of redo applied. If the standby database's redo applied point is within that many seconds of the primary database's redo generation point, a fast-start failover will be allowed. The `FastStartFailoverLagLimit` configuration property is only used by the broker when enabling fast-start failover for configurations operating in maximum performance mode. The default value is 30 seconds and the lowest possible value is 10 seconds.

In addition to setting the configuration protection mode to maximum performance, you will also need to ensure that the `LogXptMode` database property for both the primary and target standby database is set to `ASYNCR`. For example:

```
DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY LogXptMode=ASYNCR;
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY LogXptMode=ASYNCR;
DGMGRL> EDIT CONFIGURATION SET PROTECTION MODE AS MaxPerformance;
DGMGRL> EDIT CONFIGURATION SET PROPERTY FastStartFailoverLagLimit=45;
```

Alternatively, use the `RedoRoutes` property to set the redo transport mode to `ASYNCR` for the target standby and the database currently in the primary role.

Step 4 Set the FastStartFailoverThreshold configuration property.

Fast-start failover will occur if both the observer and the target standby database lose connection to the primary database for the period of time specified by the `FastStartFailoverThreshold` configuration property.

Set the `FastStartFailoverThreshold` property to specify the number of seconds you want the observer and target standby database to wait (after detecting the primary database is unavailable) before initiating a failover. For example:

```
DGMGRL> EDIT CONFIGURATION SET PROPERTY FastStartFailoverThreshold = 45;
```

The default value for the `FastStartFailoverThreshold` property is 30 seconds and the lowest possible value is 6 seconds. If you have an Oracle RAC primary database, consider specifying a higher value to minimize the possibility of a false failover in the event of an instance failure.

The time interval starts when the observer first loses its connection to the primary database. If the observer is unable to regain a connection to the primary database within the specified time, then the observer begins a fast-start failover provided the standby database is ready to fail over. Although the default value of 30 seconds is typically adequate for detecting outages and failures on most configurations, you can adjust failover sensitivity with this property to decrease the probability of false failovers in a temporarily unstable environment.

If the `FastStartFailoverPmyShutdown` configuration property is set to `TRUE`, the primary database will shut down after `FastStartFailoverThreshold` seconds has elapsed if redo generation has been stalled and the primary database is unable to reestablish connectivity with either the observer or target standby database.

Note that the `FastStartFailoverThreshold` property can be changed even when fast-start failover is enabled.

See Also: [Section 8.1.8](#) for reference information about the `FastStartFailoverThreshold` property

Step 5 Set other properties related to fast-start failover (optional).

You can optionally set the database properties described in the following table:

Property Name	Description	Default Value
<code>FastStartFailoverPmyShutdown</code>	<p>This configuration property causes the primary database to shut down if fast-start failover is enabled and <code>V\$DATABASE.FS_FAILOVER_STATUS</code> indicates the primary has been <code>STALLED</code> for longer than <code>FastStartFailoverThreshold</code> seconds. A value of <code>TRUE</code> helps to ensure that an isolated primary database cannot satisfy user queries.</p> <p>This property cannot be used to prevent the primary database from shutting down if a fast-start failover occurred because a user configuration condition was detected or was requested by an application by calling the <code>DBMS_DG.INITIATE_FS_FAILOVER</code> function.</p>	<code>TRUE</code>

Property Name	Description	Default Value
FastStartFailoverLagLimit	<p>This configuration property establishes an acceptable limit, in seconds, that the standby is allowed to fall behind the primary in terms of redo applied, beyond which a fast-start failover will not be allowed. The lowest possible value is 10 seconds.</p> <p>This property is used when fast-start failover is enabled and the configuration is operating in maximum performance mode.</p>	30 seconds
FastStartFailoverAutoReinstate	<p>This configuration property causes the former primary database to be automatically reinstated if a fast-start failover was initiated because the primary database was either isolated or had crashed. To prevent automatic reinstatement of the former primary database in these cases, set this configuration property to <code>FALSE</code>. The broker never automatically reinstates the former primary database if a fast-start failover was initiated because a user configuration condition was detected or was requested by an application calling the <code>DBMS_DG.INITIATE_FS_FAILOVER</code> function.</p>	TRUE
ObserverConnectIdentifier	<p>This database property is used to specify how the observer should connect to and monitor the primary and standby database. Set this property for the primary and target standby database if you want the observer to use a different connect identifier than that used to ship redo data (that is, the connect identifier specified by the <code>DGConnectIdentifier</code> property).</p>	Observer uses the value of the <code>DGConnectIdentifier</code> property to connect to and monitor the primary and target standby databases.
ObserverOverride	<p>The <code>ObserverOverride</code> configuration property, when set to <code>TRUE</code>, allows an automatic failover to occur when the observer has lost connectivity to the primary, even if the standby has a healthy connection to the primary.</p>	FALSE
ObserverReconnect	<p>The <code>ObserverReconnect</code> configuration property specifies how often the observer establishes a new connection to the primary database. When this property is set to the default value of 0, it prevents the observer from periodically establishing a new connection with the primary database. While this eliminates the processing overhead associated with periodically establishing a new observer connection to the primary database, it also prevents the observer from detecting that it is not possible to create new connections to the primary database. Oracle recommends that this property be set to a value that is small enough to allow timely detection of faults at the primary database, but large enough to limit the overhead associated with periodic observer connections to an acceptable level.</p>	0 (zero)

Step 6 Enable additional fast-start failover conditions (optional)

By default, a fast-start failover is done when both the observer and the standby cannot reach the primary after the configured time threshold (`FastStartFailoverThreshold`) has passed.

You can optionally indicate the database health conditions that should cause fast-start failover to occur. These conditions are described in the following table:

Health Condition	Description	Enabled by Default
Datafile Offline	A datafile is offline because of a write error.	Yes
Corrupted Dictionary	Dictionary corruption of a critical database. Currently, this state can be detected only when the database is open	Yes
Corrupted Controlfile	Controlfile is permanently damaged because of a disk failure.	Yes
Inaccessible Logfile	LGWR is unable to write to any member of the log group because on an I/O error	No
Stuck Archiver	Archiver is unable to archive a redo log because the device is full or unavailable.	No

In Oracle RAC configurations, the Inaccessible Logfile and Stuck Archiver health conditions may only be applicable to a single instance. Careful consideration should be given before enabling fast-start failover for either of these conditions because doing so will supersede availability options provided by Oracle Clusterware.

You can specify particular conditions for which a fast-start failover should occur using either Cloud Control or the DGMGRL `ENABLE FAST_START FAILOVER CONDITION` and `DISABLE FAST_START FAILOVER CONDITION` commands.

Step 7 Enable fast-start failover.

Use the Cloud Control Fast-Start Failover wizard or the DGMGRL `ENABLE FAST_START FAILOVER` command to enable fast-start failover. To enable fast-start failover, both the primary and target standby databases must be running and have connectivity, and satisfy all of the prerequisite conditions listed in [Section 5.5.1](#).

Enable Fast-Start Failover Using Cloud Control

To enable fast-start failover in Cloud Control, use the Fast-Start Failover wizard. On the Oracle Data Guard Overview page next to the Fast-Start Failover status field, click **Disabled** to invoke the Fast-Start Failover page. Then, on the Fast-Start Failover Change Mode page, click **Enabled**. Cloud Control will start the observer. Then, on the Fast-Start Failover Configure page, select the standby database that should be the target of a failover. See [Section 5.2, "Choosing a Target Standby Database"](#) for helpful advice. This page will not allow you to alter the protection mode. Rather, fast-start failover will be enabled in accordance with the current protection mode. If the currently configured mode is maximum protection, Cloud Control will downgrade the mode to maximum availability.

Enable Fast-Start Failover Using DGMGRL

To enable fast-start failover with DGMGRL, issue the `ENABLE FAST_START FAILOVER` command while connected to any database in the broker configuration, including on the observer computer. For example:

```
DGMGRL> ENABLE FAST_START FAILOVER;
Enabled.
```

See Also:

- [ENABLE FAST_START FAILOVER](#) on page 7-41
- [Section 6.7, "Scenario 6: Enabling Fast-Start Failover and Starting the Observer"](#) on page 6-11

Note: Administration at the target standby site should be as comprehensive as that at the primary site because the standby database may assume the primary role without prior notice. Staff support, hardware and software, security (both software and site), network connections, and bandwidth should be equivalent at both sites.

Step 8 Start the Observer.

The primary database must be running in order to start the observer.

You can start the observer before or after you enable fast-start failover. If fast-start failover is already enabled, the observer immediately begins monitoring the status and connections to the primary and target standby databases. If fast-start failover is not already enabled, the observer waits until fast-start failover gets enabled and then begins monitoring.

Starting the Observer Using Cloud Control

If the Cloud Control agent is installed on the observer computer, it automatically starts the observer when you enable fast-start failover through Cloud Control. If the agent is not present, you must start the observer manually using the following instructions for the DGMGRL command-line interface.

Starting the Observer Using DGMGRL

To start the observer with DGMGRL, issue the following command on the observer computer:

```
DGMGRL> START OBSERVER;
```

The observer is a continuously executing process that is created when the `START OBSERVER` command is issued. Thus, the command-line prompt on the observer computer does not return until you issue the `STOP OBSERVER` command from another DGMGRL session. To issue commands and interact with the broker configuration, you must connect through another DGMGRL client session.

See the [START OBSERVER](#) command on page 7-72 for more information.

Step 9 Verify the fast-start failover environment.

To verify the readiness of the fast-start failover configuration, issue the DGMGRL `SHOW CONFIGURATION VERBOSE` command or the `SHOW FAST_START FAILOVER` command on the primary database. For example:

```
DGMGRL> SHOW FAST_START FAILOVER;
```

```
Fast-Start Failover: ENABLED
Threshold: 60 seconds
Target: South_Sales
Observer: observer.example.com
Lag Limit: 30 seconds (not in use)
Shutdown Primary: TRUE
Auto-reinstate: TRUE
Observer Reconnect: (none)
Observer Override: FALSE
```

```
Configurable Failover Conditions
Health Conditions:
  Corrupted Controlfile YES
  Corrupted Dictionary YES
```



```
Inaccessible Logfile NO
Stuck Archiver NO
Datafile Offline YES
```

```
Oracle Error Conditions:
(none)
```

The following sections provide more information about the fast-start failover environment:

- [When Fast-Start Failover Is Enabled and the Observer Is Running](#)
- [Restrictions When Fast-Start Failover is Enabled](#)
- [Shutting Down the Primary Database When Fast-Start Failover Is Enabled](#)
- [Performing Manual Role Changes When Fast-Start Failover Is Enabled](#)

5.5.2.1 When Fast-Start Failover Is Enabled and the Observer Is Running

Once you enable fast-start failover and start the observer, the observer continuously monitors the environment to ensure the primary database is available. This section lists the steps the observer takes to determine if a fast-start failover is needed and then to perform one, if necessary.

Step 1 Monitor the environment to ensure the primary database is available.

The observer waits the number of seconds specified by the `FastStartFailoverThreshold` configuration property before attempting a fast-start failover when the primary database has crashed or has lost connectivity with the observer, as in the following situations:

- The primary database loses its connections with both the observer and target standby database
- Instance failures

If a single-instance primary database (either Oracle RAC or non-Oracle RAC), or if all instances of an Oracle RAC primary database fail, the observer attempts a fast-start failover.
- Shutdown abort

If a single-instance primary database (either Oracle RAC or non-Oracle RAC), or if all instances of an Oracle RAC primary database are shut down with the `ABORT` option, the observer attempts a fast-start failover. Fast-start failover will not be attempted for the other types of database shutdown (`NORMAL`, `IMMEDIATE`, `TRANSACTIONAL`).

The observer never waits for the threshold to expire to perform a fast-start failover in the following situations:

- User-configurable condition

If the observer determines that any of the user-configurable conditions has been detected, the observer attempts a fast-start failover.
- Application calls to `DBMS_DG.INITIATE_FS_FAILOVER`

If an application has called this function and it has received a status of `SUCCESS`, the observer attempts a fast-start failover.

Step 2 Reconnect within the time specified by FastStartFailoverThreshold.

If the observer detects an availability problem with the primary database, the observer typically attempts to reconnect to the primary database within the time specified by the `FastStartFailoverThreshold` configuration property. The `FastStartFailoverThreshold` time interval starts when the observer first detects there might be a failure with the primary database.

The time interval specified by the `FastStartFailoverThreshold` property is ignored if the observer detects that a user-configurable condition has occurred or if a fast-start failover has been requested by the `DBMS_DG.INITIATE_FS_FAILOVER` function.

If the primary database is an Oracle Real Application Clusters (Oracle RAC) database, the observer will attempt to connect to one of the remaining primary instances. Fast-start failover will not occur unless all instances comprising the Oracle RAC primary database are perceived to have failed. The observer uses the value specified by either the `DGConnectIdentifier` or `ObserverConnectIdentifier` database properties to connect to the primary and fast-start failover target standby databases. The value specified for either of these properties should allow the observer to connect to any instance of an Oracle RAC database.

Step 3 Verify the target standby database is ready for failover.

If fast-start failover is initiated, the observer verifies the target standby database is ready to fail over to the primary database role.

Fast-start failover *cannot* occur if:

- Fast-start failover is no longer enabled
- The observer cannot connect to the target standby database

See Also: [Section 5.5.7.3, "What Happens if the Observer Fails?"](#) if the observer is not running

- The observer and the target standby database are inconsistent with regard to the current state of the broker configuration
- The observer is not running
- If the protection mode is maximum availability and the target standby database was not synchronized with the primary database at the time the primary database failed
- If the protection mode is maximum performance and the apply point of the target standby database lags the redo generation point of the primary database by more than the amount specified by the `FastStartFailoverLagLimit` configuration property at the time the primary database failed
- The target standby database has contact with the primary database. However, failover is attempted if the `ObserverOverride` configuration property is set to `TRUE`.
- The `FS_FAILOVER_STATUS` column in the `V$DATABASE` view for the target standby database displays a reason why fast-start failover cannot occur
- A manual failover is already in progress. See [Section 5.4](#) for complete information about manual failovers.
- The primary database was shut down without using the `ABORT` option

Step 4 Initiate a fast-start failover.

If the target standby database is ready for failover, the observer immediately directs the target standby database to fail over to the primary database role. If failover is not possible for some reason, the observer will continue checking whether the standby database is ready to fail over. But it will also continue trying to reconnect to the primary database indefinitely. If it reconnects to the primary database before the standby agrees to fail over, the observer will stop attempting to initiate a fast-start failover.

Step 5 Reinstate the former primary database as a new standby database.

After the fast-start failover completes successfully, the observer will attempt to reinstate the former primary database as a new standby database when a connection to the former primary database is reestablished, and the `FastStartFailoverAutoReinstate` configuration property is set to `TRUE`. If the `FastStartFailoverPmyShutdown` configuration property is set to `TRUE`, the former primary database will have been automatically shut down and must be manually restarted before the observer can attempt to reinstate it.

Note that these properties only affect whether primary shutdown and automatic reinstatement are performed if a fast-start failover occurs because the primary crashed or was isolated from the observer and target standby database.

See Also: [Section 5.5.8](#) for more information about reinstatement

5.5.2.2 Restrictions When Fast-Start Failover is Enabled

When fast-start failover is enabled, you cannot:

- Change:
 - The configuration protection mode
 - The redo transport mode used to send redo to the target standby database or the database currently in the primary role
 - The `FastStartFailoverTarget` configuration property on the primary or target standby databases
 - The `RedoRoutes` property on the primary or target standby databases
 - The `RedoRoutes` property on a far sync instance if it is being used to receive redo from the primary database and ship redo to the target standby database
- Disable or delete:
 - The broker configuration
 - The standby database that is the target of fast-start failover
 - A far sync instance if it is being used to receive redo from the primary database and ship redo to the target standby database
- Perform a manual failover:
 - Unless the conditions listed in [Section 5.5.2.4](#) have been met
 - To a standby database that is not configured as the fast-start failover target

To determine if the configuration is ready for fast-start failover to occur, issue the `DGMGRL SHOW DATABASE <target-standby-database>` command, or query the `V$DATABASE` view on either the primary or target standby databases. The column value for `V$DATABASE.FS_FAILOVER_STATUS` will be `SYNCHRONIZED` in a configuration operating in maximum availability mode, and it will be

TARGET UNDER LAG LIMIT in a configuration operating in maximum performance mode when ready to fast-start failover. The FS_FAILOVER_OBSERVER_PRESENT column displays YES for the target standby database.

- Perform a switchover to a standby database that is not configured as the fast-start failover target
- Perform a switchover to the target standby database in a configuration operating in maximum availability mode, unless the standby database is synchronized with the primary database
- Perform a switchover to the target standby database in a configuration operating in maximum performance mode, unless the standby database is within the lag limit of the primary database
- Attempt to open the primary database, or the following error may be returned:

```
ORA-16649: possible failover to another database prevents this database from being opened
```

This error may return if the fast-start failover validity check fails or does not complete in under two minutes.
- Use the SQL `ALTER DATABASE MOVE DATAFILE` command to rename or relocate an online data file on a physical standby that is a fast-start failover target if the standby is mounted, but not open.

5.5.2.3 Shutting Down the Primary Database When Fast-Start Failover Is Enabled

Fast-start failover will not be triggered if the primary or standby database is shut down normally (using `SHUTDOWN NORMAL`, `SHUTDOWN IMMEDIATE`, or `SHUTDOWN TRANSACTIONAL`). A normal shutdown will prevent fast-start failover until the primary database and standby database are connected and communicating again.

5.5.2.4 Performing Manual Role Changes When Fast-Start Failover Is Enabled

If fast-start failover is enabled you can still perform a switchover or a manual failover as long as the following conditions are met:

- The role change is directed to the same standby database that was specified for the `FastStartFailoverTarget` database property on the primary database.
- The target standby database is synchronized with the primary database if it is a configuration operating in maximum availability mode, or the target standby database is within the lag limit if it is a configuration operating in maximum performance mode.
- For manual failover, the observer is started and communicating with the target standby database. You must ensure that the primary database is shut down prior to performing a manual failover.

Note: You can disable fast-start failover if necessary, by using the `FORCE` option. See [Section 5.5.5, "Disabling Fast-Start Failover"](#).

See Also: [Section 5.3](#) and [Section 5.4](#) for more information about switchovers and manual failovers, respectively

5.5.3 Directing a Fast-Start Failover From an Application

You can customize fast-start failover setup for a specific application by using the `DBMS_DG` PL/SQL package. When a serious condition uniquely known to an application is detected, the application can call the `DBMS_DG.INITIATE_FS_FAILOVER` function to initiate an immediate fast-start failover. This function can be called from a connection to either the primary or any standby in the configuration. The database on which the procedure is called notifies the observer. The observer immediately initiates a fast-start failover, as long as the failover target database is in a valid fast-start failover state ("observed" and either "synchronized" or "within lag") to accept a failover. Once the observer has initiated a fast-start failover, the primary database shuts down automatically. The observer does not attempt to reinstate the former primary database.

If the configuration is not failable, the `DBMS_DG.INITIATE_FS_FAILOVER` function returns an ORA error number (it does not signal an exception) informing the caller that a fast-start failover could not be performed.

Note: An application should use caution when calling the `DBMS_DG.INITIATE_FS_FAILOVER` function because the observer will initiate failover, if at all possible.

See Also: *Oracle Database PL/SQL Packages and Types Reference* for more information about the `DBMS_DG` package

5.5.4 Viewing Fast-Start Failover Configuration Statistics and Status

To verify the observer is started and the configuration is ready for fast-start failover, you can issue the `DGMGRL SHOW DATABASE <target-standby-database>` command or query the `V$DATABASE` view on the target standby database.

You can also query the `V$FS_FAILOVER_STATS` view to display statistics about fast-start failover occurring on the system.

The rest of this section provides examples of using `DGMGRL SHOW` commands to display fast-start failover information and includes sections describing the following views:

- [V\\$DATABASE View](#)
- [V\\$FS_FAILOVER_STATS View](#)

Example 1 SHOW FAST-START FAILOVER

The `DGMGRL SHOW FAST-START FAILOVER` command displays all the fast-start failover related information. For example:

```
DGMGRL> SHOW FAST_START FAILOVER;

Fast-Start Failover: ENABLED
Threshold:           60 seconds
Target:             South_Sales
Observer:           observer.example.com
Lag Limit:          30 seconds (not in use)
Shutdown Primary:  TRUE
Auto-reinstate:     TRUE
Observer Reconnect: (none)
Observer Override:  FALSE
```

Configurable Failover Conditions

```

Health Conditions:
  Corrupted Controlfile      YES
  Corrupted Dictionary      YES
  Inaccessible Logfile      NO
  Stuck Archiver            NO
  Datafile Offline          YES

```

```

Oracle Error Conditions:
  (none)

```

Example 2 SHOW CONFIGURATION VERBOSE

The following example shows the fast-start failover information for the DRSolution configuration:

```

Configuration - DRSolution

Protection Mode: MaxAvailability
Databases:
North_Sales - Primary database
South_Sales - (*) Physical standby database

(*) Fast-Start Failover target

Properties:
  FastStartFailoverThreshold = '60'
  OperationTimeout           = '30'
  TraceLevel                 = 'USER'
  FastStartFailoverLagLimit  = '30'
  CommunicationTimeout       = '180'
  ObserverReconnect          = '0'
  FastStartFailoverAutoReinstate = 'TRUE'
  FastStartFailoverPmyShutdown = 'TRUE'
  BystandersFollowRoleChange = 'ALL'
  ObserverOverride           = 'FALSE'
  ExternalDestination1      = ''
  ExternalDestination2      = ''
  PrimaryLostWriteAction    = 'CONTINUE'

Fast-Start Failover: ENABLED

  Threshold: 30 seconds
  Target: South_Sales
  Observer: observer.example.com
  Lag Limit: 30 seconds (not in use)
  Shutdown Primary: TRUE
  Auto-reinstate: TRUE
  Observer Reconnect: (none)
  Observer Override: FALSE

Configuration Status:
SUCCESS

```

5.5.4.1 V\$DATABASE View

You can query the V\$DATABASE view to verify that the observer is started and the configuration is ready for fast-start failover. When querying the V\$DATABASE view, pay special attention to the following:

- The FS_FAILOVER_STATUS column, which can contain the values described in [Table 5-1](#). Note that if the V\$DATABASE.FS_FAILOVER_STATUS column has a value of

DISABLED, then any values returned for the remaining columns related to fast-start failover (V\$DATABASE.FS_FAILOVER_*) become irrelevant.

- The FS_FAILOVER_OBSERVER_PRESENT column, which indicates whether the observer is running and actively pinging the database.

Table 5–1 FS_FAILOVER_STATUS Column of the V\$DATABASE View

Column Value	Description	Fast-Start Failover ...
BYSTANDER	Fast-start failover is enabled, but this standby database is not the target of the fast-start failover. The database cannot provide fast-start failover status information.	Is enabled
DISABLED	Fast-start failover is disabled.	Is not possible
LOADING DICTIONARY	Displays only on a logical standby database that has not yet completed loading a copy of the primary database's data dictionary.	Is not possible
PRIMARY UNOBSERVED	Displays only on the target standby database when it is SYNCHRONIZED with or is TARGET UNDER LAG LIMIT of the primary database, has connectivity to the observer, but the primary database does not have a connection to the observer.	Is not possible
REINSTATE FAILED	Reinstatement of the failed primary database as a new standby database failed. See Section 9.1 for details about the broker's drc* log files.	Has completed
REINSTATE REQUIRED	The failed primary database requires reinstatement as a new standby database to the new primary. The observer automatically starts the reinstatement process. REINSTATE REQUIRED is present only after fast-start failover has occurred and shows on both the new primary database and the database undergoing reinstatement. This is cleared on both when the reinstatement has been completed.	Has completed
STALLED	Displays on the primary database after loss of connectivity to the target standby database and the change to the UNSYNCHRONIZED state (maximum availability mode) or to the TARGET OVER LAG LIMIT state (maximum performance mode) cannot be confirmed by either the target standby database or the observer. Note that the value of the FastStartFailoverPmyShutdown configuration property must be FALSE for the primary to stall indefinitely under these conditions. With a value of TRUE for this property, the primary will shut down after being stalled for the number of seconds specified by the FastStartFailoverThreshold property. It shuts down or stalls because it is likely a failover has occurred. Note: this state also occurs on the primary during startup when fast-start failover is possible and neither the target standby database nor the observer are present to confirm it is okay to continue opening the database.	Is possible

Table 5-1 (Cont.) FS_FAILOVER_STATUS Column of the V\$DATABASE View

Column Value	Description	Fast-Start Failover ...
TARGET OVER LAG LIMIT	Displays if the standby database's redo applied point lags the primary database's redo generation point by more than the number of seconds specified by the FastStartFailoverLagLimit configuration property and the configuration is operating in maximum performance mode.	Is not possible
TARGET UNDER LAG LIMIT	Displays if the standby database's redo applied point does not lag the primary database's redo generation point by more than the number of seconds specified by the FastStartFailoverLagLimit configuration property and the configuration is operating in maximum performance mode.	Is possible
SUSPENDED	Displays only on the target standby database when either the primary or target standby database was shut down in a controlled fashion (using the NORMAL, IMMEDIATE, or TRANSACTIONAL, options, but not the ABORT option). Fast-start failover is inhibited in this case. SUSPENDED is cleared when connectivity with the primary database is restored.	Is not possible
SYNCHRONIZED	Displays when the primary and target standby databases are synchronized and the configuration is operating in maximum availability mode.	Is possible if the target standby database displays SYNCHRONIZED and the FS_FAILOVER_OBSERVER_PRESENT column displays YES
UNSYNCHRONIZED	Displays when the target standby database does not have all of the primary database redo data and the configuration is operating in maximum availability mode.	Is not possible

5.5.4.2 V\$FS_FAILOVER_STATS View

Because fast-start failovers are fully automated and can occur at any time, it is useful to query this view on the primary database to display statistics about fast-start failovers that have occurred on the system, including:

- LAST_FAILOVER_TIME that shows the timestamp of last fast-start failover
- LAST_FAILOVER_REASON that shows the reason for the last fast-start failover

The following is an example of querying the V\$FS_FAILOVER_STATS view:

```
SQL> SELECT LAST_FAILOVER_TIME, LAST_FAILOVER_REASON FROM V$FS_FAILOVER_STATS;
```

```
LAST_FAILOVER_TIME
-----
LAST_FAILOVER_REASON
-----
```

```
02/13/2007 16:53:10
Primary Disconnected
```

5.5.5 Disabling Fast-Start Failover

Disabling fast-start failover prevents the observer from initiating a failover to the target standby database. In this case, manual failover may still be possible. See

[Section 5.4](#) for information about manual failover.

Note: Disabling fast-start failover does not stop the observer. To stop the observer, see [Section 5.5.7.5, "Stopping the Observer"](#).

To disable fast-start failover, use the Fast-Start Failover wizard in Cloud Control or the `DGMGRL DISABLE FAST_START FAILOVER [FORCE]` command. The `FORCE` option disables fast-start failover on the database to which you are connected even when errors occur. Whether or not you need the `FORCE` option depends mostly on if the primary and target standby database have network connectivity:

- If the primary and target standby database have network connectivity, and the database to which you are connected has network connectivity with the primary database, the `FORCE` option has no effect. Simply use `DISABLE FAST_START FAILOVER`. This method will disable fast-start failover on all databases in the broker configuration.

If errors occur during the disable operation, the broker returns an error message and stops the disable operation.

- If the primary and target standby databases do not have network connectivity or if the database to which you are connected does not have network connectivity with the primary database, consider using `DISABLE FAST_START FAILOVER` *with* the `FORCE` option.

The broker may not be able to disable fast-start failover on all databases in the broker configuration when you issue the `DISABLE FAST_START FAILOVER FORCE` command. As a result, there is no guarantee that the observer will not perform a fast-start failover to the target standby database if the observer determines that conditions warrant a failover. The following list indicates the extent to which fast-start failover is disabled in the broker configuration when the `DISABLE FAST_START FAILOVER FORCE` command is issued on the primary database, target standby database, and a standby database that is not the fast-start failover target.

If you issue this command on:

- The target standby database when it does not have connectivity with the primary database, fast-start failover is disabled only on the target standby database. In this case, the observer cannot perform a fast-start failover even if conditions warrant a failover. Disabling fast-start failover with the `FORCE` option when connected to the target standby database guarantees that fast-start failover will not occur.

When the primary database and the target standby database regain network connectivity, the broker will disable fast-start failover for the entire broker configuration.

- The primary database, it attempts to disable fast-start failover on as many databases in the configuration with which it has a network connection. If the primary database does not have connectivity with the target standby database, fast-start failover remains enabled on the target standby database and the observer may still attempt a fast-start failover if conditions warrant a failover.

Caution: This action may result in two databases in the configuration simultaneously assuming the primary database role should fast-start failover occur. For this reason, you should first issue this command on the target standby database.

- Another standby database that does not have connectivity with the primary database, fast-start failover is disabled for this database. Because fast-start failover was not disabled on the target standby database, the observer may still attempt a fast-start failover to the target standby database should conditions warrant a failover.

When the primary database and the (non-target) standby database regain network connectivity, the broker will propagate its current fast-start failover setting (ENABLED or DISABLED) to the non-target standby.

Caution: When you are experiencing network disconnections and you issue the `DISABLE FAST_START FAILOVER FORCE` command on the primary database or a standby database that does not have connectivity with the primary database, fast-start failover may not be disabled for all databases in the broker configuration. As a result the observer may still initiate fast-start failover to the target standby database, if conditions warrant a failover. This may result in two databases in the configuration simultaneously assuming the primary database role.

Conditions Requiring the FORCE Option

Disabling fast-start failover without the `FORCE` option can succeed only if the database on which the command is issued has a network connection with the primary database and if the primary database and target standby database have a network connection. This is the recommended method for disabling fast-start failover.

However, there may be situations in which you must disable fast-start failover when the primary database and the target standby database do not have a network connection, or the database on which you issued the disable fast-start failover command does not have a network connection to the primary database. In cases where there is a lost network connection, be aware that the observer may attempt a fast-start failover to the target standby database if conditions warrant a failover.

The `FORCE` option may be the preferred method for disabling fast-start failover when:

- A network outage isolates the primary database from the observer and the target standby database before conditions exist that warrant a failover.

In this case, the primary database stalls and prevents any further transactions from committing because a fast-start failover may have occurred while it was isolated. If you expect the network to be disconnected for a long time and you need to make the primary database available, first confirm that a fast-start failover has not occurred to the target standby database. Then, disable fast-start failover with the `FORCE` option on the primary database.

If possible, confirm that fast-start failover has not occurred to the target standby database prior to disabling fast-start failover with the `FORCE` option on the primary database.

Caution: This action may result in two databases in the configuration simultaneously assuming the primary database role. This can be avoided by first disabling fast-start failover with the `FORCE` option on the target standby.

- You want to conduct a manual failover to any standby database in the configuration (for example, because a failure occurred on the primary database at a time when the primary and target standby database were not ready to failover).

In this case fast-start failover cannot occur because the databases are not ready to failover. You cannot perform a manual failover to the target standby database for the same reason. To proceed, you must first disable fast-start failover using the `FORCE` option, and then perform a manual failover.

Caution: This action will result in loss of data and the possibility of two databases in the configuration simultaneously assuming the primary database role. This can be avoided by first disabling fast-start failover with the `FORCE` option on the target standby.

- A fast-start failover to the target standby database fails.
If the failover fails for any reason, it could leave the target standby database inoperable, regardless of whether the target standby database is ready to failover. If there is another standby database that is available for failover, you can perform a manual failover to that standby database after you first disable fast-start failover using the `FORCE` option on that standby database.
- You want to prevent fast-start failover from occurring because the primary database will resume service soon.

In this case, disable fast-start failover using the `FORCE` option on the target standby database. Once the primary database regains connectivity with the target standby database, fast-start failover will be disabled for all the databases in the configuration.

Disabling Fast-Start Failover Using Cloud Control

Click **Disable** in the Fast-Start Failover wizard. Then, click **Continue** to proceed to the next page. See the Cloud Control online help for more information.

Disabling Fast-Start Failover Using DGMGRL

Issue the `DISABLE FAST_START FAILOVER` command or the `DISABLE FAST_START FAILOVER FORCE` command. See the "[DISABLE FAST_START FAILOVER](#)" command in [Chapter 7](#) for more information.

5.5.6 Performance Considerations for Fast-Start Failover

Consider the following recommendations to obtain better performance when using fast-start failover:

- The failover time is dependent upon whether the target standby database (physical or logical standby database) has applied all of the redo data it has received from the primary database.
- Enabling fast-start failover in a configuration operating in maximum performance mode provides better overall performance on the primary database because redo data is sent asynchronously to the target standby database. Note that this does not guarantee no data will be lost.
- Fast-start failover is faster when you take steps to optimize recovery so that the application of redo data to the standby database is kept up to date with the primary database's rate of redo application. To optimize the log apply rate:

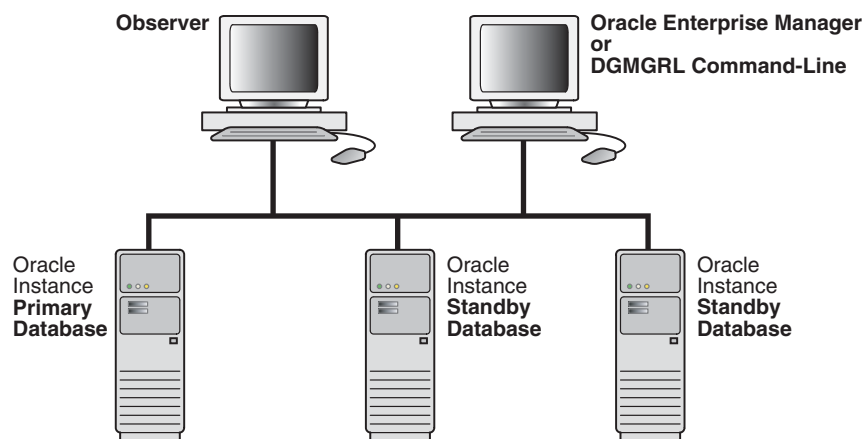
- Do not configure the `DelayMins` database property to delay applying archived redo log files to the standby database (see [Section 4.5](#) for more information).
- See *Oracle Data Guard Concepts and Administration* for information about tuning the log apply rate for a physical standby database.
- See the *Oracle Maximum Availability Architecture* white papers at: <http://www.oracle.com/goto/maa>
- When setting the `FastStartFailoverLagLimit` configuration property, consider these tradeoffs between performance and potential data-loss:
 - A low lag limit will minimize data loss but may impact the performance of the primary database.
 - A high lag limit may lead to more data loss but may lessen the performance impact of the primary database.

5.5.7 Managing the Observer

The observer is integrated in the DGMGRL client-side component of the broker and typically runs on a different computer from the primary or standby databases and from the computer where you manage the broker configuration. The observer continuously monitors the fast-start failover environment to ensure the primary database is available (described in [Section 5.5.2.1](#)). The observer's main purpose is to enhance high availability and *lights out* computing by reducing the human intervention required by the manual failover process that can add minutes or hours to downtime.

You can manage the observer through either the Oracle Data Guard Overview pages in Cloud Control or using DGMGRL commands. [Figure 5–2](#) shows the observer monitoring a fast-start failover configuration.

Figure 5–2 The Observer in the Fast-Start Failover Environment



The following sections provide information about managing the observer:

- [Installing and Starting the Observer](#)
- [Viewing Information About the Observer](#)
- [What Happens if the Observer Fails?](#)
- [Stopping the Observer](#)

- [Moving the Observer to Another Computer](#)
- [How the Observer Maintains Fast-Start Failover Configuration Information](#)
- [Patching an Environment When the Observer Is Running and Fast-start Failover Is Enabled](#)

5.5.7.1 Installing and Starting the Observer

The observer should be installed and run on a computer system that is separate from the primary and standby systems. Installing and starting the observer is an integral part of using fast-start failover and is described in detail in these sections:

- [Section 2.1](#) explains that you can either install only the Oracle Client Administrator or you can install the complete Oracle Database Enterprise Edition or Personal Edition on the observer system.
- [Section 5.5.2](#) describes how to start the observer as a part of the step-by-step process to enable fast-start failover. Examples of starting the observer using DGMGRL are included in [Section 6.7](#).

There can be only one observer monitoring the broker configuration. If you attempt to start another one, the broker returns the following error message:

```
ORA-16647: could not start more than one observer
```

To start the observer, you must be able to log in to DGMGRL with an account that has the SYSDBG or SYSDBA privilege. The observer is an OCI client that connects to the primary and target standby databases using the same SYS credentials you used when you connected to the Oracle Data Guard configuration with DGMGRL.

See Also:

- The My Oracle Support note 1625597.1 at <http://support.oracle.com> for information about compatibility requirements between the observer and DGMGRL

Starting Multiple Observers On a Single Host

If you want to use one Oracle home to start multiple observers, with each observer monitoring a different fast-start failover configuration, use the FILE qualifier to specify a unique observer configuration file location for each configuration to be monitored. If you want to capture any logging generated by the observer, use the LOGFILE option and ensure that file name is unique as well. For example:

```
% dgmgrl -logfile $ORACLE_HOME/rdbms/log/config1.log
DGMGRL> CONNECT /@primary1;
DGMGRL> START OBSERVER FILE=$ORACLE_HOME/dbs/config1.dat;

% dgmgrl -logfile $ORACLE_HOME/rdbms/log/config2.log
DGMGRL> CONNECT /@primary2;
DGMGRL> START OBSERVER FILE=$ORACLE_HOME/dbs/config2.dat;
```

5.5.7.2 Viewing Information About the Observer

You can find information about the observer by querying the following columns in the V\$DATABASE view:

- FS_FAILOVER_OBSERVER_HOST shows the name of the computer on which the observer is running

- FS_FAILOVER_OBSERVER_PRESENT shows whether or not the observer is connected to the local database

Table 5–2 FS_FAILOVER_OBSERVER_PRESENT Column of the V\$DATABASE View

Column Value ¹	Description
YES	Observer is currently connected to the local database
NO	Observer is not connected to the local database

¹ This value is consistent across instances in an Oracle Real Applications Clusters (Oracle RAC) environment. That is, if the observer is connected to any instance in the Oracle RAC, all instances will show a value of YES.

For example, to determine if fast-start failover can occur, the FS_FAILOVER_STATUS column displays either SYNCHRONIZED or TARGET UNDER LAG LIMIT and the FS_FAILOVER_OBSERVER_PRESENT column displays YES for the target standby database. For example:

Database	FS_FAILOVER_STATUS	Protection Mode	FS_FAILOVER_OBSERVER_PRESENT
Primary	SYNCHRONIZED	Maximum Availability	YES
Standby	SYNCHRONIZED	Maximum Availability	YES
Primary	TARGET UNDER LAG LIMIT	Maximum Performance	YES
Standby	TARGET UNDER LAG LIMIT	Maximum Performance	YES

In the following example, assume the network between the primary database and the observer has failed. In this case, the FS_FAILOVER_STATUS and FS_FAILOVER_OBSERVER_PRESENT columns will appear as shown in the following table and fast-start failover will not occur:

Database	FS_FAILOVER_STATUS	FS_FAILOVER_OBSERVER_PRESENT
Primary	SYNCHRONIZED	NO
Standby	PRIMARY UNOBSERVED	YES

5.5.7.3 What Happens if the Observer Fails?

If the primary and target standby databases stay connected but the connection to the observer is lost, then the broker reports that the configuration is not observed. The configuration and database status report that the observer is not running and return one of the following status messages:

```
ORA-16658: unobserved fast-start failover configuration
ORA-16820: fast-start failover observer is no longer observing this database
```

While the configuration is in the unobserved state, fast-start failover cannot happen. Therefore, the primary database can continue processing transactions, even if the target standby database fails. The configuration status returns the SUCCESS status after the observer reestablishes its connection to the primary database, which then notifies the target standby database.

See Also: [Section 5.5.4, "Viewing Fast-Start Failover Configuration Statistics and Status"](#)

5.5.7.4 Managing Observer's Connection to the Primary

The `ObserverOverride` and `ObserverReconnect` properties allow you additional control over the connection to the primary.

When the observer loses its connection to the primary database for a period of time greater than that specified by the `FastStartFailoverThreshold` property, it attempts a failover to the standby database. However, if the standby has had contact from the primary within the period of time specified by the `FastStartFailoverThreshold` property, the standby prevents the failover attempt.

To override this behavior and allow a fast-start failover to occur if the observer is unable to contact the primary for more than `FastStartFailoverThreshold` seconds, set the `ObserverOverride` property to `TRUE`. For example:

```
DGMGRL> EDIT CONFIGURATION SET PROPERTY ObserverOverride=TRUE;
```

Ordinarily the observer connects once to the primary and does not attempt to reconnect unless the connection has failed. However, if you want the observer to reconnect to the primary database periodically as a means of testing the health of the network connection to the primary, then use the `ObserverReconnect` configuration property. This specifies how often the observer establishes a new connection to the primary database. In the following example, `ObserverReconnect` is set to 30 seconds. This results in the observer establishing a new connection to the primary database every 30 seconds.

```
DGMGRL> EDIT CONFIGURATION SET PROPERTY ObserverReconnect=30;
```

5.5.7.5 Stopping the Observer

You may want to stop the observer when you no longer want to use fast-start failover (see [Section 5.5.5, "Disabling Fast-Start Failover"](#)) or if you want to move the observer to a different host machine (see [Section 5.5.7.6, "Moving the Observer to Another Computer"](#)).

To stop the observer when fast-start failover *is enabled*, the primary database and target standby database must be connected and communicating with each other. Stopping the observer does not disable fast-start failover. However, fast-start failover cannot occur when the target standby database is in the unobserved state.

To stop the observer when fast-start failover *is enabled*, but the primary and standby are isolated from each other, you must first disable fast-start failover by using the `FORCE` option, and then stop the observer. (See [Section 5.5.5, "Disabling Fast-Start Failover"](#) for important considerations when using the `FORCE` option.)

To stop the observer when fast-start failover *is not enabled*, the primary database must be running.

You can stop the observer while connected to any database in the broker configuration that has network connectivity to the primary database, as follows:

- Using Cloud Control
 - Choose the **Stop Observer** option on the first page of the fast-start failover wizard and click **Continue** at the bottom of the page. See the Cloud Control online help for more information.
- Using DGMGRL

Issue the following command:

```
DGMGRL> STOP OBSERVER;
```

See the [STOP OBSERVER](#) command on page 7-77 for more information.

Note: The observer does not stop immediately when you issue the `STOP OBSERVER` command. After the broker receives the `STOP OBSERVER` request, the request is passed to the observer the next time the observer contacts the broker, and the observer then stops itself.

5.5.7.6 Moving the Observer to Another Computer

To move the observer to another computer:

1. Stop the observer from any computer system in the broker configuration, as described in [Section 5.5.7.5](#).
2. Start the observer on the new computer system, as described in Step 8 of [Section 5.5.2](#).

There is no need to disable fast-start failover when you move the observer.

5.5.7.7 How the Observer Maintains Fast-Start Failover Configuration Information

The observer persistently maintains information about the fast-start failover configuration in a binary file created in the working directory where you started the observer. By default, the observer creates this file in the current working directory when it is started and names the file `fsfo.dat`. This file contains connect identifiers to both the primary and the target standby databases.

Ensure this file cannot be read by unauthorized users.

Once the observer is started, you cannot change the file's name and location. However, you can change the name or the location of the file if you start the observer using the `DGMGRL START OBSERVER` command and include the `FILE` qualifier. See the [START OBSERVER](#) command on page 7-72 for more information.

Note: If the observer is stopped abnormally (for example, by typing `CTRL/C`), restart it and reference the existing `fsfo.dat` file with the `FILE` qualifier.

5.5.7.8 Patching an Environment When the Observer Is Running and Fast-start Failover Is Enabled

To patch an environment where the Observer is running and fast-start failover is enabled, follow these steps prior to applying the patch:

1. Stop the observer using the `DGMGRL STOP OBSERVER` command. The primary and target standby must have connectivity for this command to complete successfully.
2. Disable fast-start failover using the `DGMGRL DISABLE FAST_START FAILOVER` command. Note the primary and target standby must have connectivity for this command to complete successfully.

After the patch has been successfully applied to all databases, take the following steps to enable fast-start failover and start the observer.

1. Enable fast-start failover using the DGMGRL `ENABLE FAST_START FAILOVER` command. The primary and target standby must have connectivity for this command to complete successfully.
2. Start the observer using the DGMGRL `START OBSERVER` command.

5.5.8 Reinstating the Former Primary Database in the Broker Configuration

If a fast-start failover was initiated because the primary database had crashed or lost connectivity with the observer and target standby database, the observer automatically attempts to *reinst*ate the former primary database as a standby database, if the `FastStartFailoverAutoReinst` configuration property is set to `TRUE`.

Reinstatement restores high availability to the broker configuration so that, in the event of a failure of the new primary database, another fast-start failover can occur. The reinstated database acts as the fast-start failover target for the new primary database, making a subsequent fast-start failover possible. The new standby database is a viable target of a failover when it begins receiving redo data received from the new primary database.

To allow the observer to automatically reinstate the former primary database, the database must be started and mounted, but it cannot be opened. The observer will restart the former primary database to the mounted state if it is open, prior to reinstating the database. The broker reinstates the database as a standby database of the same type as the former standby database of the new primary database.

If the former primary database cannot be reinstated automatically, you can manually reinstate it using either the DGMGRL `REINSTATE` command or Cloud Control. Step-by-step instructions for manual reinstatement are described in [Section 5.4.3](#).

5.5.8.1 Requirements

Reinstatement is supported only after failover in a broker configuration. It also requires Flashback Database to be enabled on both the primary and target standby databases. [Section 5.5.1](#) provides complete information about all of the fast-start failover and reinstatement requirements.

5.5.8.2 Restrictions on Reinstatement

The broker cannot automatically reinstate the former primary database if:

- A fast-start failover occurred because a user-configurable condition was detected or was requested by an application by calling the `DBMS_DG.INITIATE_FS_FAILOVER` function.
- `FastStartFailoverAutoReinst` is set to `FALSE`
- Another failover or switchover occurred *after* the fast-start failover completed but *before* the former primary database restarted
- Fast-start failover was disabled
- The observer cannot connect to the former primary database
- The former primary database cannot connect to the new primary database
- The former primary database and the new primary database are not configured in the same fast-start failover environment
- The former primary database was disabled because of a manual failover when fast-start failover was disabled

Note: Standby databases that are disabled during switchover, manual failover, or fast-start failover will not be automatically reinstated.

If automatic reinstatement fails, the broker will log errors and the former primary database will remain in the mounted state. At this point, you can either:

- Disable fast-start failover (described in [Section 5.5.5](#)) and attempt to open the former primary database
- Manually reinstate the former primary database, as described in [Section 5.4.3](#)

5.5.8.3 How the Broker Handles a Failed Reinstatement

If a failure occurs once a reinstatement operation (automatic or manual) is underway, the broker logs the appropriate information in the broker configuration files and broker log files. The former primary database is disabled. Most in-progress failures cannot be restarted (for example, archived redo log file corruption on the primary database). You must manually re-create the database as a standby database and then reenale it.

See Also: [Section 5.4.3.2, "How to Re-create and Reenable a Disabled Database"](#)

5.5.9 Shutting Down Databases In a Fast-Start Failover Environment

Perform the following steps if you need to shut down the primary or target standby database:

1. Stop the observer and wait for the `FS_FAILOVER_OBSERVER_PRESENT` column in the `V$DATABASE` fixed view to contain the value "NO" for both the primary and target standby databases. This ensures that a fast-start failover will not occur while you are shutting down the primary database.
2. Shut down the primary database and the target standby database using either `DGMGRL SHUTDOWN` command or the SQL*Plus `SHUTDOWN` statement.

When restarting the databases, you may restart them in any order. When both databases have been restarted, you may restart the observer.

Bystander standby databases can be shut down at any time in any order without impacting fast-start failover.

5.6 Database Client Considerations

This section describes the event notification and database connection failover support that is available to database clients connected to local database services when a broker-managed failover occurs. For information about event notification and database connection failover support for global services, see the *Oracle Database Global Data Services Concepts and Administration Guide*.

After a failover, the broker publishes Fast Application Notification (FAN) events. These FAN events can be used in the following ways:

- Applications can use FAN without programmatic changes if they use one of these Oracle integrated database clients: Oracle Database JDBC, Oracle Database Oracle Call Interface (OCI), Oracle Data Provider for .NET (ODP.NET), or Universal

Connection Pool for Java. These clients can be configured for Fast Connection Failover (FCF) to automatically connect to a new primary database after a failover.

- JAVA applications can use FAN programmatically by using the JDBC FAN application programming interface to subscribe to FAN events and to execute event handling actions upon the receipt of an event.
- FAN server-side callouts can be configured on the database tier.

FAN events are published using Oracle Notification Services (ONS) for all Oracle integrated database clients in Oracle Database 12c and later. In previous releases, OCI and ODP.NET clients receive FAN notifications via Oracle Advanced Queuing (AQ).

Note: A single-instance database must be registered with Oracle Restart in order to publish FAN events via ONS.

See Also:

- *Oracle Real Application Clusters Administration and Deployment Guide* for more information about configuring FAN, FCF, and ONS on an Oracle Real Application Clusters (Oracle RAC) database
- *Oracle Database Administrator's Guide* for information about configuring Oracle Restart, FAN, FCF, and ONS on a single-instance database

5.6.1 Oracle Data Guard Specific FAN and FCF Configuration Requirements

This section describes configuration requirements that must be met in order to publish and properly handle FAN events generated as the result of a broker-managed failover.

These requirements are supplemental to those described in the documents previously referenced and in the following client-specific guides:

- *Oracle Database JDBC Developer's Guide*
- *Oracle Call Interface Programmer's Guide*
- *Oracle Data Provider for .NET Developer's Guide for Microsoft Windows*

5.6.1.1 Oracle Net Configuration Requirements

For FCF to occur, a client must be able to locate the new primary database after a failover. This section describes how to configure an Oracle Net connect descriptor that meets this requirement.

The connect descriptor can be configured in one of two ways:

1. Configure the connect descriptor for connect-time failover. Add the primary database and each standby database to the address list. Set the `CONNECT_TIMEOUT` parameter to a small value to minimize the delay experienced if a network address is not available. Increase the value of this parameter if resource contention causes connection timeouts to occur during normal operation.

For example:

```
sales =
  (DESCRIPTION=
    (FAILOVER=ON)
    (CONNECT_TIMEOUT=5)
    (ADDRESS_LIST=
```

```
(ADDRESS= (HOST=boston-scan) (PORT=1521))  
(ADDRESS= (HOST=dallas-scan) (PORT=1521))  
(CONNECT_DATA= (SERVICE_NAME=sales))
```

2. Configure the connect descriptor with a single network name that is registered with a global naming service such as DNS or LDAP. Create a trigger based on the `DB_ROLE_CHANGE` system event that changes the network address associated with the network name to the network address of the new primary database after a failover.

See Also:

- *Oracle Database PL/SQL Language Reference* for more information about the `DB_ROLE_CHANGE` system event

The connect descriptor must contain the `SERVICE_NAME` parameter in either case.

5.6.1.2 Database Service Configuration Requirements

Note: The examples shown in this section do not necessarily show the specific attributes you might need to use in your own environment. The required attributes vary depending on your configuration (including whether your environment is Oracle RAC-based or single-instance). Refer to the appropriate Oracle RAC or Oracle Restart documentation for further information.

Database services can be configured to be active in specific database roles on Oracle RAC databases and on single-instance databases managed by Oracle Restart. The broker interacts with Oracle Clusterware or Oracle Restart to ensure that the appropriate database services are active and that the appropriate FAN events are published after a role change.

FAN events are always published through ONS. However, the event notifying a failover is only published for database services that have been configured to be active while the database is in the primary role on the new primary database.

Services that must be active in any given database role (primary, physical standby, logical standby, or snapshot standby) must be configured with the `SRVCTL` utility explicitly on each database where the service must be active. In the following example commands, a service named `PAYROLL` is configured to be active in the `PRIMARY` role on the primary database `NORTH`. The service is then configured to be active in the `PRIMARY` role on the standby database `SOUTH`, so that it will be active on that database after a role transition. In these sample commands, the ellipse (...) signifies any other `add service` options you wish to supply.

On primary database `NORTH`, execute the following:

```
srvctl add service -db NORTH -service PAYROLL -role PRIMARY ...
```

On standby database `SOUTH`, execute the following:

```
srvctl add service -db SOUTH -service PAYROLL -role PRIMARY ...
```

Services that are to be active while the database is in the physical standby role must also be created and started on the current primary database regardless of whether the service will be started on that database or not. This is to ensure that the service definition gets propagated to the physical standby database via the redo stream and

thus allows for the service to be started on the physical standby database. The service can be started on the physical standby only after the redo generated by starting the service has been applied. It is important that all SRVCTL `add service` options be identical on all the databases so that the services behave the same way before and after a role change.

If all the databases do not have the same values, SRVCTL attempts to override the values, which will fail on the physical standby database because it is open read-only. In the following example, a service named `sales` is configured to be active in the `PHYSICAL_STANDBY` role on the primary database `NORTH`. It is then started and stopped on the primary database. It could optionally also be removed from the primary database if there is no intention to ever run this service on the current primary database. It is then configured to be active in the `PHYSICAL_STANDBY` role on the physical standby database `SOUTH`.

Execute the following on primary database `NORTH`:

```
srvctl add service -dd NORTH -service sales -role PHYSICAL_STANDBY ...

srvctl start service -db NORTH -service sales

srvctl stop service -db NORTH -service sales
```

Execute the following on the physical standby database `SOUTH`:

```
srvctl add service -dd SOUTH -service sales -role PHYSICAL_STANDBY ...
```

Note: If the service has been configured to start automatically (`-policy AUTOMATIC`), the service will automatically start only after a database role change.

Note: In an Oracle Data Guard configuration, the SRVCTL `-startoption` for a standby database is always set to `OPEN` after a switchover.

See Also:

- For single-instance databases managed by Oracle Restart, see *Oracle Database Administrator's Guide* for more information about configuring database services with the SRVCTL utility
- For Oracle RAC databases, see *Oracle Real Application Clusters Administration and Deployment Guide* for more information about configuring database services with the SRVCTL utility

5.6.1.3 ONS Configuration Requirements

If client-side ONS configuration is used, the client-side ONS configuration file must specify the hostname and port of the ONS daemon(s) of the primary database and each standby database.

If the client uses remote ONS subscription, the client must specify the hostname and port of the ONS daemon(s) of the primary database and each standby database.

5.6.1.4 Application Continuity

Application Continuity is an Oracle Database feature that enables rapid and nondisruptive replays of requests against the database after a recoverable error that made the database session unavailable.

Application Continuity is supported for Oracle Data Guard switchovers to physical standby databases. It is also supported for fast-start failover to physical standbys in maximum availability data protection mode. Note that primary and standby databases must be licensed for Oracle RAC or Oracle Active Data Guard in order to use Application Continuity.

See Also:

- *Oracle Database Development Guide* for information about Application Continuity

Scenarios Using the DGMGRL Command-Line Interface

This chapter describes the prerequisites for getting started using the Oracle Data Guard command-line interface (DGMGRL). It also describes scenarios that demonstrate how to use DGMGRL to create, manage, and monitor a broker configuration.

This chapter contains the following sections:

- [Prerequisites for Getting Started](#)
- [Scenario 1: Creating a Configuration](#)
- [Scenario 2: Setting Database Properties](#)
- [Scenario 3: Enabling the Configuration and Databases](#)
- [Scenario 4: Setting the Configuration Protection Mode](#)
- [Scenario 5: Setting up Maximum Availability Mode with a Far Sync Instance](#)
- [Scenario 6: Enabling Fast-Start Failover and Starting the Observer](#)
- [Scenario 7: Enabling Fast-Start Failover When a Far Sync Instance Is In Use](#)
- [Scenario 8: Performing Routine Management Tasks](#)
- [Scenario 9: Performing a Switchover Operation](#)
- [Scenario 10: Performing a Manual Failover Operation](#)
- [Scenario 11: Reinstating a Failed Primary Database](#)
- [Scenario 12: Converting a Physical Standby to a Snapshot Standby](#)
- [Scenario 13: Monitoring a Data Guard Configuration](#)

6.1 Prerequisites for Getting Started

One of the prerequisites for using DGMGRL is that a primary database and any standby databases must already exist. The `DG_BROKER_START` initialization parameter must be set to `TRUE` for all databases in the configuration. You must use a server parameter file with the broker (see [Section 2.1](#) and [Section 7.1.3](#)).

Convert the initialization parameter files (PFILE) on both primary and standby databases into server parameter files (SPFILE), if necessary. Use the following SQL*Plus command:

```
CREATE SPFILE='spfilename' FROM PFILE='pfilename';
```

If an instance was not started with a server parameter file, then you must shut down the instance and restart it using the server parameter file.

After starting the Oracle instance, set the `DG_BROKER_START=TRUE` initialization parameter using the `SQL ALTER SYSTEM` statement. The parameter value will be saved in the server parameter file. The next time you start the Oracle instance, the broker is started automatically, and you do not need to issue the `SQL ALTER SYSTEM` statement again.

See Also: *Oracle Database Administrator's Guide* for detailed information about creating server parameter files

The following assumptions are made for the scenarios in this chapter:

- TCP/IP is used to connect to primary and standby databases.
- The standby database has been created from backups of the primary database control files and datafiles as described in the *Oracle Data Guard Concepts and Administration*.
- The scenarios in this chapter assume the following broker configuration:
 - The configuration name is `DRSolution`.
 - The database unique name (`DB_UNIQUE_NAME`) of the primary database is `North_Sales`.
 - The database unique name (`DB_UNIQUE_NAME`) of the remote standby database is `South_Sales`.
 - The protection mode is maximum performance mode.
 - There are standby redo log files configured for both the primary and standby database. The transport mode for both databases is `ASYNC`.
 - The standby database is a physical standby database.

6.2 Scenario 1: Creating a Configuration

This section provides examples that create a broker configuration named `DRSolution` that includes a primary and standby database named `North_Sales` and `South_Sales`.

The following steps show how to create a configuration and add one physical standby database.

Step 1 Invoke DGMGRL.

To start DGMGRL, enter `dgmgrl` at the command-line prompt on a system where Oracle Data Guard is installed:

```
$ dgmgrl
```

The DGMGRL prompt is displayed:

```
DGMGRL>
```

Step 2 Connect to the primary database.

Before you specify any command (other than the `HELP`, `EXIT`, or `QUIT`), you must first connect to the primary database using the DGMGRL `CONNECT` command.

The account from which you connect to the database (*SYS* in this example) must have *SYSDG* or *SYSDBA* privileges on the primary and standby databases.

Note: You do not have to include `AS SYSDBA` on the `CONNECT` command; it is the default setting.

The following examples show two variations of the `CONNECT` command. [Example 6–1](#) shows how to connect to the default database on the local system, and [Example 6–2](#) includes the Oracle Net Services connect identifier (`North_Sales.example.com`) to make a connection to a database located on a remote system. In both examples, you are prompted for a password.

Example 6–1 Connecting to the Primary Database on the Local System

```
DGMGRL> CONNECT sysdg;
Password: password
Connected as SYSDG.
```

Example 6–2 Connecting to the Primary Database on a Remote System

```
DGMGRL> CONNECT sysdg@North_Sales.example.com;
Password: password
Connected as SYSDG.
```

Step 3 Clear existing remote redo transport destinations.

You must clear any remote redo transport destinations on the primary database that do not have the `NOREGISTER` attribute, before a configuration can be created. Otherwise, the following error message is returned when you attempt to create the configuration:

```
ORA-16698: LOG_ARCHIVE_DEST_n parameter set for object to be added
```

Failed.

To clear `LOG_ARCHIVE_DEST_n` settings, use the `ALTER SYSTEM SET LOG_ARCHIVE_DEST_n=" " SQL*`Plus command.

Step 4 Create the broker configuration.

A broker configuration is initially created with just a primary database, which in this case is called `North_Sales`. In a later command, you will add the standby database, `South_Sales`.

Note: The names for the primary and standby databases must match their database unique names. Fetch these from their `DB_UNIQUE_NAME` initialization parameter as follows:

```
SQL> SHOW PARAMETER DB_UNIQUE_NAME;
```

Use the `CREATE CONFIGURATION` command to create the `DRSolution` configuration and define the primary database, `North_Sales`:

```
DGMGRL> CREATE CONFIGURATION 'DRSolution' AS
> PRIMARY DATABASE IS 'North_Sales'
> CONNECT IDENTIFIER IS North_Sales.example.com;
```

DGMGRL returns the following information:

Configuration "DRSolution" created with primary database "North_Sales"

The name `North_Sales` is the value of this database's `DB_UNIQUE_NAME` initialization parameter.

Step 5 Show the configuration information.

Use the `SHOW CONFIGURATION` command to display a brief summary of the configuration:

```
DGMGRL> SHOW CONFIGURATION;
```

DGMGRL returns the following information:

```
Configuration - DRSolution

Protection Mode: MaxPerformance
Databases:
  North_Sales - Primary database

Fast-Start Failover: DISABLED

Configuration Status:
DISABLED
```

Step 6 Add a standby database to the configuration.

To add a standby database to the `DRSolution` configuration, use the `ADD DATABASE` command.

The following command defines `South_Sales` as a standby database, which is the standby database associated with the primary database called `North_Sales`:

```
DGMGRL> ADD DATABASE 'South_Sales' AS
> CONNECT IDENTIFIER IS South_Sales.example.com;
```

DGMGRL returns the following information:

```
Database "South_Sales" added
```

The name `South_Sales` is the value of the database's `DB_UNIQUE_NAME` initialization parameter.

Use the `SHOW CONFIGURATION` command to verify that the `South_Sales` database was added to the `DRSolution` configuration:

```
DGMGRL> SHOW CONFIGURATION;
```

DGMGRL returns the following information:

```
Configuration - DRSolution

Protection Mode: MaxPerformance
Databases:
  North_Sales - Primary database
  South_Sales - Physical standby database

Fast-Start Failover: DISABLED

Configuration Status:
DISABLED

DGMGRL>
```

6.3 Scenario 2: Setting Database Properties

After you create the configuration with DGMGRL, you can set database properties at any time. For example, the following statements set the `LogArchiveFormat` and `StandbyArchiveLocation` database properties for the `South_Sales` standby database:

```
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY 'LogArchiveFormat'='log_%t_%s_%r_%d.arc';
Property "LogArchiveFormat" updated.
```

```
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY 'StandbyArchiveLocation'='/archfs/arch/';
Property "StandbyArchiveLocation" updated.
```

Use the `SHOW DATABASE VERBOSE` command to view all properties and their values for a database. The following example shows the properties for the `South_Sales` database.

```
DGMGRL> SHOW DATABASE VERBOSE 'South_Sales';
```

```
Database - South_Sales
```

```
Role:                PHYSICAL STANDBY
Intended State:      OFFLINE
Transport Lag:       (unknown)
Apply Lag:           (unknown)
Apply Rate:          (unknown)
Real Time Query:    OFF
Instance(s):
  south_sales1
```

```
Properties:
```

```
DGConnectIdentifier          = 'South_Sales.example.com'
ObserverConnectIdentifier    = ''
LogXptMode                   = 'ASYNC'
RedoRoutes                   = ''
DelayMins                    = '0'
Binding                      = 'optional'
MaxFailure                   = '0'
MaxConnections               = '1'
ReopenSecs                   = '300'
NetTimeout                   = '30'
RedoCompression              = 'DISABLE'
LogShipping                   = 'ON'
PreferredApplyInstance       = ''
ApplyInstanceTimeout         = '0'
ApplyLagThreshold            = '0'
TransportLagThreshold        = '0'
TransportDisconnectedThreshold = '0'
ApplyParallel                 = 'AUTO'
StandbyFileManagement        = 'AUTO'
ArchiveLagTarget              = '0'
LogArchiveMaxProcesses       = '5'
LogArchiveMinSucceedDest     = '1'
DbFileNameConvert            = 'dbs/t, dbs/ct, dbs/bt, dbs/ct'
LogFileNameConvert           = 'dbs/t, dbs/ct, dbs/bt, dbs/ct'
FastStartFailoverTarget      = ''
InconsistentProperties        = '(monitor)'
InconsistentLogXptProps      = '(monitor)'
SendQEntries                  = '(monitor)'
LogXptStatus                  = '(monitor)'
RecvQEntries                  = '(monitor)'
StaticConnectIdentifier       = ''
'(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=south_sales1.example.com)(PORT=2840))
```

```
(CONNECT_DATA=(SERVICE_NAME=South_Sales_DGMGRL.example.com)(INSTANCE_NAME=south_
sales1)(SERVER=DEDICATED)))'
StandbyArchiveLocation      = 'USE_DB_RECOVERY_FILE_DEST'
AlternateLocation           = ''
LogArchiveTrace             = '255'
LogArchiveFormat            = 'db3r_%d_%t_%s_%R.arc'
TopWaitEvents               = '(monitor)'
```

```
Database Status:
DISABLED
```

If broker management of the database is enabled, setting a database property value causes the underlying parameter value to be changed in the corresponding database, and the value for the changed parameter is reflected in the server parameter file. Thus, if the database is shut down and restarted outside of Oracle Enterprise Manager Cloud Control (Cloud Control) and DGMGRL (such as from the SQL*Plus interface), the database uses the new parameter values from the updated server parameter file when it starts. However, you should not make changes to the redo transport services initialization parameters through SQL statements. Doing so will cause an inconsistency between the database and the broker.

Note: The database properties are typically displayed in mixed-case (for example, `LogArchiveFormat`) typeface to help you visually differentiate database properties (from the corresponding initialization parameter, SQL statement, or PL/SQL procedure), which are typically documented in UPPERCASE typeface. However, the commands to manage properties are not case sensitive; you can issue commands in uppercase, lowercase, or mixed-case.

You can change a property if the database is enabled or disabled. However, if the database is disabled when you change a property, the change does not take effect until the database is enabled.

6.4 Scenario 3: Enabling the Configuration and Databases

So far, the `DRSolution` configuration is disabled, which means it is not under the control of the Data Guard broker. When you finish configuring the databases into a broker configuration and setting any necessary database properties, you must enable the configuration to allow the Data Guard broker to manage it.

You can enable:

- The entire configuration, including all of its databases
- A standby database

Enable the entire configuration.

You can enable the entire configuration, including all of the databases, with the following command:

```
DGMGRL> ENABLE CONFIGURATION;
Enabled.
```

Show the configuration.

Use the `SHOW` command to verify that the configuration and its databases were successfully enabled:

```
DGMGRL> SHOW CONFIGURATION;
```

DGMGRL returns the following information:

```
Configuration - DRSolution
```

```
Protection Mode: MaxPerformance
Databases:
  North_Sales - Primary database
  South_Sales - Physical standby database
```

```
Fast-Start Failover: DISABLED
```

```
Configuration Status:
SUCCESS
```

Enable the database.

This step is unnecessary except if the standby database was previously disabled with the `DISABLE DATABASE` command. Normally, enabling the configuration also enables the standby database.

```
DGMGRL> ENABLE DATABASE 'South_Sales';
Enabled.
```

Show the database.

```
DGMGRL> SHOW DATABASE 'South_Sales';
```

```
Database - South_Sales
```

```
Role: PHYSICAL STANDBY
Intended State: APPLY-ON
Transport Lag: 0 seconds (computed 1 second ago)
Apply Lag: 0 seconds (computed 1 second ago)
Apply Rate: 1.54 MByte/s
Real Time Query: OFF
Instance(s):
  south_sales1
```

```
Database Status:
SUCCESS
```

6.5 Scenario 4: Setting the Configuration Protection Mode

You can change the protection mode of the configuration at any time. However, it is best if you do this when there is no activity occurring in the configuration if you are moving to the maximum protection or maximum availability modes.

Note: If you change the protection mode from maximum performance mode to maximum protection mode, the broker automatically restarts the primary database. If you wish to avoid restarting the database, first change the protection mode to maximum availability mode and then change the protection mode to maximum protection mode.

A restart of the primary database is not required when changing the protection mode from:

- maximum performance to maximum availability
 - maximum availability to maximum protection
-

This scenario sets the protection mode of the configuration to the `MAXAVAILABILITY` mode. Note that this protection mode requires that there be at least one standby configured to use standby redo log files and it must receive redo via `SYNC` or `FASTSYNC` mode if it receives redo directly from the primary database. If the standby receives redo via a far sync instance, then the far sync instance must receive redo via `SYNC` or `FASTSYNC` mode and the standby must receive redo from the far sync instance via `ASYN` mode.

Step 1 Configure standby redo log files, if necessary

Because you will be setting the protection mode to the `MAXAVAILABILITY` mode, it is important to ensure that sufficient standby redo log files are configured on the standby database.

Step 2 Configure redo transport mode appropriately.

Configure the standby to receive redo via `SYNC` or `FASTSYNC` mode, if the standby receives redo directly from the primary database. If the standby receives primary redo via a far sync instance, then configure the far sync instance to receive redo via `SYNC` or `FASTSYNC` mode and configure the standby to receive redo via `ASYN` mode. For example:

```
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY 'LogXptMode'='SYNC';  
Property "LogXptMode" updated
```

The broker will not allow this command to succeed unless the standby database is configured with standby redo log files in the configuration.

Step 3 Change the overall protection mode for the configuration.

Use the `EDIT CONFIGURATION` command to upgrade the broker configuration to the `MAXAVAILABILITY` protection mode:

```
DGMGRL> EDIT CONFIGURATION SET PROTECTION MODE AS MAXAVAILABILITY;  
Succeeded.
```

If the configuration is disabled when you enter this command, the actual protection mode change is not applied until you enable the configuration with the `ENABLE CONFIGURATION` command. The broker will not allow you to enable the configuration if it does not find a standby database in the configuration that can support the requirements of the protection mode.

Step 4 Verify the protection mode has changed.

Use the `SHOW CONFIGURATION` command to display the current protection mode for the configuration:

```
DGMGRL> SHOW CONFIGURATION;

Configuration - DRSolution

Protection Mode: MaxAvailability
Databases:
  North_Sales - Primary database
  South_Sales - Physical standby database

Fast-Start Failover: DISABLED

Configuration Status:
SUCCESS
```

See Also: [Section 4.6, "Managing Data Protection Modes"](#) on page 4-20

6.6 Scenario 5: Setting up Maximum Availability Mode with a Far Sync Instance

A far sync instance can be used with maximum availability protection mode if the primary and standby database are geographically far enough apart to make the use of synchronous transport mode impractical. The example in this section shows how to add a far sync instance to the configuration and then set up the `RedoRoutes` property for all members of the configuration. Setting of `RedoRoutes` property for the far sync instance enables it to send redo data based on either the `North_Sales` or `South_Sales` database being the primary.

Step 1 Add the far sync instance to the configuration

Issue the following commands to add the far sync instance named `FS` to the broker configuration:

```
DGMGRL> ADD FAR_SYNC 'FS' AS CONNECT IDENTIFIER IS FS.EXAMPLE.COM;
DGMGRL> ENABLE FAR_SYNC 'FS';
DGMGRL> SHOW CONFIGURATION;
```

```
Configuration - DRSolution

Protection Mode: MaxPerformance
Databases:
  North_Sales - Primary database
  South_Sales - Physical standby database
  FS - Far Sync

Fast-Start Failover: DISABLED

Configuration Status:
SUCCESS
```

In this output, `South_Sales` and `FS` are indented under `North_Sales`. This indicates that the primary is sending redo data to both `South_Sales` and `FS`.

Step 2 Set up the RedoRoutes property for North_Sales and FS

Issue the following commands so that the current primary (North_Sales) sends redo data to the far sync instance only, and the far sync instance sends redo data to South_Sales when North_Sales is a primary:

```
DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY RedoRoutes='(LOCAL : FS SYNC)';
DGMGRL> EDIT FAR_SYNC 'FS' SET PROPERTY RedoRoutes='(North_Sales : South_Sales)';
DGMGRL> SHOW CONFIGURATION;
```

```
DGMGRL> SHOW CONFIGURATION;
```

```
Configuration - DRSolution
```

```
Protection Mode: MaxPerformance
Databases:
North_Sales - Primary database
  FS - Far Sync
  South_Sales - Physical standby database
```

```
Fast-Start Failover: DISABLED
```

```
Configuration Status:
SUCCESS
```

The indentation scheme in the output above indicates that North_Sales sends redo data to FS and FS sends redo data to South_Sales.

Step 3 Upgrade to maximum availability protection mode

Issue the following commands to upgrade the protection mode to maximum availability:

```
DGMGRL> EDIT CONFIGURATION SET PROTECTION MODE AS MaxAvailability;
DGMGRL> SHOW CONFIGURATION;
```

```
Configuration - DRSolution
```

```
Protection Mode: MaxAvailability
Databases:
North_Sales - Primary database
  FS - Far Sync
  South_Sales - Physical standby database
```

```
Fast-Start Failover: DISABLED
```

```
Configuration Status:
SUCCESS
```

Actions Required When Roles Change

If South_Sales ever becomes the primary and North_Sales becomes a standby database that receives redo data from the far sync instance, FS, then the RedoRoutes property must be set for South_Sales. Additionally, a new rule must be included in the RedoRoutes property for FS to allow it to send redo data to North_Sales when South_Sales is the primary database.

If these rules are not created, then North_Sales will not be able to receive redo data. To check whether the proper rules have been set, use the `SHOW CONFIGURATION WHEN PRIMARY IS` command to see what the redo transport configuration would be if South_Sales were the primary database. For example:


```
DGMGRL> SHOW CONFIGURATION WHEN PRIMARY IS 'South_Sales';
```

```
Configuration when South_Sales is primary - DRSolution
```

```
Members:
```

```
South_Sales - Primary database
FS          - Far Sync
North_Sales - Physical standby database
```

```
Members Not Receiving Redo:
```

```
North_FS - Physical standby database
Warning: ORA-16685: database does not receive redo data
```

To correct this error, set the RedoRoutes property for South_Sales and FS as follows:

```
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY RedoRoutes='(LOCAL : FS SYNC)';
DGMGRL> EDIT FAR_SYNC 'FS' SET PROPERTY RedoRoutes=('North_Sales : South_
Sales)(South_Sales : North_Sales)';
```

After the change to the RedoRoutes property for South_Sales and FS is complete, use the SHOW CONFIGURATION WHEN PRIMARY IS command to confirm that the error has been cleared:

```
DGMGRL> SHOW CONFIGURATION WHEN PRIMARY IS 'South_Sales';
```

```
Configuration when South_Sales is primary - DRSolution
```

```
Members:
```

```
South_Sales - Primary database
FS          - Far Sync
North_Sales - Physical standby database
```

6.7 Scenario 6: Enabling Fast-Start Failover and Starting the Observer

You can enable fast-start failover from any site, including the observer site, while connected to any database in the broker configuration. Enabling fast-start failover does not trigger a failover. Instead, it allows the observer that is monitoring the configuration to initiate a fast-start failover if conditions warrant a failover.

Fast-start failover can be enabled in configurations operating in either maximum performance or maximum availability protection mode. This section describes the steps to enable fast-start failover and start the observer where the configuration protection mode is set to maximum availability mode.

Step 1 Ensure standby redo logs are configured on the primary and target standby databases.

Standby redo logs must be configured on the primary and standby databases. You must stop log apply services prior to configuring standby redo logs.

See Also: *Oracle Data Guard Concepts and Administration* for instructions on configuring standby redo log files

Step 2 Configure redo transport mode appropriately.

Configure the primary and standby databases to receive redo via SYNC or FASTSYNC mode, if they receive redo directly rather than via a far sync instance. If either database receives redo via a far sync instance, then configure the far sync instance to receive redo via SYNC or FASTSYNC mode and configure the database to receive redo via ASYNC mode. For example:

```
DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY 'LogXptMode'='SYNC';
Property "LogXptMode" updated
```

```
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY 'LogXptMode'='SYNC';
Property "LogXptMode" updated
```

The broker does not allow these commands to succeed unless the databases are configured with standby redo log files.

Step 3 Set the FastStartFailoverTarget configuration property.

If you have two or more standby databases, set up the `FastStartFailoverTarget` configuration property on the primary database to indicate the desired target standby database. The broker reciprocally sets this property for the target standby database to indicate the primary database as its future target standby database when fast-start failover is actually enabled. There is no need for you set this property on the target standby as this is done for you automatically.

For example:

```
DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY FastStartFailoverTarget='South_
Sales';
Property "FastStartFailoverTarget" updated
```

Step 4 Upgrade the protection mode to MAXAVAILABILITY, if necessary.

If it is necessary to upgrade the protection mode, use the following DGMGRL EDIT CONFIGURATION command. For example:

```
DGMGRL> EDIT CONFIGURATION SET PROTECTION MODE AS MAXAVAILABILITY;
```

Step 5 Enable Flashback Database on the primary and target standby databases, if necessary.

If it is not already enabled on the primary and standby databases, enable Flashback Database by issuing the following statements on each database:

```
ALTER SYSTEM SET UNDO_RETENTION=3600 SCOPE=SPFILE;
ALTER SYSTEM SET UNDO_MANAGEMENT='AUTO' SCOPE=SPFILE;
SHUTDOWN IMMEDIATE;
STARTUP MOUNT;
SHOW PARAMETER UNDO;
ALTER SYSTEM SET DB_FLASHBACK_RETENTION_TARGET=4320 SCOPE=BOTH;
ALTER DATABASE ARCHIVELOG;
ALTER SYSTEM SET db_recovery_file_dest_size=<size>;
ALTER SYSTEM SET db_recovery_file_dest=<directory-specification>;
ALTER DATABASE FLASHBACK ON;
ALTER DATABASE OPEN;
```

Ensure the `UNDO_RETENTION` and `DB_FLASHBACK_RETENTION_TARGET` initialization parameters are set to sufficiently large values so that reinstatement is still possible after a prolonged outage.

Step 6 Start the observer.

Start the observer by logging into the observer computer and running DGMGRL. Connect to the configuration as a user who has the `SYSDG` or `SYSDBA` privilege and then issue the `START OBSERVER` command. Note that the command does not return; that is you will not get the DGMGRL prompt after issuing the command.

```
DGMGRL> CONNECT sysdg@North_Sales.example.com;
Password: password
```

```

Connected as SYSDBA.
DGMGRL> START OBSERVER;
Observer started

```

When starting the observer interactively, Oracle recommends that connection credentials not be supplied as command line parameters to the DGMGRL connect command. This practice prevents other users on the system from using a utility (for example, the UNIX ps utility) to display connection credentials. It also prevents clear-text passwords from being visible on the user's terminal.

When starting the observer from a script, Oracle recommends that you use a method that supports 'connect /', so that database connection credentials do not have to be embedded within the script. If you choose to use a client-side Oracle Wallet as a secure external password store (see *Oracle Database Enterprise User Security Administrator's Guide*), be sure to add credentials for both the primary and fast-start failover target standby databases. The database connect string that you specify when adding the credentials for each database must match the ObserverConnectIdentifier or DGConnectIdentifier database property.

Step 7 Enable fast-start failover.

You can enable fast-start failover while connected to any database system in the broker configuration. For example:

```

DGMGRL> ENABLE FAST_START FAILOVER;
Enabled.

```

Step 8 Verify the fast-start failover configuration.

Use the SHOW FAST_START FAILOVER command to display the fast-start failover settings:

```

DGMGRL> SHOW FAST_START FAILOVER;

Fast-Start Failover: DISABLED

Threshold: 30 seconds
Target: South_Sales
Observer: observer.example.com
Lag Limit: 30 seconds (not in use)
Shutdown Primary: TRUE
Auto-reinstate: TRUE
Observer Reconnect: (none)
Observer Override: FALSE

Configurable Failover Conditions
Health Conditions:
  Corrupted Controlfile YES
  Corrupted Dictionary YES
  Inaccessible Logfile NO
  Stuck Archiver NO
  Datafile Offline YES

Oracle Error Conditions:
(none)

```

The following commands show that the FastStartFailoverTarget property is set up reciprocally once fast-start failover is enabled. The first command, issued for the current primary database North_Sales, shows the value of the FastStartFailoverTarget property to be the current target standby, South_Sales. The second command, issued for the target standby South_Sales, shows the current

primary, North_Sales, as the target standby's future target standby should it ever take over as a primary.

```
DGMGRL> SHOW DATABASE 'North_Sales' FastStartFailoverTarget;  
FastStartFailoverTarget='South_Sales';
```

```
DGMGRL> SHOW DATABASE 'South_Sales' FastStartFailoverTarget;  
FastStartFailoverTarget='North_Sales';
```

6.8 Scenario 7: Enabling Fast-Start Failover When a Far Sync Instance Is In Use

Fast-start failover can be enabled in maximum availability mode when the fast-start failover target is a logical or physical standby database that receives redo data from a far sync instance.

To enable fast-start failover when a far sync instance is used to ship redo data to the standby database, the `FastStartFailoverTarget` property must first be set on both the primary and target standby database, as follows:

```
DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY FastStartFailoverTarget='South_ Sales';  
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY FastStartFailoverTarget='North_ Sales';
```

Then, fast-start failover can be enabled, as follows:

```
DGMGRL> ENABLE FAST_START FAILOVER;
```

Note that the far sync instance database is not specified as the fast-start failover target for either North_Sales or South_Sales.

6.9 Scenario 8: Performing Routine Management Tasks

There may be situations in which you want to change the state or properties of the databases in a broker configuration to perform routine maintenance on one or more databases. You might also need to temporarily disable broker management of databases in a configuration.

6.9.1 Changing Properties and States

As you monitor the configuration, you might need to dynamically modify the states of the databases or their properties. The following sections show how to change the state or properties of the databases in the configuration.

6.9.1.1 Alter a Database Property

You can modify the values of database properties at any time—if the database is enabled or disabled.

[Example 6-3](#) shows how to use the `EDIT DATABASE` command to change the `LogXptMode` database property to the value `ASYNC` for the North_Sales database.

Example 6-3 Altering a Database Property

```
DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY 'LogXptMode'=ASYNC;
```

DGMGRL returns the following message to indicate that the `LogXptMode` property was updated successfully in the Data Guard configuration file:

Property "LogXptMode" updated

If the configuration is currently disabled, the database does not use the new property value until you enable the broker configuration with the `ENABLE CONFIGURATION` command.

6.9.1.2 Reset a Property to Its Default Value

You can reset a configuration or configurable property to its default value at any time if the database or configuration is enabled or disabled.

[Example 6-4](#) shows how to use the `EDIT DATABASE` command to reset the `LogXptMode` database configurable property to its default value for the `North_Sales` database.

Example 6-4 *Resetting LogXptMode to Its Default*

```
EDIT DATABASE 'North_Sales' RESET PROPERTY LogXptMode;
```

[Example 6-5](#) shows how to use the `EDIT CONFIGURATION` command to reset the `TraceLevel` configuration property to its default value.

Example 6-5 *Resetting TraceLevel to Its Default Value*

```
EDIT CONFIGURATION RESET PROPERTY TraceLevel;
```

6.9.1.3 Alter the State of a Standby Database

You might want to temporarily stop Redo Apply on a physical standby. To change the state of the standby database to `APPLY-OFF`, enter the `EDIT DATABASE` command as shown in [Example 6-6](#).

Example 6-6 *Altering a Standby Database State*

```
DGMGRL> EDIT DATABASE 'South_Sales' SET STATE='APPLY-OFF';
Succeeded.
```

Redo data is still being received when you put the physical standby database in the `APPLY-OFF` state.

6.9.1.4 Alter the State of a Primary Database

You might want to stop the transmittal of redo data to the standby database. To change the state of the primary database to accommodate this, use the following command:

```
DGMGRL> EDIT DATABASE North_Sales SET STATE=TRANSPORT-OFF;

Succeeded.
```

To change the state of the primary database back to `TRANSPORT-ON`, do the following:

```
DGMGRL> EDIT DATABASE North_Sales SET STATE=TRANSPORT-ON;

Succeeded.
```

6.9.2 Disabling the Configuration and Databases

When you disable the broker configuration or any of its databases, you are disabling the broker's management of them and are effectively removing your ability to use `DGMGRL` to manage and monitor them. However, disabling the broker's management of a broker configuration does not affect the actual operation of the underlying Oracle

Data Guard configuration or the databases. For example, the redo transport services and log apply services in the Oracle Data Guard configuration continue to function unchanged, but you can no longer manage them.

6.9.2.1 Disable a Configuration

You must use the `DISABLE CONFIGURATION` command to disable management of the entire broker configuration including the primary database as shown in [Example 6-7](#).

Example 6-7 Disabling the Configuration and Primary Database

```
DGMGRL> DISABLE CONFIGURATION;
```

The only way to disable broker management of the primary database is to use the `DISABLE CONFIGURATION` command; the `DISABLE DATABASE` command only disables management of a standby database. Likewise, the `DISABLE FAR_SYNC` command only disables management of a far sync instance.

Note: If you disable management of a configuration while connected to the standby database or far sync instance, you must connect to the primary database (that is, a database whose control file role is primary) to reenble the configuration.

Disabling the broker's management of a configuration member does not remove the member from the broker configuration file. You can reenble your ability to use DGMGRL (or Cloud Control) to manage the member by entering the appropriate `ENABLE CONFIGURATION` or `ENABLE DATABASE` command.

6.9.2.2 Disable a Standby Database

You use the `DISABLE DATABASE` command when you temporarily do not want the broker to manage and monitor a standby database.

You can explicitly disable broker management of a standby database to prevent it from being enabled when the rest of the configuration is enabled. [Example 6-8](#) shows how to disable the `South_Sales` standby database.

Example 6-8 Disabling a Standby Database

```
DGMGRL> DISABLE DATABASE 'South_Sales';  
Disabled.
```

Note: You cannot disable a standby database from the configuration if fast-start failover is enabled and the database to be disabled is the target standby database.

Note: If you disable management of a standby database while connected to that standby database, you must connect to the primary database or another enabled standby database to reenble broker-management of the standby database.

Caution: If you disable broker management of a standby database in the broker configuration, that standby database cannot be used by the broker as a failover target in the event of loss of the primary database.

When operating under either maximum protection mode or maximum availability mode, the broker prevents you from disabling the last standby database that supports the protection mode.

6.9.2.3 Disabling a Far Sync Instance

Use the `DISABLE FAR_SYNC` command when you temporarily do not want the broker to manage and monitor a far sync instance.

You can explicitly disable broker management of a far sync instance to prevent it from being enabled when the rest of the configuration is enabled. [Example 6–6](#) shows how to disable the far sync instance.

Example 6–9 Disabling a Far Sync Instance

```
DGMGRL> DISABLE FAR_SYNC 'FS';
Disabled.
```

Note: The following restrictions apply when disabling a far sync instance:

- You cannot disable a far sync instance if it is specified in the `RedoRoutes` property of any other configuration member.
 - If you disable management of a far sync instance while connected to that far sync instance, you must connect to the primary database or another enabled standby database to reenable broker management of the far sync instance.
-
-

Caution: If you disable broker management of a far sync instance in the broker configuration, that far sync instance cannot be specified in a `RedoRoutes` property for any other configuration member.

6.9.3 Removing the Configuration, a Standby Database, or a Far Sync Instance

When you use the `REMOVE CONFIGURATION`, `REMOVE DATABASE`, or `REMOVE FAR_SYNC` command, you effectively delete the configuration, standby database, or far sync instance from the broker configuration file, removing the ability of Oracle Data Guard broker to manage them.

A remove operation with the `PRESERVE DESTINATIONS` clause *does not* remove or delete the actual Oracle Data Guard configuration underneath, nor does it affect the operation of the actual Oracle Data Guard configuration and its databases.

Caution: After you use the `REMOVE CONFIGURATION`, `REMOVE DATABASE`, or `REMOVE FAR_SYNC` command, you must reissue the command(s) that you originally issued if you decide to re-create the deleted object. You must go through the steps in [Section 6.2](#) as necessary, to create a broker configuration that can be managed with DGMGRL (or Cloud Control).

Note: The following restrictions apply:

- You cannot remove a standby database from the configuration if fast-start failover is enabled and the database to be removed is the target standby database.
 - You cannot remove a standby database or a far sync instance if it is specified in the `RedoRoutes` property for any other member in the configuration.
-
-

Step 1 Remove a standby database from the configuration.

When you use the `REMOVE DATABASE` command, broker management and monitoring of the database ceases and the database is deleted from the broker configuration file.

Show the configuration before deletion of the `South_Sales` standby database:

```
DGMGRL> SHOW CONFIGURATION;

Configuration - DRSolution

  Protection Mode: MaxPerformance
  Databases:
    North_Sales - Primary database
    FS - Far Sync
    South_Sales - Physical standby database

Fast-Start Failover: DISABLED

Configuration Status:
SUCCESS
```

Issue the DGMGRL `REMOVE DATABASE` command to remove the `South_Sales` database information from the Data Guard configuration file:

```
DGMGRL> REMOVE DATABASE 'South_Sales';
Removed database "South_Sales" from the configuration
```

Show the configuration after deletion of the `South_Sales` standby database:

```
DGMGRL> SHOW CONFIGURATION;

Configuration - DRSolution

  Protection Mode: MaxPerformance
  Databases:
    North_Sales - Primary database
    FS - Far Sync

Fast-Start Failover: DISABLED
```



```
Configuration Status:
SUCCESS
```

When operating under either maximum protection mode or maximum availability mode, the broker prevents you from deleting the last standby database that supports the protection mode.

Step 2 Remove the Far Sync Instance

Use the following command to remove the FS far sync instance information from the Oracle Data Guard configuration file:

```
DGMGRL> REMOVE FAR_SYNC 'FS';
Removed far sync instance "FS" from the configuration
```

Show the configuration after deletion of the FS far sync instance:

```
DGMGRL> SHOW CONFIGURATION;

Configuration - DRSolution

Protection Mode: MaxPerformance
Databases:
North_Sales - Primary database

Fast-Start Failover: DISABLED

Configuration Status:
SUCCESS
```

Step 3 Remove the broker configuration.

Use the following command to remove the entire configuration from management and monitoring by the broker:

```
DGMGRL> REMOVE CONFIGURATION;
```

Note: You cannot remove the configuration if fast-start failover is enabled.

DGMGRL returns the following message to indicate the command successfully removed all of the configuration information from the Data Guard configuration file:

```
Removed configuration

DGMGRL> SHOW CONFIGURATION;
Error: ORA-16532: Data Guard broker configuration does not exist

Configuration details cannot be determined by DGMGRL
```

6.10 Scenario 9: Performing a Switchover Operation

You can switch the role of the primary database and a standby database using the SWITCHOVER command. Before you issue the SWITCHOVER command, you must ensure:

- The state of the primary and standby databases are TRANSPORT-ON and APPLY-ON, respectively.
- All participating databases are in good health, without any errors or warnings present.

- The standby database properties were set on the primary database, so that the primary database can function correctly when transitioning to a standby database (shown in the following examples in boldface type).
- Standby redo log files are configured on the primary database.
- If the configuration is in maximum availability mode, then the current primary is configured to receive redo via SYNC or FASTSYNC mode if it will receive redo directly from the new primary. If it will receive redo via a far sync instance, then the far sync instance is configured to receive redo via SYNC or FASTSYNC mode and the current primary is configured to receive redo via ASYNC mode. If the configuration is in maximum protection mode, then the current primary is configured to receive redo via SYNC mode.
- If fast-start failover is enabled, you can perform a switchover only to the standby database that was specified as the target standby database.

Step 1 Check the primary database.

Use the `SHOW DATABASE VERBOSE` command to check the state, health, and properties of the primary database, as follows:

```
DGMGRL> SHOW DATABASE VERBOSE 'North_Sales';
```

```
Database - North_Sales
```

```
Role:                PRIMARY
Intended State:      TRANSPORT-ON
Instance(s):
  north_sales1
```

```
Properties:
  DGConnectIdentifier           = 'North_Sales.example.com'
  ObserverConnectIdentifier     = ''
  LogXptMode                   = 'ASYNC'
  RedoRoutes                   = ''
  DelayMins                    = '0'
  Binding                      = 'optional'
  MaxFailure                   = '0'
  MaxConnections               = '1'
  ReopenSecs                   = '300'
  NetTimeout                   = '30'
  RedoCompression              = 'DISABLE'
  LogShipping                  = 'ON'
  PreferredApplyInstance       = ''
  ApplyInstanceTimeout         = '0'
  ApplyLagThreshold            = '0'
  TransportLagThreshold        = '0'
  TransportDisconnectedThreshold = '0'
  ApplyParallel                = 'AUTO'
  StandbyFileManagement        = 'AUTO'
  ArchiveLagTarget             = '0'
  LogArchiveMaxProcesses       = '5'
  LogArchiveMinSucceedDest     = '1'
  DbFileNameConvert          = 'dbs/bt, dbs/t, dbs/ct, dbs/t'
  LogFileNameConvert        = 'dbs/bt, dbs/t, dbs/ct, dbs/t'
  FastStartFailoverTarget      = 'South_Sales'
  InconsistentProperties        = '(monitor)'
  InconsistentLogXptProps      = '(monitor)'
  SendQEntries                 = '(monitor)'
  LogXptStatus                 = '(monitor)'
```

```

RecvQEntries                = '(monitor)'
StaticConnectIdentifier      =
'DESCRIPTION=(ADDRESS=(PROTOCOL=tcp) (HOST=north_sales1.example.com) (PORT=2840))
(CONNECT_DATA=(SERVICE_NAME=North_Sales_DGMGRL.example.com) (INSTANCE_NAME=north_
sales1) (SERVER=DEDICATED)))'
StandbyArchiveLocation    = 'USE_DB_RECOVERY_FILE_DEST'
AlternateLocation            = ''
LogArchiveTrace              = '255'
LogArchiveFormat          = 'db1r_%d_%t_%s_%R.arc'
TopWaitEvents                = '(monitor)'

```

```

Database Status:
SUCCESS

```

In particular, you should examine the boldface properties and the current status of the primary database. See [Chapter 4](#) for information about managing databases.

Step 2 Check the standby database that is the target of the switchover.

Use the `SHOW DATABASE` command to check the status of the standby database that is the target of the switchover. For example:

```
DGMGRL> SHOW DATABASE 'South_Sales';
```

```
Database - South_Sales
```

```

Role: PHYSICAL STANDBY
Intended State: APPLY-ON
Transport Lag: 0 seconds (computed 0 seconds ago)
Apply Lag: 0 seconds (computed 0 seconds ago)
Apply Rate: 1.44 MByte/s
Real Time Query: OFF
Instance(s):
  south_sales1

```

```

Database Status:
SUCCESS

```

Step 3 Confirm that the database is ready for a role change.

Prior to performing a role change, you can use the `VALIDATE DATABASE` command to perform an exhaustive set of checks on the database to confirm that it is ready for a role change.

The examples shown in this step use the `VALIDATE DATABASE` command for all three databases in the `DRSolution` configuration: a primary, logical standby, and physical standby database. The configuration looks as follows:

```
DGMGRL> SHOW CONFIGURATION;
```

```
Configuration - DRSolution
```

```

Protection Mode: MaxAvailability
Databases:
North_Sales - Primary database
West_Sales - Logical standby database
South_Sales - Physical standby database

```

```
Fast-Start Failover: DISABLED
```

```

Configuration Status:
SUCCESS

```

Example 6–10 Validate the Primary Database

Validate the primary database, as follows:

```
DGMGRL> VALIDATE DATABASE 'North_Sales';
```

```
Database Role: Primary database
```

```
Ready for Switchover: Yes
```

Example 6–11 Validate the Logical Standby Database

Validate the logical standby database, as follows:

```
DGMGRL> VALIDATE DATABASE 'West_Sales';
```

```
Database Role: Logical standby database
```

```
Primary Database: North_Sales
```

```
Ready for Switchover: Yes
```

```
Ready for Failover: Yes (Primary Running)
```

```
Warning: Physical and snapshot standby databases will  
be disabled if a role change is performed to this database
```

Example 6–12 Validate the Physical Standby Database

Validate the physical standby database, as follows:

```
DGMGRL> VALIDATE DATABASE 'South_Sales';
```

```
Database Role: Physical standby database
```

```
Protection Mode: MaxAvailability
```

```
Error: Switchover to this standby is not possible since there  
are no other standbys that can support the protection mode
```

```
Primary Database: North_Sales
```

```
Ready for Switchover: No
```

```
Ready for Failover: Yes (Primary Running)
```

```
Transport-Related Property Settings:
```

Property	North_Sales Value	South_Sales Value
LogXptMode	ASync	SYnc

Because the configuration protection is set to maximum availability mode and database `South_Sales` is the only one that has its `LogXptMode` property set to `SYnc`, an error is displayed to indicate that a switchover is not possible. Note that when a database receives redo from a database or a far sync instance that has the `RedoRoutes` property configured with a transport mode, that mode overrides the transport mode specified by `LogXptMode`.

Step 4 Issue the switchover command.

Issue the `SWITCHOVER` command to swap the roles of the primary and standby databases. The following example shows how the broker automatically shuts down and restarts the old primary database as a part of the switchover. (See the usage notes in [Section 7.1.3](#) for information about how to set up the broker environment so that DGMGRL can automatically restart the primary and standby databases for you.)

```
DGMGRL> SWITCHOVER TO 'South_Sales';
```

```
Performing switchover NOW, please wait...
```

```
Operation requires a connection to instance "south_sales1" on database "South_
```

```

Sales"
Connecting to instance "south_sales1"...
Connected as SYSDBA.
New primary database "South_Sales" is opening...
Operation requires startup of instance "north_sales1" on database "North_Sales"
Starting instance "north_sales1"...
ORACLE instance started.
Database mounted.
Switchover succeeded, new primary is "South_Sales"

```

After the switchover completes, use the `SHOW CONFIGURATION` and `SHOW DATABASE` commands to verify that the switchover operation was successful.

Step 5 Show the configuration.

Issue the `SHOW CONFIGURATION` command to verify that the switchover was successful.

```

DGMGRL> SHOW CONFIGURATION;

Configuration - DRSolution

Protection Mode: MaxAvailability
Databases:
  South_Sales - Primary database
  West_Sales - Logical standby database
  North_Sales - Physical standby database

Fast-Start Failover: DISABLED

Configuration Status:
SUCCESS

```

6.11 Scenario 10: Performing a Manual Failover Operation

You invoke a failover operation in response to an emergency situation, usually when the primary database cannot be accessed or is unavailable. See [Section 5.2](#) before you fail over to decide which standby database should be the target of the failover. The following scenario describes a failover to the remote database called `South_Sales`.

Note: If fast-start failover is enabled, you can perform a manual failover only to the standby database that was specified as the target of a fast-start failover and only when the observer is running and currently has connectivity with the standby database.

If you want to perform a manual failover to a standby database that is not the fast-start failover target standby database, you must first disable fast-start failover using the `FORCE` option on the standby database you want to fail over. See [Section 5.5.5, "Disabling Fast-Start Failover"](#) for more information about the `FORCE` option.

Step 1 (Optional) Check the readiness of the target standby.

To validate the target standby database to ensure that it's ready to become the new primary database, use the `VALIDATE DATABASE` command, as shown in the following example:

```

DGMGRL> VALIDATE DATABASE 'South_Sales';

Database Role: Physical standby database

```

```
Primary Database: South_Sales
```

```
Ready for Switchover: Yes
```

```
Ready for Failover: Yes
```

Step 2 Connect to the target standby database.

To perform the failover operation, you must connect to the standby database to which you want to fail over as a user that has the SYSDG or SYSDBA privilege. For example:

```
DGMGRL> CONNECT sysdg@South_Sales.example.com;
```

```
Password: password
```

```
Connected as SYSDG.
```

Step 3 Issue the failover command.

Now you can issue the failover command to make the target standby database the new primary database for the configuration.

```
DGMGRL> FAILOVER TO 'South_Sales';
```

```
Performing failover NOW, please wait...
```

```
Failover succeeded, new primary is "South_Sales"
```

Note that after the failover completes, the original primary database cannot be used as a standby database of the new primary database unless it is reinstated or re-created (as described in [Section 5.4.3](#)).

Step 4 Show the configuration.

Issue the SHOW CONFIGURATION command to verify the failover.

```
DGMGRL> SHOW CONFIGURATION;
```

```
Configuration - DRSolution
```

```
Protection Mode: MaxAvailability
```

```
Databases:
```

```
South_Sales - Primary database
```

```
Warning: ORA-16629: database reports a different protection level from the protection mode
```

```
North_Sales - Physical standby database (disabled)
```

```
ORA-16661: the standby database needs to be reinstated
```

```
Fast-Start Failover: DISABLED
```

```
Configuration Status:
```

```
WARNING
```

Note that in this example, the configuration was operating in maximum availability mode. The protection mode was preserved after the failover. The configuration also has a warning status. The SHOW DATABASE command for the new primary shows that the warning is the result of not having an enabled physical standby database. As a result, the warning status indicates that the protection level of the configuration is not the same as the configured mode.

Step 5 Show the new primary database.

```
DGMGRL> SHOW DATABASE South_Sales;
```

```
Database - South_Sales
```

```
Role: PRIMARY
```

```

Intended State:  TRANSPORT-ON
Instance(s):
  south_sales1

Database Warning(s):
  ORA-16629: database reports a different protection level from the protection
mode

Database Status:
WARNING

```

Step 6 Show the old primary database.

Issue the `SHOW DATABASE` command to see that the former (failed) primary database was disabled by the broker as a consequence of the failover. It must be reinstated (as described in [Section 5.4.3](#)).

```

DGMGRL> SHOW DATABASE 'North_Sales';
Database - North_Sales

Role: PHYSICAL STANDBY
Intended State: APPLY-ON
Transport Lag: (unknown)
Apply Lag: (unknown)
Apply Rate: (unknown)
Real Time Query: OFF
Instance(s):
  north_sales1

Database Status:
ORA-16661: the standby database needs to be reinstated

```

6.12 Scenario 11: Reinstating a Failed Primary Database

If your former primary database was configured with Flashback Database, you can easily reinstate the failed primary database as a standby database of the new primary database. The failed primary database will be reinstated as a standby type that matches the old standby database. For example, if you failed over to a physical standby database, the old primary will be reinstated as a physical standby database.

To reinstate the failed primary database, start it to the mounted state. Then run `DGMGRL`, connect to the new primary database and reinstate the old primary database.

Step 1 Restart the Old Primary Database.

```

% dgmgrl connect sysdg
Password: password
Connected as SYSDBG.
DGMGRL> startup mount;
ORACLE instance started.
Database mounted.

```

Step 2 Reinstat the old primary database.

```

% dgmgrl connect sysdg
Password: password
Connected as SYSDBG.
DGMGRL> REINSTATE DATABASE 'North_Sales';
Reinstating database "North_Sales", please wait...
Reinstatement of database "North_Sales" succeeded

```

```
Database dismounted.
ORACLE instance shut down.
Operation requires startup of instance "north_sales1" on database "North_Sales"
Starting instance "north_sales1"...
ORACLE instance started.
Database mounted.
Continuing to reinstate database "North_Sales" ...
Reinstatement of database "North_Sales" succeeded
```

After the primary has been reinstated, issue the `SHOW CONFIGURATION` and `SHOW DATABASE` commands to confirm that the old primary has been successfully reinstated.

Step 3 Show the Configuration and Databases.

```
DGMGRL> SHOW CONFIGURATION
```

```
Configuration - DRSolution
```

```
Protection Mode: MaxAvailability
Databases:
  South_Sales - Primary database
  North_Sales - Physical standby database
```

```
Fast-Start Failover: DISABLED
```

```
Configuration Status:
SUCCESS
```

```
DGMGRL> SHOW DATABASE 'South_Sales';
```

```
Database - South_Sales
```

```
Role: PRIMARY
Intended State: TRANSPORT-ON
Instance(s):
  south_sales1
```

```
Database Status:
SUCCESS
```

```
DGMGRL> SHOW DATABASE 'North_Sales'
```

```
Database - 'North_Sales'
```

```
Role: PHYSICAL STANDBY
Intended State: APPLY-ON
Transport Lag: 0 seconds (computed 1 second ago)
Apply Lag: 0 seconds (computed 1 second ago)
Apply Rate: 0 Byte/s
Real Time Query: OFF
Instance(s):
  north_sales1
```

```
Database Status:
SUCCESS
```


6.13 Scenario 12: Converting a Physical Standby to a Snapshot Standby

If you have a physical standby database that you would like to convert to a snapshot standby database, use the `DGMGRL CONVERT DATABASE` command. Redo data will continue to be received by the database while it is operating as a snapshot standby database, but it will not be applied until the snapshot standby is converted back into a physical standby database.

A physical standby database must be configured with a fast recovery area to convert it to a snapshot standby database. This is because a guaranteed restore point is created during the conversion process, and guaranteed restore points require a fast recovery area.

```
DGMGRL> convert database 'South_Sales' to snapshot standby;
Converting database "South_Sales" to a Snapshot Standby database, please wait...
Database "South_Sales" converted successfully
```

```
DGMGRL> SHOW CONFIGURATION;
```

```
Configuration - DRSolution
```

```
Protection Mode: MaxPerformance
Databases:
  North_Sales - Primary database
  South_Sales - Snapshot standby database
```

```
Fast-Start Failover: DISABLED
```

```
Configuration Status:
SUCCESS
```

When you are ready to revert the database back to a physical standby database, use the `DGMGRL CONVERT DATABASE` command again as follows. Any updates made to the database while it was operating as a snapshot standby database will be discarded. All accumulated redo data will be applied by Redo Apply services after the database is converted back to a physical standby database.

```
DGMGRL> CONVERT DATABASE 'South_Sales' to PHYSICAL STANDBY;

Converting database "South_Sales" to a Physical Standby database, please wait...
Operation requires shutdown of instance "south_sales1" on database "South_Sales"
Shutting down instance "south_sales1"...
Database closed.
Database dismounted.
ORACLE instance shut down.
Operation requires startup of instance "south_sales1" on database "South_Sales"
Starting instance "south_sales1"...
ORACLE instance started.
Database mounted.
Continuing to convert database "South_Sales" ...
Database "South_Sales" converted successfully
```

6.14 Scenario 13: Monitoring a Data Guard Configuration

The scenario in this section demonstrates how to use the `SHOW` command and monitorable properties to identify and resolve a failure situation.

Step 1 Check the configuration status.

The status of the broker configuration is an aggregated status of all databases and instances in the broker configuration. You can check the configuration status first to determine whether or not any further action needs to be taken. If the configuration status is SUCCESS, everything in the broker configuration is working properly. However, if you see a status of WARNING or ERROR, then something is wrong in the configuration.

For example, in the following display, you can see that the primary database has multiple warnings:

```
DGMGRL> SHOW CONFIGURATION;
Configuration - DRSolution

Protection Mode: MaxPerformance
Databases:
  North_Sales - Primary database
    Warning: ORA-16809: multiple warnings detected for the database

  South_Sales - Physical standby database

Fast-Start Failover: DISABLED

Configuration Status:
WARNING
```

Step 2 Check the database status.

To identify the warnings on the primary database, show its status using the SHOW DATABASE command:

```
DGMGRL> SHOW DATABASE 'North_Sales';
Database - North_Sales

Role:          PRIMARY
Intended State: TRANSPORT-ON
Instance(s):
  north_sales1
    Warning: ORA-16737: the redo transport service for standby "South_Sales" has
an error
    Warning: ORA-16714: the value of property LogArchiveTrace is inconsistent
with the database setting
    Warning: ORA-16715: redo transport-related property ReopenSecs of standby
database "South_Sales" is inconsistent

Database Status:
WARNING
```

Step 3 Check the LogXptStatus monitorable property.

The SHOW DATABASE output in step 2 shows a Warning for error ORA-16737. To identify the exact transport error, use the LogXptStatus monitorable property:

```
DGMGRL> SHOW DATABASE 'North_Sales' 'LogXptStatus';
LOG TRANSPORT STATUS
PRIMARY_INSTANCE_NAME  STANDBY_DATABASE_NAME  STATUS
  north_sales1          South_Sales  ORA-12541: TNS:no listener
```

The output shows that the listener for the physical standby database is not running. To fix this error, start the listener for the physical standby database South_Sales.

Step 4 Check the InconsistentProperties monitorable property.

The SHOW DATABASE output in step 2 also shows a Warning for error ORA-16714. To identify the exact error, use the InconsistentProperties monitorable property:

```
DGMGRL> SHOW DATABASE 'North_Sales' 'InconsistentProperties';
INCONSISTENT PROPERTIES
  INSTANCE_NAME  PROPERTY_NAME  MEMORY_VALUE  SPFILE_VALUE  BROKER_VALUE
north_sales1    LogArchiveTrace  511           255           255
```

The current database memory value (511) is different from both the server parameter file (SPFILE) value (255) and the Oracle Data Guard broker's property value (255). If you decide the database memory value is correct, you can update the Oracle Data Guard broker's property value using the following command:

```
DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY 'LogArchiveTrace'=511;
Property "LogArchiveTrace" updated
```

This command will result in the broker updating the SPFILE value so that the value of LogArchiveTrace is kept consistent.

Step 5 Check the InconsistentLogXptProps monitorable property.

Another warning shown in the SHOW DATABASE display in Step 2 is ORA-16715. To identify the inconsistent values for the redo transport database property, ReopenSecs, you can use the InconsistentLogXptProps monitorable property:

```
DGMGRL> SHOW DATABASE 'North_Sales' 'InconsistentLogXptProps';
INCONSISTENT LOG TRANSPORT PROPERTIES
  INSTANCE_NAME  STANDBY_NAME  PROPERTY_NAME  MEMORY_VALUE  BROKER_VALUE
south_sales1    South_Sales    ReopenSecs     600           300
```

The current database memory value (600) is different from the Oracle Data Guard broker's property value (300). If you think the broker's property value is correct, you can fix the inconsistency by re-editing the property of the standby database with the same value, as shown in the following example:

```
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY 'ReopenSecs'=300;
Property "ReopenSecs" updated
```

You can also reenable the standby database or reset the state of the primary database to TRANSPORT-ON to fix this inconsistency.

Oracle Data Guard Command-Line Interface Reference

The Data Guard command-line interface (DGMGRL) enables you to manage a Data Guard broker configuration and its various members directly from the command line, or from batch programs or scripts. You can use the Data Guard command-line interface as an alternative to Oracle Enterprise Manager Cloud Control (Cloud Control) for managing a Data Guard configuration.

This chapter contains the following sections, which provide reference information for the Data Guard command-line interface:

- [Starting the Data Guard Command-Line Interface](#)
- [Exiting the Data Guard Command-Line Interface](#)

7.1 Starting the Data Guard Command-Line Interface

To run DGMGRL, you must have `SYSDG` or `SYSDBA` administrative privilege.

Start the command-line interface by entering `dgmgrl` at the command-line prompt on a system where Oracle is installed:

```
% dgmgrl
```

The DGMGRL command prompt is displayed:

```
DGMGRL>
```

7.1.1 DGMGRL Optional Parameters

You can supply optional parameters on the command line to indicate how you want the Data Guard command-line interface to display output such as command prompts, banners, and messages.

Additionally, a single command mode is available. In this mode, DGMGRL executes one command and exits upon the completion of the command. The exit code is the result of the command. If the exit code is 0, the command completed successfully. Otherwise, there was an error.

The command line of DGMGRL appears as follows:

```
% dgmgrl [<options>] [<logon> [<command>] ]
```

Specify any of the following keywords when you invoke the DGMGRL command-line interface:

- `<options>` can be one of the following choices:

- `-echo`

Displays command input and output to the default display device. If you do not use this parameter, only the output from the command is displayed.
- `-logfile <file-spec> "<dgmgrl-command>"`

Specifies a file into which you can capture the actions of the DGMGRL command-line interface. This is particularly useful when DGMGRL is being invoked to serve as the fast-start failover *observer*. See the ["START OBSERVER"](#) command for more information.

See Also:

- The ["START OBSERVER"](#) command
- [Section 9.5.3, "Capturing Observer Actions in the Observer Log File"](#)

- `-silent`

Suppresses the display of the DGMGRL (`DGMGRL>`) command prompt on your default display device. This option is useful if you are directing the command output to a file or to another display tool.
- `<logon>` is:
 - `username [@connect-identifier]`

To connect to the database, enter a `username` and optionally, a `connect-identifier`. You will then be prompted for a password. The `connect-identifier` is a fully specified connect descriptor or a name to be resolved by an Oracle naming method (for example, TNS).

WARNING: Including a password on the command line when invoking DGMGRL is a security risk. This risk can be avoided either by omitting the password when invoking DGMGRL and entering it when prompted, or by using an external authentication method.

- You can connect as `'/'` when using operating-system authentication (remote database restarts will not work), Secure Sockets Layer (SSL) protocol, or database credentials stored in a wallet.

- `<command>` is a single command.

For example:

```
% dgmgrl sys/ "show database 'North_Sales'"
```

Password: *password*

The following subsections specify the command format that you enter at the `DGMGRL>` command prompt.

7.1.2 DGMGRL Command Format and Parameters

The DGMGRL commands allow you to create and maintain one broker configuration at a time.

After you invoke the command-line interface, you can enter any of the DGMGRL commands listed in [Table 7-1](#). Each command and its associated parameters are described in detail in later sections of this chapter.

Table 7-1 Summary of DGMGRL Commands

Command	Effect
ADD DATABASE	Adds a new standby database to the existing broker configuration.
ADD FAR_SYNC	Adds an existing far sync instance to an Oracle Data Guard broker configuration.
CONNECT	Connects to the specified database using the specified username.
CONVERT DATABASE	Converts the specified database to either a snapshot standby database or a physical standby database.
CREATE CONFIGURATION	Creates a broker configuration and adds a primary database to that configuration.
DISABLE CONFIGURATION	Disables broker management of a configuration so that the configuration and all of its databases are no longer managed by the broker.
DISABLE DATABASE	Disables broker management of the named standby database.
DISABLE FAR_SYNC	Disables broker management of a far sync instance.
DISABLE FAST_START FAILOVER	Disables fast-start failover.
DISABLE FAST_START FAILOVER CONDITION	Allows a user to remove conditions for which a fast-start failover should be performed.
EDIT CONFIGURATION (Property)	Changes the value of a property for the broker configuration.
EDIT CONFIGURATION (Protection Mode)	Changes the current protection mode setting for the broker configuration.
EDIT CONFIGURATION (RENAME)	Changes the configuration name.
EDIT CONFIGURATION RESET (Property)	Resets the specified configuration property to its default value.
EDIT DATABASE (Property)	Changes the value of a property for the named database.
EDIT DATABASE (Rename)	Changes the name used by the broker to refer to the specified database.
EDIT DATABASE (State)	Changes the state of the specified database.
EDIT DATABASE RESET (Property)	Resets the specified property for the named database to its default value.
EDIT FAR_SYNC	Changes the name, state, or properties of a far sync instance.
EDIT FAR_SYNC RESET (Property)	Resets the specified property for the named far sync instance to its default value.
EDIT INSTANCE (AUTO PFILE)	Sets the name of the initialization parameter file for the specified instance.
EDIT INSTANCE (Property)	Changes the value of a property for the specified instance.
EDIT INSTANCE RESET (Property)	Resets an instance-specific property for the specified instance(s) to its default value.

Table 7–1 (Cont.) Summary of DGMGRL Commands

Command	Effect
ENABLE CONFIGURATION	Enables broker management of the broker configuration and all of its databases.
ENABLE DATABASE	Enables broker management of the specified database.
ENABLE FAR_SYNC	Enables broker management of the specified far sync instance.
ENABLE FAST_START FAILOVER	Enables the broker to automatically failover from the primary database to a target standby database.
ENABLE FAST_START FAILOVER CONDITION	Allows a user to add conditions for which a fast-start failover should be performed.
EXIT	Exits the Data Guard command-line interface.
FAILOVER	Performs a database failover operation in which the standby database, to which DGMGRL is currently connected, fails over to the role of primary database.
HELP	Displays online help for the Data Guard command-line interface.
QUIT	Quits the Data Guard command-line interface.
REINSTATE DATABASE	Reinstates the database after a failover.
REMOVE CONFIGURATION	Removes the broker configuration and ends broker management of its members.
REMOVE DATABASE	Removes the specified standby database from the broker configuration.
REMOVE FAR_SYNC	Removes a far sync instance from an Oracle Data Guard broker configuration.
REMOVE INSTANCE	Removes an instance from the broker configuration.
SHOW CONFIGURATION	Displays information about the broker configuration.
SHOW CONFIGURATION WHEN PRIMARY IS	Shows the redo transport configuration that would be in effect if the specified database were the primary database.
SHOW DATABASE	Displays information about the specified database.
SHOW FAR_SYNC	Shows information about a far sync instance.
SHOW FAST_START FAILOVER	Displays all fast-start failover related information.
SHOW INSTANCE	Displays information about the specified instance.
SHUTDOWN	Shuts down a currently running Oracle database.
SQL	Allows you to enter SQL statements from the Data Guard command-line interface (DGMGRL).
START OBSERVER	Starts the observer.
STARTUP	Starts an Oracle instance with the same options as SQL*Plus, including mounting and opening a database.
STOP OBSERVER	Stops the observer.
SWITCHOVER	Performs a switchover operation in which the current primary database becomes a standby database, and the specified standby database becomes the primary database.
VALIDATE DATABASE	Performs a comprehensive set of database checks prior to a role change.

Table 7–1 (Cont.) Summary of DGMGRL Commands

Command	Effect
<code>VALIDATE FAR_SYNC</code>	Performs a comprehensive set of checks for a far sync instance.

7.1.3 DGMGRL Command Usage Notes

To use DGMGRL, the following must be true:

- The `DG_BROKER_START` dynamic initialization parameter is set to `TRUE`.
- To enable broker operations that require restarting instances without manual intervention, Oracle Net Services must be configured on each of the hosts that contain the primary and standby database instances. Specifically, the `listener.ora` file must contain static configuration information about the instance. The `GLOBAL_DBNAME` attribute must be set to `db_unique_name_DGMGRL.db_domain`. See [Section 2.2](#) for additional information.
- The connect identifier used while creating the configuration or adding a database, must be resolvable from any of the hosts in the configuration.
- You must have `SYSDG` or `SYSDBA` privileges to use the Oracle Data Guard command-line interface. Do not include `AS SYSDG` or `AS SYSDBA` on the `CONNECT` command. DGMGRL first attempts an `AS SYSDG` connection; if that fails, it then attempts an `AS SYSDBA` connection.
- If you specify more than one option on the command, you can specify the options in any order.
- A semicolon is required at the end of each DGMGRL command.
- Characters specified in a DGMGRL command string value are interpreted as lowercase characters, unless enclosed in double (") or single (') quotation marks. For example, `database` and `DatAbaSe` are equivalent, but `"database"` and `"DatAbaSe"` are not.
- You can use the backslash (\) to escape a single quotation mark ('), a double quotation mark ("), and the backslash character (\) itself if these characters appear in a character string.
- Some operations on a broker configuration may require that one or more databases be shut down and restarted. In most cases, DGMGRL will automatically shut down and restart a given database for you if the following are true:
 - The *instance-name* is the SID (this applies to Cloud Control as well as DGMGRL).
 - The broker must be able to connect to the database using the same credentials given in the last `CONNECT` command, even if the last `CONNECT` command was used to connect to another database.

Command Examples

Example 1

This example demonstrates how to connect to the DGMGRL command-line interface on a local system.

```
% dgmgrl
.
.
```

```
.  
Welcome to DGMGRL, type "help" for information.
```

```
DGMGRL> CONNECT sysdg;  
Password: password  
Connected as SYSDBG.
```

Example 2

This example demonstrates how to connect to the Oracle Data Guard (DGMGRL) command-line interface on a remote system.

```
DGMGRL> CONNECT sysdg@remote-stby;  
Password: password  
Connected as SYSDBG.
```

7.2 Exiting the Data Guard Command-Line Interface

When you are done working with the command-line interface and want to return to the operating system, enter the `EXIT` or `QUIT` command at the DGMGRL command prompt. For example:

```
DGMGRL> EXIT;
```

ADD DATABASE

Adds a standby database to an existing broker configuration.

Format

```
ADD DATABASE database-name
AS CONNECT IDENTIFIER IS connect-identifier
```

Command Parameters

database-name

The name that will be used by the broker to refer to this standby database. It must match (case-insensitive) the value of the corresponding database `DB_UNIQUE_NAME` initialization parameter.

connect-identifier

A fully specified connect descriptor or a name to be resolved by an Oracle Net Services naming method (for example, TNS). The value you specify is also used as the initial value of the `DGConnectIdentifier` database property.

Usage Notes

- To issue this command, you must connect to the primary database or to an enabled standby database that is already in the configuration.
- The broker uses the specified `connect-identifier` to communicate with the specified database from other databases. Therefore, you must ensure that the `connect-identifier` can be used to address the specified database from all databases in your configuration. For example, if TNS is used as the naming method, you must ensure that the `tnsnames.ora` file on every database and instance that is part of the configuration contains an entry for the `connect-identifier`. The connect identifier must resolve to the same connect descriptor. If the database that is being added is an Oracle RAC database, the `connect-identifier` provided here must reach all instances of the Oracle RAC, preferably with `FAILOVER` attributes set.

See Also: *Oracle Database Net Services Administrator's Guide*

- If the connection cannot be made, the broker does not add the new database to the configuration.
- You must clear any remote redo transport destinations on the standby database before it can be added to the configuration.

Command Example

Example 1

The following example shows how to add a database named `South_Sales`.

```
DGMGRL> ADD DATABASE South_Sales AS CONNECT IDENTIFIER IS South_Sales.example.com;
Database "South_Sales" added
```

ADD FAR_SYNC

Adds an existing far sync instance to an Oracle Data Guard broker configuration.

The far sync instance is disabled after creation. You must explicitly enable it before the broker can ship redo to and from it.

Format

```
ADD FAR_SYNC far_sync_instance_name AS CONNECT IDENTIFIER IS connect-identifier;
```

Command Parameters

far_sync_instance_name

The name that will be used by the broker to refer to this far sync instance. It must match (case-insensitive) the value of the corresponding far sync instance `DB_UNIQUE_NAME` initialization parameter.

connect-identifier

A fully specified connect descriptor or a name to be resolved by an Oracle Net Services naming method (for example, TNS). The value you specify is also used as the initial value of the `DGConnectIdentifier` property.

Usage Notes

- The far sync instance must already exist before you can add it to a broker configuration.
- You must clear any remote redo transport destinations on the far sync instance before it can be added to the configuration.

Command Example

Example 1

The following example adds a far sync instance named `chicago` to the configuration.

```
DGMGRL> ADD FAR_SYNC chicago AS CONNECT IDENTIFIER IS chicago.example.com;
```

CONNECT

Connects a given username to the specified configuration member.

Format

```
CONNECT username [@connect-identifier];
```

Command Parameters

username

Represents the username with which you want to connect to the configuration member. You will be prompted for a password after you enter a username and optionally, a connect-identifier.

connect-identifier

This parameter is optional. It is an Oracle Net Services connect identifier for the configuration member to which you want to connect. The exact syntax depends upon the Oracle Net Services communications protocol your Oracle installation uses.

Usage Notes

- The username and password must be valid for the configuration member to which you are trying to connect.

The username you specify must have the SYSDBG or SYSDBA privilege. Do not include AS SYSDBG or AS SYSDBA on the CONNECT command. DGMGRL first attempts an AS SYSDBG connection; if that fails, it then attempts an AS SYSDBA connection.
- If the CONNECT command returns an error, check to see that you specified a valid connect-identifier.

Command Examples

Example 1

This example connects to the default database or far sync instance on the local system.

```
DGMGRL> CONNECT sysdg;  
Password: password  
Connected as SYSDBG.
```

Example 2

This example connects to a remote configuration member whose connect-identifier is North_Sales.example.com.

```
DGMGRL> CONNECT sysdg@North_Sales.example.com;  
Password: password  
Connected as SYSDBG.
```

Example 3

This example connects to a configuration member using CONNECT '/' so that connection credentials are not visible on the command line:

```
DGMGRL> CONNECT /@North_Sales.example.com;
```

You must set up Oracle Wallet or SSL to use `CONNECT '/'`. By setting up Oracle Wallet or SSL, you can write a script to securely start and run the observer as a background job without specifying database credentials in the script.

CONVERT DATABASE

Converts a physical standby database to a snapshot standby database, or reverts the snapshot standby database back to a physical standby database.

A snapshot standby database is a fully updatable standby database. Like a physical or logical standby database, a snapshot standby database receives and archives redo data from a primary database. Unlike a physical or logical standby database, a snapshot standby database does not apply the redo data that it receives. The redo data received by a snapshot standby database is not applied until the snapshot standby is converted back into a physical standby database, after first discarding any local updates made to the snapshot standby database.

A snapshot standby database is best used in scenarios that require a temporary, updatable snapshot of a physical standby database. Note that because redo data received by a snapshot standby database is not applied until it is converted back into a physical standby, the time needed to perform a role transition is directly proportional to the amount of redo data that needs to be applied.

See *Oracle Data Guard Concepts and Administration* for additional information about snapshot standby databases.

Format

```
CONVERT DATABASE db_unique_name TO {SNAPSHOT | PHYSICAL} STANDBY;
```

Usage Notes

- A physical standby database cannot be converted to a snapshot standby database if it is the target of a fast-start failover. The `ORA-16668: operation cannot be performed on the fast-start failover target standby database error` will be returned.
- A physical standby database cannot be converted to a snapshot standby database if its `RedoRoutes` configurable property is set to non-NULL value.
- Use the DGMGRL [ADD DATABASE](#) command to import an existing snapshot standby database into an Oracle Data Guard broker configuration.
- A snapshot standby database cannot be the target of a switchover or a fast-start failover.
- A snapshot standby database can be the target of a manual failover if fast-start failover is disabled.
- You can use the `SHOW CONFIGURATION` or `SHOW DATABASE` command to verify the conversion result. For example:

```
DGMGRL> SHOW CONFIGURATION;

Configuration - DRSolution

Protection Mode: MaxPerformance
Databases:
  North_Sales - Primary database
  South_Sales - Snapshot standby database

Fast-Start Failover: DISABLED

Configuration Status:
```

- SUCCESS

■ After a snapshot standby database is converted back to a physical standby database, it will be in the default state for a physical standby database, APPLY-ON.

Command Examples

Example 1

Issue the following to convert a physical standby database to a snapshot standby database:

```
DGMGRL> CONVERT DATABASE 'South_Sales' to SNAPSHOT STANDBY;
Converting database "South_Sales" to a Snapshot Standby database, please wait...
Database "South_Sales" converted successfully
```

Example 2

Issue the following to convert the snapshot standby database back to a physical standby database:

```
DGMGRL> CONVERT DATABASE 'South_Sales' to PHYSICAL STANDBY;
Converting database "South_Sales" to a Physical Standby database, please wait...
Operation requires shutdown of instance "south_sales1" on database "South_Sales"
Shutting down instance "south_sales1"...
Database closed.
Database dismounted.
ORACLE instance shut down.
Operation requires startup of instance "south_sales1" on database "South_Sales"
Starting instance "south_sales1"...
ORACLE instance started.
Database mounted.
Continuing to convert database "South_Sales" ...
Database "South_Sales" converted successfully
```


CREATE CONFIGURATION

Creates a new broker configuration that includes the specified primary database.

Format

```
CREATE CONFIGURATION configuration-name AS
  PRIMARY DATABASE IS database-name
  CONNECT IDENTIFIER IS connect-identifier;
```

Command Parameters

configuration-name

A user-friendly name for the configuration you are creating. Valid names contain any alphanumeric characters. If spaces are included in the name, the name must be enclosed in double or single quotation marks. The name must consist of 30 or fewer bytes.

database-name

The name that will be used by the broker to refer to the primary database. It must match (case-insensitive) the value of the primary database `DB_UNIQUE_NAME` initialization parameter.

connect-identifier

A fully specified connect descriptor or a name to be resolved by an Oracle Net Services naming method (for example, TNS). The value you specify is also used as the initial value of the `DGConnectIdentifier` database property.

Usage Notes

- A broker configuration is a named collection of one or more databases that you want to manage as a group. You must specify a value for each of the command parameters. There are no default values.
- You must connect to the primary database to issue this command.
- The broker uses the specified `connect-identifier` to communicate with the specified database from other databases. Therefore, you must ensure that the `connect-identifier` can be used to address the specified database from all databases in your configuration. For example, if TNS is used as the naming method, you must ensure that the `tnsnames.ora` file on every database and instance that is part of the configuration contains an entry for the `connect-identifier`. The connect identifier must resolve to the same connect descriptor. If the database that is being added is an Oracle RAC database, the `connect-identifier` provided here must reach all instances of the Oracle RAC, preferably with `FAILOVER` attributes set.

See Also: *Oracle Database Net Services Administrator's Guide*

- To add standby databases after you create the broker configuration, use the [ADD DATABASE](#) command.
- You must clear any remote redo transport destinations on the primary database that do not have the `NOREGISTER` attribute, before a configuration can be created.

Command Example

Example 1

The following example creates a new broker configuration named DRSolution with a primary database named North_Sales.

```
DGMGRL> CREATE CONFIGURATION 'DRSolution' AS
> PRIMARY DATABASE IS 'North_Sales'
> CONNECT IDENTIFIER IS North_Sales.example.com;
Configuration "DRSolution" created with primary database "North_Sales"
```

DISABLE CONFIGURATION

Disables broker management of a configuration so that the configuration and all of its databases are no longer managed by the broker.

Format

```
DISABLE CONFIGURATION;
```

Command Parameters

None.

Usage Notes

- A disabled configuration and all of its constituent databases are no longer managed by the broker.
- The only way to disable broker management of the primary database is to use the `DISABLE CONFIGURATION` command.
- This command does not remove the broker configuration from the configuration file. See the [REMOVE CONFIGURATION](#) command for more information about removing the configuration.
- You can edit database properties and modify the configuration's protection mode while the configuration is disabled. However, any changes made to properties or to the protection mode will not take effect until the configuration is enabled.
- This command cannot be executed if fast-start failover is enabled.

Command Example

Example 1

The following example disables management of the broker configuration and all of its databases.

```
DGMGRL> DISABLE CONFIGURATION;  
Disabled.
```

DISABLE DATABASE

Disables broker management of the named standby database. This means that broker directed state changes will be disallowed for this database, and the broker will not monitor the database for health status or for monitorable properties.

Format

```
DISABLE DATABASE database-name;
```

Command Parameter

database-name

Name of the standby database to be disabled.

Usage Notes

- You cannot specify the name of a primary database.
- Use the `DISABLE CONFIGURATION` command to disable the primary and all standby databases.
- If the sole standby database is disabled, you have no failover option. This standby database is not viable for failover until it is reenabled.
- This command cannot be used to disable the fast-start failover target database when fast-start failover is enabled.

Command Example

Example 1

The following example shows how to disable a database named `South_Sales`.

```
DGMGRL> DISABLE DATABASE 'South_Sales';  
Disabled.
```

DISABLE FAR_SYNC

Disables broker management of a far sync instance.

Format

```
DISABLE FAR_SYNC far_sync_instance_name;
```

Command Parameter

far_sync_instance_name

The name of the far sync instance to be disabled.

Usage Notes

- A far sync instance that has its RedoRoutes property set cannot be disabled.

Command Example

Example 1

The following example disables broker management of a far sync instance named *chicago*.

```
DGMGRL> DISABLE FAR_SYNC 'chicago';
```

DISABLE FAST_START FAILOVER

Disables fast-start failover and prevents the observer from initiating a failover to the target standby database. See [Section 5.5.5, "Disabling Fast-Start Failover"](#) for additional information.

Format

```
DISABLE FAST_START FAILOVER [ FORCE ];
```

Command Parameters

None.

Usage Notes

- If the primary and target standby database have a network connection, use `DISABLE FAST_START FAILOVER` *without* the `FORCE` option to disable fast-start failover on all databases in the broker configuration. If errors occur during the disable operation, the broker returns an error message and stops the disable operation. You may need to reissue the `DISABLE FAST_START FAILOVER` command with the `FORCE` option to override the error conditions and disable fast-start failover on the database to which you are connected. See [Section 5.5.5, "Disabling Fast-Start Failover"](#) for more information.
- Use `DISABLE FAST_START FAILOVER` *with* the `FORCE` option when the network between the primary and target standby databases is disconnected or when the database upon which the command is received does not have a connection with the primary database. The `FORCE` option disables fast-start failover on the database to which you are connected, even when errors occur.
- Disabling fast-start failover with the `FORCE` option on a primary database that is disconnected from the observer and the target standby database does not prevent the observer from initiating a fast-start failover to the target standby database.
- You can disable fast-start failover while connected to any database in the broker configuration so long as connectivity exists between that database and the primary.
- If disabled by force at the target standby database and the connection subsequently resumes with the primary database, fast-start failover is disabled on all databases in the configuration.
- Disabling fast-start failover with the `FORCE` option while connected to the primary will disable fast-start failover on the target standby database if there is network connectivity between both databases.

Command Example

Example 1

The following example shows how to disable fast-start failover.

```
DGMGR> DISABLE FAST_START FAILOVER;  
Disabled.
```

Example 2

The following example uses the `FORCE` option which disables fast-start failover on the database to which you are connected.

```
DGMGR> DISABLE FAST_START FAILOVER FORCE;  
Disabled.
```

DISABLE FAST_START FAILOVER CONDITION

Allows a user to remove conditions for which a fast-start failover should be performed.

Format

DISABLE FAST_START FAILOVER CONDITION *value*;

Command Parameters

value

Possible values are any conditions for which a fast-start failover has been enabled.

Usage Notes

If the condition has not been set or if it is an unrecognized condition, then an error is raised.

Command Example

Example 1

```
DISABLE FAST_START FAILOVER CONDITION "Corrupted Controlfile";
```

This specifies that the detection of a corrupted controlfile does not automatically initiate an immediate fast-start failover.

EDIT CONFIGURATION (Property)

Changes the value of a property for the broker configuration.

Format

```
EDIT CONFIGURATION SET PROPERTY property-name = value;
```

Command Parameter

property-name

The name of a configuration property.

value

The new value for the property.

See Also: [Chapter 4](#) and [Chapter 8](#) for information about configuration properties

Usage Notes

- Issue this command while connected to the primary database or to any standby database in the broker configuration having connectivity to the primary database.
- Use the `SHOW CONFIGURATION` command to display the current property information for the configuration.

Command Example

Example 1

The following example shows how to set the `FastStartFailoverThreshold` configuration property to 90 seconds.

```
DGMGRL> EDIT CONFIGURATION SET PROPERTY FastStartFailoverThreshold=90;
```

EDIT CONFIGURATION (Protection Mode)

Edits the current protection mode setting for the broker configuration.

Format

EDIT CONFIGURATION SET PROTECTION MODE AS *protection-mode*;

Command Parameter

protection-mode

The data protection mode in which you want the configuration to run when the configuration is enabled. The possible protection modes are:

MAXPROTECTION
MAXAVAILABILITY
MAXPERFORMANCE

Usage Notes

- Before you use the EDIT CONFIGURATION command to set the protection mode to either MAXPROTECTION or MAXAVAILABILITY mode, ensure that at least one standby is configured to receive redo via SYNC or FASTSYNC mode if it receives redo directly from the primary. If the standby receives redo via a far sync instance, the far sync instance must be configured to receive redo via SYNC or FASTSYNC mode and the standby must be configured to receive redo via ASYNC mode.
- The following table shows the configuration protection modes and the minimum corresponding settings for redo transport services:

Protection Mode	Redo Transport	Standby Redo Log Files Needed?	Usable with Fast-Start Failover?
MAXPROTECTION	SYNC	Yes	No
MAXAVAILABILITY	SYNC or FASTSYNC	Yes	Yes
MAXPERFORMANCE	ASYNC	Yes	Yes

The default protection mode for the configuration is MAXPERFORMANCE.

See Also: [Chapter 4](#) for more information about the protection modes and redo transport services

- MAXAVAILABILITY or MAXPERFORMANCE is required in order to enable fast-start failover.
- This command cannot be executed if fast-start failover is enabled.
- If you want to avoid a restart of the primary database when upgrading from maximum performance mode to maximum protection mode, then first upgrade to maximum availability. Then wait for the PROTECTION_LEVEL column of V\$DATABASE to contain a value of MAXIMUM AVAILABILITY. Once the column contains that value, you can upgrade to maximum protection mode.
- Use the SHOW CONFIGURATION command to display the current protection mode for the configuration.

```
DGMGRL> SHOW CONFIGURATION;

Configuration - DRSolution

Protection Mode: MaxPerformance
Databases:
  North_Sales - Primary database
  South_Sales - Physical standby database

Fast-Start Failover: DISABLED

Configuration Status:
SUCCESS
```

If broker management of the configuration is disabled when you enter the `EDIT CONFIGURATION` command, the protection mode of the configuration does not take effect until the next time you enable the configuration with the `ENABLE CONFIGURATION` command.

Command Example

Example 1

The following example shows how to upgrade the broker configuration to the `MAXAVAILABILITY` protection mode.

Verify that standby redo log files are configured on the standby database and that the redo transport service is set to `SYNC`, for example:

```
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY 'LogXptMode'='SYNC';
Property "LogXptMode" updated

DGMGRL> EDIT CONFIGURATION SET PROTECTION MODE AS MAXAVAILABILITY;
Succeeded.
```

EDIT CONFIGURATION (RENAME)

Changes the configuration name.

Format

```
EDIT CONFIGURATION RENAME TO new-configuration-name ;
```

Command Parameter

new-configuration-name

The new name for the configuration.

Command Example

Example 1

The following example shows how to rename a configuration named DR_Sales to HA_Sales.

```
DGMGRL> show configuration

Configuration - DR_Sales

Protection Mode: MaxPerformance
Databases:
  North_Sales - Primary database
  South_Sales - Physical standby database

Fast-Start Failover: DISABLED

Configuration Status:
DISABLED

DGMGRL> edit configuration rename to "HA_Sales";
Succeeded.
DGMGRL> enable configuration
Enabled.
DGMGRL> show configuration

Configuration - HA_Sales

Protection Mode: MaxPerformance
Databases:
  North_Sales - Primary database
  South_Sales - Physical standby database

Fast-Start Failover: DISABLED

Configuration Status:
SUCCESS
```

EDIT CONFIGURATION RESET (Property)

Resets the specified configuration property to its default value.

Format

```
EDIT CONFIGURATION RESET PROPERTY property-name ;
```

Command Parameter

property-name

The name of an existing configuration property.

Usage Notes

- Issue this command while connected to the primary database or to any standby database in the broker configuration having connectivity to the primary database.
- Use the `SHOW CONFIGURATION` command to display the current property information for the configuration.

Command Example

Example 1

The following example shows how to reset the `BystandersFollowChange` property.

```
DGMGRL> EDIT CONFIGURATION RESET PROPERTY BystandersFollowChange;  
Succeeded.
```

EDIT DATABASE (Property)

Changes the name used by the broker for the specified database.

Format

```
EDIT DATABASE database-name  
SET PROPERTY property-name = value;
```

Command Parameters

database-name

The name of the database for which you want to change a property value.

property-name

The name of an existing database-specific property. If this is an Oracle RAC database, this property change affects all instances of the database.

See Also: [Chapter 4](#) and [Chapter 8](#) for information about properties.

value

The new value for the property.

Caution: This command can be used to change the value of an instance-specific property if and only if just one instance is known by the broker for the named database. An attempt to use this command to change an instance-specific property when the broker knows of multiple instances of the database will be rejected. It is recommended to only use `EDIT INSTANCE (property)` to change the value of an instance-specific property.

Command Examples

Example 1

Edit a configurable property at the database level.

```
DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY 'ArchiveLagTarget'=1200;  
Property "ArchiveLagTarget" updated
```

Example 2

Edit a configurable instance-specific property of a non-Oracle RAC database.

```
DGMGRL> EDIT DATABASE 'South_Sales' SET PROPERTY  
> 'StandbyArchiveLocation'='/archfs/arch/';  
Property "StandbyArchiveLocation" updated
```

Example 3

Edit a configurable instance-specific property of an Oracle RAC database. This will not succeed because it is not clear to which instance the property change should be applied.

```
DGMGRL> EDIT DATABASE 'North_Sales' SET PROPERTY
```

```
> 'StandbyArchiveLocation'='/archfs/arch/';  
Error: ORA-16587: ambiguous object specified to Data Guard broker Failed.
```

EDIT DATABASE (Rename)

Changes the name used by the broker to refer to the specified database.

Format

```
EDIT DATABASE database-name
      RENAME TO new-database-name;
```

Command Parameters

database-name

The name of the database that you want to change.

new-database-name

The new name of the database.

Usage Notes

- Use this command to track changes to the DB_UNIQUE_NAME initialization parameter for this database.

Caution: The *database-name* must always match the value for that database's DB_UNIQUE_NAME initialization parameter.

- This command can only be done when broker management of the database that you are renaming is disabled.

Command Example

Example 1

The following example shows how to edit and rename a database.

```
DGMGRL> DISABLE DATABASE 'South_Sales_typo';
Disabled.
```

```
DGMGRL> EDIT DATABASE 'South_Sales_typo' RENAME TO 'South_Sales';
Succeeded.
```

```
DGMGRL> ENABLE DATABASE 'South_Sales';
Enabled.
```

EDIT DATABASE (State)

Changes the state of the specified database.

Format

```
EDIT DATABASE database-name
SET STATE = state
[WITH APPLY INSTANCE = instance-name];
```

Command Parameters

database-name

The name of the database for which you want to change the state.

state

The state in which you want the database to be running. The possible states are:

TRANSPORT-ON (primary database only)
TRANSPORT-OFF (primary database only)
APPLY-ON (physical or logical standby database only)
APPLY-OFF (physical or logical standby database only)

instance-name

The name of the instance you want to become the apply instance if this is an Oracle RAC standby database.

Usage Notes

- If the target state is APPLY-ON and this database is currently a physical or logical standby database, the optional WITH APPLY INSTANCE clause specifies which instance will become the apply instance.
- If the target state is not APPLY-ON or if the database is currently in the primary role, the WITH APPLY INSTANCE clause is ignored even if it is specified.
- You cannot change the state of a snapshot standby database.
- All instances of an Oracle RAC database are affected by this database state change.

Command Example

The following examples show how to change the state of a database.

Example 1

```
DGMGRL> EDIT DATABASE 'South_Sales' SET STATE='APPLY-ON';
Succeeded.
```

EDIT DATABASE RESET (Property)

Resets the specified property for the named database back to its default value.

Format

```
EDIT DATABASE database-name RESET PROPERTY property-name ;
```

Command Parameter

database-name

The name of the database for which you want to reset the property value back to its default.

property-name

The name of an existing database-specific configurable property.

Command Example

Example 1

The following example shows how to reset the `NetTimeout` property for the database named `South_Sales`.

```
DGMGR> EDIT DATABASE 'South_Sales' RESET PROPERTY NetTimeout;  
Succeeded.
```

EDIT FAR_SYNC

Changes the name or properties of a far sync instance.

Format

```
EDIT FAR_SYNC far_sync_instance_name RENAME TO new_far_sync_instance_name;
```

```
EDIT FAR_SYNC far_sync_instance_name SET PROPERTY property_name = value;
```

Command Parameters

far_sync_instance_name

The name of the far sync instance for which you want to edit information. It must match (case-insensitive) the value of the corresponding database DB_UNIQUE_NAME initialization parameter.

new_far_sync_instance_name

The new name of the far sync instance.

property_name

The name of an existing far sync instance-specific configurable property.

Command Example

Example 1

The following example renames a far sync instance named `chicago` to the name `dallas`.

```
DGMGRL> DISABLE FAR_SYNC 'chicago_typo';  
EDIT FAR_SYNC 'chicago_typo' RENAME TO 'chicago';  
ENABLE FAR_SYNC 'chicago';
```

EDIT FAR_SYNC RESET (Property)

Resets the specified property for the named far sync instance to its default value.

Format

```
EDIT FAR_SYNC far_sync_instance_name RESET PROPERTY property-name;
```

Command Parameters

far_sync_instance_name

The name of the far sync instance for which you want to edit information. It must match (case-insensitive) the value of the corresponding database DB_UNIQUE_NAME initialization parameter.

property_name

The name of the property to be reset to its default value.

Command Example

Example 1

The following example shows how to reset the ReopenSecs property back to its default value for the far sync instance named dallas.

```
DGMGR> EDIT FAR_SYNC 'dallas' RESET PROPERTY ReopenSecs;
```

EDIT INSTANCE (AUTO PFILE)

Sets the name of the initialization parameter file for the specified instance.

Format

```
EDIT INSTANCE instance-name
      [ON DATABASE database-name]
      SET AUTO PFILE [= { initialization-file | OFF }];
```

Command Parameters

instance-name

The name of the instance (SID) for which you want to specify its initialization parameter file.

database-name

The name of the database to which the *instance-name* is associated.

initialization-file

Executes the startup operation for the instance when a subsequent broker operation requires the instance to be started automatically. If `SET AUTO PFILE` is set to `OFF`, automatic restart of that instance is disabled. When a subsequent operation needs to start that instance, you must start it manually. If you do not specify `SET AUTO PFILE` for the instance, the automatic startup operation looks for the initialization parameter file at the default location.

Usage Notes

- The *instance-name* can be unique across the configuration. If *instance-name* is not unique, you must specify both the *database-name* and the *instance-name* to fully identify the instance.
- `SET AUTO PFILE` is valid only for the duration of the current DGMGRL session. You must specify `SET AUTO PFILE` again if you quit and reenter DGMGRL.

Command Example

Example 1

The following example shows how to change the parameter file used to start an instance of a database.

```
DGMGRL> EDIT INSTANCE 'south_sales1' ON DATABASE 'South_Sales'
> SET AUTO PFILE='initsales1.ora';
Instance 'south_sales1' updated
```

EDIT INSTANCE (Property)

Changes the value of an instance-specific property for the specified instance(s).

Format

```
EDIT INSTANCE
  { {instance-name [ON {DATABASE database-name | FAR_SYNC far-sync-name}] } } |
  { * ON {DATABASE database-name | FAR_SYNC far-sync-name } } }
SET PROPERTY property-name = value;
```

Command Parameters

instance-name

The name of the instance (SID) for which you want to change an instance-specific property's value. If an asterisk is specified for *instance-name*, then either the ON DATABASE clause or the ON FAR_SYNC clause must also be specified and this change will be applied to the specified property for each instance associated with that database or far sync. See [Chapter 8](#) for the instance-specific properties that can be changed using this command.

database-name

The name of the database with which the *instance-name* is associated. This must be specified if an asterisk is specified for *instance-name*, or if *instance-name* is not unique across the configuration

far-sync-name

The name of the far sync with which the *instance-name* is associated.

The option of specifying a far sync is available only in Oracle Database 12c release 1 (12.1.0.2) and later.

property-name

The name of the instance-specific property for which you want to set a new value.

See Also: [Chapter 4](#) and [Chapter 8](#) for information about properties.

value

The new value for the property.

Usage Notes

- The *instance-name* can be unique across the configuration. If *instance-name* is not unique, you must specify both the *database-name* (or *far-sync-name*) and the *instance-name* to fully identify the instance.
- This command cannot be used to change a database-specific property.

Command Examples

Example 1

Edit an instance-specific property.

```
DGMGRL> EDIT INSTANCE 'north_sales1' ON DATABASE 'North_Sales'  
> SET PROPERTY 'StandbyArchiveLocation'='/archfs/arch/';  
Property "StandbyArchiveLocation" updated.
```

Example 2

Edit a database-specific property. This will not be allowed.

```
DGMGRL> EDIT INSTANCE 'north_sales1' ON DATABASE 'North_Sales'  
> SET PROPERTY 'LogXptMode'='SYNC';  
Error: ORA-16586: cannot change database property with EDIT INSTANCE command  
  
Failed.
```

Example 3

Change the value of an instance-specific property for all instances of an Oracle RAC database:

```
DGMGRL> EDIT INSTANCE * ON DATABASE 'North_Sales'  
> SET PROPERTY LogArchiveTrace=4095;
```

Example 4

Change the value of an instance-specific property for all instances of an Oracle RAC far sync instance:

```
DGMGRL> EDIT INSTANCE * ON FAR_SYNC 'dallas' SET PROPERTY LogArchiveTrace=4095;
```

EDIT INSTANCE RESET (Property)

Resets an instance-specific property for the specified instance(s) to its default value.

Format

```
EDIT INSTANCE
{ {instance-name [ON {DATABASE database-name | FAR_SYNC far-sync-name}] } |
{* ON {DATABASE database-name | FAR_SYNC far-sync-name} } }
RESET PROPERTY property-name;
```

Command Parameters

instance-name

The name of the instance (SID) for which you want to reset an instance-specific property's value to the default. If an asterisk is specified for instance-name, the ON DATABASE clause must also be specified and the property will be reset to the default for each instance associated with that database.

database-name

The name of the database with which the instance-name is associated. This must be specified if an asterisk is specified for instance-name, or if instance-name is not unique across the configuration.

property-name

The name of the instance-specific property that you want to reset to its default value.

See Also: [Chapter 4](#) and [Chapter 8](#) for information about properties.

value

The new value for the property.

Usage Notes

- The *instance-name* can be unique across the configuration. If *instance-name* is not unique, you must specify both the *database-name* and the *instance-name* to fully identify the instance.
- This command cannot be used to change a database-specific property.

Command Examples

Example 1

The following command shows how to reset the LogArchiveTrace property for the specific instance south1 on the database named South_Sales.

```
DGMGRL> EDIT INSTANCE 'south1' ON DATABASE 'South_Sales'
> RESET PROPERTY LogArchiveTrace;
Succeeded.
```


Example 2

The following example shows how to reset the LogArchiveTrace property for all instances of the database named South_Sales.

```
DGMGRL> EDIT INSTANCE * ON DATABASE 'South_Sales' RESET PROPERTY LogArchiveTrace;  
Succeeded.
```

ENABLE CONFIGURATION

Enables the broker to actively manage the broker configuration including all of its databases.

Format

```
ENABLE CONFIGURATION;
```

Command Parameters

None.

Usage Notes

- Use this command to enable broker management of the primary database and all members of the configuration, if these members are not explicitly disabled by the user.
- To issue this command, you must connect to a database whose control file role is primary.
- By default, broker management of the configuration's databases is enabled in the `TRANSPORT-ON` state with redo transport services turned on at the primary database and `APPLY-ON` with log apply services started at the standby databases. Far sync instances will be enabled such that they receive redo data and send redo data. You can change the state of a database using the [EDIT DATABASE \(State\)](#) command, but not when the database or the entire configuration is disabled. You cannot change the state of a far sync instance.
- Use the [SHOW CONFIGURATION](#) command to display information about the configuration.
- Use this command to update the roles stored in the broker configuration if a failover or switchover was performed using SQL*Plus instead of DGMGRL or Cloud Control.

Command Example

Example 1

The following example enables management of a broker configuration.

```
DGMGRL> ENABLE CONFIGURATION;  
Enabled.
```

ENABLE DATABASE

Enables broker management of the specified standby database.

Caution: Do not issue the `ENABLE DATABASE` command on a standby database that needs to be reinstated. See [Section 5.4.3](#) for more details.

Format

`ENABLE DATABASE database-name;`

Command Parameter

database-name

The name of the standby database for which you want to enable broker management.

Usage Notes

- You must connect to the primary database or to an already enabled standby database to issue this command.
- A standby database may have been disabled by the broker as a consequence of a prior failover or switchover operation. See [Section 5.4.3](#) to understand how the database can be reinstated or re-created.
- By default, broker management of the physical or logical standby database is enabled in the `APPLY-ON` state with log apply services enabled. You can change the state of the standby database using the [EDIT DATABASE \(State\)](#) command, but only when the database is enabled.
- Use the [SHOW DATABASE](#) command to display information about the database.
- For an Oracle RAC database, only one instance is required to be started and mounted for this command to succeed.

Command Example

Example 1

The following example shows how to enable a database named `South_Sales`.

```
DGMGRL> ENABLE DATABASE 'South_Sales';
Enabled.
```

ENABLE FAR_SYNC

Enables broker management of the specified far sync instance.

Format

```
ENABLE FAR_SYNC far_sync_instance_name ;
```

Command Parameter

far_sync_instance_name

The name of the far sync instance for which you want to enable broker management.

Command Example

Example 1

The following example enables broker management of a far sync instance named dallas.

```
DGMGRL> ENABLE FAR_SYNC 'dallas';
```

ENABLE FAST_START FAILOVER

Enables the broker to fail over to a specifically-chosen standby database in the event of loss of the primary database, without requiring you to perform any manual steps to invoke the failover. See [Section 5.5.2, "Enabling Fast-Start Failover"](#) for complete information.

Format

```
ENABLE FAST_START FAILOVER;
```

Command Parameters

None.

Usage Notes

- The prerequisites described in [Section 5.5.1](#) must be met before you issue this command to enable fast-start failover.
- Issuing the `ENABLE FAST_START FAILOVER` command does not trigger a failover, it only allows the observer that is monitoring the configuration to initiate a fast-start failover if conditions warrant a failover.
- You can enable fast-start failover while connected to any database in the broker configuration.
- If you do not start the observer after you have enabled fast-start failover, the `ORA-16819` warning is displayed for the primary and target standby databases. For example:

```
DGMGRL> SHOW DATABASE 'South_Sales';
Database - South_Sales

Role:                PRIMARY
Intended State:      TRANSPORT-ON
Instance(s):
  south_sales1

Database Warning(s):
  ORA-16819: fast-start failover observer not started

Database Status:
WARNING
```

- To enable fast-start failover for a broker configuration with multiple standby databases, the `FastStartFailoverTarget` configuration property on the primary database must be set to point to the desired target standby database. Both the primary database and the target standby database must have:
 - Standby redo logs configured
 - Redo transport must be properly configured at both databases for the configured protection mode
 - Flashback Database enabled on both the primary and standby databases
 Step 2 in [Section 5.5.2](#) and [Section 8.3.10](#) provide more information about the `FastStartFailoverTarget` configuration property.

- Once you have enabled fast-start failover, you must comply with the restrictions described in [Section 5.5.2.2, "Restrictions When Fast-Start Failover is Enabled"](#).

Command Example

Example 1

The following example enables fast-start failover.

```
DGMGRL> ENABLE FAST_START FAILOVER;  
Enabled.
```

Example 2

The following example shows that fast-start failover was successfully enabled when the configuration is operating in maximum performance mode.

```
DGMGRL> SHOW FAST_START FAILOVER;
```

```
Fast-Start Failover: ENABLED
```

```
Threshold: 30 seconds  
Target: South_Sales  
Observer: (none)  
Lag Limit: 30 seconds  
Shutdown Primary: TRUE  
Auto-reinstate: TRUE  
Observer Reconnect: (none)  
Observer Override: FALSE
```

```
Configurable Failover Conditions
```

```
Health Conditions:  
Corrupted Controlfile YES  
Corrupted Dictionary YES  
Inaccessible Logfile NO  
Stuck Archiver NO  
Datafile Offline YES
```

```
Oracle Error Conditions:  
(none)
```

ENABLE FAST_START FAILOVER CONDITION

Specifies additional conditions for which a fast-start failover should be performed.

Format

ENABLE FAST_START FAILOVER CONDITION *value*;

Command Parameters

value

Possible values are those described in the `SHOW FAST_START FAILOVER` command as health conditions. The Oracle error `ORA-00240` can also be named as a condition by specifying `240` as the value.

Usage Notes

- [Table 7-2](#) lists some examples of health conditions maintained by the database health-check facility.
- An error is raised if the specified value is not recognized or if the condition has already been set.

Table 7-2 Examples of Health Conditions

Health Condition	Description
"Datafile Offline"	Data file offline due to a write error. This condition is enabled by default.
"Corrupted Controlfile"	Corrupted controlfile. This condition is enabled by default.
"Corrupted Dictionary"	Dictionary corruption of a critical database object. This condition is enabled by default.
"Inaccessible Logfile"	LGWR is unable to write to any member of a log group due to an I/O error.
"Stuck Archiver"	Archiver is unable to archive a redo log because device is full or unavailable.

- You can display these configurable conditions with the `SHOW FAST_START FAILOVER` command.

Command Example

Example 1

```
ENABLE FAST_START FAILOVER CONDITION "Corrupted Controlfile";
```

This specifies that a fast-start failover should be done if a corrupted controlfile is detected.

Example 2

```
ENABLE FAST_START FAILOVER CONDITION 240;
```

This specifies that a fast-start failover should be done if an `ORA-00240` error is raised.

EXIT

Exits (quits) the command-line interface.

Format

EXIT;

Command Parameters

None.

Usage Notes

- This command has the same effect as the [QUIT](#) command.
- A database connection is not required to execute this command. However, if you are connected, this command breaks the connection.

Command Example

Example 1

The following example demonstrates how to exit (quit) the command-line interface.

```
DGMGRL> EXIT;
```

FAILOVER

Invokes a failover that transitions the named (target) standby database into the role of a primary database. This type of failover is referred to as *manual failover*. See [Section 5.4, "Manual Failover"](#) for more information.

Note: Because a failover results in a transition of a standby database to the primary role, it should be performed when the primary database has failed or is unreachable and cannot be recovered in a timely manner. Failover may or may not result in data loss depending on the protection mode in effect at the time of the failover and whether the target standby database was synchronized with the primary database.

Use the `SWITCHOVER` command if the primary database has not failed and you want the current primary database and a standby database to switch roles with no data loss.

Format

```
FAILOVER TO database-name
[ IMMEDIATE ];
```

Command Parameters

database-name

The name of a physical, logical, or snapshot standby database that you want to fail over to the primary database role.

Usage Notes

- Always try to perform a complete failover first unless Redo Apply has stopped at the failover target due to an `ORA-752` or `ORA-600 [3020]` error. If one of these errors has occurred, then before proceeding follow the guidelines in "Resolving `ORA-752` or `ORA-600 [3020]` During Standby Recovery" in My Oracle Support Note 1265884.1 at <http://support.oracle.com>. An immediate failover should only be performed when a complete failover is unsuccessful or in the error case just noted.
- The specified standby database must be enabled before the primary database fails. However, an enabled standby database that was shut down can be a candidate for the failover operation. In this case, restart the standby database using `DGMGRL STARTUP` command, then issue the `FAILOVER` command.
- The failover operates on the specified standby database and changes its role to a primary database. Bystander standby databases (those not involved in the failover) remain in the standby role.
- Before you issue the `FAILOVER` command, verify that you are connected to the standby database that will become the new primary database. If necessary, issue a `CONNECT` command to connect to the standby database to which you want to failover.

- If the `FAILOVER` command is issued without any options, the standby database chosen as the failover target applies all unapplied redo it has received before changing to the primary role. This is referred to as a complete failover.
- If the broker configuration is operating in maximum protection mode, a manual failover operation will force the protection mode to be maximum performance. The redo transport service settings are unaffected. You need to restore the desired protection mode for the resulting configuration after the failover operation.

Note: With fast-start failover, the broker preserves the protection mode that was in effect prior to the failover. The broker configuration cannot be in maximum protection mode while fast-start failover is enabled.

- If the `FAILOVER` command is issued with the `IMMEDIATE` option, no attempt is made to apply any unapplied redo that has been received. This option more likely results in lost application data even when standby redo log files are configured on the standby database. Additionally, any remaining standby databases in the configuration cannot function as such until they are reinstated or re-created. See [Section 5.4.3](#) for more information.
- You can perform a manual failover or set up the broker to perform a fast-start failover. See the `ENABLE FAST_START FAILOVER` command for information about allowing the broker to automatically invoke failover, when conditions warrant a failover.
- If fast-start failover is enabled, you can perform a *complete* manual failover only to the fast-start failover target standby database and only if the fast-start failover target standby database is synchronized with, or within the lag limit of, the primary database, and only when the observer is started. You cannot perform an *immediate* manual failover when fast-start failover is enabled.
- If Flashback Database was enabled on the former (failed) primary database prior to the failover, the former primary database can be reinstated using the broker's `REINSTATE` command (see the `REINSTATE DATABASE` command on page 7-50).

If failover was performed to a physical standby database, any other physical standby databases that were disabled by the failover can be reinstated if Flashback Database was enabled on the standby database and there are sufficient flashback logs available. See [Section 5.4.3, "Reenabling Disabled Databases After a Role Change"](#) for step-by-step instructions.

- The original primary database can only participate in the configuration as a standby database after it is reinstated or re-created.

Caution: You should shut down the original primary database if it still has any active instances running prior to failing over.

See Also: [Section 5.4.3](#) about reenabling the original primary database so that it could serve as a standby database to the primary database

Command Examples

Example 1

The following example performs a failover in which the standby database, South_Sales, transitions to the primary role:

```
DGMGRL> FAILOVER TO 'South_Sales';
Performing failover NOW, please wait...
Failover succeeded, new primary is "South_Sales"

DGMGRL> SHOW CONFIGURATION;
Configuration - DRSolution

Protection Mode: MaxPerformance
Databases:
  South_Sales - Primary database
  North_Sales - Physical standby database (disabled)
    ORA-16661: the standby database needs to be reinstated

Fast-Start Failover: DISABLED

Configuration Status:
WARNING
```

HELP

Displays online help for the Data Guard command-line interface.

Format

```
HELP [command_name];
```

Command Parameter

topic

The topic for which you want to display help information. If you do not specify a topic, the command lists all of the topics and the format. The following commands are available:

```
add      Adds a standby database to the broker configuration
connect  Connects to an Oracle database instance
convert  Converts a database from one type to another
create   Creates a broker configuration
disable  Disables a configuration, a database, or fast-start failover
edit     Edits a configuration, database, or instance
enable   Enables a configuration, a database, or fast-start failover
exit     Exits the program
failover Changes a standby database to be the primary database
help     Displays description and syntax for a command
quit     Exits the program
reinstat Changes a database marked for reinstatement into a viable standby
rem      Comment to be ignored by DGMGRL
remove   Removes a configuration, database, or instance
show     Displays information about a configuration, database, or instance
shutdown Shuts down a currently running Oracle database instance
sql      Executes a SQL statement
start    Starts the fast-start failover observer
startup  Starts an Oracle database instance
stop     Stops the fast-start failover observer
switchover Switches roles between a primary and standby database
validate Performs an exhaustive set of validations for a database
```

Enter `help command_name` to see syntax for individual commands.

Usage Notes

- A database connection is not required to execute this command.

Command Examples

Example 1

The following example gets help on the EDIT commands.

```
DGMGRL> HELP EDIT
```

QUIT

Quits (exits) the Data Guard command-line interface.

Format

QUIT;

Command Parameters

None.

Usage Notes

- This command has the same effect as the [EXIT](#) command.
- A database connection is not required to execute this command. However, if you are connected, this command breaks the connection.

Command Example

Example 1

The following example shows how to quit (exit) the command-line interface.

```
DGMGRL> QUIT;
```

REINSTATE DATABASE

Reinstates a database as a new standby database in the broker configuration for the current primary database.

Format

```
REINSTATE DATABASE database-name;
```

Command Parameter

database-name

The name of the database that is to be reinstated in the broker configuration.

Usage Notes

- If the conditions for reinstatement described in [Section 5.5.8](#) are not satisfied, the reinstatement will fail with an appropriate error status and the specified database will remain disabled.
- If the *database-name* specified is that of the old primary and fast-start failover is enabled, the old primary database will be reinstated as a standby to the new primary, and the fast-start failover environment will be updated to reflect the availability of the new standby database. It will accept redo data from the new primary database and be the target of a fast-start failover should the new primary database fail. Reinstatement occurs automatically if the observer is running unless the `FastStartFailoverAutoReinstate` configuration property is set to `FALSE`.
- This command does not require that fast-start failover be enabled. It can be used to reinstate an old primary database after a complete manual failover has been performed. It can also be used to reinstate a bystander standby database that had been disabled after either a complete or immediate failover.
- Issue this command while connected to any database in the broker configuration, except the database that is to be reinstated.

Command Examples

Example 1

The following example reinstates the `North_Sales` database as a standby database in the broker configuration.

```
DGMGRL> REINSTATE DATABASE 'North_Sales';  
Reinstating database "North_Sales", please wait...  
Reinstatement of database "North_Sales" succeeded
```

REMOVE CONFIGURATION

Removes the Oracle Data Guard broker configuration and ends broker management of all members in the configuration.

Format

```
REMOVE CONFIGURATION [ PRESERVE DESTINATIONS ];
```

Command Parameters

None.

Usage Notes

- When you remove a broker configuration, management of all of the members associated with that configuration is disabled.
- By default, the command removes the corresponding broker settings of the LOG_ARCHIVE_DEST_n initialization parameter on the primary database and the LOG_ARCHIVE_CONFIG initialization parameters on all members of the configuration. To preserve these settings, use the PRESERVE DESTINATIONS option.
- This command does not remove or affect the actual primary or standby database instances, databases, far sync instances, data files, control files, initialization parameter files, server parameter files, or log files of the underlying Oracle Data Guard configuration.
- You cannot remove the configuration when fast-start failover is enabled.

Command Example

The following examples show a successful and an unsuccessful REMOVE CONFIGURATION command.

Example 1 Successful REMOVE CONFIGURATION Command

The following command shows how to remove configuration information from the configuration file.

```
DGMGRL> REMOVE CONFIGURATION;
Removed configuration
DGMGRL> SHOW CONFIGURATION;
Error: ORA-16532: Data Guard broker configuration does not exist
```

Configuration details cannot be determined by DGMGRL

Example 2 Unsuccessful REMOVE CONFIGURATION Command

The following command is unsuccessful because fast-start failover is enabled.

```
DGMGRL> REMOVE CONFIGURATION;
Error: ORA-16654: fast-start failover is enabled
```

Failed.

```
DGMGRL> SHOW CONFIGURATION;
```

Configuration - DRSolution

REMOVE CONFIGURATION

Protection Mode: MaxAvailability

Databases:

North_Sales - Primary database

South_Sales - (*) Physical standby database

Fast-Start Failover: ENABLED

Configuration status:

SUCCESS

REMOVE DATABASE

Removes the specified standby database from the broker configuration and terminates broker management of that standby database.

Format

```
REMOVE DATABASE database-name [ PRESERVE DESTINATIONS ];
```

Command Parameter

database-name

The name of the standby database that you want to remove from the broker configuration.

Usage Notes

- An error is returned if you specify the name of the primary database in the broker configuration.
- By default, this command removes all references to the specified database from all redo transport initialization parameters at each member of the configuration. To preserve these settings, use the `PRESERVE DESTINATIONS` option.
- This command cannot be executed if fast-start failover is enabled and *database-name* specifies the name of the target standby database.

Command Example

Example 1

The following example shows how to remove a database from the Oracle Data Guard broker configuration.

```
DGMGRL> SHOW CONFIGURATION;

Configuration - DRSolution

Protection Mode: MaxPerformance
Databases:
  North_Sales - Primary database
  South_Sales - Physical standby database

Fast-Start Failover: DISABLED

Configuration status:
SUCCESS

DGMGRL> REMOVE DATABASE 'South_Sales';
Removed database "South_Sales" from the configuration.

Configuration - DRSolution

Protection Mode: MaxPerformance
Databases:
  North_Sales - Primary database
```

Fast-Start Failover: DISABLED

Configuration status:
SUCCESS

REMOVE FAR_SYNC

Removes a far sync instance from an Oracle Data Guard broker configuration.

Format

```
REMOVE FAR_SYNC far_sync_instance_name;
```

Command Parameters

far_sync_instance_name

The name of the far sync instance to be removed from the broker configuration. It must match (case-insensitive) the value of the corresponding database `DB_UNIQUE_NAME` initialization parameter.

Usage Notes

- A far sync instance that has its `RedoRoutes` property set cannot be removed.

Command Example

Example 1

The following example removes a far sync instance named `dallas` from the broker configuration.

```
DGMGRL> REMOVE FAR_SYNC 'dallas';
```

REMOVE INSTANCE

Removes the specified instance from the broker configuration.

Format

```
REMOVE INSTANCE instance-name  
[ON { DATABASE | FAR_SYNC } object-name];
```

Command Parameters

instance-name

The name of the instance (SID) that you want to remove from the broker configuration.

object-name

The name of the database or the far sync with which the *instance-name* is associated. The option of specifying a far sync is available only in Oracle Database 12c Release 1 (12.1.0.2) and later.

Usage Notes

- The broker automatically adds started instances to the broker configuration. However, the broker does not automatically remove instances from the database. The `REMOVE INSTANCE` command can be used to manually remove any instance that no longer exists from the configuration.
- If the *instance-name* is not unique within the configuration, then you must specify both the database-name or far sync-name, and the *instance-name* to fully identify the instance.
- This command is rejected for an instance that is currently active in the broker configuration.
- This command is rejected if this is the only instance currently associated with a database or far sync.

Command Example

Example 1

The following example shows how to remove an instance of the database.

```
DGMGRL> REMOVE INSTANCE 'south_sales3' ON DATABASE 'South_Sales';  
Removed instance "south_sales3" from the database "South_Sales"
```

SHOW CONFIGURATION

Displays a summary and status of the broker configuration. The summary lists all members included in the broker configuration and other information pertaining to the broker configuration itself, including the fast-start failover status.

Format

```
SHOW CONFIGURATION [VERBOSE | property-name];
```

Command Parameters

property-name

The name of the property for which you want to display summary information.

See [Chapter 8, "Oracle Data Guard Broker Properties"](#) for complete information about properties.

Usage Notes

- Use the `SHOW CONFIGURATION VERBOSE` command (or the `SHOW FAST_START FAILOVER` command) to show the properties related to fast-start failover.
- You can optionally specify either `VERBOSE` or `property-name`, but not both.
- The `SHOW CONFIGURATION` command displays the status of the configuration and its members as of the last time the health was evaluated. (The health of the configuration and its members is evaluated once a minute.)

Specifying the `VERBOSE` keyword forces an immediate health evaluation of the configuration and its members before the health information is displayed.

Command Examples

Example 1 Showing a Summary of the DRSolution Configuration

The following example provides a summary of the `DRSolution` configuration for which fast-start failover is disabled. The output shows a far sync instance named `FS` in the broker configuration. The `North_Sales` database is shipping to `FS`, and `FS` is shipping to `South_Sales`.

```
DGMGRL> SHOW CONFIGURATION;

Configuration - DRSolution

Protection Mode: MaxAvailability
Members:
North_Sales - Primary database
FS- Far sync instance
South_Sales- Physical standby database

Fast-Start Failover: DISABLED

Configuration Status:
SUCCESS (status updated 20 seconds ago)
```

Example 2 Showing Detailed Description of the DRSolution Configuration

The following example provides detailed information about the DRSolution configuration, including configuration properties, and fast-start failover-related information:

```
DGMGRL> SHOW CONFIGURATION VERBOSE;

Configuration - DRSolution

Protection Mode: MaxAvailability
Members:

North_Sales - Primary database
  FS- Far sync instance
  South_Sales- (*) Physical standby database

(*) Fast-Start Failover target

Properties:
FastStartFailoverThreshold      = '30'
OperationTimeout                = '30'
TraceLevel                      = 'USER'
FastStartFailoverLagLimit       = '30'
CommunicationTimeout            = '180'
ObserverReconnect               = '0'
FastStartFailoverAutoReinstate  = 'TRUE'
FastStartFailoverPmyShutdown    = 'TRUE'
BystandersFollowRoleChange      = 'ALL'
ObserverOverride                 = 'FALSE'
ExternalDestination1            = ''
ExternalDestination2            = ''
PrimaryLostWriteAction          = 'CONTINUE'

Fast-Start Failover: ENABLED

Threshold:      30 seconds
Target:         South_Sales
Observer:       observer.example.com
Lag Limit:      30 seconds (not in use)
Shutdown Primary: TRUE
Auto-reinstate: TRUE
Observer Reconnect: (none)
Observer Override: FALSE

Configuration Status:
WARNING
```

SHOW CONFIGURATION WHEN PRIMARY IS

Displays the redo transport configuration that would be in effect if the specified database were the primary database. The display lists all members, but unlike `SHOW CONFIGURATION`, it does not provide status or any other configuration information. (This command is available in Oracle Database 12c Release 1 (12.1.0.2) and later.)

Format

`SHOW CONFIGURATION WHEN PRIMARY IS database-name;`

Command Parameters

database-name

The name of the database for which you want to see what the redo transport configuration would be if it were the primary database.

Usage Notes

- Use the `SHOW CONFIGURATION WHEN PRIMARY IS` command to show the redo transport configuration that would be in effect if the specified database were the primary database. You can use this information to identify ahead of time any redo transport configurations that would be incorrect after a role change.

Command Examples

Example 1 Showing the DRSolution configuration before and after a role change to South_Sales

The following example provides a summary of the DRSolution configuration before and after a role change to the `South_Sales` database.

```
DGMGRL> SHOW CONFIGURATION;
```

```
Configuration - DRSolution
```

```
Protection Mode: MaxAvailability
```

```
Members:
```

```
North_Sales - Primary database
```

```
North_FS    - Far Sync
```

```
South_Sales - Physical standby database
```

```
Members Not Receiving Redo:
```

```
South_FS    - Far Sync
```

```
Fast-Start Failover: DISABLED
```

```
Configuration Status:
```

```
SUCCESS
```

```
DGMGRL> SHOW CONFIGURATION WHEN PRIMARY IS 'South_Sales';
```

```
Configuration when South_Sales is primary - DRSolution
```

```
Members:
```

```
South_Sales - Primary database
```

```
South_FS      - Far Sync
North_Sales   - Physical standby database
```

```
Members Not Receiving Redo:
North_FS - Far Sync
```

These displays are based upon the RedoRoutes property being set as follows for each member:

```
DGMGRL> SHOW DATABASE 'North_Sales' RedoRoutes;
RedoRoutes = '(LOCAL : North_FS)'
```

```
DGMGRL> SHOW FAR_SYNC 'North_FS' RedoRoutes;
RedoRoutes = '(North_Sales : South_Sales)'
```

```
DGMGRL> SHOW DATABASE 'South_Sales' RedoRoutes;
RedoRoutes = '(LOCAL : South_FS)'
```

```
DGMGRL> SHOW FAR_SYNC 'South_FS' RedoRoutes;
RedoRoutes = '(South_Sales : North_Sales)'
```

SHOW DATABASE

Displays information or property values of the specified database and its instances.

Format

```
SHOW DATABASE [VERBOSE] database-name [property-name];
```

Command Parameters

database-name

The name of the database for which you want to display information. The `VERBOSE` keyword, if used, must come before the *database-name* or an error is returned.

property-name

The name of the property for which you want to display a value. If a property name is specified, the output shows only the specified property (not all properties of the database), regardless of whether or not the `VERBOSE` keyword is specified.

See Also: [Chapter 4](#) and [Chapter 8](#) for information about properties.

Usage Notes

- The `SHOW DATABASE` command shows a brief summary of the database. `SHOW DATABASE VERBOSE` shows properties of the database in addition to the brief summary. They both show the status of the database.
- The `SHOW DATABASE VERBOSE` command shows database-specific properties and instance-specific properties. For a non-Oracle RAC database, the values of the instance-specific properties are those of the only instance of the database. For an Oracle RAC database, the values of the instance-specific properties will not be shown, although the property names are still listed. To see the instance-specific values of these properties, use the `SHOW INSTANCE` command.
- The properties that the `SHOW DATABASE VERBOSE` command shows depend on the database role and the configuration composition:
 - For the primary database, properties specific to physical or snapshot standby databases are shown only if there is at least one physical or snapshot standby database in the configuration. The properties specific to logical standby databases are shown only if there is at least one logical standby database in the configuration.
 - For physical and snapshot standby databases, properties specific to logical standby databases are not shown.
 - For logical standby databases, properties specific to physical and snapshot standby databases are not shown.
- This command is rejected if you use the `SHOW DATABASE database-name property-name` command to show an instance-specific property in an Oracle RAC database.

Command Examples

Example 1

Shows database information in an abbreviated format.

```
DGMGRL> SHOW DATABASE South_Sales;
```

```
Database - South_Sales
```

```
Role:                PHYSICAL STANDBY
Intended State:      APPLY-ON
Transport Lag:       (unknown)
Apply Lag:           0 seconds (computed 0 seconds ago)
Apply Rate:          1.73 MByte/s
Real Time Query:    OFF
Instance(s):
    south_sales1
```

```
Database Status:
SUCCESS
```

Example 2

Shows database information in an extended format.

```
DGMGRL> SHOW DATABASE VERBOSE 'South_Sales';
```

```
Database - South_Sales
```

```
Role:                PHYSICAL STANDBY
Intended State:      OFFLINE
Transport Lag:       (unknown)
Apply Lag:           (unknown)
Apply Rate:          (unknown)
Real Time Query:    OFF
Instance(s):
    south_sales1
```

```
Properties:
```

```
DGConnectIdentifier      = 'South_Sales.example.com'
ObserverConnectIdentifier = ''
LogXptMode               = 'ASYNC'
RedoRoutes               = ''
DelayMins                = '0'
Binding                  = 'optional'
MaxFailure                = '0'
MaxConnections           = '1'
ReopenSecs               = '300'
NetTimeout               = '30'
RedoCompression          = 'DISABLE'
LogShipping              = 'ON'
PreferredApplyInstance   = ''
ApplyInstanceTimeout     = '0'
ApplyLagThreshold        = '0'
TransportLagThreshold    = '0'
TransportDisconnectedThreshold = '0'
ApplyParallel            = 'AUTO'
StandbyFileManagement    = 'AUTO'
ArchiveLagTarget         = '0'
LogArchiveMaxProcesses   = '5'
LogArchiveMinSucceedDest = '1'
```

```
DbFileNameConvert          = 'dbs/t, dbs/ct, dbs/bt, dbs/ct'
LogFileNameConvert         = 'dbs/t, dbs/ct, dbs/bt, dbs/ct'
FastStartFailoverTarget    = ''
InconsistentProperties      = '(monitor)'
InconsistentLogXptProps    = '(monitor)'
SendQEntries               = '(monitor)'
LogXptStatus               = '(monitor)'
RecvQEntries               = '(monitor)'
StaticConnectIdentifier     = '(DESCRIPTION= (ADDRESS= (PROTOCOL=tcp)
(HOST=south_      sales1.example.com) (PORT=2840))
(CONNECT_DATA=(SERVICE_NAME=South_Sales_DGMGRL.example.com)
(INSTANCE_NAME=south_sales1) (SERVER=DEDICATED)))'
StandbyArchiveLocation     = 'USE_DB_RECOVERY_FILE_DEST'
AlternateLocation          = ''
LogArchiveTrace            = '255'
LogArchiveFormat           = 'db3r_%d_%t_%s_%R.arc'
TopWaitEvents              = '(monitor)'
```

```
Database Status:
SUCCESS
```

SHOW FAR_SYNC

Shows information about a far sync instance.

Format

```
SHOW FAR_SYNC [VERBOSE] far_sync_instance_name [property-name];
```

Command Parameters

far_sync_instance_name

The name of the far sync instance for which the broker will show information. It must match (case-insensitive) the value of the corresponding database `DB_UNIQUE_NAME` initialization parameter.

property-name

The name of the property for which you want to display a value. If a property name is specified, the output shows only the specified property (not all properties of the far sync), regardless of whether or not the `VERBOSE` keyword is specified.

See Also: [Chapter 4](#) and [Chapter 8](#) for information about properties.

Command Example

Example 1

The following example shows sample output from this command:

```
DGMGRL> SHOW FAR_SYNC FS;

Far Sync - FS

Transport Lag: 0 seconds (computed 1 second ago)
Instance(s):
  fs1

Far Sync Status:
SUCCESS
```

Example 2

The following example shows sample output from this command when the `VERBOSE` option is used:

```
DGMGRL> SHOW FAR_SYNC VERBOSE FS;

Far Sync - FS

Transport Lag: 0 seconds (computed 0 seconds ago)
Instance(s):
  b02

Properties:
  DGConnectIdentifier      = 'fs.example.com'
  LogXptMode              = 'sync'
  RedoRoutes               = '(North_Sales : South_Sales)
(South_Sales : North_Sales)'
```

```

Binding = 'optional'
MaxFailure = '0'
MaxConnections = '1'
ReopenSecs = '300'
NetTimeout = '30'
RedoCompression = 'DISABLE'
LogShipping = 'ON'
TransportLagThreshold = '0'
TransportDisconnectedThreshold = '0'
ArchiveLagTarget = '0'
LogArchiveMaxProcesses = '5'
LogArchiveMinSucceedDest = '1'
InconsistentProperties = '(monitor)'
InconsistentLogXptProps = '(monitor)'
LogXptStatus = '(monitor)'
StaticConnectIdentifier = '(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)
(HOST=fs.example.com)(PORT=2202))
(CONNECT_DATA=(SERVICE_NAME=FS_DGMGRL.example.com)
(INSTANCE_NAME=fs1)(SERVER=DEDICATED)))'
StandbyArchiveLocation = 'USE_DB_RECOVERY_FILE_DEST'
AlternateLocation = ''
LogArchiveTrace = '255'
LogArchiveFormat = 'db2r_%d_%t_%s_%R.arc'
TopWaitEvents = '(monitor)'

```

```

Far Sync Status:
SUCCESS

```

SHOW FAST_START FAILOVER

Displays all fast-start failover related information.

Format

```
SHOW FAST_START FAILOVER;
```

COMMAND PARAMETERS

None.

Usage Notes

The `SHOW FAST_START FAILOVER` command shows a summary of the fast-start failover configuration.

Command Example

```
DGMGRL> SHOW FAST_START FAILOVER;
```

```
Fast-Start Failover: DISABLED
```

```
Threshold: 30 seconds  
Target: (none)  
Observer: (none)  
Lag Limit: 30 seconds  
Shutdown Primary: TRUE  
Auto-reinstate: TRUE  
Observer Reconnect: (none)  
Observer Override: FALSE
```

```
Configurable Failover Conditions
```

```
Health Conditions:  
Corrupted Controlfile YES  
Corrupted Dictionary YES  
Inaccessible Logfile NO  
Stuck Archiver NO  
Datafile Offline YES
```

```
Oracle Error Conditions:  
(none)
```

SHOW INSTANCE

Displays information or property values for the specified instance.

Format

```
SHOW INSTANCE [VERBOSE] instance-name [property-name]
[ON {DATABASE database-name | FAR_SYNC far-sync-name }];
```

Command Parameters

instance-name

The name of the instance for which you want to display information. The `VERBOSE` keyword, if used, must come before the instance name.

property-name

The name of the property for which you want to display a value. If a property name is specified, the output shows only the specified property (not all properties), regardless of whether or not the `VERBOSE` keyword is specified.

See Also: [Chapter 4](#) and [Chapter 8](#) for information about properties.

database-name | far-sync-name

The name of the database or far sync associated with the instance for which you want to show information.

Usage Notes

- The `SHOW INSTANCE` command shows a brief summary of the instance. `SHOW INSTANCE VERBOSE` shows properties of the instance in addition to the brief summary. They both show the status of the instance.
- The `SHOW INSTANCE VERBOSE` command only shows instance-specific properties.
- The properties that the `SHOW INSTANCE VERBOSE` command shows depend on the database role and the configuration composition:
 - For instances of the primary database, properties specific to physical or snapshot standby instances are shown only if there is at least one physical or snapshot standby database in the configuration. The properties specific to logical standby instances are shown only if there is at least one logical standby database in the configuration.
 - For instances of physical or snapshot standby databases, properties specific to logical standby instances are not shown.
 - For instances of logical standby databases, properties specific to physical and snapshot standby instances are not shown.
- The *instance-name* can be unique across the configuration. If *instance-name* is not unique, you must specify both the *database-name* and the *instance-name* to fully identify the instance.

Command Example

Example 1

The following example shows information about a specific instance of a database.

```
DGMGRL> SHOW INSTANCE 'north_sales1';

Instance 'north_sales1' of database 'North_Sales'

Instance Status:
SUCCESS
```

Example 2

Shows instance information in an extended format.

```
DGMGRL> SHOW INSTANCE VERBOSE 'north_sales1';

Instance 'north_sales1' of database 'North_Sales'

Host Name: north.example.com
PFILE:
Properties:
  SidName                = 'north_sales1'
  StaticConnectIdentifier = '(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)
(HOST=north.example.com)(PORT=2094))
(CONNECT_DATA=(SERVICE_NAME=North_Sales_DGMGRL.example.com)
(INSTANCE_NAME=north_sales1)(SERVER=DEDICATED)))'
  StandbyArchiveLocation = 'USE_DB_RECOVERY_FILE_DEST'
  AlternateLocation      = ''
  LogArchiveTrace        = '255'
  LogArchiveFormat        = 'r_%d_%t_%s_%R.arc'
  LatestLog               = '(monitor)'
  TopWaitEvents           = '(monitor)'

Instance Status:
SUCCESS
```

SHUTDOWN

Shuts down a currently running Oracle instance.

Format

```
SHUTDOWN [ ABORT | IMMEDIATE | NORMAL ];
```

Command Parameters

None.

Usage Notes

- Using the `SHUTDOWN` command with no arguments is equivalent to using the `SHUTDOWN NORMAL` command.
- The following list describes the options to the `SHUTDOWN` command:

- `ABORT`

Proceeds with the fastest possible shutdown of the database without waiting for calls to complete or for users to disconnect from the database. Uncommitted transactions are not rolled back. Client SQL statements being processed are terminated. All users connected to the database are implicitly disconnected, and the next database startup will require instance recovery. You must use this option if a background process terminates abnormally.

Caution: If you use the `ABORT` option on the primary database when fast-start failover is enabled and the observer is running, a fast-start failover may ensue. Use the `IMMEDIATE` or `NORMAL` option to prevent an unexpected fast-start failover from occurring.

- `IMMEDIATE`

Does not wait for current calls to complete or users to disconnect from the database. Further connections are prohibited. The database is closed and dismounted. The instance is shut down, and no instance recovery is required on the next database startup.

- `NORMAL`

This is the default option. The process waits for users to disconnect from the database. Further connections are prohibited. The database is closed and dismounted. The instance is shut down, and no instance recovery is required on the next database startup.

Command Example

Example 1

The following command shuts down the primary database in normal mode.

```
DGMGRL > SHUTDOWN;
```

```
Database closed.  
Database dismounted.
```

SHUTDOWN

Oracle instance shut down.

SQL

Executes a SQL statement or a PL/SQL stored procedure.

Format

SQL "*sql_statement*"

Command Parameters

sql_statement

The SQL statement or PL/SQL stored procedure to be executed.

Usage Notes

- The SQL statement or PL/SQL stored procedure to be executed must be enclosed in double quotation marks.
- If the string that DGMGRL passes to PL/SQL contains a filename, then the filename must be enclosed in single quotation marks and the entire string following the SQL command must be enclosed in double quotation marks. For example, use the following syntax:

```
SQL "CREATE TABLESPACE temp1 DATAFILE '?/oradata/trgt/temp1.dbf' SIZE 10M  
TEMPORARY"
```

- You cannot execute `SELECT` statements.
- Do not include a semi-colon (;) after the SQL statement to be executed.

Command Example

Example 1

The following command opens the database.

```
DGMGRL> SQL "alter database open"
```

START OBSERVER

Starts the fast-start failover observer.

Format

```
START OBSERVER [ FILE=observer_configuration_filename ];
```

Command Parameters

observer_configuration_filename

Specifies an explicit directory path and file name on the observer computer.

Usage Notes

- The Oracle Client Administrator kit, or the full Oracle Database Enterprise Edition or Oracle Personal Edition kit must be installed on the observer computer to monitor a broker configuration for which fast-start failover is to be enabled. See [Section 5.5.1](#) for more information.
- The `START OBSERVER` command must be issued on the observer computer. Once the observer is successfully started, control is not returned to the user until the observer is stopped (for example, by issuing the `STOP OBSERVER` command from a different client connection). If you want to perform further interaction with the broker configuration, you must connect through another client.
- The observer runs autonomously once it has been successfully started. For this reason, it is recommended that when invoking DGMGRL for the purpose of issuing the `START OBSERVER` command, specify the `-logfile` optional parameter on the command line so that output generated while acting as the observer is not lost. See [Section 7.1.1](#) for more information about this parameter and see [Section 9.5.3](#) for an example use of the `-logfile` option.
- If a directory path is not specified with the `FILE` parameter, the observer searches the current working directory for the `fsfo.dat` file. If an `fsfo.dat` file is not found and this is the first time the `START OBSERVER` command is issued, the observer creates a `fsfo.dat` file.
- The primary and target standby database `DB_UNIQUE_NAME` initialization parameter and connect identifiers are stored in the `fsfo.dat` configuration file. Oracle recommends you ensure this file is protected from unauthorized access.
- Fast-start failover does not need to be enabled before you issue this command.
 - If fast-start failover is enabled, the observer will retrieve primary and target standby connect identifiers from the broker configuration and begin monitoring the configuration.
 - If fast-start failover is not enabled, the observer continually monitors for when fast-start failover is enabled.
- Only the primary database needs to be running when you issue this command; the standby database that will be the target of a fast-start failover does not need to be running in order for this command to complete successfully.
- Use the `SHOW FAST_START FAILOVER` command, the `SHOW CONFIGURATION VERBOSE` command, or query the `FS_FAILOVER_*` columns in the `V$DATABASE` view on the primary database to see the status of the observer and its host computer.

- The `SHOW FAST_START FAILOVER` command indicates whether the broker configuration believes that an observer has already been started. If the command displays the following, then you can issue the `START OBSERVER` command on any computer to start the observer:

```
Observer:          (none)
```

If the `SHOW FAST_START FAILOVER` command shows a value for "Observer:", but the observer at that observer location is no longer running for some reason, you can do either of the following:

- issue the `START OBSERVER` command on the same observer computer where it was started originally, with the observer configuration file used when the observer was first started
- issue the `STOP OBSERVER` command and then the `START OBSERVER` command on any computer to start the observer

If the `SHOW FAST_START FAILOVER` command shows a value for "Observer:" and the observer is already running at that location, an attempt to start an observer at that location again will fail with the following error:

```
DGM-16954: Unable to open and lock the Observer configuration file
```

If the `SHOW FAST_START FAILOVER` command shows a value for "Observer:" and you attempt to start an observer at a different location, the command will fail with the following error:

```
ORA-16647: could not start more than one observer
```

- If the primary and target standby databases stay connected but they lose the connection to the observer, then the primary database goes into an unobserved state. This state is reported by the broker's health check capability.

Command Examples

Example 1

The following example shows how to start the observer.

```
DGMGRL> CONNECT sysdg@North_Sales.example.com;
Password: password
Connected as SYSDG.
DGMGRL> START OBSERVER;
Observer started
```

Example 2

The following example shows how to start the observer using `CONNECT '/'` so that connection credentials are not visible on the command line:

```
DGMGRL> CONNECT /@North_Sales.example.com;
DGMGRL> START OBSERVER;
Observer started.
```

You must set up Oracle Wallet or SSL to use `CONNECT '/'`. By setting up Oracle Wallet or SSL, you can write a script to securely start and run the observer as a background job without specifying database credentials in the script. When using Oracle Wallet as a secure external password store, be sure to add credentials for both the primary and fast-start failover target standby databases. The database connect string that you

specify when adding the credentials for each database must match the
ObserverConnectIdentifier or DGConnectIdentifier database property.

STARTUP

Starts an Oracle database instance with any of the following options:

- **FORCE**: shuts down the current Oracle instance in the `SHUTDOWN ABORT` mode before restarting it.
- **RESTRICT**: allows only Oracle users with the `RESTRICTED SESSION` system privilege to connect to the instance.
- **PFILE**: specifies the `PFILE` initialization parameter file to be used when the database instance is started.
- **MOUNT**: mounts the specified database without opening it.
- **OPEN**: mounts and opens the specified database.
- **NOMOUNT**: starts the specified database instance without mounting the database.

Format

```
STARTUP [FORCE] [RESTRICT] [PFILE=filename] [MOUNT | OPEN [open_options] | NOMOUNT];
```

Command Parameters

filename

The name of the initialization parameter file to be used when starting the database instance. If you do not specify the `PFILE` parameter option, then the default server parameter file (specific to your operating system) is used.

open-options

The mode of access in which you want the specified database to start. The possible modes are:

```
READ ONLY
READ WRITE
```

Usage Notes

- Using the `STARTUP` command with no arguments is equivalent to using the `STARTUP OPEN` command.
- If you do not use the `FORCE` clause when you use the `STARTUP` command and the current database instance is running, an error results. The `FORCE` clause is useful when you are debugging or when error conditions are occurring. Otherwise, it should not be used.
- Use the `RESTRICT` clause to allow only Oracle users with the `RESTRICTED SESSION` system privilege to connect to the instance. Later, you can use the `ALTER SYSTEM` command through SQL*Plus to disable the restricted session feature.
- If you do not use the `PFILE` clause to specify the initialization parameter file, the `STARTUP` command uses the default server parameter file, if it exists. Otherwise, the `STARTUP` command uses the default initialization parameter file. The default files are platform specific.

See your operating system-specific documentation for more information about the default parameter files.
- Use the `OPEN` clause to mount and open the specified database.

- The `NOMOUNT` clause starts the database instance without mounting the database. You cannot use the `NOMOUNT` clause with the `MOUNT` or `OPEN` options.

Command Examples

Example 1

The following examples show two different methods for starting a database instance. Each command starts a database instance using the standard parameter file, mounts the default database in exclusive mode, and opens the database.

```
DGMGRL> STARTUP;  
DGMGRL> STARTUP OPEN;
```

Example 2

The following command shuts down the current instance, immediately restarts it without mounting or opening the database, and allows only users with restricted session privileges to connect to it.

```
DGMGRL > STARTUP FORCE RESTRICT NOMOUNT;
```

Example 3

The following command starts an instance using the parameter file `testparm` without mounting the database.

```
DGMGRL > STARTUP PFILE=testparm NOMOUNT;
```

Example 4

The following example starts and mounts a database instance, but does not open it.

```
DGMGRL> STARTUP MOUNT;
```

STOP OBSERVER

Stops the fast-start failover observer.

Format

```
STOP OBSERVER;
```

Command Parameters

None.

Usage Notes

- You can issue this command while connected to any database in the broker configuration.
- This command does not disable fast-start failover, but a fast-start failover cannot be initiated in the absence of an observer.
- Fast-start failover does not need to be enabled when you issue this command.
- If fast-start failover *is enabled* when you issue the `STOP OBSERVER` command, then the primary and standby databases must be connected and communicating with each other. Otherwise the following error is returned:

```
ORA-16636 fast-start failover target standby in error state, cannot stop observer
```

If connectivity does not exist between the primary and standby databases, you can issue the `DISABLE FAST_START FAILOVER FORCE` command on the primary database and then issue the `STOP OBSERVER` command. Note that disabling fast-start failover with the `FORCE` option on a primary database that is disconnected from the observer and the target standby database does not prevent the observer from initiating a fast-start failover to the target standby database.

- If fast-start failover *is not enabled* when you issue the `STOP OBSERVER` command, then only the primary database must be running when you stop the observer.
- The observer does not stop immediately when the `STOP OBSERVER` command is issued. The observer does not discover it has been stopped until the next time the observer contacts the broker.

As soon as you have issued the `STOP OBSERVER` command, you may enter the `START OBSERVER` command again on any computer. You can start a new observer right away, even if the old observer has not yet discovered it was stopped. Any attempt to restart the old observer will fail, because a new observer has been started for the broker configuration.

Command Examples

Example 1

The following example stops the observer.

```
DGMGR> STOP OBSERVER;
```

SWITCHOVER

A switchover operation is a planned transition in which the primary database exchanges roles with one of the standby databases. When you issue the `SWITCHOVER` command, the current primary database becomes a standby database, and the specified standby database becomes the primary database.

Format

```
SWITCHOVER TO database-name;
```

Command Parameter

database-name

The name of the standby database you want to change to the primary database role.

Usage Notes

- If fast-start failover is enabled, you may switch over only to the fast-start failover target standby database.
- The broker verifies that the primary and standby databases are in the following states before starting the switchover:
 - The primary database must be enabled and in the `TRANSPORT-ON` state so redo transport services are started.
 - The standby database must be enabled and in the `APPLY-ON` state, with log apply services started.
- The broker allows the switchover to proceed as long as there are no redo transport services errors for the standby database that you selected to participate in the switchover. However, errors occurring for any other bystander standby database will not prevent the switchover from proceeding.
- Switchover to a logical standby database is not allowed when the configuration is operating in maximum protection mode.
- If the broker configuration is operating in either maximum protection mode or maximum availability mode, the switchover maintains the protection mode even after the operation (described in [Section 5.3.1](#)). The switchover will not be allowed if the mode cannot be maintained because the target standby database of the switchover was the only standby that satisfied the protection mode requirement.
- If the standby database that is assuming the primary role is a physical standby database, then the old primary database will be restarted after the switchover completes. If the standby database is a logical standby database, then neither the primary database nor the logical standby database is restarted.
- If the standby database that is assuming the primary role is a physical standby database, then the original primary becomes a physical standby database.
- If the standby database that is assuming the primary role is a logical standby database, then the original primary becomes a logical standby database.
- If an Oracle RAC primary database is becoming a physical standby database, all but one instance of the primary database will be shut down before performing the switchover. See [Section 5.3](#) for details.

- You cannot switchover to a snapshot standby database.
- If the standby database that is assuming the primary role is a logical standby database and there are physical standby databases in the configuration, after the switchover, the physical standby databases will be disabled.

Caution: For this reason, Oracle generally recommends that you specify your physical standby database for switchover instead of your logical standby database. If you must switch over to your logical standby database, see [Section 5.4.3](#) to re-create your physical standby database.

If you intend to switch back to the original primary database relatively soon, you may allow the physical and snapshot standbys to remain disabled. Once you have completed the switchover back to the original primary, you may then reenable the physical and snapshot standby databases since they are still viable standbys for the original primary database.

Command Examples

Example 1

The following example shows a successful switchover in which the physical standby database, `South_Sales`, transitions into the primary role.

```
DGMGRL> switchover to 'South_Sales';
Performing switchover NOW, please wait...
New primary database "South_Sales" is opening...
Operation requires shutdown of instance "north_sales1" on database "North_Sales"
Shutting down instance "north_sales1"...
ORA-01109: database not open

Database dismounted.
ORACLE instance shut down.
Operation requires startup of instance "north_sales1" on database "North_Sales"
Starting instance "north_sales1"...
ORACLE instance started.
Database mounted.
Switchover succeeded, new primary is "South_Sales"
```

Example 2

If you connect to the database using operating system authentication, you can use any username and password to connect. However, DGMGRL may not be able to shut down and start up the primary and standby database automatically because it cannot remotely authenticate itself.

The following example shows a switchover that succeeded but returns an error because DGMGRL cannot shut down and start up the primary and standby databases.

```
DGMGRL> SWITCHOVER TO 'South_Sales';
Performing switchover NOW, please wait...
New primary database "South_Sales" is opening...
Operation requires shutdown of instance "north_sales1" on database "North_Sales"
Shutting down instance "north_sales1"...
ORA-01031: insufficient privileges

Warning: You are no longer connected to ORACLE.
```

Please complete the following steps to finish switchover:

```
shut down instance "north_sales1" of database "North_Sales"  
start up and mount instance "north_sales1" of database "North_Sales"
```

Note: For DGMGRL to restart instances automatically, you must connect to the database using the same credentials given in the last `CONNECT` command, even if the last `CONNECT` command was used to connect to another database.

You must manually issue the [SHUTDOWN](#) and [STARTUP](#) commands to restart the new primary and any standby instances that may have been shut down.

VALIDATE DATABASE

Performs a comprehensive set of database checks prior to a role change. The checks use information available in various Oracle Data Guard views as well as the Automatic Diagnostic Repository.

Format

```
VALIDATE DATABASE [VERBOSE] database-name;
```

Command Parameters

database-name

The name of the database for which you want to display information. The `VERBOSE` keyword, if used, must come before the `database-name` or an error is returned.

Usage Notes

The `VALIDATE DATABASE` command shows a brief summary of the database, and reports any errors or warnings that were detected. `VALIDATE DATABASE VERBOSE` shows everything in the brief summary plus all items that were validated.

Command Examples

The examples in this section show what the `VALIDATE DATABASE` command output might look like in both the brief and verbose forms for primary and standby databases.

Example 1

The following example shows brief output for a primary database:

```
DGMGRL> VALIDATE DATABASE 'South_Sales';
```

```
Database Role:      Primary database
```

```
Ready for Switchover:  Yes
```

Example 2

The following example shows brief output for a logical standby database:

```
DGMGRL> VALIDATE DATABASE 'West_Sales';
```

```
Database Role:      Logical standby database
```

```
Primary Database:  South_Sales
```

```
Ready for Switchover:  Yes
```

```
Ready for Failover:   Yes (Primary Running)
```

```
Warning: Physical and snapshot standby databases will  
be disabled if a role change is performed to this database
```

Example 3

The following example shows brief output for a physical standby database:

```
DGMGRL> VALIDATE DATABASE 'North_Sales';
```

```
Database Role:      Physical standby database
```

```

Primary Database:  South_Sales

Ready for Switchover:  Yes
Ready for Failover:   Yes (Primary Running)
    
```

Example 4

The following example shows verbose output for a primary database:

```
DGMGRL> VALIDATE DATABASE VERBOSE 'South_Sales';
```

```

Database Role:      Primary database

Ready for Switchover:  Yes

Capacity Information:
  Database      Instances      Threads
  South_Sales   2              2

Temporary Tablespace File Information:
  South_Sales TEMP Files:  2

Flashback Database Status:
  South_Sales:  On

Data file Online Move in Progress:
  South_Sales:  No

Transport-Related Information:
  Transport On:  Yes

Log Files Cleared:
  South_Sales Standby Redo Log Files:  Cleared

Automatic Diagnostic Repository Errors:
  Error                      South_Sales
  No logging operation        NO
  Control file corruptions    NO
  System data file missing    NO
  System data file corrupted  NO
  System data file offline    NO
  User data file missing       NO
  User data file corrupted     NO
  User data file offline      NO
  Block Corruptions found     NO
    
```

Example 5

The following command shows verbose output for a logical standby database:

```
DGMGRL> VALIDATE DATABASE VERBOSE 'West_Sales';
```

```

Database Role:      Logical standby database
Primary Database:   South_Sales

Ready for Switchover:  Yes
Ready for Failover:   Yes (Primary Running)
  Warning: Logical and snapshot standby databases will
  be disabled if a role change is performed to this database

Capacity Information:
  Database      Instances      Threads
  South_Sales   2              2
    
```

```

West_Sales      2      2

Temporary Tablespace File Information:
  South_Sales TEMP Files:  2
  West_Sales TEMP Files:  2

Flashback Database Status:
  South_Sales:  On
  West_Sales:  On

Data file Online Move in Progress:
  South_Sales:  No
  West_Sales:  No

Standby Apply-Related Information:
  Apply State:      Running
  Apply Lag:        0 seconds
  Apply Delay:      0 minutes

Transport-Related Information:
  Transport On:     Yes
  Gap Status:       No Gap
  Transport Lag:    0 seconds
  Transport Status: Success

Log Files Cleared:
  South_Sales Standby Redo Log Files: Cleared
  West_Sales Online Redo Log Files:  Not applicable for this database
  West_Sales Standby Redo Log Files:  Available

Current Log File Groups Configuration:
  Thread #  Online Redo Log Groups  Standby Redo Log Groups  Status
           (South_Sales)         (West_Sales)
  1         4                     5                        Sufficient SRLs
  2         4                     5                        Sufficient SRLs

Future Log File Groups Configuration:
  Thread #  Online Redo Log Groups  Standby Redo Log Groups  Status
           (West_Sales)         (South_Sales)
  1         4                     5                        Sufficient SRLs
  2         4                     5                        Sufficient SRLs

Current Configuration Log File Sizes:
  Thread #  Smallest Online Redo  Smallest Standby Redo
           Log File Size  Log File Size
           (South_Sales)  (West_Sales)
  1         25 MBytes     25 MBytes
  2         25 MBytes     25 MBytes

Future Configuration Log File Sizes:
  Thread #  Smallest Online Redo  Smallest Standby Redo
           Log File Size  Log File Size
           (West_Sales)  (South_Sales)
  1         25 MBytes     25 MBytes
  2         25 MBytes     25 MBytes

Apply-Related Property Settings:
  Property                South_Sales Value  West_Sales Value
  DelayMins                0                  0
  LsbyMaxEventsRecorded    0                  0

```

```

LsbyPreserveCommitOrder      <empty>          <empty>
LsbyRecordSkipErrors         <empty>          <empty>
LsbyRecordSkipDdl            <empty>          <empty>
LsbyRecordAppliedDdl         <empty>          <empty>

```

Transport-Related Property Settings:

Property	South_Sales Value	West_Sales Value
LogXptMode	SYNC	SYNC
Dependency	<empty>	<empty>
DelayMins	0	0
Binding	optional	optional
MaxFailure	0	0
MaxConnections	1	1
ReopenSecs	30	30
NetTimeout	300	300
RedoCompression	DISABLE	DISABLE
LogShipping	ON	ON

Automatic Diagnostic Repository Errors:

Error	South_Sales	West_Sales
No logging operation	NO	NO
Control file corruptions	NO	NO
SRL Group Unavailable	NO	NO
System data file missing	NO	NO
System data file corrupted	NO	NO
System data file offline	NO	NO
User data file missing	NO	NO
User data file corrupted	NO	NO
User data file offline	NO	NO
Block Corruptions found	NO	NO

Example 6

The following command shows verbose output for a physical standby database:

```

DGMGRL> VALIDATE DATABASE VERBOSE North_Sales';
Database Role:      Physical standby database
Primary Database:  South_Sales

Ready for Switchover:  Yes
Ready for Failover:   Yes (Primary Running)

Capacity Information:
Database      Instances      Threads
South_Sales   2              2
North_Sales   2              2

Temporary Tablespace File Information:
South_Sales TEMP Files:  2
North_Sales TEMP Files:  2

Flashback Database Status:
South_Sales:  On
North_Sales:  On

Data file Online Move in Progress:
South_Sales:  No
North_Sales:  No

Standby Apply-Related Information:
Apply State:      Running

```



```

Apply Lag:          0 seconds
Apply Delay:        0 minutes

```

Transport-Related Information:

```

Transport On:       Yes
Gap Status:         No Gap
Transport Lag:      0 seconds
Transport Status:   Success

```

Log Files Cleared:

```

South_Sales Standby Redo Log Files:  Cleared
North_Sales Online Redo Log Files:   Cleared
North_Sales Standby Redo Log Files:   Available

```

Current Log File Groups Configuration:

Thread #	Online Redo Log Groups (South_Sales)	Standby Redo Log Groups (North_Sales)	Status
1	4	5	Sufficient SRLs
2	4	5	Sufficient SRLs

Future Log File Groups Configuration:

Thread #	Online Redo Log Groups (North_Sales)	Standby Redo Log Groups (South_Sales)	Status
1	4	5	Sufficient SRLs
2	4	5	Sufficient SRLs

Current Configuration Log File Sizes:

Thread #	Smallest Online Redo Log File Size (South_Sales)	Smallest Standby Redo Log File Size (North_Sales)
1	25 MBytes	25 MBytes
2	25 MBytes	25 MBytes

Future Configuration Log File Sizes:

Thread #	Smallest Online Redo Log File Size (North_Sales)	Smallest Standby Redo Log File Size (South_Sales)
1	25 MBytes	25 MBytes
2	25 MBytes	25 MBytes

Apply-Related Property Settings:

Property	South_Sales Value	North_Sales Value
DelayMins	0	0
ApplyParallel	AUTO	AUTO

Transport-Related Property Settings:

Property	South_Sales Value	North_Sales Value
LogXptMode	SYNC	SYNC
Dependency	<empty>	<empty>
DelayMins	0	0
Binding	optional	optional
MaxFailure	0	0
MaxConnections	1	1
ReopenSecs	30	30
NetTimeout	300	300
RedoCompression	DISABLE	DISABLE
LogShipping	ON	ON

Automatic Diagnostic Repository Errors:

Error	South_Sales	North_Sales
-------	-------------	-------------

No logging operation	NO	NO
Control file corruptions	NO	NO
SRL Group Unavailable	NO	NO
System data file missing	NO	NO
System data file corrupted	NO	NO
System data file offline	NO	NO
User data file missing	NO	NO
User data file corrupted	NO	NO
User data file offline	NO	NO
Block Corruptions found	NO	NO

VALIDATE FAR_SYNC

Performs a comprehensive set of checks for a far sync instance. The checks use information available in various Oracle Data Guard views.

Format

```
VALIDATE FAR_SYNC [VERBOSE] far_sync_instance_name [WHEN PRIMARY IS database-name];
```

Command Parameters

far_sync_instance_name

The name of the far sync instance for which you want to display information. The `VERBOSE` keyword, if used, must come before the `far_sync_instance_name` or an error is returned.

database-name

The validation of the far sync instance is performed based on the specified database being the primary database.

Usage Notes

The `VALIDATE FAR_SYNC` command shows a brief summary of the far sync instance and reports any errors or warnings that were detected. The `VALIDATE FAR_SYNC VERBOSE` command shows everything in the brief summary plus redo transport-related information.

Command Examples

The examples in this section show what the `VALIDATE FAR_SYNC` command output might look like in various scenarios.

Example 1

The following example shows brief output for a far sync instance:

```
DGMGRL> VALIDATE FAR_SYNC FS;
```

```
Member Role:          Far Sync Instance
When Primary Is:     North_Sales

Active Redo Source: North_Sales
Redo Destinations:
                    South_Sales

Thread #  Online Redo Log Groups  Standby Redo Log Groups  Status
        1      4                5
        North_Sales              FS
                                Sufficient SRLs
```

Example 2

The following example shows verbose output for a far sync instance:

```
DGMGRL> VALIDATE FAR_SYNC VERBOSE FS;
```

```
Member Role:          Far Sync Instance
When Primary Is:     North_Sales
```

Active Redo Source: North_Sales

Redo Destinations:

South_Sales

Thread #	Online Redo Log Groups	Standby Redo Log Groups	Status
1	North_Sales 4	FS 5	Sufficient SRLs

Transport-Related Information:

Transport On: Yes

Gap Status: No Gap

Transport Lag: 0 seconds (computed 0 seconds ago)

Transport Status: Success

Example 3

The following example shows the output when the South_Sales database is specified as the primary database:

```
DGMGRL> VALIDATE FAR_SYNC FS WHEN PRIMARY IS 'South_Sales';
```

Member Role: Far Sync Instance

When Primary Is: South_Sales

Redo Destinations:

North_Sales

Thread #	Online Redo Log Groups	Standby Redo Log Groups	Status
1	South_Sales 4	FS 5	Sufficient SRLs

Oracle Data Guard Broker Properties

This chapter describes the various configuration and database properties that help you view and control the behavior of entire broker configurations, individual configuration members, redo transport services, and log apply services.

This chapter discusses the following topics:

- [Configuration Properties](#)
- [Monitorable \(Read-Only\) Properties](#)
- [Configurable Properties](#)

Properties have either configuration-wide scope, database-wide scope, far sync instance-wide scope, or instance-specific scope. Configuration-wide properties control the behavior of the broker on all members of the configuration. The values of such properties apply uniformly across all members of the configuration.

Database-wide properties allow you to view or control the behavior of a specific database. If the database (primary or standby) is an Oracle RAC database consisting of multiple instances, the value of such a property applies uniformly across all instances of that database.

Far sync instance-wide properties allow you to view or control the behavior of a far sync instance.

Instance-specific properties allow you to view or control the behavior of an individual database instance. Such a property exists for all instances of an Oracle RAC database, but its value may differ from one specific instance to another.

Note: This chapter presents properties primarily from the point of view of the Oracle Data Guard command-line interface (DGMGRL). Using DGMGRL, the properties described in this chapter may be viewed or modified using discrete DGMGRL commands.

Oracle Enterprise Manager Cloud Control (Cloud Control) explicitly presents some of these properties on the Edit Properties page. Information from other properties may be implicitly incorporated into other Web pages displayed by Cloud Control. Each property's description in this chapter indicates how Cloud Control presents that property.

8.1 Configuration Properties

Configuration properties control the behavior of the broker configuration. You can view and dynamically update the values of these properties using either DGMGRL or Cloud Control. However, some properties can only be updated through DGMGRL.

A configuration property has configuration-wide scope; meaning that the value you set for the property applies uniformly to each member in the configuration.

Configuration properties are set using the `EDIT CONFIGURATION SET PROPERTY` command, as shown in the following examples.

Example 1

This example sets the `FastStartFailoverThreshold` configuration property to 90 seconds.

```
DGMGRL> EDIT CONFIGURATION SET PROPERTY FastStartFailoverThreshold=90;
```

Example 2

This example sets the `ExternalDestination1` configuration property to point to a redo transport destination with a service name of `Sales` and a `DB_UNIQUE_NAME` of `chicago`.

```
EDIT CONFIGURATION SET PROPERTY ExternalDestination1='SERVICE=Sales DB_UNIQUE_
NAME=chicago';
```

Note: In general, quotation marks are not necessary around property values unless the case you specify needs to be preserved. By default, DGMGRL lowercases any string you supply, unless it is placed within quotation marks.

Quotation marks are required for any property values that include spaces or punctuation characters (for example, `RedoRoutes`, `LogFileNameConvert`, or `DBFileNameConvert`).

The following sections describe the configuration properties:

- [BystandersFollowRoleChange](#)
- [CommunicationTimeout](#)
- [ExternalDestination1](#)
- [ExternalDestination2](#)
- [FastStartFailoverAutoReinstate](#)
- [FastStartFailoverLagLimit](#)
- [FastStartFailoverPmyShutdown](#)
- [FastStartFailoverThreshold](#)
- [ObserverOverride](#)
- [ObserverReconnect](#)
- [OperationTimeout](#)
- [PrimaryLostWriteAction](#)
- [TraceLevel](#)

8.1.1 BystandersFollowRoleChange

The `BystandersFollowRoleChange` configuration property establishes whether bystander standby databases are evaluated *during* failover (value = `ALL`) or *after* failover (value = `NONE`).

- `ALL` - During the failover process, the broker determines whether the bystander standby databases are ahead of or behind the failover target standby (that is, the standby that will be the new primary).

If the bystander standbys are ahead, they will be disabled with a status of `ORA-16661` as part of the failover operation. The bystander standbys must be reinstated after failover completes. The broker reinstates a standby through a flashback to the SCN at which the target standby became a primary database, and sets up the redo transport configuration from the new primary to the standby.

If the bystander standbys are behind, then the broker simply sets up the redo transport configuration from the soon-to-be-new-primary to these standbys and completes the failover process.

- `NONE` - During the failover process, the broker does NOT evaluate the status of the bystander standbys as part of the failover operation. They are marked as disabled with a status of `ORA-16661` so that they can be evaluated later. The broker simply completes the failover to produce a new primary database as soon as possible.

After the failover is completed, you can reinstate the bystander standbys. During reinstatement of a bystander, the broker determines whether the bystander is ahead of, or behind, the new primary. If the bystander is ahead of the new primary, then the broker automatically flashes back the standby to the SCN at which the target standby became a primary database and sets up redo transport from the new primary to the standby. (Even if a flashback is not required, the broker sets up the redo transport configuration from the new primary to these standbys.)

The `NONE` option decreases the processing time for failover, but disables broker management of all bystander databases in the configuration. If fast-start failover is enabled, then the observer automatically reinstates the standby databases after failover has completed. Otherwise, you will have to manually reinstate the standby databases after failover has completed.

Category	Description
Datatype	String
Valid value	<code>ALL</code> or <code>NONE</code>
Broker default	<code>ALL</code>
Imported?	No
Parameter class	Not applicable
Role	Primary and standby
Standby type	Not applicable
Corresponds to	Not applicable
Scope	Broker configuration. This property will be consumed by broker on the database that is the target of a complete failover.
Cloud Control name	Not applicable

8.1.2 CommunicationTimeout

The `CommunicationTimeout` configuration property allows you to decide how many seconds the broker should wait before timing out its network communication between members in the configuration. A value of zero indicates that a network communication should never be timed out.

Category	Description
Datatype	Integer
Valid values	>= 0
Broker default	180 seconds
Imported?	No
Parameter class	Not applicable
Role	Primary and standby
Standby type	Not applicable
Corresponds to	Not applicable
Scope	Broker configuration. This property will be consumed by broker on all databases in the configuration.
Cloud Control name	Not applicable

8.1.3 ExternalDestination1

Notes:

- The `Externaldestination1` configuration property is available only in Oracle Database 11g Release 2 (11.2.0.4) and in Oracle Database 12c Release 1 (12.1.0.2) and later.
 - As of Oracle Database 12c release 1 (12.1.0.2), the `ExternalDestination1` configuration property supports Zero Data Loss Recovery Appliance (Recovery Appliance) as a redo transport destination.
-
-

The `ExternalDestination1` configuration property is used to specify a redo transport destination that can receive redo data from the current primary database. To set up transport of redo data to the specified destination, the broker uses the values specified for this parameter to define a `LOG_ARCHIVE_DEST_n` initialization parameter on the primary database. The broker also monitors the health of the transport to the specified destination.

After a role change, the broker automatically sets up a `LOG_ARCHIVE_DEST_n` initialization parameter on the new primary database to ship redo data to the specified destination.

Category	Description
Datatype	String

Category	Description
Valid values	Any LOG_ARCHIVE_DEST_1 attributes, except the following: <ul style="list-style-type: none"> ▪ ALTERNATE ▪ DELAY ▪ LOCATION ▪ MANDATORY ▪ MAX_FAILURE ▪ NET_TIMEOUT ▪ SYNC ▪ VALID_FOR <p>See <i>Oracle Data Guard Concepts and Administration</i> for more information about the LOG_ARCHIVE_DEST_1 parameter.</p>
Broker default	Empty string
Imported?	No
Parameter class	Not applicable
Role	Primary
Standby type	Not applicable
Corresponds to	Not applicable
Scope	Configuration
Cloud Control name	Not applicable

8.1.4 ExternalDestination2

Notes:

- The Externaldestination2 configuration property is available only in Oracle Database 11g Release 2 (11.2.0.4) and in Oracle Database 12c Release 1 (12.1.0.2) and later.
 - As of Oracle Database 12c release 1 (12.1.0.2), the ExternalDestination2 configuration property supports Zero Data Loss Recovery Appliance (Recovery Appliance) as a redo transport destination.
-
-

The ExternalDestination2 configuration property is used to specify a redo transport destination that can receive redo data from the current primary database. To set up transport of redo data to the specified destination, the broker uses the values specified for this parameter to define a LOG_ARCHIVE_DEST_1 initialization parameter on the primary database. The broker also monitors the health of the transport to the specified destination.

After a role change, the broker automatically sets up a LOG_ARCHIVE_DEST_1 initialization parameter on the new primary database to ship redo data to the specified destination.

Category	Description
Datatype	String
Valid values	Any LOG_ARCHIVE_DEST_n attributes, except the following: <ul style="list-style-type: none"> ■ ALTERNATE ■ DELAY ■ LOCATION ■ MANDATORY ■ MAX_FAILURE ■ NET_TIMEOUT ■ SYNC ■ VALID_FOR <p>See <i>Oracle Data Guard Concepts and Administration</i> for more information about the LOG_ARCHIVE_DEST_n parameter.</p>
Broker default	Empty string
Imported?	No
Parameter class	Not applicable
Role	Primary
Standby type	Not applicable
Corresponds to	Not applicable
Scope	Configuration
Cloud Control name	Not applicable

8.1.5 FastStartFailoverAutoReinstate

The `FastStartFailoverAutoReinstate` configuration property causes the former primary database to be automatically reinstated if a fast-start failover was initiated because the primary database was either isolated or had crashed. To prevent automatic reinstatement of the former primary database in these cases, set this configuration property to `FALSE`.

The broker never automatically reinstates the former primary database if a fast-start failover was initiated because a user configuration condition was detected or was requested by an application calling the `DBMS_DG.INITIATE_FS_FAILOVER` function.

Category	Description
Datatype	Boolean
Valid value	TRUE or FALSE
Broker default	TRUE
Imported?	No
Parameter class	Not applicable
Role	Primary and standby
Standby type	Not applicable
Corresponds to	Not applicable

Category	Description
Scope	Broker configuration. This property will be consumed by the observer after fast-start failover has been enabled.
Cloud Control name	Automatically Reinstate Primary

8.1.6 FastStartFailoverLagLimit

The `FastStartFailoverLagLimit` configuration property establishes an acceptable limit, in seconds, that the standby is allowed to fall behind the primary in terms of redo applied, beyond which a fast-start failover will not be allowed. The lowest possible value is 10 seconds.

This property is used when fast-start failover is enabled and the configuration is operating in maximum performance mode.

Category	Description
Datatype	Integer
Valid value	Integral number of seconds. Must be greater than, or equal to, 10.
Broker default	30 seconds
Imported?	No
Parameter class	Not applicable
Role	Primary and standby
Standby type	Not applicable
Corresponds to	Not applicable
Scope	Broker configuration. This property will be consumed by the primary database after fast-start failover has been enabled.
Cloud Control name	Lag Limit

8.1.7 FastStartFailoverPmyShutdown

The `FastStartFailoverPmyShutdown` configuration property causes the primary database to shut down if fast-start failover is enabled and `V$DATABASE.FS_FAILOVER_STATUS` indicates the primary has been `STALLED` for longer than `FastStartFailoverThreshold` seconds. In such a situation, it is likely that the primary has been isolated and a fast-start failover has already occurred. A value of `TRUE` helps to ensure that an isolated primary database cannot satisfy user queries.

Setting this property to `FALSE` will not prevent the primary database from shutting down if a fast-start failover occurred because a user configuration condition was detected or was requested by an application by calling the `DBMS_DG.INITIATE_FS_FAILOVER` function.

Category	Description
Datatype	Boolean
Valid value	TRUE or FALSE
Broker default	TRUE
Imported?	No

Category	Description
Parameter class	Not applicable
Role	Primary and standby
Standby type	Not applicable
Corresponds to	Not applicable
Scope	Broker configuration. This property will be consumed by the primary database after fast-start failover has been enabled.
Cloud Control name	Automatically Shutdown Primary

8.1.8 FastStartFailoverThreshold

The `FastStartFailoverThreshold` configuration property defines the number of seconds the observer attempts to reconnect to the primary database before initiating a fast-start failover to the target standby database. The time interval starts when the observer first loses connection with the primary database. If the observer is unable to regain a connection to the primary database within the specified time, then the observer initiates a fast-start failover. See Step 4 in [Section 5.5.2](#) for more information about setting this property.

The observer ignores the threshold completely if a configurable fast-start failover condition is detected or an application has requested that fast-start failover be initiated.

For help in determining an appropriate value for this property, you can use the information provided in the `V$FS_OBSERVER_HISTOGRAM` view. This view displays statistics that are based on the frequency of successful pings between the observer and primary database for different time intervals. (This view is available in Oracle Database 12c Release 1 (12.1.0.2) and later. It is also available in Oracle Database 11g Release 2 (11.2.0.4), but not in Oracle Database 12c Release 1 (12.1.0.1).) See *Oracle Database Reference* for a description of the `V$FS_OBSERVER_HISTOGRAM` view.

Category	Description
Datatype	Integer
Valid value	Integral number of seconds. Must be greater than, or equal to, 6.
Broker default	30 seconds
Imported?	No
Parameter class	Not applicable
Role	Target standby database that is about to fail over to the primary role
Standby type	Not applicable
Corresponds to	Not applicable
Scope	Broker configuration. This property will be consumed by the observer after fast-start failover has been enabled.
Cloud Control name	Cloud Control presents this as "Failover Threshold" on the Oracle Data Guard Overview page.

8.1.9 ObserverOverride

The `ObserverOverride` configuration property, when set to `TRUE`, allows an automatic failover to occur when the observer has lost connectivity to the primary, even if the standby has a healthy connection to the primary.

Category	Description
Datatype	Boolean
Valid values	TRUE or FALSE
Broker default	FALSE
Imported?	No
Parameter class	Not applicable
Role	Primary and standby
Standby type	Not applicable
Corresponds to	Not applicable
Scope	Broker configuration. This property will be consumed by the observer after fast-start failover has been enabled.
Cloud Control name	Not applicable

8.1.10 ObserverReconnect

The `ObserverReconnect` configuration property specifies how often the observer establishes a new connection to the primary database. When this property is set to the default value of 0, the observer creates and maintains a connection to the primary database, but it does not periodically create a new connection to the primary database. While this eliminates the processing overhead associated with periodically establishing a new observer connection to the primary database, it also prevents the observer from detecting that it is not possible to create new connections to the primary database. Oracle recommends that this property be set to a value that is small enough to allow timely detection of faults at the primary database, but large enough to limit the overhead associated with periodic observer connections to an acceptable level.

Category	Description
Datatype	Integer
Valid values	>=0
Broker default	0
Imported?	No
Parameter class	Not applicable
Role	Primary and standby
Standby type	Not applicable
Corresponds to	Not applicable
Scope	Broker configuration. This property will be consumed by the observer after fast-start failover has been enabled.
Cloud Control name	Not applicable

8.1.11 OperationTimeout

The `OperationTimeout` configuration property specifies the maximum amount of time the broker should wait for health check, get monitorable property, and set property operations to complete.

Category	Description
Datatype	Integer
Valid values	>= 30 and <= 600
Broker default	30 seconds
Imported?	No
Parameter class	Not applicable
Role	Primary and standby
Standby type	Not applicable
Corresponds to	Not applicable
Scope	Broker configuration. This property will be consumed by broker on all databases in the configuration.
Cloud Control name	Not applicable

8.1.12 PrimaryLostWriteAction

Note: The `PrimaryLostWriteAction` configuration property is available only in Oracle Database 11g Release 2 (11.2.0.4) and in Oracle Database 12c Release 1 (12.1.0.2) and later.

The `PrimaryLostWriteAction` configuration property determines what action is taken if a standby database detects that a lost write has occurred at the primary database:

- **CONTINUE** - The primary database continues operating even if a standby database detects that a lost write has occurred at the primary database. This is the default action.
- **SHUTDOWN** - If a standby database detects that a lost write has occurred at the primary database, then fast start failover is disabled and the primary database performs a shutdown abort. Automatic failover will not occur.
- **FAILOVER** - If fast-start failover is enabled in maximum performance mode, and the configuration is within the acceptable limit specified for the `FastStartFailoverLagLimit` property, then the observer initiates a failover. (The `FAILOVER` option is available only in Oracle Database 12c Release 1 (12.1.0.2) and later.)
- **FORCEFAILOVER** - If fast-start failover is enabled (in either maximum performance or maximum availability mode), then the observer initiates a failover. This option results in a data loss failover. (The `FORCEFAILOVER` option is available only in Oracle Database 12c Release 1 (12.1.0.2) and later.)

For both the `FAILOVER` and `FORCEFAILOVER` options, if fast-start failover is disabled then no failover occurs, but the primary is shut down.

Fast start failover is disabled if a standby detects that a lost write has occurred at the primary database. Automatic failover will therefore not occur if a standby detects that a lost write has occurred at the primary database, even if `PrimaryLostWriteAction` is set to `SHUTDOWN`.

Diagnostic information is written to the database alert and broker logs at the primary database and at the standby database where the lost write was detected.

If a lost write occurs at the primary database, then follow the guidelines in "Resolving ORA-752 or ORA-600 [3020] During Standby Recovery" in My Oracle Support Note 1265884.1 located at <http://support.oracle.com>.

Note: The `DB_LOST_WRITE_PROTECT` database initialization parameter must be set to `TYPICAL` or `FULL` at the primary database and at each standby database in the configuration to ensure that lost primary writes can be detected by all standby databases in the configuration.

Category	Description
Datatype	String
Valid values	CONTINUE or SHUTDOWN
Broker default	CONTINUE
Imported?	No
Parameter class	Not applicable
Role	Primary and standby
Standby type	Not applicable
Corresponds to	Not applicable
Scope	Broker configuration
Cloud Control name	Not applicable

8.1.13 TraceLevel

The `TraceLevel` configuration property is used to control the amount of tracing performed by the broker for every member in the configuration. Setting the property to `USER` limits the tracing to completed operations and to any warning or error messages resulting from an operation or health check. Setting the property to `SUPPORT` increases the amount of tracing to include lower-level information needed by Oracle Support Services.

Category	Description
Datatype	String
Valid values	USER, SUPPORT
Broker default	USER
Imported?	No
Parameter class	Not applicable
Role	Primary and standby
Standby type	Not applicable

Category	Description
Corresponds to	Not applicable
Scope	Database
Cloud Control name	Not applicable

8.2 Monitorable (Read-Only) Properties

Monitorable properties allow you to view information related to the database, database instance, or far sync instance, but you cannot change the values of these properties. You can view all of the monitorable properties using the DGMGRL `SHOW` commands.

Note: Information for monitorable properties can be seen only when broker management of the entity is enabled. Cloud Control displays the information obtained from these properties on the Property page.

If the database is an Oracle RAC database, the output values of some properties may also show instance-specific information. For example if the primary database is an Oracle RAC database, `LogXptStatus` may show `Instance1` transmitting redo data to `Standby2` has an error and `Instance2` transmitting redo data to `Standby4` has an error.

The following sections describe the monitorable properties:

- [InconsistentLogXptProps \(Inconsistent Redo Transport Properties\)](#)
- [InconsistentProperties \(Inconsistent Properties\)](#)
- [LogXptStatus \(Redo Transport Status\)](#)
- [LsbyFailedTxnInfo \(Logical Standby Failed Transaction Information\)](#)
- [LsbyParameters \(Logical Standby Parameters\)](#)
- [RecvQEntries \(Receive Queue Entries\)](#)
- [SendQEntries \(Send Queue Entries\)](#)
- [TopWaitEvents](#)

8.2.1 InconsistentLogXptProps (Inconsistent Redo Transport Properties)

The `InconsistentLogXptProps` monitorable property returns a table that shows all properties related to redo transport services whose values are inconsistent between the broker configuration file and the runtime value.

Although the properties reported in this table are database-specific or far sync instance-specific properties, the inconsistency is reported on an instance-specific basis. A database or far sync instance-specific property only ensures that there is one value in the broker's configuration file for all instances of the database or far sync instance, but the runtime values can be different. This means that a database or far sync instance-specific property may be inconsistent only on some instances.

This property pertains to the primary database, a physical standby database that ships redo data, or a far sync instance. The table contains the following columns:

- `INSTANCE_NAME`

The value identifying the SID for the instance on the primary database.

- STANDBY_DATABASE_NAME

The database unique name (DB_UNIQUE_NAME) of the standby database or far sync instance.

- ERROR

The text of the redo transport error. If there is no error, the field is empty.

Each entry in the table indicates the status of redo transport services on one redo source to one redo destination.

The error status can be an empty string, which indicates there is no error.

In the following example, the STATUS from South_Sales is empty because there is no error for the South_Sales destination. The South_Report destination returned the ORA-01034 message.

```
DGMGRL> SHOW DATABASE 'North_Sales' 'LogXptStatus' ;
LOG TRANSPORT STATUS
PRIMARY_INSTANCE_NAME STANDBY_DATABASE_NAME          STATUS
                        north_sales1                South_Sales
                        north_sales1                South_Report  ORA-01034: ORACLE not available
```

8.2.4 LsbyFailedTxnInfo (Logical Standby Failed Transaction Information)

The LsbyFailedTxnInfo monitorable property identifies a failed transaction that caused log apply services to stop. This property contains a string with the following values from the DBA_LOGSTDBY_EVENTS view:

- XIDUSN: Transaction ID undo segment number
- XIDSLT: Transaction ID slot number
- XIDSQN: Transaction ID sequence number
- STATUS_CODE: Status (or Oracle error code) belonging to the STATUS message
- STATUS: Description of the current activity of the process or the reason why log apply services stopped

The transaction IDs and status information are separated by a string of number signs (###).

This property pertains to a logical standby database.

8.2.5 LsbyParameters (Logical Standby Parameters)

The LsbyParameters monitorable property contains a string that identifies the value of MAX_SGA (maximum system global area) and MAX_SERVERS (maximum number of parallel query servers) specifically reserved for log apply services. These values are separated by a string of number signs (###) in the LsbyParameters property.

This property pertains to a logical standby database.

8.2.6 RecvQEntries (Receive Queue Entries)

The RecvQEntries monitorable property returns a table indicating all log files that were received by the standby database but have not yet been applied. If no rows are returned, it implies all log files received have been applied. This property pertains to a standby database.

The table contains the following columns in the order shown:

- STATUS

The STATUS column is set to one of the following values for a log file on a logical standby database:

- NOT_APPLIED: No redo records in this log file have been applied.
- PARTIALLY_APPLIED: Some of the redo records in this log file have been applied while others have not.
- COMMITTED_TRANSACTIONS_APPLIED: This status value only applies to a logical standby database. All redo records belonging to the committed transactions have been applied. Redo records belonging to uncommitted transactions have not been read by LogMiner and may still be needed when the transactions are committed in the future. Therefore, it is not safe yet to discard this online redo log file.

- RESETLOGS_ID

Resetlogs identifier associated with the archived redo log file

- THREAD

The redo thread number

- LOG_SEQ

The online redo log file sequence number

- TIME_GENERATED

The first time when the online redo log file was written to the primary database

- TIME_COMPLETED

The next time when the log file was archived on the primary database (corresponds to the NEXT_CHANGE# column)

- FIRST_CHANGE#

First change number in the archived redo log file

- NEXT_CHANGE#

First change in the next log file

- SIZE (KBs)

The SIZE of the online redo log file in kilobytes

For example:

```
DGMGR> SHOW DATABASE 'South_Sales' 'RecvQEntries' ;
          STATUS      RESETLOGS_ID      THREAD
LOG_SEQ   TIME_GENERATED      TIME_COMPLETED      FIRST_CHANGE#
NEXT_CHANGE#  SIZE (KBs)
          NOT_APPLIED      497198843           1           5
06/20/2003 14:55:38 06/20/2003 16:31:26      202138
210718      7364
          NOT_APPLIED      497198843           1           6
06/20/2003 16:31:26 06/20/2003 16:31:39      210718
210753      13
          NOT_APPLIED      497198843           1           7
06/20/2003 16:31:39 06/20/2003 16:31:54      210753
210758      1
          NOT_APPLIED      497198843           1           8
```

06/20/2003 16:31:54 06/20/2003 16:31:59 210758
 210789 11

Note: Cloud Control displays this information on the Log File Details page.

8.2.7 SendQEntries (Send Queue Entries)

The `SendQEntries` monitorable property returns a table that shows all log files on the primary database that were not successfully archived to one or more standby databases. This property pertains to the primary database.

The table contains the following columns:

- `STANDBY_NAME`

The value can be empty or it can contain the database unique name (`DB_UNIQUE_NAME`) of a standby database. If empty, the `STATUS` column will contain a value of `CURRENT` or `NOT_ARCHIVED`.

- `STATUS`

The `STATUS` column is set to one of the following values:

- `CURRENT`: A log file to which online redo is currently being written.
- `NOT_ARCHIVED`: A completed online redo log file that has not been archived locally.
- `ARCHIVED`: A completed log file that has been archived locally but has not been transmitted to the standby database specified in the `STANDBY_NAME` column.

The table contains exactly one row with the value of `STATUS=CURRENT`. There can be multiple rows with the value `STATUS=ARCHIVED` or `STATUS=NOT_ARCHIVED`.

- `RESETLOGS_ID`

Resetlogs identifier associated with the archived redo log file

- `THREAD`

The redo thread number.

- `LOG_SEQ`

The log sequence number. Multiple rows may have the same `LOG_SEQ` value (for different `STANDBY_NAME` values).

- `TIME_GENERATED`

The first time when the online redo log file was written to the primary database.

- `TIME_COMPLETED`

The next time when the log file was archived on the primary database (corresponds to the `NEXT_CHANGE#` column).

- `FIRST_CHANGE#`

First change number in the archived redo log file.

- `NEXT_CHANGE#`

First change in the next log file.

- `SIZE (KBs)`

The SIZE of the online redo log file in kilobytes.

For example, the following shows output from a SHOW DATABASE command:

```
DGMGRL> SHOW DATABASE 'North_Sales' 'SendQEntries' ;
PRIMARY_SEND_QUEUE
      STANDBY_NAME      STATUS      RESETLOGS_ID
THREAD      LOG_SEQ      TIME_GENERATED      TIME_COMPLETED
FIRST_CHANGE#      NEXT_CHANGE#      SIZE (Kbs)
      South_Sales      ARCHIVED      497198843
1           9 06/20/2003 16:31:59 06/20/2003 16:39:57
210789      211411      186
      South_Sales      ARCHIVED      497198843
1           10 06/20/2003 16:39:57 06/20/2003 16:40:01
211411      211415      1
      South_Sales      ARCHIVED      497198843
1           11 06/20/2003 16:40:01 06/20/2003 16:40:07
211415      211418      1
                        CURRENT      497198843
1           12 06/20/2003 16:40:07
211418      1
```

Note: Cloud Control displays this information on the Log File Details page.

8.2.8 TopWaitEvents

The TopWaitEvents monitorable property specifies the 5 events with the longest waiting time in the specified instance. The events and their waiting time are retrieved from V\$SYSTEM_EVENT. Each instance in the configuration has this property. This property is an instance-specific monitorable property. The table contains the following columns in the order shown:

- Event
 - The system wait event.
- Wait Time
 - The total amount of time waited for this event in hundredths of a second.

The following example shows output from a SHOW INSTANCE command:

```
DGMGRL> SHOW INSTANCE north_sales1 'TopWaitEvents';
TOP SYSTEM WAIT EVENTS
      Event      Wait Time
      rdbms ipc message      671350
SQL*Net message from client      62390
      pmon timer      47897
Queue Monitor Wait      43016
wakeup time manager      38508
```

8.3 Configurable Properties

Configurable properties control the behavior of broker configuration members. You can view and dynamically update the values of these properties using either DGMGRL or Cloud Control. However, some properties can only be updated through DGMGRL.

These properties control certain database initialization parameters and SQL statements that the broker uses to manage an Oracle Data Guard broker configuration. Therefore,

you should not manually set those initialization parameters or issue those SQL statements on a broker configuration. See "[Broker Controlled Database Initialization Parameters and SQL Statements](#)" on page 8-19.

In most cases, the configurable property is said to apply to the entire member, meaning the value you set for the property applies to each instance of the member (that is, database or far sync instance). However, in a few cases, the configurable property is said to have instance-specific scope which means that for a multiple-instance database or far sync instance, it is possible that the values of some particular properties may differ between instances. [Table 8-1](#) lists each configurable property and indicates whether the scope of the property is member-wide or instance-specific. If the Scope column contains:

- Member—The value of the property is database or far sync instance-wide, not instance or configuration specific.
- Instance—The value of the property is instance specific.

If there is an asterisk (*) present, it indicates that the property value can be set for all instances of an Oracle RAC database using the `EDIT INSTANCE * ON DATABASE` command.

- Configuration—The value of the property is configuration wide, not instance or member specific.

Table 8-1 Configurable Properties

Configurable Property Name	Scope	Pertains To
AlternateLocation	Instance	Redo transport services
ApplyInstanceTimeout	Member	Redo Apply and SQL Apply
ApplyLagThreshold	Member	Redo transport services
ApplyParallel	Member	Redo Apply
ArchiveLagTarget	Member	Redo transport services
Binding	Member	Redo transport services
DbFileNameConvert	Member	Redo Apply
DelayMins	Member	Redo Apply and SQL Apply
DGConnectIdentifier	Member	Broker communication, Redo transport services
FastStartFailoverTarget	Member	Fast-start failover
LogArchiveFormat	Instance *	Redo transport services
LogArchiveMaxProcesses	Member	Redo transport services
LogArchiveMinSucceedDest	Member	Redo transport services
LogArchiveTrace	Instance *	Diagnosis
LogFileNamesConvert	Member	Redo Apply
LogShipping	Member	Redo transport services
LogXptMode	Member	Redo transport services
LsbyMaxEventsRecorded	Member	SQL Apply
LsbyMaxServers	Instance *	SQL Apply
LsbyMaxSga	Instance *	SQL Apply
LsbyPreserveCommitOrder	Member	SQL Apply

Table 8–1 (Cont.) Configurable Properties

Configurable Property Name	Scope	Pertains To
LsbyRecordAppliedDdl	Member	SQL Apply
LsbyRecordSkipDdl	Member	SQL Apply
LsbyRecordSkipErrors	Member	SQL Apply
MaxConnections	Member	Redo transport services
MaxFailure	Member	Redo transport services
NetTimeout	Member	Redo transport services
ObserverConnectIdentifier	Member	Fast-start failover
PreferredApplyInstance	Member	Redo Apply and SQL Apply
RedoCompression	Member	Redo transport services
RedoRoutes	Member	Redo transport services
ReopenSecs	Member	Redo transport services
SidName	Instance	Instance identification
StandbyArchiveLocation	Instance	Redo transport services
StandbyFileManagement	Member	Redo Apply and SQL Apply
StaticConnectIdentifier	Instance	Instance Startup and Shutdown
TransportDisconnectedThreshold	Member	Redo transport services
TransportLagThreshold	Member	Redo transport services

See Also: [Chapter 4](#) for more information about property management

Note: When a broker configuration with its primary database is created and members are added to the configuration, the broker imports existing settings from the members to set many of the properties. If importing an existing setting fails, or if a property value is not imported, then the broker uses a broker default value. The default values and whether or not a property is imported is indicated within each property description.

Broker Controlled Database Initialization Parameters and SQL Statements

The following database initialization parameters are controlled by broker configurable properties. Therefore, you should not set these parameters manually:

- LOG_ARCHIVE_DEST_n
- LOG_ARCHIVE_DEST_STATE_n
- ARCHIVE_LAG_TARGET
- DB_FILE_NAME_CONVERT
- LOG_ARCHIVE_FORMAT
- LOG_ARCHIVE_MAX_PROCESSES
- LOG_ARCHIVE_MIN_SUCCEED_DEST

- LOG_ARCHIVE_TRACE
- LOG_FILE_NAME_CONVERT
- STANDBY_FILE_MANAGEMENT

The broker also uses configurable property settings to manage how apply is to be started. Therefore, the following SQL statements are managed automatically by the broker:

- ALTER DATABASE RECOVER MANAGED STANDBY DATABASE
- ALTER DATABASE START LOGICAL STANDBY APPLY IMMEDIATE

The broker sets the following apply parameters on a logical standby automatically; you should not manually set them:

- MAX_EVENTS_RECORDED
- MAX_SERVERS
- MAX_SGA
- PRESERVE_COMMIT_ORDER
- RECORD_APPLIES_DDL
- RECORD_SKIP_DDL
- RECORD_SKIP_ERRORS

8.3.1 AlternateLocation

The AlternateLocation configurable property specifies an alternate disk location to store the archived redo log files in the standby when the location specified by the StandbyArchiveLocation configurable property fails. The property has instance-specific scope, and the location it specifies has to be accessible by the instance.

Category	Description
Datatype	String
Valid values	Directory specification on system where the standby instance is located
Broker default	Empty string
Imported?	No
Parameter class	Dynamic
Role	Standby database, far sync instance
Standby type	Physical, logical, or snapshot standby, or a far sync instance
Corresponds to	<ul style="list-style-type: none"> ■ On the standby database or the far sync instance, the LOCATION attribute for the LOG_ARCHIVE_DEST_n initialization parameter that represents an alternate destination of the local destination that matches the configurable property StandbyArchiveLocation.
Scope	Instance
Cloud Control name	Alternate Standby Location

Note: On a logical standby database, Oracle recommends the LOCATION attribute of the LOG_ARCHIVE_DEST_# initialization parameter for the local destination be different from the value of AlternateLocation configurable property.

8.3.2 ApplyInstanceTimeout

The ApplyInstanceTimeout configurable property specifies the number of seconds the broker waits after detecting the current apply instance failed before initiating the apply instance failover.

Category	Description
Datatype	Integer
Valid values	>=0 (seconds)
Broker default	0 (results in immediate apply instance failover)
Imported?	No
Parameter class	Not applicable
Role	Standby
Standby type	Physical, logical
Corresponds to	Not applicable
Scope	Database
Cloud Control name	Not applicable

8.3.3 ApplyLagThreshold

The ApplyLagThreshold configurable property generates a warning status for a logical or physical standby when the member's apply lag exceeds the value specified by the property. The property value is expressed in seconds. A value of 0 seconds results in no warnings being generated when an apply lag exists.

Category	Description
Datatype	Number
Valid values	>=0
Broker default	0 seconds
Imported?	No
Parameter class	Not applicable
Role	Standby
Standby type	Physical, logical
Corresponds to	Not applicable
Scope	Database
Cloud Control name	Not applicable

8.3.4 ApplyParallel

The `ApplyParallel` configurable property specifies whether Redo Apply should use multiple processes to apply redo data to the physical standby database. If Redo Apply is shut off, then setting the property has no immediate effect. However, when Redo Apply is running again, the value of the property is used to determine the parallel apply behavior of Redo Apply.

Category	Description
Datatype	String
Valid values	<ul style="list-style-type: none"> ▪ <code>AUTO</code>—the number of parallel processes used for Redo Apply is automatically determined by Oracle based on the number of CPUs that the system has. ▪ <code>NO</code>—no parallel apply ▪ <code>2, 3, and so on</code>—manually specify the number of parallel processes used for Redo Apply. (Specifying <code>0</code> is the same as specifying <code>NO</code>; specifying <code>1</code> is the same as specifying <code>AUTO</code>.)
Broker default	<code>AUTO</code>
Imported?	No
Parameter class	Not applicable
Role	Standby
Standby type	Physical
Corresponds to	<ul style="list-style-type: none"> ▪ <code>AUTO</code> corresponds to the <code>PARALLEL</code> clause of the <code>ALTER DATABASE RECOVER MANAGED STANDBY DATABASE</code> statement ▪ <code>NO</code> corresponds to the <code>NOPARALLEL</code> clause of the <code>ALTER DATABASE RECOVER MANAGED STANDBY DATABASE</code> statement ▪ <code>2, 3, and so on</code> corresponds to the <code>PARALLEL n</code> clause of the <code>ALTER DATABASE RECOVER MANAGED STANDBY DATABASE</code> statement
Scope	Database
Cloud Control name	Not applicable

8.3.5 ArchiveLagTarget

The `ArchiveLagTarget` configurable property limits the amount of data that can be lost and, effectively, increases the availability of the standby database by forcing a log switch after the amount of time you specify (in seconds) elapses. That way, the standby database will not miss redo records generated from a time range longer than the value set for the `ARCHIVE_LAG_TARGET` initialization parameter.

Category	Description
Datatype	Number
Valid values	Seconds (either 0 seconds, or any number from 60 to 7200 seconds)
Broker default	0 (disabled)
Imported?	Yes, from the <code>ARCHIVE_LAG_TARGET</code> initialization parameter
Parameter class	Dynamic
Role	Primary database, far sync instance
Standby type	Not applicable

Category	Description
Corresponds to	ARCHIVE_LAG_TARGET= <i>seconds</i> initialization parameter
Scope	Database
Cloud Control name	Archive Lag Target

8.3.6 Binding

The Binding configurable property specifies whether the destination is MANDATORY or OPTIONAL.

Category	Description
Datatype	String
Valid values	<ul style="list-style-type: none"> ■ MANDATORY You can specify a policy for reuse of online redo log files using the MANDATORY value. If the archiving operation of a mandatory destination fails, online redo log files cannot be overwritten. ■ OPTIONAL You can specify a policy for reuse of online redo log files using the OPTIONAL value. If the archiving operation of an optional destination fails, the online redo log files are overwritten.
Broker default	OPTIONAL
Imported?	No
Parameter class	Dynamic
Role	Standby database, far sync instance ¹
Standby type	Physical, logical, or snapshot standby, or a far sync instance
Corresponds to	<ul style="list-style-type: none"> ■ MANDATORY and OPTIONAL attributes for the LOG_ARCHIVE_DEST_<i>n</i> initialization parameter of the database or far sync instance that is sending redo data ■ BINDING column of the V\$ARCHIVE_DEST view of the database or far sync instance that is sending redo data
Scope	Database, far sync instance
Cloud Control name	Not applicable

¹ Although this property is set for the redo destination, it is indirectly related to the redo transport services for the database or far sync instance that is sending redo data. The broker propagates the setting you specify to the corresponding attributes of the LOG_ARCHIVE_DEST_*n* value of the database or far sync instance that is sending redo data.

8.3.7 DbFileNameConvert

The DbFileNameConvert configurable property distinguishes physical standby datafile filenames from primary datafile filenames. If you add a datafile to the primary database, this property converts the datafile name on the primary database to the datafile on the physical standby database.

This property is used in the following situations:

- At physical standby mount time, it is used to rename primary datafile filenames to standby datafile filenames if the datafile file path on the physical standby system is different from the primary database system.

- When a new data file is created on the primary database, a corresponding new data file will be created on the physical standby database if the `StandbyFileManagement` configurable property is set to 'AUTO'. Oracle uses the data-file file-path mapping information from the `DbFileNameConvert` property to determine the standby file path of the new standby data file. If the `StandbyFileManagement` property is set to 'MANUAL', you must add a corresponding file to the physical standby database.

Note: When a database is added to the configuration, the broker sets the initial value of this property to the in-memory value of the `DB_FILE_NAME_CONVERT` initialization parameter. It is possible that the in-memory value and server parameter file (SPFILE) value of this parameter will differ. If you want to use the parameter's in-memory value, then enable the database and the broker will ensure that the SPFILE value of the parameter is set to the in-memory value. If you want to use the SPFILE value, then set the property value to be the parameter's value stored in the SPFILE. Then enable the database.

Category	Description
Datatype	String
Valid values	<p>Set the value of this property to a list of string pairs:</p> <ol style="list-style-type: none"> The first string is the substring found in the datafile names on the primary database. The second string is the substring found in the datafile names on the standby database. <p>For example, ('string1', 'string2', 'string3', 'string4',...) Where:</p> <ul style="list-style-type: none"> string1 is the substring of the primary database filename. string2 is the substring of the standby database filename. string3 is the substring of the primary database filename. string4 is the substring of the standby database filename.
Broker default	Empty string
Imported?	Yes, from the <code>DB_FILE_NAME_CONVERT</code> initialization parameter
Parameter class	Static
Role	Standby
Standby type	Physical
Corresponds to	<code>DB_FILE_NAME_CONVERT</code> initialization parameter
Scope	Database
Cloud Control name	DB File Name Convert

8.3.8 DelayMins

The `DelayMins` configurable property specifies the number of minutes log apply services will delay applying the archived redo log data on the standby database. When the `DelayMins` property is set to the default value of 0 minutes, log apply services apply redo data as soon as possible.

If the `DelayMins` property is set to 0, start log apply services as follows:

- Start Redo Apply on physical standby databases using the following SQL statement:

```
ALTER DATABASE RECOVER MANAGED STANDBY DATABASE;
```

- Start SQL Apply on logical standby databases using the following SQL statement:

```
ALTER DATABASE START LOGICAL STANDBY APPLY IMMEDIATE;
```

Category	Description
Datatype	Integer
Valid values	>=0 (minutes)
Broker default	0
Imported?	No
Parameter class	Dynamic
Role	Standby ¹
Standby type	Physical, logical
Corresponds to	<ul style="list-style-type: none"> ■ <code>DELAY</code> attribute for the <code>LOG_ARCHIVE_DEST_n</code> initialization parameter of the database or far sync instance that is sending redo data ■ <code>DELAY_MINS</code> column of the <code>V\$ARCHIVE_DEST</code> view of the database or far sync instance that is sending redo data ■ Options used to start Redo Apply and SQL Apply
Scope	Database
Cloud Control name	Apply Delay (mins)

¹ Although this property is set for the standby database, it is indirectly related to the redo transport services for the database or far sync instance that is sending redo data. The broker propagates the setting you specify on the standby database to the corresponding attributes of the `LOG_ARCHIVE_DEST_n` value of the database or far sync instance that is sending redo data.

8.3.9 DGConnectIdentifier

The `DGConnectIdentifier` configurable property specifies the connection identifier the broker uses when making connections to a configuration member. If using `DGMGRL`, you supply the value when you enter the `CREATE CONFIGURATION`, `ADD DATABASE`, or `ADD FAR_SYNC` command. If you are using Cloud Control, the value is supplied automatically. The connect identifier for a configuration member must:

- Allow all other members in the configuration to reach it.
- Allow the member to reach itself.
- Allow all instances of an Oracle RAC database to be reached.
- Specify a service that all instances dynamically register with the listeners so that connect-time failover on an Oracle RAC database is possible.
- Have failover attributes set to allow the primary database's Redo Transport Services to continue shipping redo data to an Oracle RAC standby database, even if the receiving instance of that standby database has failed.

The value of this property is specified in the `SERVICE` attribute of the `LOG_ARCHIVE_DEST_n` parameter when the broker configures redo transport services on the primary database.

Category	Description
Datatype	String
Valid values	A connect identifier that can be used to connect to this database
Broker default	Not applicable
Imported?	No
Parameter class	Not applicable
Role	Primary, standby, far sync instance
Standby type	Physical, logical, or snapshot standby, or a far sync instance
Corresponds to	<code>SERVICE_NAME</code> attribute of the <code>LOG_ARCHIVE_DEST_n</code> initialization parameter of the configuration member that is sending redo data
Scope	Database, far sync instance
Cloud Control name	Not applicable

See Also: *Oracle Database Net Services Administrator's Guide*

8.3.10 FastStartFailoverTarget

The `FastStartFailoverTarget` configuration property specifies the `DB_UNIQUE_NAME` of the database that will be the target of a fast-start failover when this database is the primary database. See Step 2 in [Section 5.5.2](#) for more information about setting this property.

The `FastStartFailoverTarget` configuration property can only be set to the name of a physical or logical standby. It cannot be set to the name of a snapshot standby database or far sync instance.

Category	Description
Datatype	String
Valid value	<code>DB_UNIQUE_NAME</code> of the database that is the target of the fast-start failover.
Broker default	<p>If only one physical or logical standby database exists, then the broker selects that as the default value for this property on the primary database when fast-start failover is enabled.</p> <p>If more than one physical or logical standby database exists, you must specify the <code>FastStartFailoverTarget</code> value explicitly, prior to enabling fast-start failover. The indicated target is verified when fast-start failover is enabled.</p> <p>For the target standby database, the broker automatically selects the current primary database as the value for this property when fast-start failover is enabled.</p>
Imported?	No
Parameter class	Not applicable
Role	Primary or standby

Category	Description
Standby type	Physical or logical
Corresponds to	Not applicable
Scope	Database
Cloud Control name	Cloud Control displays the value for the current primary database on the Oracle Data Guard Overview page, along with whether or not fast-start failover has been enabled.

8.3.11 LogArchiveFormat

The `LogArchiveFormat` configurable instance-specific property specifies the format for filenames of archived redo log files using a database ID (%d), thread (%t), sequence number (%s), and resetlogs ID (%r).

Note: When a member is added to the configuration, the broker sets the initial value of this property to the in-memory value of the `LOG_ARCHIVE_FORMAT` initialization parameter. It is possible that the in-memory value and server parameter file (SPFILE) value of this parameter will differ. If you want to use the parameter's in-memory value, then enable the member and the broker will ensure that the SPFILE value of the parameter is set to the in-memory value. If you want to use the SPFILE value, then set the property value to be the parameter's value stored in the SPFILE. Then enable the member.

Category	Description
Datatype	String
Valid values	%d_%t_%s_%r
Broker default	Empty string
Imported?	Yes, from the <code>LOG_ARCHIVE_FORMAT</code> initialization parameter on the primary database
Parameter class	Static
Role	Primary, standby, far sync instance
Standby type	Physical, logical, or snapshot standby, or a far sync instance
Corresponds to	<code>LOG_ARCHIVE_FORMAT</code> initialization parameter
Scope	Instance (On an Oracle RAC database, you can use the <code>EDIT INSTANCE * ON DATABASE</code> command to have all instances use the same value for this property.)
Cloud Control name	Not applicable

8.3.12 LogArchiveMaxProcesses

The `LogArchiveMaxProcesses` configurable property specifies the initial number of archiver processes (`ARCn`) that are invoked. The actual number of archiver processes in use may increase subsequently based on the archiving workload.

Category	Description
Datatype	Integer
Valid values	1 to 30
Broker default	4
Imported?	Yes, from the LOG_ARCHIVE_MAX_PROCESSES initialization parameter
Parameter class	Dynamic
Role	Primary, standby, far sync instance
Standby type	Physical, logical, or snapshot standby, or a far sync instance
Corresponds to	LOG_ARCHIVE_MAX_PROCESSES initialization parameter
Scope	Database, far sync instance
Cloud Control name	Archiver Processes

8.3.13 LogArchiveMinSucceedDest

The LogArchiveMinSucceedDest configurable property controls when online redo log files are available for reuse. For the online redo log files to be available for reuse, archiving must succeed to a minimum number of destinations.

Category	Description
Datatype	Integer
Valid values	1 to 10
Broker default	1
Imported?	Yes, from the LOG_ARCHIVE_MIN_SUCCEED_DEST initialization parameter
Parameter class	Dynamic
Role	Primary database, far sync instance
Standby type	Not applicable
Corresponds to	LOG_ARCHIVE_MIN_SUCCEED_DEST initialization parameter
Scope	Database, far sync instance
Cloud Control name	Not applicable

8.3.14 LogArchiveTrace

Set the LogArchiveTrace configurable instance-specific property to an integer value to see the progression of the archiving of online redo log files on the primary and the standby databases. The Oracle database writes an audit trail of the archived redo log files received from the primary database into process trace files.

Category	Description
Datatype	Integer

Category	Description
Valid values	<p>A valid value is the sum of any combination of any of the following values:</p> <ul style="list-style-type: none"> 0: Disable archive redo log tracing 1: Track archiving of online redo log file 2: Track archiving status of each archive redo log destination 4: Track archiving operational phase 8: Track ARCHIVELOG destination activity 16: Track detailed ARCHIVELOG destination activity 32: Track ARCHIVELOG destination parameter modifications 64: Track ARCn process state activity 128: Track FAL (fetch archive log) server related activities 256: Tracks RFS Logical Client 512: Tracks LGWR redo shipping network activity 1024: Tracks RFS physical client 2048: Tracks RFS/ARCn ping heartbeat 4096: Tracks real-time apply activity 8192: Tracks Redo Apply (media recovery or physical standby) 16384: Tracks archive I/O buffers 32768: Tracks LogMiner dictionary archiving
Broker default	255
Imported?	Yes, from the LOG_ARCHIVE_TRACE initialization parameter
Parameter class	Dynamic
Role	Primary, standby, far sync instance
Standby type	Physical, logical, or snapshot standby, or a far sync instance
Corresponds to	LOG_ARCHIVE_TRACE initialization parameter
Scope	Instance (On an Oracle RAC database, you can use the <code>EDIT INSTANCE * ON DATABASE</code> command to have all instances use the same value for this property.)
Cloud Control name	Log Archive Trace

8.3.15 LogFileNameConvert

The `LogFileNameConvert` configurable property converts the filename of an online redo log file on the primary database to the filename of a corresponding online redo log file on the physical standby database.

Note: When a database is added to the configuration, the broker sets the initial value of this property to the in-memory value of the LOG_FILE_NAME_CONVERT initialization parameter. It is possible that the in-memory value and server parameter file (SPFILE) value of this parameter will differ. If you want to use the parameter's in-memory value, then enable the database and the broker will ensure that the SPFILE value of the parameter is set to the in-memory value. If you want to use the SPFILE value, then set the property value to be the parameter's value stored in the SPFILE. Then enable the database.

Category	Description
Datatype	String
Valid values	<p>Set the value of this property to a list of an even number of string pairs, separated by commas.</p> <ol style="list-style-type: none"> 1. The first string is the substring found in the datafile names on the primary database. 2. The second string is the substring found in the datafile names on the standby database. <p>For example, ('string1', 'string2', 'string3', 'string4',...)</p> <p>Where:</p> <ul style="list-style-type: none"> ■ string1 is the substring of the primary database filename. ■ string2 is the substring of the standby database filename. ■ string3 is the substring of the primary database filename. ■ string4 is the substring of the standby database filename.
Broker default	Empty string
Imported?	Yes, from the LOG_FILE_NAME_CONVERT initialization parameter
Parameter class	Static
Role	Standby
Standby type	Physical database and far sync instance
Corresponds to	LOG_FILE_NAME_CONVERT initialization parameter
Scope	Database
Cloud Control name	Log File Name Convert

8.3.16 LogShipping

The broker uses the value of the LogShipping property when the primary database is in the TRANSPORT-ON state or when the physical standby or far sync instance forwards redo data to another member (physical, logical, or snapshot standby, or a far sync instance).

- If the primary database is in the TRANSPORT-ON state and the value of the LogShipping property is ON, then redo transport services are enabled to send redo data to the particular configuration member. If the LogShipping property is OFF, then redo transport services are disabled to that member.
- If a configuration member that forwards redo data has its LogShipping property set to ON and the member to which it sends redo data also has its LogShipping

property set to **ON**, then redo transport services are enabled from the member sending redo data to the member that is to receive redo data.

If a member that forwards redo data has its `LogShipping` property set to **ON** but the member to which it sends redo data has its `LogShipping` property set to **OFF**, then redo transport services are disabled from the member sending redo data to the member that is to receive redo data.

Category	Description
Datatype	String
Valid values	ON or OFF
Broker default	ON
Imported?	No
Parameter class	Dynamic
Role	Standby database, far sync instance ¹
Standby type	Physical, logical, or snapshot standby, or a far sync instance
Corresponds to	ENABLE and DEFER values for the <code>LOG_ARCHIVE_DEST_STATE_n</code> initialization parameter of the database or far sync instance that is sending redo data
Scope	Database
Cloud Control name	Log Shipping

¹ Although this property is set for a standby database or far sync instance, it is indirectly related to the redo transport services for the database or far sync instance that is sending redo data. The broker propagates the setting you specify on the standby database to the corresponding attributes of the `LOG_ARCHIVE_DEST_n` value of the database or far sync instance that is sending redo data.

8.3.17 LogXptMode

The `LogXptMode` configurable property enables you to set the redo transport service. You set the redo transport services on each configuration member to one of the following modes:

- **SYNC**
Configures redo transport services for this configuration member using the `SYNC` and `AFFIRM` attributes of the `LOG_ARCHIVE_DEST_n` initialization parameter. Standby redo log files are required. This mode is required for the maximum protection or maximum availability data protection modes. This redo transport service enables the highest grade of data protection to the primary database, but also incurs the highest performance impact.
- **ASYNC**
Configures redo transport services for this configuration member using the `ASYNC` and `NOAFFIRM` attributes of the `LOG_ARCHIVE_DEST_n` initialization parameter. Standby redo log files are required. This mode enables a moderate grade of data protection to the primary database, and incurs a lower performance impact than `SYNC`.
- **FASTSYNC**
Configures redo transport services for this configuration member using the `SYNC` and `NOAFFIRM` attributes of the `LOG_ARCHIVE_DEST_n` initialization parameter. This mode is only available in maximum availability protection mode. Because

FASTSYNC transport mode uses the NOAFFIRM attribute of the LOG_ARCHIVE_DEST_n parameter, data loss is possible.

Category	Description
Datatype	String
Valid values	SYNC, ASYNC, or FASTSYNC
Broker default	ASYNC
Imported?	No
Parameter class	Dynamic
Role	Standby database and far sync instance ¹
Standby type	Physical, logical, or snapshot standby, or a far sync instance
Corresponds to	<ul style="list-style-type: none"> ■ SYNC, ASYNC, AFFIRM, and NOAFFIRM attributes for the LOG_ARCHIVE_DEST_n initialization parameter of the database or far sync instance that is sending redo data ■ ARCHIVER, TRANSMIT_MODE, and AFFIRM columns of V\$ARCHIVE_DEST view of the database or far sync instance that is sending redo data
Scope	Database, far sync instance
Cloud Control name	Redo Transport Service

¹ Although this property is set for the standby database or far sync instance, it is indirectly related to the redo transport services for the database or far sync instance that is sending redo. The broker propagates the setting you specify on the standby database or far sync instance to the corresponding attributes of the LOG_ARCHIVE_DEST_n value of the database or far sync instance that is sending redo data. Note that if a database receives redo from a database or far sync instance where the RedoRoutes property has been configured with a redo transport mode, then the mode specified by that RedoRoutes property value overrides the value of the LogXptMode property.

See Also: [Chapter 4](#) for more information about setting data protection modes for redo transport services

8.3.18 LsbyMaxEventsRecorded

The LsbyMaxEventsRecorded configurable property specifies the number of events that will be stored in the DBA_LOGSTDBY_EVENTS table, which stores logical standby event information.

Category	Description
Datatype	Integer
Valid values	>=0
Broker default	0
Imported?	Yes, from the MAX_EVENTS_RECORDED row of SYSTEM.LOGSTDBY\$PARAMETERS
Parameter class	Dynamic; SQL Apply does not require restart
Role	Standby
Standby type	Logical
Corresponds to	DBMS_LOGSTDBY.APPLY_SET('MAX_EVENTS_RECORDED') and the DBMS_LOGSTDBY.APPLY_UNSET('MAX_EVENTS_RECORDED') procedures

Category	Description
Scope	Database
Cloud Control name	Max Events Recorded

8.3.19 LsbyMaxServers

The `LsbyMaxServers` configurable instance-specific property specifies the number of parallel query servers specifically reserved for SQL Apply. If the value is 0, SQL Apply uses all available parallel query servers to read the log files and apply changes.

Category	Description
Datatype	Integer
Valid values	>=0
Broker default	0
Imported?	Yes, from the <code>MAX_SERVERS</code> row of <code>SYSTEM.LOGSTDBY\$PARAMETERS</code>
Parameter class	Dynamic; SQL Apply does not require restart
Role	Standby
Standby type	Logical
Corresponds to	<code>DBMS_LOGSTDBY.APPLY_SET('MAX_SERVERS')</code> and the <code>DBMS_LOGSTDBY.APPLY_UNSET('MAX_SERVERS')</code> procedures
Scope	Instance (On an Oracle RAC database, you can use the <code>EDIT INSTANCE * ON DATABASE</code> command to have all instances use the same value for this property.)
Cloud Control name	Max Servers

8.3.20 LsbyMaxSga

The `LsbyMaxSga` configurable instance-specific property specifies the number of megabytes for the allocation of SQL Apply cache in the system global area (SGA). If the value is 0, SQL Apply uses one quarter of the value set for the `SHARED_POOL_SIZE` initialization parameter.

Category	Description
Datatype	Integer
Valid values	>=0
Broker default	0
Imported?	Yes, from the <code>MAX_SGA</code> row of <code>SYSTEM.LOGSTDBY\$PARAMETERS</code>
Parameter class	Dynamic; SQL Apply does not require restart
Role	Standby
Standby type	Logical
Corresponds to	<code>DBMS_LOGSTDBY.APPLY_SET('MAX_SGA')</code> and the <code>DBMS_LOGSTDBY.APPLY_UNSET('MAX_SGA')</code> procedures

Category	Description
Scope	Instance (On an Oracle RAC database, you can use the <code>EDIT INSTANCE * ON DATABASE</code> command to have all instances use the same value for this property.)
Cloud Control name	Max SGA (MB)

8.3.21 LsbyPreserveCommitOrder

The `LsbyPreserveCommitOrder` configurable property controls whether transactions are committed on the logical standby database in the exact same order in which they were committed on the primary database. Specify one of the following values:

- `TRUE`: Transactions are applied to the logical standby database in the exact order in which they were committed on the primary database.
- `FALSE`: Transactions containing non-overlapping sets of rows may be committed in a different order than they were committed on the primary database.

Category	Description
Datatype	String
Valid values	TRUE or FALSE
Broker default	TRUE
Imported?	Yes, from the <code>PRESERVE_COMMIT_ORDER</code> row of <code>SYSTEM.LOGSTDBY\$PARAMETERS</code>
Parameter class	Static; SQL Apply requires restart
Role	Standby
Standby type	Logical
Corresponds to	<code>DBMS_LOGSTDBY.APPLY_SET('PRESERVE_COMMIT_ORDER')</code> and <code>DBMS_LOGSTDBY.APPLY_UNSET('PRESERVE_COMMIT_ORDER')</code> procedures
Scope	Database
Cloud Control name	Preserve Commit Order

8.3.22 LsbyRecordAppliedDdl

The `LsbyRecordAppliedDdl` configurable property controls whether or not SQL statements that were applied to the logical standby database are recorded in the `DBA_LOGSTDBY_EVENTS` table. Specify one of the following values:

- `TRUE`: DDL statements applied to the logical standby database are recorded in the `DBA_LOGSTDBY_EVENTS` table. This is the default setting.
- `FALSE`: Applied DDL statements are not recorded.

Category	Description
Datatype	String
Valid values	TRUE or FALSE
Broker default	TRUE

Category	Description
Imported?	Yes, from the RECORD_APPLIED_DDL row of SYSTEM.LOGSTDBY\$PARAMETERS
Parameter class	Dynamic; SQL Apply does not require restart
Role	Standby
Standby type	Logical
Corresponds to	DBMS_LOGSTDBY.APPLY_SET('RECORD_APPLIED_DDL') and the DBMS_LOGSTDBY.APPLY_UNSET('RECORD_APPLIED_DDL') procedures
Scope	Database
Cloud Control name	Record Applied DDL

8.3.23 LsbyRecordSkipDdl

The LsbyRecordSkipDdl configurable property controls whether or not skipped DDL statements are recorded in the DBA_LOGSTDBY_EVENTS table. Specify one of the following values:

- TRUE: Skipped DDL statements are recorded in the DBA_LOGSTDBY_EVENTS table. This is the default setting.
- FALSE: Skipped DDL statements are not recorded in the DBA_LOGSTDBY_EVENTS table.

Category	Description
Datatype	String
Valid values	TRUE or FALSE
Broker default	TRUE
Imported?	Yes, from the RECORD_SKIP_DDL row of SYSTEM.LOGSTDBY\$PARAMETERS
Parameter class	Dynamic; SQL Apply does not require restart
Role	Standby
Standby type	Logical
Corresponds to	DBMS_LOGSTDBY.APPLY_SET('RECORD_SKIP_DDL') and the DBMS_LOGSTDBY.APPLY_UNSET('RECORD_SKIP_DDL') procedures
Scope	Database
Cloud Control name	Record Skip DDL

8.3.24 LsbyRecordSkipErrors

The LsbyRecordSkipErrors configurable property controls whether or not skipped errors (as described by the DBMS_LOGSTDBY.SKIP_ERROR procedure) are recorded in the DBA_LOGSTDBY_EVENTS table. Specify one of the following values:

- TRUE: Skipped errors are recorded in the DBA_LOGSTDBY_EVENTS table.
- FALSE: Skipped errors are not recorded in the DBA_LOGSTDBY_EVENTS table.

Category	Description
Datatype	String
Valid values	TRUE or FALSE
Broker default	TRUE
Imported?	Yes, from the RECORD_SKIP_ERRORS row of SYSTEM.LOGSTDBY\$PARAMETERS
Parameter class	Dynamic; SQL Apply does not require restart
Role	Standby
Standby type	Logical
Corresponds to	DBMS_LOGSTDBY.APPLY_SET('RECORD_SKIP_ERRORS') and the DBMS_LOGSTDBY.APPLY_UNSET('RECORD_SKIP_ERRORS') procedures
Scope	Database
Cloud Control name	Record Skip Errors

8.3.25 MaxConnections

The `MaxConnections` configurable property specifies how many `ARCn` processes will be used in parallel to transmit redo data from a single archived redo log on the database or far sync instance to the archived redo log at the remote site. If the `MaxConnections` property is set to a value greater than 1, then redo transport services use multiple `ARCn` processes to transmit redo data to archived redo log files at the remote destinations.

Category	Description
Datatype	Integer
Valid values	1 to 20
Broker default	1
Imported?	No
Parameter class	Dynamic
Role	Standby
Standby type	Physical, logical, snapshot
Corresponds to	<ul style="list-style-type: none"> ■ LOG_ARCHIVE_MAX_PROCESSES initialization parameter¹ ■ MAX_CONNECTIONS attribute for the LOG_ARCHIVE_DEST_{<i>n</i>} initialization parameter of the database or far sync instance that is sending redo data ■ MAX_CONNECTIONS column of the V\$ARCHIVE_DEST view of the database or far sync instance that is sending redo data
Scope	Database
Cloud Control name	Not applicable

¹ The LOG_ARCHIVE_MAX_PROCESSES initialization parameter affects the actual number of `ARCn` processes used by an instance.

8.3.26 MaxFailure

The `MaxFailure` configurable property specifies the maximum number of contiguous archiving failures before the redo transport services stop trying to transport archived redo log files to the standby database. A value of zero indicates that an unlimited number of failures are allowed.

Category	Description
Datatype	Integer
Valid values	>=0
Broker default	0
Imported?	No
Parameter class	Dynamic
Role	Standby ¹
Standby type	Physical, logical, snapshot
Corresponds to	<ul style="list-style-type: none"> ■ <code>MAX_FAILURE</code> attribute for the <code>LOG_ARCHIVE_DEST_n</code> initialization parameter of the database or far sync instance that is sending redo data ■ <code>MAX_FAILURE</code> column of the <code>V\$ARCHIVE_DEST</code> view of the database or far sync instance that is sending redo data
Scope	Database
Cloud Control name	Not applicable

¹ Although this property is set for a standby database or far sync instance, it is indirectly related to the redo transport services for the database or far sync instance that is sending redo data. The broker propagates the setting you specify on the standby database or far sync instance to the corresponding attributes of the `LOG_ARCHIVE_DEST_n` value of the database or far sync instance that is sending redo data.

8.3.27 NetTimeout

The `NetTimeout` configurable property specifies the number of seconds the LGWR waits for Oracle Net Services to respond to a LGWR request. It is used to bypass the long connection timeout in TCP.

Category	Description
Datatype	Integer
Valid values	0, 15 to 1200
Broker default	30
Imported?	No
Parameter class	Dynamic
Role	Primary, standby, far sync instance
Standby type	Physical, logical, or snapshot standby, or a far sync instance
Corresponds to	<ul style="list-style-type: none"> ■ <code>NET_TIMEOUT</code> attribute of the <code>LOG_ARCHIVE_DEST_n</code> initialization parameter of the database or far sync instance that is sending redo data ■ <code>NET_TIMEOUT</code> column of <code>V\$ARCHIVE_DEST</code> view of the database or far sync instance that is sending redo data

Category	Description
Scope	Database, far sync instance
Cloud Control name	Not applicable

8.3.28 ObserverConnectIdentifier

The `ObserverConnectIdentifier` configurable property specifies a connect identifier that can be used by the observer to connect to this database. This can pertain only to the primary database, or to the target standby database when fast-start failover is enabled.

Category	Description
Datatype	String
Valid Values	A connect identifier that observer can use to connect to this database
Broker Default	Empty String ¹
Imported?	No
Parameter Class	Not applicable
Role	Primary and Standby
Standby Type	Physical or logical
Corresponds to	Not applicable
Scope	Database
Cloud Control name	Observer Connect Identifier

¹ When this is Empty String (not set by the user), the connect identifier specified by this database's `DGConnectIdentifier` property will be used by the observer.

8.3.29 PreferredApplyInstance

The `PreferredApplyInstance` configurable property indicates that a particular instance is the preferred choice for serving log apply services. It is only used when the database is a standby Oracle RAC database. The value could be an empty string (default) which means the broker chooses the apply instance.

Category	Description
Datatype	String
Valid Values	The instance name (SID) or empty string. Note that on certain platforms, SIDs may be case-sensitive.
Broker Default	Empty string
Imported?	No
Parameter Class	Not applicable
Role	Standby
Standby Type	Physical or logical
Corresponds to	Not applicable
Scope	Database

Category	Description
Cloud Control name	Apply Instance

See Also:

- *Oracle Database Administrator's Guide* for more information about SIDs
- [Section 4.5.5](#) for information about apply services in an Oracle RAC environment

8.3.30 RedoCompression

The `RedoCompression` configurable property is used to specify whether redo data is transmitted to a standby database or far sync instance in compressed or uncompressed form.

Note: Redo transport compression is a feature of the Oracle Advanced Compression option. You must purchase a license for this option before using the redo transport compression feature.

Category	Description
Datatype	String
Valid value	DISABLE or ENABLE
Broker Default	DISABLE
Imported?	No
Parameter class	Dynamic
Role	Standby database, far sync instance ¹
Standby type	Physical, logical, or snapshot standby, or a far sync instance
Corresponds to	<ul style="list-style-type: none"> ■ COMPRESSION attribute for the LOG_ARCHIVE_DEST_n initialization parameter of the database or far sync instance that is sending redo data ■ COMPRESSION column of the V\$ARCHIVE_DEST view of the database or far sync instance that is sending redo data
Scope	Database, far sync instance
Cloud Control name	Not applicable

¹ Although this property is set for a standby database or far sync instance, it is indirectly related to the redo transport services for the database or far sync instance that is sending redo data. The broker propagates the setting you specify on the standby database or far sync instance to the corresponding attributes of the LOG_ARCHIVE_DEST_n value of the database or far sync instance that is sending redo data.

8.3.31 RedoRoutes

By default, a primary database sends the redo that it generates to every other redo transport destination in the configuration. You can use the `RedoRoutes` property to create a more complex redo transport topology, such as one in which a physical standby database or a far sync instance forwards redo received from the primary database to one or more destinations, or one in which the redo transport mode used for a given destination is dependent on which database is in the primary role.

The RedoRoutes property is set to a character string that contains one or more redo routing rules. Each rule contains one or more redo sources and one or more redo destinations. A redo routing rule becomes active when one of the redo sources in the rule is in the primary role. This results in redo from the primary database being sent to every redo destination in that rule.

Category	Description
Datatype	String
Valid values	A character string that contains one or more redo routing rules, each contained within a pair of parentheses: (redo_routing_rule_1) [(redo_routing_rule_n)] See " Redo Routing Rules " on page 8-40 for more information about redo routing rules.
Broker default	Null
Imported?	No
Parameter class	Not applicable
Role	Primary, standby, far sync instance
Standby type	Physical standby and far sync instance
Corresponds to	LOG_ARCHIVE_DEST_n
Scope	Database, far sync instance
Cloud Control Name	RedoRoutes

8.3.31.1 Redo Routing Rules

A redo routing rule contains a redo source field and a redo destination field separated by a colon:

```
(redo source : redo destination)
```

The redo source field must contain either the keyword LOCAL or ANY, or a comma-separated list of DB_UNIQUE_NAME values:

```
{LOCAL | ANY | db_unique_name_1, [,db_unique_name_n]}
```

- The LOCAL keyword is an alias for the local database name. This keyword cannot be used at a far sync instance.
- The ANY keyword is an alias for any database in the configuration.
- A database cannot be specified as a redo source in more than one redo routing rule defined at a given database, either explicitly or implicitly through use of the LOCAL keyword.

The redo destination field must contain the keyword ALL or a comma-separated list of database names, each of which can be followed by an optional redo transport attribute:

```
{ALL [attribute] | db_unique_name_1 [attribute] [,db_unique_name_n [attribute]]}
```

The ALL keyword is an alias for all possible destinations in the configuration.

The optional redo transport attribute specifies the redo transport mode to use to send redo to the associated destination. It can have one of three values:

[ASYNC | SYNC | FASTSYNC]

If the redo transport attribute is not specified, then the redo transport mode used will be the one specified by the `LogXptMode` property for the destination.

The `RedoRoutes` property can also be used to set up an alternate remote destination. To do this, you use the `ALT` keyword:

```
[ALT=(alternate db_unique_name [ASYNC | SYNC | FASTSYNC] [FALLBACK])]
```

The `ALT` keyword accepts a `DB_UNIQUE_NAME` of another (alternate) member of the configuration to which redo data can be sent directly if there is a problem sending redo data to the non-alternate member. If the optional `FALLBACK` keyword is specified, then the shipping of redo to the non-alternate member is resumed once it becomes available. Oracle recommends that you use the `FALLBACK` keyword. If you omit it and the `ALTERNATE` redo destination subsequently fails, then no redo will be shipped to the terminal standby databases serviced by the alternate destination.

The following usage notes apply:

- The `RedoRoutes` property has a default value of `NULL`, which is treated as `(LOCAL : ALL)` at a primary database.
- A redo routing rule is active if its redo source field specifies the current primary database. If a rule is active, primary database redo is sent by the database at which the rule is defined to each destination specified in the redo destination field of that rule.
- The `ASYNC` redo transport attribute must be explicitly specified for a cascaded destination to enable real-time cascading to that destination.
- The `RedoRoutes` property cannot be set on a logical or snapshot standby database.
- The `RedoRoutes` property can be set for a logical standby database only if the redo destination field is set to `LOCAL`.
- The non-alternate member must have a non-zero value for its `MaxFailure` configurable property in order for an alternate to be specified.

8.3.32 ReopenSecs

The `ReopenSecs` configurable property specifies the minimum number of seconds before the archiver process (`ARCn`, foreground, or log writer process) should try again to access a previously failed destination.

Category	Description
Datatype	Integer
Valid values	>=0 seconds
Broker default	300
Imported?	No
Parameter class	Dynamic
Role	Standby database, far sync instance ¹
Standby type	Physical, logical, or snapshot standby, or a far sync instance

Category	Description
Corresponds to	<ul style="list-style-type: none"> ■ REOPEN attribute for the LOG_ARCHIVE_DEST_# initialization parameter of the database or far sync instance that is sending redo data ■ REOPEN_SECS column of the V\$ARCHIVE_DEST view of the database or far sync instance that is sending redo data
Scope	Database, far sync instance
Cloud Control name	Not applicable

¹ Although this property is set for a standby database or far sync instance, it is indirectly related to the redo transport services for the database or far sync instance that is sending redo data. The broker propagates the setting you specify on the standby database or far sync instance to the corresponding attributes of the LOG_ARCHIVE_DEST_# value of the database or far sync instance that is sending redo data.

8.3.33 SidName

The `SidName` configurable property specifies the Oracle System Identifier (SID) of the instance. The property can only be updated when broker management of the database is disabled. You should only update the value when the SID is changed, in which case you need to disable broker management of the database, update the `SidName` property to match with the new SID, and reenable broker management of the database.

Note: If no existing instance in the broker configuration, for the database whose name matches this instance's `db_unique_name`, has a `SidName` property with a value that matches this instance's actual SID, then the broker creates a new instance for this database that has this instance's `db_unique_name`.

Category	Description
Datatype	String
Valid values	SID of the instance. Note that on certain platforms, SIDs may be case-sensitive.
Broker default	Not applicable
Imported?	Yes
Parameter class	Not applicable
Role	Primary and standby
Standby type	Physical, logical, or snapshot
Corresponds to	INSTANCE_NAME column of the V\$INSTANCE view
Scope	Instance
Cloud Control name	Not applicable

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

8.3.34 StandbyArchiveLocation

The `StandbyArchiveLocation` configurable property specifies the location of archived redo log files arriving from a redo source. Oracle recommends that you always explicitly set the value.

Category	Description
Datatype	String
Valid values	Nonempty file specification of the location of archived redo log files on the database or far sync instance. Specify <code>USE_DB_RECOVERY_FILE_DEST</code> if a database recovery area is desired.
Broker default	<code>dgsby_db_unique_name</code>
Imported?	Yes, from the <code>DESTINATION</code> column of the <code>V\$ARCHIVE_DEST</code> fixed view of the standby database or far sync instance where the destination is a local destination and where the <code>VALID_FOR</code> attribute is compatible with the string <code>(STANDBY_ROLE, STANDBY_LOGFILE)</code> ; if no such destination exists, import is from the <code>STANDBY_ARCHIVE_DEST</code> initialization parameter. Note that the <code>STANDBY_ARCHIVE_DEST</code> parameter has been deprecated and is supported for backward compatibility only.
Parameter class	Dynamic
Role	Standby database or far sync instance
Standby type	Physical, logical, or snapshot standby, or a far sync instance
Corresponds to	<ul style="list-style-type: none"> ■ <code>LOCATION</code> attribute of the <code>LOG_ARCHIVE_DEST_n</code> initialization parameter of the standby database or far sync instance with <code>VALID_FOR</code> compatible with <code>(STANDBY_ROLE, STANDBY_LOGFILE)</code> ■ <code>DESTINATION</code> column of the <code>V\$ARCHIVE_DEST</code> view of the standby database
Scope	Instance
Cloud Control name	Standby Archive Location

Note: On a logical standby database, Oracle recommends the `LOCATION` attribute of the `LOG_ARCHIVE_DEST_n` initialization parameter for the local destination be different from the value of `StandbyArchiveLocation` property, unless you are using a database recovery area.

8.3.35 StandbyFileManagement

The `StandbyFileManagement` configurable property affects how the add datafile operation on the primary database is applied on the standby database. If this property is set to `AUTO`, in conjunction with valid settings in the `DbFileNameConvert` configurable property, a corresponding new datafile is automatically created on the standby database. The location of this new standby datafile is determined by the value of the `DbFileNameConvert` property.

If this property is set to `MANUAL`, you have to create the correct new datafile on the standby database manually.

Category	Description
Datatype	String
Valid values	AUTO or MANUAL
Broker default	AUTO
Imported?	Yes, from the STANDBY_FILE_MANAGEMENT initialization parameter
Parameter class	Dynamic
Role	Standby
Standby type	Physical or snapshot
Corresponds to	STANDBY_FILE_MANAGEMENT initialization parameter
Scope	Database
Cloud Control name	Not applicable

8.3.36 StaticConnectIdentifier

The `StaticConnectIdentifier` configurable instance-specific property specifies the connection identifier that the DGMGRL client will use when starting database instances. If this property has a null value, then the `DGConnectIdentifier` value is used for operations that involve shutting down and starting up the instance.

Category	Description
Datatype	String
Valid values	A connect identifier that refers to a service that is statically registered.
Broker default	Connect descriptor that is the concatenation of: ¹ <ul style="list-style-type: none"> ■ The ADDRESS attribute value of the listener that is specified for the LOCAL_LISTENER initialization parameter ■ The value for the SERVICE_NAME attribute will be set to a concatenation of db_unique_name_DGMGRL.db_domain
Imported?	Yes, from the LOCAL_LISTENER and DB_UNIQUE_NAME initialization parameters.
Parameter class	Not applicable
Role	Primary and standby
Standby type	Physical, logical, or snapshot
Corresponds to	Not applicable
Scope	Instance
Cloud Control name	Not applicable

¹ If the instance specified by the `SidName` property is started on a different host (read from the `HOST_NAME` column of the `V$INSTANCE` view) than the host on which it had been previously started, the broker automatically updates the default value of the `StaticConnectIdentifier` property to incorporate the current `ADDRESS` attribute of the listener that is specified for the `LOCAL_LISTENER` initialization parameter.

See Also:

- See *Oracle Database Net Services Administrator's Guide* for more information about statically registered services
- See the My Oracle Support Note 1387859.1 at <http://support.oracle.com> for additional information about using static services and the `StaticConnectIdentifier` configurable property

8.3.37 TransportDisconnectedThreshold

The `TransportDisconnectedThreshold` configurable property can be used to generate a warning status for a logical, physical, or snapshot standby, or a far sync instance when the last communication from the primary database exceeds the value specified by the property. The property value is expressed in seconds. A value of 0 seconds results in no warnings being generated.

Category	Description
Datatype	Number
Valid values	>=0
Broker default	30 seconds
Imported?	No
Parameter class	Not applicable
Role	Standby database, far sync instance
Standby type	Physical, logical, or snapshot standby, or a far sync instance
Corresponds to	Not applicable
Scope	Database, far sync instance
Cloud Control name	Not applicable

8.3.38 TransportLagThreshold

The `TransportLagThreshold` configurable property can be used to generate a warning status for a logical, physical, or snapshot standby, or a far sync instance when the member's transport lag exceeds the value specified by the property. The property value is expressed in seconds. A value of 0 seconds results in no warnings being generated when a transport lag exists.

Category	Description
Datatype	Number
Valid values	>=0
Broker default	0 seconds
Imported?	No
Parameter class	Not applicable
Role	Standby database, far sync instance
Standby type	Physical, logical, or snapshot standby, or a far sync instance
Corresponds to	Not applicable

Category	Description
Scope	Database, far sync instance
Cloud Control name	Not applicable

Troubleshooting Oracle Data Guard

This chapter describes various errors and how to solve them. It contains the following topics:

- [Sources of Diagnostic Information](#)
- [General Problems and Solutions](#)
- [Troubleshooting Problems During a Switchover Operation](#)
- [Troubleshooting Problems During a Failover Operation](#)
- [Troubleshooting Problems with the Observer](#)

9.1 Sources of Diagnostic Information

The Oracle Data Guard broker provides information about its activities in several forms:

- Database status information (see [Section 4.10](#))
- Oracle alert log files

The broker records key information in the alert log file for each instance of each database in a broker configuration. You can check the alert log files for such information when troubleshooting Oracle Data Guard.

- Oracle Data Guard "broker log files"

For each instance of each database in a broker configuration, the broker DMON process records important behavior and status information in a broker log file, useful in diagnosing Oracle Data Guard failures. The `TraceLevel` configuration property (see [Section 8.1.13](#)) is used to specify the level of diagnostic information reported in the broker log files.

The broker log file is created in the same directory as the alert log and is named `drc<${ORACLE_SID}>.log`.

- Oracle Data Guard command line (DGMGRL) logfile

If the DGMGRL command-line interface was started with the `-logfile` optional parameter (see [Section 7.1.1](#)), then the resulting log file may contain a useful record of past operations, error conditions, or both.

9.2 General Problems and Solutions

This section describes general problems and solutions. This section contains the following topics:

- [ORA-16596: database not part of the Oracle Data Guard broker configuration](#)
- [Redo Accumulating on the Primary Is Not Sent to Some Standby Databases](#)
- [Many Log Files Are Received on a Standby Database But Not Applied](#)
- [The Request Timed Out or Cloud Control Performance Is Sluggish](#)
- [The Primary Database is Flashed Back](#)
- [Standby Fails to Automatically Start Up Due to Unknown Service \(ORA-12514\)](#)

9.2.1 ORA-16596: database not part of the Oracle Data Guard broker configuration

A request was issued to the broker, but the database instance through which you have connected is no longer a part of the broker configuration.

Solution Reconnect to the configuration through another database that you know is part of the broker configuration. Confirm that a database exists in the broker configuration that has a name that matches the `db_unique_name` value of the database that returned the ORA-16596 error.

This problem can also occur if you attempt to enable a configuration, but the broker configuration file for one of its databases was accidentally removed or is outdated. In this case, remove the database from the broker configuration, manually delete the configuration file for that standby database (not for the primary database), and try again to enable the configuration. After the configuration is enabled, you can either use the Cloud Control Add Standby Database wizard and choose the Add existing standby database option, or you can use the DGMGRL command-line interface and issue the `ADD DATABASE` command.

9.2.2 Redo Accumulating on the Primary Is Not Sent to Some Standby Databases

By viewing the Log File Details page in Cloud Control, you have determined that log files are accumulating on the primary database and are not being archived to some of the standby databases in the broker configuration.

Solution To narrow down the problem, do the following:

- Verify that the state of the primary database is in the `TRANSPORT-ON` state (not `TRANSPORT-OFF`).
- Verify that the value of the `LogShipping` database property of the standby database in question is `ON`.
- Check the status of the redo transport services on the primary database using the `LogXptStatus` monitorable property. If redo transport services have an error, then use the error message to determine further checking and resolution action. For example:
 - If the error indicates the standby database is not available, you need to restart the standby database.
 - If the error indicates no listener, you need to restart the listener.
 - If the error indicates the standby database has no local destination, you need to set up a standby location to store archived redo log files from the primary database.

9.2.3 Many Log Files Are Received on a Standby Database But Not Applied

By viewing the Performance page or Log File Details page in Cloud Control, you have determined that the standby database accumulates too many log files without applying them.

Solution There are many possible reasons why archived redo log files might not be applied to the standby database. Investigate why the log files are building up and rule out the valid reasons.

If the current status of the standby database is not normal:

- Determine whether or not the log apply services might be unexpectedly stopped. See the ORA-16766 (for physical standby databases) or ORA-16768 (for logical standby databases) error description and solution statement for more help.
- If this is a logical standby database, check to see if a failed transaction has occurred.
- If you want to suppress the error while you investigate the problem, you can temporarily disable broker management of the database.

See Also: [Chapter 7](#) for additional information about disabling the database using the DGMGRL command-line interface

If the current status of the standby database is normal:

- Verify the state of the standby database is APPLY-ON (not in the APPLY-OFF state).
- Verify the state of the primary database is TRANSPORT-ON (not in the TRANSPORT-OFF state).

See Also: [Chapter 8](#) for additional information about the LogShipping database property

- Check to see if log files are building up because the value of the DelayMins property is set too large. (Log apply services will delay applying the archived redo log files on the standby database for the number of minutes specified.)

See Also: [Chapter 8](#) for additional information about the DelayMins database property

- If you cannot see any errors, compare the archive rate to the apply rate on the Performance page in Cloud Control to see if the apply rate is lower than the archive rate.

9.2.4 The Request Timed Out or Cloud Control Performance Is Sluggish

If the broker requests are not completing within the normal timeout parameters, try the following actions to solve the problem:

1. Verify the network is operating appropriately.
2. Try to ping all of the nodes in the configuration.
3. Try reconnecting through another database to retry the operation.
4. Run the VERIFY command to determine on which database the broker is unable to process the requests.

9.2.5 The Primary Database is Flashed Back

If the primary database is flashed back, the standby databases in the configuration must be also be flashed back or re-created to be viable targets for switchovers or failovers. The broker will report errors for the standby databases if the primary database has been flashed back.

For more information about restoring the viability of a standby database that was disabled by the broker, see [Section 5.4.3](#).

9.2.6 Standby Fails to Automatically Start Up Due to Unknown Service (ORA-12514)

If the DGMGRL CLI fails to automatically start up an instance after a broker operation (for example, a switchover, reinstatement, convert to physical standby, or upgrade of protection mode to maximum protection mode), and you receive an `ORA-12514: TNS:listener does not currently know of service requested in connect descriptor` error, then you must manually start the instance to complete or continue the broker operation.

You can restart the instance before or after completing the following steps:

1. Issue the following DGMGRL CLI command to check the value of the `StaticConnectIdentifier` configurable property for the instance the DGMGRL CLI was unable to restart. (You will have to connect to another running instance to issue this command):

```
DGMGRL> SHOW INSTANCE VERBOSE 'instance_name' ON DATABASE db_unique_name;
```

2. The static service name specified in the value of the `StaticConnectIdentifier` instance property should be registered with the listener specified in the property value. The default value for the static service name is of the following form:

```
db_unique_name_DGMGRL.db_domain
```

See [Section 2.2](#) for more information about this, and other prerequisites, for using the broker.

3. Confirm that the static service name is registered with the listener specified in the `StaticConnectIdentifier` configurable property value by using the Listener Control utility's status command. If the static service name is properly registered with the listener, it will be included in the output generated by the following command:

```
lsnrctl status
```

4. If you are working in an Oracle RAC environment, you may want to repeat steps 1 and 2 for each instance to ensure that a static service name is properly registered with its respective listener.

9.3 Troubleshooting Problems During a Switchover Operation

If the switchover fails due to problems with the configuration, then the broker reports any problems it encounters in the alert log files or in the broker log files (see [Section 9.1](#)). If the reported problems can be corrected, you can retry the switchover operation and it will usually succeed. If the reported problems cannot be corrected or the switchover operation fails even after correcting the reported problems, then you can choose another database for the switchover or restore the configuration to its pre-switchover state and then retry the switchover.

If fast-start failover is enabled, the broker does not allow switchover to any standby database except to the target standby database. In addition, switchover to the target standby database is allowed only when the value of the `FS_FAILOVER_STATUS` column in the `V$DATABASE` view on the target standby database is either `READY` or `SUSPENDED`.

See Also: [Section 5.5.2.1, "When Fast-Start Failover Is Enabled and the Observer Is Running"](#)

9.4 Troubleshooting Problems During a Failover Operation

Although it is possible for a failover operation to fail, it is unlikely. If an error occurs, use the guidelines given in the following subsections to fix the problem and then retry the broker failover.

9.4.1 Failed Failovers to Physical Standby Databases

Use the steps below to recover from a failed broker failover to a physical standby database.

9.4.1.1 Failed Broker Complete Physical Failovers

Examine the alert and DRC log files on the target standby database to determine the cause of the failure and correct the problem. If the reported problem can be corrected, then retry the failover operation. If the reported problem cannot be corrected or if the failover operation fails again after the reported problem has been corrected, then take the following steps:

1. Connect to the target standby database and disable fast-start failover using the `FORCE` option if it is enabled.
2. Then you can either:
 - Connect to another physical standby database and attempt a broker complete failover.
 - Perform a broker immediate failover to the target physical standby database.
3. Reinstall the original primary database and any bystander physical standby databases that are disabled with a status of reinstatement required (`ORA-16661`).
4. Reenable fast-start failover if it was disabled in step 1.

9.4.1.2 Failed Broker Immediate Physical Failovers

Examine the alert and DRC log files on the target standby database to determine the cause of the failure and correct the problem. If the problem can be corrected, retry the broker immediate failover. Otherwise connect to another physical standby database and attempt either a broker complete or immediate failover.

9.4.2 Failed Failovers to Logical Standby Databases

1. Examine the alert and DRC log files on the target standby database to determine the cause of the failure and correct the problem.
2. Connect to the target standby database and disable fast-start failover using the `FORCE` option if it is enabled.
3. Retry the broker failover.
4. Reinstall the old primary database. All bystander standby databases will be re-created from a copy of the new primary database.

5. Reenable fast-start failover if it was disabled in step 1.

If broker failover continues to fail, you should stop the broker on all databases in the Oracle Data Guard configuration (set the `DG_BROKER_START` initialization parameter to `FALSE`). Remove the Oracle Data Guard broker configuration files from all databases. Attempt a manual failover using the guidelines for role transitions in *Oracle Data Guard Concepts and Administration*.

Note: You can enable or disable the broker configuration using `DGMGRL ENABLE CONFIGURATION` and `DISABLE CONFIGURATION` commands. You cannot disable the configuration using Cloud Control. You can only enable the configuration using Cloud Control if it was previously disabled using `DGMGRL`.

9.5 Troubleshooting Problems with the Observer

The observer continuously monitors the fast-start failover environment to ensure the primary database is available. Installing and starting the observer is an integral part of using fast-start failover. The following sections describe techniques for troubleshooting the observer:

- [Problems Starting the Observer](#)
- [Problems Because the Observer Has Stopped](#)
- [Capturing Observer Actions in the Observer Log File](#)

9.5.1 Problems Starting the Observer

Only one observer can be observing the broker configuration at any given time. If you attempt to start a second observer, one of the following errors is returned:

```
ORA-16647: could not start more than one observer
DGM-16954: Unable to open and lock the Observer configuration file
```

Use the `DGMGRL SHOW CONFIGURATION VERBOSE` command to determine the location of the observer that is currently associated with the broker configuration.

```
DGMGRL> SHOW CONFIGURATION VERBOSE;

Configuration - DRSolution

Protection Mode: MaxAvailability
Databases:
North_Sales- Primary database
South_Sales- Physical standby database

Properties:
FastStartFailoverThreshold      = '30'
OperationTimeout                = '30'
TraceLevel                      = 'USER'
FastStartFailoverLagLimit      = '30'
CommunicationTimeout           = '180'
ObserverReconnect               = '0'
FastStartFailoverAutoReinstate = 'TRUE'
FastStartFailoverPmyShutdown   = 'TRUE'
BystandersFollowRoleChange     = 'ALL'
ObserverOverride                = 'FALSE'
ExternalDestination1           = ''
```



```

ExternalDestination2          = ''
PrimaryLostWriteAction       = 'CONTINUE'

Fast-Start Failover: ENABLED

Threshold: 30 seconds
Target: South_Sales
Observer: observer.example.com
Lag Limit: 30 seconds (not in use)
Shutdown Primary: TRUE
Auto-reinstate: TRUE
Observer Reconnect: (none)
Observer Override: FALSE

Configuration Status:
SUCCESS

```

9.5.2 Problems Because the Observer Has Stopped

If the observer host machine crashes, the broker configuration is no longer observed and fast-start failover is no longer possible. In this case, you may have to move the observer to a new host if the original host machine cannot be repaired in a timely fashion. To move the observer, you must stop allowing the first observer to observe this broker configuration, and then start a new observer on another host.

1. Issue the `DGMGRL STOP OBSERVER` command to sever the link between the original observer and the broker configuration:

```

DGMGRL> STOP OBSERVER;
Done.

```

2. Issue the `DGMGRL SHOW CONFIGURATION VERBOSE` command to verify that the configuration is no longer being observed:

```

DGMGRL> SHOW CONFIGURATION VERBOSE;

```

```

Configuration - DRSolution

```

```

Protection Mode: MaxAvailability
Databases:
  North_Sales - Primary database
  Warning: ORA-16819: fast-start failover observer not started

  South_Sales - (*) Physical standby database
  Warning: ORA-16819: fast-start failover observer not started

```

```

(*) Fast-Start Failover target

```

```

Properties:

```

```

FastStartFailoverThreshold    = '30'
OperationTimeout              = '30'
TraceLevel                    = 'USER'
FastStartFailoverLagLimit     = '30'
CommunicationTimeout          = '180'
ObserverReconnect             = '0'
FastStartFailoverAutoReinstate = 'TRUE'
FastStartFailoverPmyShutdown  = 'TRUE'
BystandersFollowRoleChange    = 'ALL'
ObserverOverride              = 'FALSE'
ExternalDestination1          = ''
ExternalDestination2          = ''

```

```

PrimaryLostWriteAction          = 'CONTINUE'

Fast-Start Failover: ENABLED

Threshold: 30 seconds
Target: South_Sales
Observer: observer.example.com
Lag Limit: 30 seconds (not in use)
Shutdown Primary: TRUE
Auto-reinstate: TRUE
Observer Reconnect: (none)
Observer Override: FALSE

Configuration Status:
WARNING

```

3. Note that you do not need to issue the `DGMGRL SHOW CONFIGURATION` command to verify that the observer has actually stopped. Successful completion of the `DGMGRL STOP OBSERVER` command will allow a new observer to become associated with the configuration.

9.5.3 Capturing Observer Actions in the Observer Log File

You can use the `DGMGRL -logfile` option to start the observer, so that all of the troubleshooting actions performed in [Section 9.5.1](#) can be captured in a file. For example:

```
% dgmgrl -logfile observer.log / "start observer"
```

All the observer output is then recorded in a file named `observer.log` in the current working directory where you issued the `DGMGRL` command. If the specified log file is not accessible, then the observer output is sent to standard output as though a log file had not been specified.

Note that this is not only useful for troubleshooting problems with the observer, but also for troubleshooting problems with fast-start failover in general.

Oracle Data Guard Broker Upgrading and Downgrading

This appendix guides you through the process of upgrading or downgrading the Oracle databases and Oracle Enterprise Manager Cloud Control (Cloud Control) in a broker configuration. It contains the following topics:

- [Upgrading from Oracle Database 9i Release 2 \(9.2\) to Oracle Database 12c Release 1 \(12.1\)](#)
- [Upgrading from Oracle Database 10g and Oracle Database 11g to Oracle Database 12c Release 1 \(12.1\)](#)
- [Downgrading from Oracle Database 12c Release 1 \(12.1\)](#)
- [Preserving Broker Configurations During Rolling Upgrades](#)

Note: In an Oracle Data Guard configuration, all physical and snapshot standby databases must use a copy of the password file from the primary database, and that copy must be refreshed whenever an administrative privilege (SYSDBG, SYSOPER, SYSDBA, and so on) is granted or revoked, and after the password of any user with administrative privileges is changed.

See Also: [Chapter 2, "Oracle Data Guard Installation"](#)

A.1 Upgrading from Oracle Database 9i Release 2 (9.2) to Oracle Database 12c Release 1 (12.1)

If you are currently running a broker configuration in Oracle Database 9i Release 2 (9.2), then you must upgrade to Oracle Database 12c Release 1 (12.1), and then re-create the broker configuration, as follows:

1. Delete (remove) the Oracle Database 9i Release 2 (9.2) broker configuration. For example, the DGMGRL REMOVE CONFIGURATION command can be used. To perform the deletion you must use Oracle Data Guard Manager or DGMGRL running on Oracle Database 9i Release 2 (9.2).
2. If using DGMGRL, clear the LOG_ARCHIVE_DEST_ *n* initialization parameter settings by using the ALTER SYSTEM SET LOG_ARCHIVE_DEST_ *n* = " " SQL*Plus command.
3. Upgrade the database software to Oracle Database 12c Release 1 (12.1). See the Oracle Database installation documentation that is appropriate for your operating system.

4. Oracle Enterprise Manager and Oracle Data Guard Manager running on Oracle Database 9i Release 2 (9.2) cannot manage a broker configuration running on Oracle Database 12c Release 1 (12.1).

See Also: My Oracle Support note 787461.1 at <http://support.oracle.com> for information about which versions of Enterprise Manager Cloud Control (formerly called Grid Control) are required to manage the full set of Oracle Data Guard features in various Oracle Database releases

5. If you are using DGMGRL running on Oracle Database 9i Release 2 (9.2), then you must upgrade to DGMGRL running on Oracle Database 12c Release 1 (12.1):
 - DGMGRL running on Oracle Database 9i Release 2 (9.2) is not compatible with Oracle Data Guard running on Oracle Database 12c Release 1 (12.1).
 - DGMGRL running on Oracle Database 12c Release 1 (12.1) is not compatible with Oracle Data Guard running on Oracle Database 9i Release 2 (9.2).

Note: Oracle Database 12c command-line scripts are not supported in Oracle9i.

6. Invoke Enterprise Manager or the DGMGRL command-line interface, and re-create the broker configuration.

A.2 Upgrading from Oracle Database 10g and Oracle Database 11g to Oracle Database 12c Release 1 (12.1)

If you are currently running Oracle Data Guard on Oracle Database 10g or Oracle Database 11g, then you can upgrade the database software to Oracle Database 12c Release 1 (12.1) using the Oracle Database installation documentation that is appropriate for your operating system.

Note: Prior to Oracle Database 11g Release 2 (11.2), the configuration file was restricted to reside only on disks having the same sector size (physical block size) as that upon which the file was initially created. This was not a problem because there was typically a single sector size in use within a given broker configuration. In anticipation of having mixed sector sizes somewhere within a given broker configuration, the broker configuration file is now completely insensitive to the underlying sector size, so long as the sector size is 4KB or less.

Conversion of the configuration file to be insensitive to the underlying sector size occurs during the upgrade processing in Step 5 below.

Step 1 Disable fast-start failover if it is enabled.

Issue the following DGMGRL command to disable fast-start failover:

```
DGMGRL> DISABLE FAST_START FAILOVER;
```

Step 2 Shut down the 10.n or 11.n Oracle Data Guard broker, whichever version you are running.

Disable the broker's management of the configuration and then stop the broker. For example:

1. Issue the following DGMGRL command to disable the broker's active management of the databases in the Oracle Data Guard configuration:

```
DGMGRL> DISABLE CONFIGURATION;
```

2. Issue the following SQL*Plus statement to stop the broker:

```
SQL> ALTER SYSTEM SET DG_BROKER_START=FALSE;
```

Step 3 Make a copy of the current broker configuration files.

Make a copy of the current broker configuration files, as indicated by the following initialization parameters: `DG_BROKER_CONFIG_FILE1` and `DG_BROKER_CONFIG_FILE2`.

Step 4 Upgrade the Oracle Database software to Oracle Database 12c Release 1 (12.1).

For step-by-step upgrade instructions, see the Oracle Database installation documentation that is appropriate for your operating system.

The DGMGRL command-line interface must also be upgraded in order to manage and monitor a broker configuration running on Oracle Database 12c Release 1 (12.1). DGMGRL running on Oracle Database 12c Release 1 (12.1) cannot be used to manage Oracle Data Guard running on Oracle Database 10g or Oracle Database 11g.

Note: Existing DGMGRL command-line scripts for Oracle Database 10g and Oracle Database 11g are supported by the DGMGRL command-line interface available in Oracle Database 12c Release 1 (12.1).

DGMGRL command-line scripts for Oracle Database 11g are not guaranteed to be supported by Oracle Database 10g.

Step 5 Start the broker.

After the upgrade to Oracle Database 12c Release 1 (12.1), start the broker. For example:

1. Issue the following SQL*Plus statements to start the broker:

```
SQL> ALTER SYSTEM SET DG_BROKER_START=TRUE;
```

2. Issue the following DGMGRL command to enable the broker's active management of the databases in the Oracle Data Guard configuration:

```
DGMGRL> ENABLE CONFIGURATION
```

The first time the broker starts on Oracle Database 12c Release 1 (12.1) it detects the existence of the Oracle Database 10g and Oracle Database 11g broker configuration files. It automatically upgrades them to include any new properties that were introduced in Oracle Database 12c Release 1 (12.1). This automatic conversion is transparent, permanent, and occurs only once.

Note: As of this automatic upgrade that occurs in Step 5, the configuration file may henceforth reside on a disk having any supported disk sector size (physical block size) up to and including 4KB sectors.

Note: The observer that was started prior to the upgrade will automatically be stopped and unable to observe the configuration once the upgrade is complete. You must use an Oracle Database 12c version of the Oracle Observer software to observe Oracle databases running on Oracle Database 12c Release 1 (12.1) or higher.

A.3 Downgrading from Oracle Database 12c Release 1 (12.1)

If you have upgraded to Oracle Database 12c Release 1 (12.1) and want to downgrade to your prior release, then you must downgrade the database release and re-create the broker configuration as follows:

1. Delete (remove) the Oracle Database 12c Release 1 (12.1) broker configuration using Enterprise Manager or DGMGRL running on Oracle Database 12c Release 1 (12.1). For example, the `DGMGRL REMOVE CONFIGURATION` command can be used.
2. Downgrade the Oracle Database software to the prior Oracle release. See the Oracle Database documentation that is appropriate for your operating system.
3. You can downgrade the Oracle Data Guard broker to Oracle Database 10g or Oracle Database 11g as follows:
 - If you are confident that you made no changes to your broker configuration after upgrading to Oracle Database 12c Release 1 (12.1), then you can copy the broker configuration files that you created during the upgrade procedure (described in [Section A.2](#)). Then, invoke Enterprise Manager or the DGMGRL command-line interface and reenable the broker configuration as it existed at the time you made that copy.
 - Alternatively, you can invoke Enterprise Manager or the DGMGRL command-line interface to re-create the broker configuration.
4. You can downgrade the Oracle Data Guard broker to Oracle Database 9i Release 2 (9.2), as follows:
 - If you were using Oracle Enterprise Manager, you can continue to use it running on Oracle Database 10g Release 2 (10.2) to manage your Oracle Database 9i Release 2 (9.2) broker configuration. You can also downgrade to the Oracle Data Guard Manager by re-installing Oracle Database 9i Release 2 (9.2) Enterprise Manager.
 - Invoke Enterprise Manager, the Oracle Data Guard Manager, or the DGMGRL command-line interface and re-create the broker configuration.

A.4 Preserving Broker Configurations During Rolling Upgrades

If you use Oracle Data Guard broker to manage an Oracle Data Guard configuration and you want to preserve that configuration when you perform a rolling upgrade, then you must do the following:

1. Disable the broker configuration (using the `DGMGRL DISABLE CONFIGURATION` command).
2. Perform a rolling upgrade.
3. Connect to the current primary database (database whose control file role is PRIMARY) and re-enable the broker configuration (using the `DGMGRL ENABLE CONFIGURATION` command).

Following this procedure preserves your broker configuration so that you do not have to rebuild it after a rolling upgrade is complete.

This procedure has the following restrictions:

- The broker configuration must be enabled after the rolling upgrade is complete. You must ensure that the rolling upgrade is complete before enabling the configuration.
- The broker configuration does not support databases running different versions of Oracle. Any standby databases not running the same version of Oracle as the primary database are disabled with an `ORA-16673: standby database running different Oracle version` error message.
- When the broker configuration is disabled, the broker configuration can then only be enabled while connected to a database whose control file role is primary (query the `DATABASE_ROLE` column of the `V$DATABASE` view to see which one contains the value `PRIMARY`).

See Also:

- *Oracle Data Guard Concepts and Administration* for more information about rolling upgrades

Glossary

apply instance

In an Oracle Real Application Clusters (Oracle RAC) databases environment, the apply instance is the one instance applying archived redo data to a standby database.

broker

A distributed management framework that automates and simplifies most of the complex operations required to create, control, and monitor an Oracle Data Guard configuration.

broker configuration

A logical grouping of the primary and standby databases (including redo transport services and log apply services) of an Oracle Data Guard configuration.

See also [Data Guard configuration](#).

bystander standby database

A standby database that is not the target of, or directly involved in, a switchover or failover operation.

configuration object

A named collection of database objects. It is an abstraction of an Oracle Data Guard configuration.

database guard

The database guard controls whether or not modifications can be made to the tables in a logical standby database.

database object

A named object that corresponds to a primary or standby database in an Oracle Data Guard configuration. The broker uses this object to manage and control the state of a single database and to associate properties with the database.

Data Guard command-line interface

The Oracle Data Guard command-line interface (DGMGRL) enables you to control and monitor an Oracle Data Guard configuration from the DGMGRL prompt or within scripts.

Data Guard configuration

A distributed computing system that prevents or minimizes losses due to unplanned events (for example, human errors, environmental disasters, or data corruption) as well as to planned downtime (such as for routine maintenance tasks).

See also [broker configuration](#).

Data Guard environment

The physical configuration of the primary, standby databases, and far sync instances. The environment depends on many factors, including the:

- Number of standby databases and far sync instances associated with a primary database
- Number of host systems used by the databases
- Directory structures of the machines used by the databases
- Network configuration
- Redo transport services
- Log apply services

The Oracle Data Guard environment can be managed manually by a DBA, automatically using Enterprise Manager or the Data Guard command-line interface (DGMGRL) or a combination of all of these.

default state

The initial runtime state in which the database object will run when you enable broker management of the configuration. The actual default state can vary depending on the role (primary or standby) in which the database is running.

See also [intended state](#).

failover

Failover changes a standby database into the role of a primary database and disables the old primary database.

See also [fast-start failover](#) and [manual failover](#).

far sync instance

A remote Oracle Data Guard destination that accepts redo from the primary database and redistributes that redo throughout the Oracle Data Guard configuration. It is similar to a physical standby database in that it manages a control file, receives redo into Standby Redo Logs (SRLs), and archives those SRLs to local Archived Redo Logs (ARLs). But unlike a standby database, a far sync instance does not manage data files, cannot be opened for access, and cannot run redo apply.

fast-start failover

Enables a failover *automatically* when the primary database becomes unavailable. When fast-start failover is enabled, the broker determines if a failover is necessary and automatically, quickly, and reliably fails over to a designated standby without requiring that you perform any manual steps to invoke the failover.

See also [manual failover](#).

instance object

A named object; a database object is a collection of one or more named instance objects. The broker uses this object to manage and control the state of the database with which the instance is associated, and to associate instance-specific properties with this instance of the database.

intended state

The runtime state of a database object while it is enabled for management by the broker.

See also [default state](#).

lights out computing

Refers to computing equipment or software whose operations are automated, requiring little to no intervention by human administrators.

The term "lights out" originated from when computing centers were located in one room and contained a number of servers that were kept under lock and key and in the dark. Under normal operation, the room was not entered by human administrators, and all operations in the room were automated.

logical standby database

A logical standby database takes standard Oracle archived redo log files, transforms them back into SQL transactions, and then applies them to an open standby database. Although changes can be applied concurrently with end-user access, the tables being maintained through regenerated SQL transactions allow read-only access to users of the logical standby database. Because the database is open to allow application of reconstructed SQL transactions, it is physically different from the primary database. The database tables can have different indexes and physical characteristics from their primary database peers, but the tables must maintain logical consistency from an application access perspective, to fulfill their role as a standby data source.

manual failover

A failover that is initiated by the DBA who first determines a failover is necessary and then invokes one by clicking `FAILOVER` in Enterprise Manager or by issuing the `DGMGRL FAILOVER` command. The word *manual* is used to contrast this type of failover with a fast-start failover.

See also [fast-start failover](#).

observer

A DGMGRL client that continuously monitors the primary and target standby databases, evaluates whether failover is necessary, and initiates a fast-start failover when conditions warrant.

physical standby database

A physical standby database is an exact copy of a primary database. While the primary database is open and active, a physical standby database uses Redo Apply to apply redo data received from the primary database. Redo Apply can run on a physical standby database instance that is mounted. If a license for the Oracle Active Data Guard option has been purchased, Redo Apply can also run on a physical standby database instance that is open. See *Oracle Data Guard Concepts and Administration* for more details.

primary database

A production database from which one or more standby databases is created and maintained. Every standby database is associated with one and only one primary database. A single primary database can, however, support multiple standby databases. The Oracle Data Guard broker monitor (DMON) maintains the master copy of the binary configuration file with the primary database, ensuring that each standby database's copy of the file is kept up to date as changes are made.

The broker refers to this database using the value in the `DB_UNIQUE_NAME` initialization parameter which is defined to be globally unique.

read-only mode

A mode in which a database can be opened that allows queries, but disallows modifications.

A physical standby database can be opened read-only so that queries may be performed. If a license for the Oracle Active Data Guard option has been purchased, a physical standby database can be open while Redo Apply is active. This capability is known as real-time query. See *Oracle Data Guard Concepts and Administration* for more details.

Redo Apply

A physical standby database is kept synchronized with the primary database through Redo Apply, which recovers the redo data received from the primary database and applies the redo to the physical standby database.

reinstatement

Reinstatement is the process of turning a database, including the old primary database, that had been disabled after a failover operation into a viable standby database for the new primary database. Flashback database must be enabled on a database in order to reinstate it.

snapshot standby database

A snapshot standby is a fully updatable standby database that is created from a physical standby database. On snapshot standby databases, redo data is received but not applied until the snapshot standby database is converted back to a physical standby database.

SQL Apply

A logical standby database is kept synchronized with the primary database through SQL Apply, which transforms the data in the redo received from the primary database into SQL statements and then executes the SQL statements on the standby database.

standby database

A copy of a primary database created using a backup of your primary database. Standby databases are kept synchronized with the primary database by applying archived redo data over time from the primary database to each standby database. The standby database can take over processing from the primary database, providing nearly continuous database availability. A standby database has its own server parameter file, control file, and data files. It also has a copy of the broker's configuration file, kept up to date at the direction of the DMON process running in the primary database instance.

The broker refers to a standby database by its globally unique name that is stored in its `DB_UNIQUE_NAME` initialization parameter.

See also [logical standby database](#), [physical standby database](#) and [snapshot standby database](#).

A

- ADD DATABASE command, 7-7
- ADD FAR_SYNC command, 7-8
- Add Standby Database wizard
 - creating a broker configuration, 3-7
 - definition, 1-9
 - introduction, 1-9
- adding
 - standby database to the broker configuration, 6-4
- ALTER SYSTEM statement
 - starting the broker, 6-2
- altering
 - properties
 - database, 6-14
 - states
 - database, 6-15
- AlternateLocation property, 8-20
 - setting log apply services, 4-13
- application continuity
 - support in Oracle Data Guard broker, 5-44
- application integration, 1-8
- apply instance, 4-17
 - failover, 4-17
 - PreferredApplyInstance property, 4-18
 - selecting, 4-17
- ApplyInstanceTimeout property, 8-21
- ApplyLagThreshold property, 8-21
- ApplyParallel property, 8-22
- architecture
 - Data Guard broker, 1-8
- ARCHIVE_LAG_TARGET initialization parameter
 - setting in a broker configuration, 8-23
- archived redo logs
 - destination and configuration parameters, 3-1
 - in a Data Guard configuration, 3-2
 - primary database setup, 2-3
- ArchiveLagTarget property, 8-22
- ASYNC redo transport service, 4-8
- Automatic Storage Management (ASM)
 - creating a standby database that uses, 2-2

B

- background processes
 - DMON, 1-10

- banners
 - suppressing from DGMGRL command-line interface output, 7-1
- before you perform a switchover, 5-4
- before you use DGMGRL, 6-1
- benefits
 - Data Guard broker, 1-5
 - high availability, 1-6
 - scalability, 1-6
 - with Oracle Real Application Clusters, 1-6
- Binding property, 8-23
- broker
 - components, 1-8
 - configuration, 1-2
 - Data Guard configuration file, 1-12
 - drc* log files, 9-1
 - installation, 2-2
 - introduction, 1-2
 - user interfaces, 1-9
- broker configurations
 - benefits, 1-5
 - changing roles, 3-8
 - creating, 6-2
 - Data Guard configuration file, 1-10
 - disabling databases, 3-9, 7-15
 - effects of removing metadata from, 7-51
 - enabling, 6-6, 7-38
 - enabling databases, 3-8, 3-9, 6-6, 7-38
 - enabling fast-start failover, 7-41
 - health, 1-10, 3-10
 - life cycle, 3-7
 - management, 1-9, 1-10, 1-11
 - monitoring, 6-27
 - Oracle Net Services configuration, 1-7, 1-9
 - properties, 4-4
 - protection modes, 7-22
 - setting protection mode, 4-20
 - status modes, 3-10
 - supported, 3-1
- bystander standby databases, 5-4
 - reintegrating into configuration, 5-11
- BystandersFollowRoleChange property, 8-3

C

- CDBs

- Data Guard broker support, 1-4
- CFS
 - See* cluster file system
- changing
 - properties
 - databases in a broker configuration, 3-9, 4-6
 - roles
 - within the broker configuration, 3-8
 - states
 - databases in a broker configuration, 4-3
 - of databases in a broker configuration, 6-14
- See also* editing
- cluster file system (CFS)
 - broker configuration files, 3-5
- command prompts
 - suppressing from DGMGRL command-line interface output, 7-1
- command-line interface
 - See* Data Guard command-line interface (DGMGRL)
- CommunicationTimeout property, 8-4
- COMPATIBLE initialization parameter
 - requirements for setting, 2-3
- complete failover, 5-7, 7-45
- components
 - broker, 1-8
 - Data Guard configuration, 3-1
- configurable properties, 4-4
 - AlternateLocation, 8-20
 - ApplyInstanceTimeout, 8-21
 - ApplyLagThreshold, 8-21
 - ApplyParallel, 8-22
 - ArchiveLagTarget, 8-22
 - Binding, 8-23
 - database, 4-6, 8-17
 - DbFileNameConvert, 8-23
 - DelayMins, 8-24
 - DGConnectIdentifier, 8-25
 - FastStartFailoverTarget, 8-26
 - LogArchiveFormat, 8-27
 - LogArchiveMaxProcesses, 8-27
 - LogArchiveMinSucceedDest, 8-28
 - LogArchiveTrace, 8-28
 - LogFileNameConvert, 8-29
 - LogShipping, 8-30
 - LogXptMode, 8-31
 - LsbyMaxEventsRecorded, 8-32
 - LsbyMaxServers, 8-33
 - LsbyMaxSga, 8-33
 - LsbyPreserveCommitOrder, 8-34
 - LsbyRecordAppliedDdl, 8-34
 - LsbyRecordSkipDdl, 8-35
 - LsbyRecordSkipErrors, 8-35
 - managing redo transport services, 4-7
 - MaxConnections, 8-36
 - MaxFailure, 8-37
 - NetTimeout, 8-37
 - ObserverConnectIdentifier, 8-38
 - PreferredApplyInstance, 8-38
 - RedoCompression, 8-39

- RedoRoutes, 8-39
- ReopenSecs, 8-41
- SidName, 8-42
- StandbyArchiveLocation, 8-43
- StandbyFileManagement, 8-43
- StaticConnectIdentifier, 8-44
- TransportDisconnectedThreshold, 8-45
- TransportLagThreshold, 8-45
- configuration files
 - fast-start failover configuration (fsfo.dat) file, 5-38
 - See Also* Data Guard configuration file
- configuration properties
 - BystandersFollowRoleChange, 8-3
 - CommunicationTimeout, 8-4
 - FastStartFailoverAutoReinstate, 8-6
 - FastStartFailoverLagLimit, 8-7
 - FastStartFailoverPmyShutdown, 8-7
 - FastStartFailoverThreshold, 8-8
 - ObserverOverride, 8-9
 - ObserverReconnect, 8-9
 - OperationTimeout, 8-10
 - PrimaryLostWriteAction, 8-10
 - Tracelevel, 8-11
- CONNECT command, 6-3, 7-9
- connecting
 - privileges required for Data Guard broker configurations, 7-5
 - starting the observer, 6-12
 - to the primary database, 6-3
- controlling
 - available SGA memory, 4-17
- CONVERT DATABASE command, 7-11
- CREATE CONFIGURATION command, 6-3, 7-13
- creating
 - a broker configuration, 6-2, 6-3
 - with the Add Standby Database wizard, 3-7
 - a standby database, 3-7

D

- Data Guard
 - troubleshooting, 9-1
- Data Guard broker
 - See Also* broker
- Data Guard command-line interface
 - commands
 - ENABLE FAST_START FAILOVER, 7-41
 - SET STATE, 4-18
 - DG_BROKER_START initialization parameter, 2-2
 - enabling a database, 6-6
 - introduction, 1-10
- Data Guard command-line interface (DGMGRL)
 - banners from output, 7-1
 - commands
 - ADD DATABASE, 7-7
 - ADD FAR_SYNC, 7-8
 - CONNECT, 7-9
 - CONVERT DATABASE, 7-11

- CREATE CONFIGURATION, 7-13
- DISABLE CONFIGURATION, 7-15
- DISABLE DATABASE, 7-16
- DISABLE FAR_SYNC, 7-13, 7-17
- DISABLE FAST_START FAILOVER, 5-31, 7-18
- DISABLE FAST_START FAILOVER CONDITION, 7-20
- EDIT CONFIGURATION (property), 7-21
- EDIT CONFIGURATION (protection mode), 7-22
- EDIT CONFIGURATION (RENAME), 7-13, 7-24
- EDIT CONFIGURATION RESET (Property), 7-13, 7-25
- EDIT DATABASE (Property), 7-13, 7-26
- EDIT DATABASE (rename), 7-28
- EDIT DATABASE (state), 7-29
- EDIT DATABASE RESET (Property), 7-30
- EDIT FAR_SYNC, 7-31
- EDIT FAR_SYNC RESET (Property), 7-32
- EDIT INSTANCE (AUTO PFILE), 7-33
- EDIT INSTANCE (property), 7-34, 7-36
- EDIT INSTANCE RESET (Property), 7-36
- ENABLE CONFIGURATION, 7-38
- ENABLE DATABASE, 7-39
- ENABLE FAR_SYNC, 7-40
- ENABLE FAST_START FAILOVER, 7-41
- ENABLE FAST_START FAILOVER CONDITION, 7-43
- EXIT, 7-44
- FAILOVER, 6-23
- HELP, 7-48
- QUIT, 7-49
- REINSTATE DATABASE, 7-50
- REMOVE CONFIGURATION, 7-51
- REMOVE DATABASE, 7-53
- REMOVE FAR_SYNC, 7-55
- REMOVE INSTANCE, 7-56
- SHOW CONFIGURATION, 7-57, 7-59
- SHOW DATABASE, 7-61
- SHOW FAR_SYNC, 7-64
- SHOW FAST_START FAILOVER, 7-66
- SHOW INSTANCE, 7-67
- SHUTDOWN, 7-69
- SQL, 7-71
- START OBSERVER, 7-72
- STARTUP, 7-75
- STOP OBSERVER, 7-77
- SWITCHOVER, 6-19, 7-78
- VALIDATE DATABASE, 7-81
- VALIDATE FAR_SYNC, 7-87
- creating a configuration, 6-2
- DG_BROKER_START initialization parameter, 7-5
- enabling the configuration, 6-6
- FSFO.DAT file creation, 7-72
- monitoring broker configurations, 6-27
- performing routine management tasks, 6-14
- prerequisites, 6-1
 - setting critical database properties, 6-5
 - setting up standby redo log files, 4-22
 - single command mode, 7-1
 - starting, 6-2, 7-1
 - stopping, 7-6
 - string values, 7-5
 - summary of commands, 7-3
 - suppressing command prompts, 7-1
- Data Guard configuration file
 - for an Oracle RAC database, 3-5
 - in a CFS area, 3-5
 - inconsistent values from server parameter file, 8-13
 - relationship to DMON process, 1-10
 - setting up, 3-3
- Data Guard configurations
 - automated creation of, 1-7
 - background, 3-2
- Data Guard monitor (DMON), 1-10
 - interaction with the Oracle database, 1-10
 - managing objects, 3-2, 3-8, 6-6
 - Oracle database background processes, 1-10
 - overview, 1-10
 - starting with the DG_BROKER_START parameter, 3-6
 - two-way network communication, 1-11
- data protection modes
 - See* protection modes
- database resources
 - monitoring, 1-8
- databases
 - changing the state of, 6-14
 - connecting to, 6-2
 - creating and adding to a broker configuration, 1-9
 - dependencies, 4-1
 - disabling management of, 7-15
 - during
 - fast-start failover, 5-14
 - manual failover, 5-7
 - switchover, 5-4
 - enabling, 6-6
 - health, 4-32
 - installation for broker management, 2-2
 - monitorable properties, 4-6
 - network setup, 1-11, 2-2
 - prerequisites for broker configurations, 2-2
 - properties, 1-9, 4-6, 8-17 to 8-43
 - property management, 1-12
 - reenabling after a role transition, 5-12
 - removing
 - from a broker configuration, 6-18
 - restarting
 - after fast-start failover, 5-40
 - shutting down
 - effect on fast-start failover, 5-23, 5-26
 - states, 4-1
 - dependencies, 4-1
 - transitions, 4-3
 - status, 4-32

- DB_FILE_NAME_CONVERT initialization parameter
 - initialization parameters
 - setting DB_FILE_NAME_CONVERT in a
 - broker configuration, 8-23
- DBA_LOGSTDBY_EVENTS table
 - managing, 4-17
- DbFileNameConvert property, 8-23
- DBMS_DG package
 - using to direct fast-start failover, 5-27
- delaying
 - application of redo data with DelayMins
 - property, 4-16
- DelayMins property, 8-24
- destinations
 - archived redo log file parameters, 3-1
 - viewing the LogXptStatus property, 8-13
- DG_BROKER_CONFIG_FILEn file, 3-5
 - in a CFS area, 3-5
- DG_BROKER_START initialization parameter, 2-2, 6-1, 7-5
- DGConnectIdentifier property, 8-25
- DGMGRL
 - reinstating a database, 5-14
- diagnostic information
 - sources, 9-1
- DISABLE CONFIGURATION command, 7-15
 - example, 6-16
- DISABLE DATABASE command, 7-16
 - example, 6-16
- DISABLE FAR_SYNC command, 7-13, 7-17
- DISABLE FAST_START FAILOVER command, 5-31, 7-18
- DISABLE FAST_START FAILOVER CONDITION command, 7-20
- disabling
 - broker configuration, 6-15, 7-15
 - broker management of standby database, 7-16
 - databases, 3-9, 6-16
 - fast-start failover, 5-30, 7-18
 - on a standby database, 5-31
 - See also* each DISABLE command
- disaster protection
 - benefits, 1-5
- displaying
 - configuration information, 7-57
 - help for DGMGRL commands, 7-48
 - properties, 4-4
- distributed management framework, 1-2
- DMON
 - See* Data Guard monitor (DMON)
- downgrading
 - Data Guard, A-4
 - Oracle Enterprise Manager, A-4
 - protection mode, 4-24
- drc* log files
 - broker diagnostic information, 9-1
 - during reinstatement, 5-29
 - recording failed reinstatement, 5-40

E

- EDIT CONFIGURATION (Property) command, 7-21
- EDIT CONFIGURATION (protection mode)
 - command, 7-22
- EDIT CONFIGURATION (RENAME)
 - command, 7-13, 7-24
- EDIT CONFIGURATION RESET (Property)
 - command, 7-13, 7-25
- EDIT DATABASE (Property) command, 7-13, 7-26
- EDIT DATABASE (property) command
 - example, 6-14
- EDIT DATABASE (rename) command, 7-28
- EDIT DATABASE (state) command, 7-29
 - example, 6-15
- EDIT DATABASE RESET (Property) command, 7-30
- EDIT FAR_SYNC command, 7-31
- EDIT FAR_SYNC RESET (Property) command, 7-32
- EDIT INSTANCE (AUTO PFILE) command, 7-33
- EDIT INSTANCE (property) command, 7-34, 7-36
- EDIT INSTANCE RESET (Property) command, 7-36
- editing
 - configurable properties, 7-21
 - database instance, 7-33, 7-34, 7-36
 - database name, 7-28
 - database state, 7-29
 - protection modes, 7-22
- effect of switchovers on, 5-7
- e-mail
 - reporting events, 1-9
- ENABLE CONFIGURATION command, 6-6, 7-38
- ENABLE DATABASE command, 7-39
- ENABLE FAR_SYNC command, 7-40
- ENABLE FAST_START FAILOVER command, 7-41
- ENABLE FAST_START FAILOVER CONDITION command, 7-43
- enabling
 - broker configuration, 3-8, 6-6
 - databases in a broker configuration, 3-9
 - fast-start failover, 5-16, 5-21
 - See also* each ENABLE command
- Enterprise Edition database
 - installation, 2-1
- Enterprise Manager
 - Fast-Start Failover wizard
 - disabling fast-start failover, 5-31
 - wizards
 - automate standby database creation, 1-3
- error messages
 - ORA-16795, 5-12, 5-13
- error status, 3-10
- events
 - managing
 - in a logical standby database, 4-17
 - monitoring with Oracle Enterprise Manager, 1-4
 - Oracle Enterprise Manager, 1-8
 - reporting, 1-9
 - responding to, 1-8
- EXIT command, 7-44
 - See also* QUIT command

F

- failover
 - failing over to a standby database, 7-45
 - Oracle Enterprise Manager, 1-9
- FAILOVER command, 6-23, 7-45
- failovers
 - and FastStartFailoverTarget property, 6-12
 - benefits, 1-7
 - broker tasks, 5-10
 - choosing a target standby database, 5-2
 - complete, 5-10
 - fast-start failover, 5-25
 - effect on data protection mode, 5-16
 - V\$DATABASE, 5-28
 - immediate, 5-12
 - managing, 5-1
 - fast-start, 5-14
 - manual, 5-7
 - manual
 - complete option, 5-7
 - effect on data protection mode, 5-9
 - immediate option, 5-8
 - reenabling disabled databases after, 5-11
 - starting manually, 5-8
 - using DGMGRL, 6-23
- failures
 - observer, 5-36
- far sync instances
 - managing, 4-27
- fast-start failover, 5-1
 - configuration information in the fsfo.dat file, 5-38
 - defined, 5-14
 - DGMGRL client as the observer, 1-10
 - directed from an application, 5-27
 - disabling, 5-30, 5-31
 - to perform manual failover, 5-33
 - enabling, 5-21, 7-41
 - instance failures, 5-23
 - monitoring by the observer, 5-23
 - observer monitoring, 5-23
 - observer overview, 5-15
 - performing manual failovers when enabled, 5-26
 - performing switchover when enabled, 5-26
 - querying V\$DATABASE, 5-25
 - reinstating the former primary database, 5-39
 - requirements, 5-16
 - restarting databases after shut down, 5-40
 - setting the FastStartFailoverTarget property, 5-17, 6-12
 - setting the FastStartFailoverThreshold property, 5-19
 - shutdown abort, 5-23, 5-26
 - to Oracle RAC database instances, 5-24
 - unobserved, 5-36
 - verifying the environment, 5-22
 - viewing failover configuration statistics, 5-28
- FastStartFailoverAutoReinstate property, 8-6
- FastStartFailoverLagLimit property, 8-7
- FastStartFailoverPmyShutdown property, 8-7
- FastStartFailoverTarget property, 8-26

- setting, 5-17, 6-12
- FastStartFailoverThreshold property, 8-8
 - setting, 5-19
- FASTSYNC redo transport mode, 8-31
- FASTSYNC redo transport service, 4-9
- files
 - naming the server parameter file, 1-12
- Flashback Database
 - reinstating databases, 5-13
 - use after failover, 5-11
- flashback logs
 - reinstating a failed primary database, 5-13
- FORCE option
 - disabling fast-start failover, 5-31, 7-18
 - performing a manual failover, 5-33
- FS_FAILOVER_OBSERVER_HOST column
 - of V\$DATABASE view, 5-35
- FS_FAILOVER_OBSERVER_PRESENT column
 - of V\$DATABASE view, 5-36
- FS_FAILOVER_STATUS column
 - of V\$DATABASE view
 - failed reinstatement, 5-29
- FSFO.DAT file
 - overview, 5-38
 - START OBSERVER command, 7-72

H

- health check, 1-8
 - monitoring, 1-10, 1-11
 - on primary database, 4-32
 - on standby database, 4-32
 - revealed by configuration status, 3-10
- HELP command, 7-48
- high availability
 - benefits, 1-6
 - levels of data protection, 4-20, 8-31
 - LogXptMode property, 8-31
 - restoring after fast-start failover, 5-39

I

- immediate failover, 5-8, 7-45
- InconsistenLogXptProps property, 8-12
- InconsistentProperties property, 8-13
- increased scalability
 - benefits, 1-6
- initialization parameters
 - COMPATIBLE, 2-3
 - database configurable properties, 4-6
 - DG_BROKER_CONFIG_FILEn, 3-5
 - DG_BROKER_START, 2-2, 6-1
 - dynamic, 4-6
 - inconsistent, 8-13
 - LOG_ARCHIVE_DEST_n, 4-14
 - static, 4-6
 - See also* server parameter file, 2-2
- installation
 - ARCHIVELOG mode setup, 2-3
 - Data Guard, 2-1

- Oracle Enterprise Manager, 2-1
- Oracle Instant Client, 2-1, 7-72
- prerequisites, 2-2
- instances
 - failures, 5-23
 - removing, 7-20, 7-43
 - shutting down, 7-69
 - starting, 7-75
- intended state
 - configuration health check, 3-10
- invoking
 - the Data Guard command-line interface, 6-2

L

- life cycle of a broker configuration, 3-7
- log apply services
 - and apply instance failover, 4-17
 - and parallel apply, 4-16
 - apply instance, 4-17
 - configuring, 1-9
 - Data Guard configuration, 3-1
 - delayed application of redo data, 4-16
 - delaying, 8-24
 - in an Oracle RAC database, 4-17
 - managing, 4-15
 - selecting the apply instance, 4-17
 - verifying, 1-7
- log files
 - broker, 5-40
 - FSFO.DAT
 - creating, 7-72
- LOG_ARCHIVE_DEST_n initialization parameter
 - setting redo transport services, 4-14
 - setting the DELAY attribute, 8-24
 - setting the ENABLE and DEFER attributes, 8-30
 - setting the MANDATORY or OPTIONAL attributes, 8-23
- LOG_ARCHIVE_FORMAT initialization parameter, 8-27
- LOG_ARCHIVE_MAX_PROCESSES initialization parameter, 8-27
- LOG_ARCHIVE_MIN_SUCCEED initialization parameter, 8-28
- LOG_ARCHIVE_TRACE initialization parameter
 - setting LogArchiveTrace property, 8-28
- LOG_FILE_NAME_CONVERT initialization parameter
 - setting LogFileNameConvert property, 8-29
- LogArchiveFormat property, 8-27
- LogArchiveMaxProcesses property, 8-27
- LogArchiveMinSucceedDest property, 8-28
- LogArchiveTrace property, 8-28
- LogFileNameConvert property, 8-29
- logical standby databases
 - managing events, 4-17
 - SQL Apply, 4-17
 - state transitions, 4-4
 - switchover to the primary role, 5-4
- LogShipping property, 8-30

- LogXptMode property, 8-31
- LogXptStatus property, 8-13
- LsbyFailedTxnInfo property, 8-14
- LsbyMaxEventsRecorded property, 8-32
- LsbyMaxServers property, 8-33
- LsbyMaxSga property, 8-33
- LsbyParameters property, 8-14
- LsbyPreserveCommitOrder property, 8-34
- LsbyRecordAppliedDdl property, 8-34
- LsbyRecordSkipDdl property, 8-35
- LsbyRecordSkipErrors property, 8-35

M

- management
 - benefits, 1-7
 - Oracle Enterprise Manager, 1-9
- managing
 - a broker configuration, 3-1, 6-1
 - databases, 4-1
 - DBA_LOGSTDBY_EVENTS table, 4-17
 - events
 - in a logical standby database, 4-17
 - fast-start failover, 5-14
 - local operations, 1-2
 - log apply services, 4-15
 - manual failover, 5-7
 - parallel apply in a physical standby database, 4-16
 - remote operations, 1-2
 - roles, 5-1
 - fast-start failover, 5-14
 - manual failover, 5-7
 - switchover, 5-4
- MANDATORY attribute
 - set with the Binding property, 8-23
- manual failover, 1-7, 5-1
 - complete option, 5-7
 - effect on protection modes, 4-25
 - immediate option, 5-8
 - performing when fast-start failover is enabled, 5-26, 5-33
 - starting, 5-8
- manual role changes
 - performing when fast-start failover is enabled, 5-26
- MaxConnections property, 8-36
- MaxFailure property, 8-37
- maximize availability, 1-7, 3-9, 8-31
- maximize data protection, 1-7, 3-9, 8-31
- maximize performance, 1-7, 3-9, 8-31
- maximum availability
 - data protection mode, 4-21
- maximum performance
 - data protection mode, 4-21
- maximum protection
 - data protection mode, 4-21
- monitorable properties, 4-4, 4-6, 8-12
- InconsistentLogXptProps, 8-12

- InconsistentProperties, 8-13
- LogXptStatus, 8-13
- LsbyFailedTxnInfo, 8-14
- LsbyParameters, 8-14
- RecvQEntries, 8-14
- SendQEntries, 8-16
- TopWaitEvents, 8-17
- monitoring
 - broker configurations, 1-10, 6-1, 6-27
 - local and remote databases, 1-8
 - observer
 - fast-start failover environment, 5-23
 - Oracle Enterprise Manager performance page, 3-9
 - through the command-line interface, 1-10
- multitenant container databases
 - See* CDBs

N

- NetTimeout property, 8-37
- networks
 - Data Guard configuration, 3-1
 - setting up files, 2-2
 - two-way communication, 1-11
- normal status, 3-10

O

- objects
 - broker configuration, 3-2
 - disabling, 6-15
 - properties for databases, 1-12
- observer, 5-15
 - and FastStartFailoverThreshold property, 5-19
 - connecting to the configuration, 6-12
 - connection to the local computer, 5-36
 - detecting
 - database shutdown, 5-23, 5-26
 - instance failures, 5-23
 - failures, 5-36
 - finding host machine name, 5-35
 - installing Oracle Instant Client kit, 2-1, 7-72
 - lost connection, 5-36
 - maintaining fast-start failover configuration information, 5-38
 - monitoring, 5-23
 - shutting down, 5-40
 - using Oracle Wallet for connection credentials, 7-10
- ObserverConnectIdentifier property, 8-38
- ObserverOverride property, 8-9
- ObserverReconnect property, 8-9
- operations
 - complete failover, 7-45
 - disable broker management
 - effect on protection modes, 4-26
 - downgrade
 - effect on protection modes, 4-24
 - effect on protection modes, 4-24

- enable broker management
 - effect on protection modes, 4-26
- failover, 6-23
- immediate failover, 7-45
- manual failover
 - effect on protection modes, 4-25
 - removing a database from the configuration
 - effect on protection modes, 4-26
- OperationTimeout property, 8-10
- OPTIONAL attribute
 - set with the Binding property, 8-23
- ORA-16661 message
 - reinstating a database, 5-12, 5-13
- ORA-16795 message
 - re-creating a database, 5-13
- Oracle Clusterware, 1-6
 - and instances of an Oracle RAC database, 1-6
 - integration with Data Guard broker, 1-6
 - recover failed instances, 1-6
- Oracle Enterprise Manager, 1-9
 - Add Standby Database wizard, 1-9
 - database property pages, 1-9
 - downgrading, A-4
 - event management system, 1-8
 - integration, 1-9
 - integration with the Data Guard monitor, 1-9
 - introduction, 1-2, 1-8
 - making Oracle Net Services configuration changes, 1-7, 1-9
 - monitoring events, 1-4
 - performance tools and graphs, 1-9
 - setting up standby redo log files, 4-22
 - upgrading, A-2
 - wizards
 - creating standby databases, 1-9
- Oracle Instant Client
 - installing, 2-1, 7-72
- Oracle Managed Files (OMF)
 - creating a standby database that uses, 2-2
- Oracle Net Services
 - configuration changes, 1-7, 1-9
 - installation prerequisites, 2-2
 - supported configuration, 3-1
 - two-way communication, 1-11
- Oracle RAC One Node
 - supported by Oracle Data Guard broker, 1-1
- Oracle Real Application Clusters
 - and log apply services, 4-17
 - and setting the apply instance, 4-17
 - availability of instances with Oracle Clusterware, 1-6
 - benefits, 1-6
 - fast-start failover, 5-24
 - observer behavior in, 5-24
 - observer connection to instances, 5-36
- Oracle Clusterware, 1-6
- Oracle Universal Installer
 - installing Oracle Instant Client, 2-1, 7-72
- Oracle Wallet
 - starting the observer, 7-10

P

- parallel apply
 - and log apply services, 4-16
 - managing in a physical standby database, 4-16
- performance
 - Oracle Enterprise Manager tools, 1-9
- physical standby databases
 - managing parallel apply, 4-16
 - state transitions, 4-3
 - switchover to the primary role, 5-4
- PreferredApplyInstance property, 8-38
- prerequisites
 - installation, 2-2
 - switchover, 5-4
- primary database
 - ARCHIVELOG mode, 2-3
 - connecting to, 6-2, 6-3
 - constructing a standby database, 3-7, 6-2
 - Data Guard configuration, 3-1
 - during failover, 1-9
 - during switchover, 5-6
 - Flashback Database, 5-13
 - health check, 4-32
 - preparing for switchover, 5-5
 - reinstating after a fast-start failover, 5-39
 - standby redo log files on, 4-22
 - state transitions, 4-3
 - switching over to the standby role, 7-78
- PrimaryLostWriteAction property, 8-10
- processes
 - DMON, 1-10
 - Oracle database, 1-10
- properties, 4-6
 - configurable, 4-4
 - database, 4-4
 - managing, 1-12
 - monitorable, 4-4
 - setting, 3-9, 6-5
 - in server parameter file, 1-12, 2-2
- property pages
 - database, 1-9
- protection modes
 - after a failover, 4-25, 5-9, 5-16
 - benefits, 1-7
 - configuration, 7-22, 7-24, 7-25
 - downgrading, 4-24
 - redo transport services setup, 8-31
 - setting for a broker configuration, 4-20
 - updating, 3-9
 - upgrading, 4-24

Q

- QUIT command, 7-49
 - See also* EXIT command

R

- RecvQEntries property, 8-14
- Redo Apply, 3-2, 5-7

- redo transport services
 - ASYNCR mode, 4-8, 8-31
 - configuring, 1-9
 - data protection modes, 8-31
 - Data Guard configuration, 3-1
 - FASTSYNC mode, 4-9, 8-31
 - managing, 4-7
 - SYNC mode, 4-8, 8-31
 - tuning, 4-14
 - turning off, 4-13
 - turning on, 4-12
 - verifying, 1-7
- RedoCompression property, 8-39
- RedoRoutes property, 8-39
- REINSTATE DATABASE command, 7-50
- reinstatement, 5-39
 - failure logged in broker drc* log files, 5-29
 - Flashback Database requirement, 5-13
 - flashback logs requirement, 5-13
 - how to, 5-13
 - logged in broker log files, 5-40
 - ORA-16661 message, 5-12, 5-13
 - requires standby redo log files, 4-22
 - using DGMGRL, 5-14
 - using Enterprise Manager, 5-13
- REMOVE CONFIGURATION command, 7-51
 - example, 6-19
- REMOVE DATABASE command, 7-53
- REMOVE FAR_SYNC command, 7-55
- REMOVE INSTANCE command, 7-56
- removing
 - a standby database, 4-26
 - See* each REMOVE command
- ReopenSecs property, 8-41
- requests
 - passing between sites, 1-11
- requirements
 - for enabling fast-start failover, 5-16
- restarting
 - databases
 - when fast-start failover is enabled, 5-40
- role transitions
 - changing, 3-8
 - managing, 5-1
 - fast-start failover, 5-14
 - manual failover, 5-7
 - switchover, 5-4

S

- scripts
 - using Data Guard command-line interface (DGMGRL), 7-1
- selecting the apply instance, 4-17
- SendQEntries property, 8-16
- server parameter file
 - broker property management, 1-12, 2-2
 - filenames, 1-12
 - inconsistent values from Data Guard configuration file, 8-13

- server-side software, 1-10
- SET STATE command
 - and setting the apply instance, 4-18
- setting
 - configuration protection mode, 4-20
 - database properties, 6-5
 - log apply services, 4-13
 - redo transport services, 4-14
- SGA memory
 - and SQL Apply, 4-17
- SHOW CONFIGURATION command, 6-4, 6-7, 7-57, 7-59
- SHOW CONFIGURATION VERBOSE
 - example, 5-28
- SHOW DATABASE command, 7-61
- SHOW DATABASE VERBOSE command
 - displaying properties, 4-4
- SHOW FAR_SYNC command, 7-64
- SHOW FAST_START FAILOVER command, 7-66
- SHOW FAST-START FAILOVER command
 - example, 5-27
- SHOW INSTANCE command, 7-67
- showing
 - See each SHOW command
- shutdown
 - FS_FAILOVER_OBSERVER_PRESENT view, 5-40
- shutdown abort, 5-23
 - effect on fast-start failover, 5-26
- SHUTDOWN command, 7-69
- shutting down an Oracle instance, 7-69
- SidName property, 8-42
- single command mode
 - for Data Guard command-line interface, 7-1
- SQL Apply, 3-2
 - and logical standby databases, 4-17
 - effect of switchovers on, 5-7
- SQL command, 7-71
- SQL statements
 - executing from DGMGRL command line, 7-71
- standby databases
 - choosing a target standby database, 5-2
 - constructing from backups, 3-7, 6-2
 - creating, 1-9
 - if primary database uses ASM or OMF, 2-2
 - health check, 4-32
 - not involved in a switchover, 5-7
 - reenabling after failover, 5-11
 - removing, 4-26
 - require standby redo log files, 4-22
 - specifying the location of archived redo logs, 4-13
 - switching over to the primary role, 7-78
- standby redo log files
 - setting up, 4-22
 - setting up with DGMGRL, 4-22
- StandbyArchiveLocation property, 8-43
 - setting log apply services, 4-13
- StandbyFileManagement property, 8-43
- START OBSERVER command
 - fast-start failover configuration file (fsfo.dat), 5-38
- starting
 - Data Guard command-line interface (DGMGRL), 6-2, 7-1
 - Data Guard monitor (DMON), 3-6
 - manual failover, 5-8
 - Oracle instance, 7-75
 - switchover, 5-6
- STARTUP command, 7-75
- state transitions
 - effect on database states, 4-3
 - logical standby database, 4-4
 - physical standby database, 4-3
 - primary database, 4-3
- states, 3-9
 - changing, 6-14
 - database, 4-1
 - database transitions, 4-3
 - logical standby database
 - APPLY-OFF, 4-2
 - APPLY-ON, 4-2
 - physical standby database
 - APPLY-OFF, 4-2
 - APPLY-ON, 4-2
 - primary database
 - TRANSPORT-OFF, 4-2
 - TRANSPORT-ON, 4-2
- StaticConnectIdentifier property, 8-44
- status
 - configuration, 3-10
 - health check on primary database, 4-32
 - health check on standby database, 4-32
 - health of the database, 4-32
 - intended state of a configuration, 3-10
- status messages
 - ORA-16661, 5-12
 - ORA-16795, 5-13
- STOP OBSERVER command, 7-72, 7-77
- string values
 - Data Guard command-line interface, 7-5
- SWITCHOVER command, 6-19, 7-78
- switchovers
 - benefits, 1-7
 - broker tasks, 5-6
 - choosing a target standby database, 5-2
 - DGMGRL SWITCHOVER command, 6-19
 - effect on
 - database startup, 7-78
 - primary database, 5-6
 - protection modes, 4-25
 - standby databases not involved in the switchover, 5-7
 - managing, 5-1, 5-4
 - overview, 5-1
 - Oracle Enterprise Manager, 1-9
 - performing when fast-start failover is enabled, 5-26
 - preparing the primary database, 5-5
 - prerequisites, 5-4
 - reenabling disabled standby databases, 5-12

- starting, 5-6
- transitioning
 - a logical standby database to the primary role, 5-4
 - a physical standby database to the primary role, 5-4
- using the DGMGRL SWITCHOVER command, 7-78
- SYNC redo transport mode, 4-8
- SYSDBA privileges
 - to connect to the database, 7-5
- SYSDG privileges, 6-3

T

- target standby database
 - choosing, 5-2
 - disabling fast-start failover on, 5-31
 - setting with the FastStartFailoverTarget property, 5-17, 6-12
- TopWaitEvents property, 8-17
- TraceLevel property, 8-11
- transaction commit order
 - controlling, 4-17
- TransportDisconnectedThreshold property, 8-45
- TransportLagThreshold property, 8-45
- TRANSPORT-ON state
 - setting LogShipping property, 8-30
- troubleshooting
 - Data Guard, 9-1
 - diagnostics logged in broker drc* log files, 9-1
- tuning
 - redo transport services, 4-14
- two-way communication channel, 1-11

U

- unobserved configuration, 5-36
- updating configuration properties, 4-4
- upgrading
 - Data Guard, A-1, A-2
 - Oracle Enterprise Manager, A-2
 - protection mode, 4-24
- user interfaces
 - overview, 1-9

V

- V\$DATABASE view
 - fast-start failover columns, 5-25, 5-28
 - FS_FAILOVER_OBSERVER_HOST column, 5-35
 - FS_FAILOVER_OBSERVER_PRESENT column, 5-36
 - when shutting down databases, 5-40
- verifying
 - fast-start failover environment, 5-22
- viewing
 - fast-start failover statistics, 5-28
- V\$FS_FAILOVER_STATS view, 5-30
- VALIDATE DATABASE command, 7-81
- VALIDATE FAR_SYNC command, 7-87

- verifying
 - fast-start failover environment, 5-22
- viewing
 - property information about databases, 4-6, 8-12

W

- warning status, 3-10
- wizards
 - Add Standby Database, 1-9