

Oracle® Database

Global Data Services Concepts and Administration Guide



19c
E96443-04
June 2023

The Oracle logo, consisting of a solid red square with the word "ORACLE" in white, uppercase, sans-serif font centered within it.

ORACLE®

Oracle Database Global Data Services Concepts and Administration Guide, 19c

E96443-04

Copyright © 2013, 2023, Oracle and/or its affiliates.

Primary Author: Virginia Beecher

Contributing Authors: Janet Stern, Richard Strohm

Contributors: Steve Ball, Srinagesh Battula, Nourdine Benadjaoud, Laurence Clarke, David Colello, Mark Dilman, Shahab Hamid, Wei Hu, Bob McGuirk, Joseph Meeks, Leonid Novak, Cris Pedregal-Martin, Nick Wagner, Jean Zeng

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

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

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

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

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

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

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

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

Contents

Preface

Audience	ix
Documentation Accessibility	ix
Related Documents	ix
Conventions	ix

Changes in This Release for Oracle Database Global Data Services Concepts and Administration Guide

Changes in Oracle Database 19c	xi
Changes in Oracle Database 18c Release 1 (18.1)	xii
Changes in Oracle Database 12c Release 2 (12.2.0.1)	xii

1 Introduction to Global Data Services

Introduction to Global Data Services	1-1
Global Data Services Architecture	1-2
Global Data Services Pool	1-3
Global Data Services Region	1-3
Global Service Manager	1-4
Global Data Services Catalog	1-4
Oracle Notification Service Servers	1-4
Overview of Global Services	1-5
Global Service Attributes	1-6
Global Services in an Oracle RAC Database	1-7
Global Services in an Oracle Data Guard Broker Configuration	1-7
Database Placement of a Global Service	1-8
Replication Lag and Global Services	1-9
Global Services and Distributed Transaction Processing	1-11
Global Services in Multitenant Architecture	1-11
Global Connection Load Balancing	1-12
Client-Side Load Balancing	1-12
Server-Side Load Balancing	1-12

Region Affinity for Global Services	1-13
Any-Region Affinity	1-13
Affinity to a Local Region	1-13
Affinity to a Local Region with Interregion Failover	1-13
Global Run-Time Connection Load Balancing	1-14
Affinity	1-15
Disk I/O and CPU Thresholds	1-15
Global Services Failover	1-15
Global Service Manager Process Suite Architecture	1-17
Global Data Services Use Cases	1-18
Load Balancing for Replicated Databases	1-18
Service Failover for Replicated Databases	1-19
Region Affinity in Oracle GoldenGate Multi-Master	1-21
Load Balancing in Oracle GoldenGate Multi-Master	1-23
Balancing Oracle Active Data Guard and Oracle GoldenGate Reader Farms	1-24

2 Configuring the Global Data Services Framework

Planning an Installation	2-1
What You Need to Know About Installing a Global Service Manager	2-1
Installing a Global Service Manager	2-2
Performing a Silent Install of Global Service Manager	2-3
What You Need to Know About Upgrading Global Data Services	2-4
Upgrading Global Data Services	2-5
GSM Out-of-Place Update/Patching Examples	2-7
GSM_HOME to 19.18.0.0.0 DBRU, Move Existing GSM to New Home on Same Host	2-12
GSM_HOME to 19.18.0.0.0 DBRU on a Different Host	2-15
Using GDSCTL	2-19
Operational Notes	2-19
GDSCTL Command Syntax and Objects	2-21
GDSCTL Connections	2-22
What You Need to Know About Creating the Global Data Services Catalog	2-23
Creating the Global Data Services Catalog	2-24
Adding a Global Service Manager to the Global Data Services Catalog	2-24
Connecting to the Global Data Services Catalog	2-25
What You Need to Know About Adding a Global Data Services Pool	2-27
Adding a Global Data Services Pool	2-27
What You Need to Know About Adding a Global Data Services Region	2-28
Adding a Global Data Services Region	2-28
Adding a Database to a Global Data Services Pool	2-28
Valid Node Checking for Registration	2-29

Adding a Service to a Global Data Services Pool	2-29
Starting a Global Service	2-30
Database Client Configuration	2-31
What You Need to Know About Exporting the GDS Catalog Data for Logical Backups	2-33
Exporting the GDS Catalog Data for Logical Backups	2-33
Restoring Logical Backup of the GDS Catalog into the Same Catalog Database	2-33
Restoring Logical Backup of the GDS Catalog into a new Catalog Database	2-34
Changing the GSMCATUSER Password	2-34

3 Administering Global Data Services Configurations

Overview of Global Data Services Administration	3-1
Managing Database Pools	3-3
Adding Oracle Data Guard Broker Managed Databases to a Database Pool	3-3
Managing Global Services	3-3
Creating a Global Service	3-4
Starting a Global Service	3-4
Stopping a Global Service	3-5
Disabling a Global Service	3-6
Enabling a Global Service	3-6
Modifying Global Service Attributes	3-7
Deleting a Global Service	3-7
Managing the GDS Stack	3-7
Starting Up the GDS Stack	3-7
Shutting Down the GDS Stack	3-8

4 Troubleshooting Global Data Services

Troubleshooting Oracle Error Codes	4-1
ORA-01045: user GSMADMIN_INTERNAL lacks CREATE SESSION privilege; logon denied	4-1
ORA-12514: TNS:listener does not currently know of service requested in connect descriptor	4-1
ORA-12516: TNS:listener could not find available handler with matching protocol stack	4-1
ORA-12541: TNS:no listener	4-1
GSM-45034: Connection to GDS catalog is not established	4-1
GSM-45054: GSM error or NET-40006: unable to start GSM	4-2
Solutions for General Issues	4-2
Connecting to GDS Configuration Databases When No Global Service Managers Are Running	4-2
Connecting to Catalog Databases When No Global Service Managers Are Running	4-3
Obtaining the Running Status of Global Data Services Components	4-3

Viewing Static Configuration Information for Global Data Services Components	4-3
Enabling and Disabling Tracing on a Global Service Manager	4-4
Using Global Service Manager Log and Trace Files	4-4
Using SYS_CONTEXT Parameters in a GDS Environment	4-5

A GDSCTL Commands Used For Oracle Sharding

B GDSCTL Commands Used For Global Data Services

C Global Data Services Control Utility (GDSCTL) Command Reference

add brokerconfig	C-1
add cdb	C-2
add credential	C-3
add database	C-4
add file	C-6
add gdspool	C-7
add gsm	C-8
add invitednode (add invitedsubnet)	C-10
add region	C-11
add service	C-12
add shard	C-18
add shardgroup	C-20
add shardspace	C-22
config	C-23
config cdb	C-23
config chunks	C-24
config credential	C-25
config database	C-26
config file	C-26
config gdspool	C-27
config gsm	C-28
config region	C-29
config sdb	C-30
config service	C-31
config shard	C-33
config shardgroup	C-33
config shardspace	C-34
config table family	C-35
config vncr	C-36

configure	C-36
connect	C-37
create catalog	C-39
create gdscatalog	C-41
create shard	C-43
create shardcatalog	C-46
databases	C-50
delete catalog	C-51
deploy	C-52
disable service	C-53
enable service	C-54
exit	C-55
export catalog	C-55
help	C-56
import catalog	C-56
modify catalog	C-57
modify cdb	C-59
modify credential	C-60
modify database	C-61
modify file	C-62
modify gdspool	C-63
modify gsm	C-63
modify region	C-65
modify service	C-66
modify shard	C-74
modify shardgroup	C-75
modify shardspace	C-76
move chunk	C-77
quit	C-78
recover shard	C-78
relocate service	C-79
remove brokerconfig	C-80
remove cdb	C-81
remove credential	C-82
remove database	C-82
remove file	C-83
remove gdspool	C-84
remove gsm	C-84
remove invitednode (remove invitedsubnet)	C-85
remove region	C-86
remove service	C-86

remove shard	C-87
remove shardgroup	C-88
remove shardspace	C-89
services	C-89
set gsm	C-91
set inbound_connect_level	C-92
set inbound_connect_timeout	C-92
set log_level	C-93
set log_status	C-94
set outbound_connect_level	C-95
set outbound_connect_timeout	C-95
set trace_level	C-96
set trc_level	C-97
show ddl	C-98
split chunk	C-99
sql	C-100
start gsm	C-100
start observer	C-101
start service	C-102
status	C-103
status database	C-104
status gsm	C-105
status service	C-107
stop gsm	C-108
stop service	C-108
sync brokerconfig (synchronize brokerconfig)	C-110
sync database (synchronize database)	C-111
validate catalog	C-112
validate	C-113

Glossary

Index

Preface

Oracle Database Global Data Services provides dynamic load balancing, failover, and centralized service management for a set of replicated databases that offer common services. The set of databases can include an Oracle Sharding architecture, Oracle Real Application Clusters (Oracle RAC) and noncluster Oracle databases interconnected through Oracle Data Guard, Oracle GoldenGate, or any other replication technology.

This Preface contains these topics:

Audience

This document is intended for database administrators, Architects, and engineers who manage replicated databases.

Documentation Accessibility

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

Access to Oracle Support

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

Related Documents

For more information, see the following documents:

- *Oracle Data Guard Broker*
- *Oracle Data Guard Concepts and Administration*
- *Oracle Database Net Services Administrator's Guide*
- *Oracle Database Net Services Reference*
- *Oracle Real Application Clusters Administration and Deployment Guide*
- *Using Oracle Sharding*

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 Database Global Data Services Concepts and Administration Guide

This preface contains:

Changes in Oracle Database 19c

The following are changes in *Oracle Database Global Data Services Concepts and Administration Guide* for Oracle Database 19c.

New Features

The following features are new in this release:

Multiple Table Family Support for System-Managed Sharding

The Oracle Sharding feature for Oracle Database 18c supported only one table family (a set of related tables sharing the same sharding key) for each sharded database. In Oracle Database 19c, Oracle Sharding includes support for multiple table families where all data from different table families reside in the same chunks. This feature applies to system-managed sharded databases only. Different applications accessing different table families can now be hosted on one sharded database.

There is one new GDSCTL command, `CONFIG TABLE FAMILY`, and several other commands are extended to support this feature: `ADD SERVICE`, `MODIFY SERVICE`, `CONFIG SERVICE`, `CONFIG CHUNKS`, `STATUS ROUTING`, and `VALIDATE CATALOG`.

There are no new SQL keywords or statements introduced with this feature; however, some restrictions are changed with the use of `CREATE SHARDED TABLE`, `DUPLICATED TABLE`, and `TABLESPACE SET`.

See

- [Global Data Services Control Utility \(GDSCTL\) Command Reference](#)
- *Using Oracle Sharding*
- *Oracle Database SQL Language Reference*

GSMROOTUSER

A new user called GSMROOTUSER is used to log into CDB\$ROOT for CDBs in a sharding configurations (this user is not used in GDS configurations). Any connections to CDB\$ROOT in a CDB will now be with GSMROOTUSER.

See

- [Global Data Services Control Utility \(GDSCTL\) Command Reference](#)

Deprecation and Desupport

The following features are deprecated or desupported in this release:

Desupport of Setting Passwords in GDSCTL Command Line

To enhance security, starting with Oracle Database 19c, the ability to specify passwords from the Global Data Services Control Utility (GDSCTL) command-line when called from the operating system prompt is no longer supported.

This desupport applies only to password changes where GDSCTL is called from a user command-line prompt. For example, the following command is desupported:

```
$ gdsctl add database -connect inst1 -pwd gsm_password
```

Specifying the password from the GDSCTL utility itself is still valid. For example, the following command is valid:

```
GDSCTL> add database -connect inst1 -pwd gsm_password
```

This deprecation addresses the security vulnerability when specifying passwords in GDSCTL commands called from the operating system prompt.

Changes in Oracle Database 18c Release 1 (18.1)

The following are changes in *Oracle Database Global Data Services Concepts and Administration Guide* for Oracle Database 18c Release 1 (18.1)

New Features

The following features are new in this release:

- ADD SHARD is extended and new commands ADD CDB, MODIFY CDB, CONFIG CDB, and REMOVE CDB are implemented so that Oracle Sharding can support a multitenant architecture.

See [add shard](#), [add cdb](#), [modify cdb](#), [config cdb](#), and [remove cdb](#).

- With the release of Oracle GoldenGate 18c, the composite sharding method is supported with GoldenGate replication. The `add shardgroup` and `create shardcatalog` command documentation is updated accordingly.

See [add shardgroup](#) and [create shardcatalog](#).

Changes in Oracle Database 12c Release 2 (12.2.0.1)

The following are changes in *Oracle Database Global Data Services Concepts and Administration Guide* for Oracle Database 12c Release 2 (12.2.0.1).

New Features

The following features are new in this release:

- Oracle Sharding

Oracle Sharding is a scalability and availability feature that supports distribution and replication of data across a pool of discrete Oracle databases that share no hardware or software. In this release, global service managers are used as shard directors which route connection requests to individual shards, the GDS catalog is extended to support a shard catalog which contains the metadata for the sharded database, and GDSCTL is enhanced with several new commands to facilitate the creation, monitoring, and lifecycle management of a sharded database.

See [GDSCTL Commands Used For Oracle Sharding](#)

1

Introduction to Global Data Services

This chapter provides an overview of the Global Data Services architecture.

This chapter includes the following topics:

Introduction to Global Data Services

Many enterprises consolidate their information technology infrastructure to improve business efficiency. Database consolidation is a critical part of this process. However, most organizations must still maintain local and remote replicas of their databases for reasons that include:

- Business continuity and disaster recovery
- High availability
- Performance optimization for local clients
- Content localization and caching
- Compliance with local laws

In any set of database replicas, some database servers may have a slow query response time because of high load or high network latency, while other servers capable of offering a faster response time may be under-utilized. Optimal query performance and resource utilization across a set of database replicas requires a workload management solution that provides dynamic load balancing of client connections and workload requests across the replicas.

Many enterprises use a home-grown solution for workload management across database replicas. These solutions cannot provide critical functionality such as run-time load balancing and reliable database service failover between replicas because they are not fully integrated with Oracle software.

Oracle Database provides a powerful workload management feature called database services. Database services are named representations of one or more database instances. Database services allow you to group database workloads, ensure that client requests are routed to the optimal instance that offers a service, and provide high availability by transparently failing client connections over to surviving instances when a planned or unplanned instance outage occurs.

Oracle Global Data Services (GDS) implements the Oracle Database service model across a set of replicated databases known as a **Global Data Services configuration**.

A Global Data Services configuration looks like a virtual multi-instance database to database clients. It provides client access through **global services**, which are functionally similar to the local database services provided by single-instance or Oracle Real Application Clusters (Oracle RAC) databases. Local and global services both provide load balancing, high availability, and resource management. The essential difference between global services and local services is that global services span the instances of multiple databases, whereas local services span the instances of a single database.

A Global Data Services configuration and its global services are created and managed using the **GDSCTL** command-line interface, which is similar to the **SRVCTL** command-line interface used to manage an Oracle RAC database and its services.

A Global Data Services configuration can consist of any combination of multi-instance or single-instance Oracle databases hosted on heterogeneous or non-heterogeneous server platforms. Oracle Data Guard, Oracle GoldenGate, or any other database replication technology, can be used to synchronize the databases in a Global Data Services configuration.

Global Data Services is a highly effective solution for automatic workload management across a set of replicated databases, whether used with many widely distributed databases and clients or with a single database, a local replica and a few clients.

 **Note:**

Global Data Services is primarily intended for applications that are replication-aware. A replication-aware application is one that has any of the following characteristics:

- Uses read-only global services exclusively
- Uses read/write global services, but is programmed to resolve update conflicts if those services are simultaneously offered by more than one database
- Can tolerate replicated data that is potentially stale due to replication lag

Applications that are not replication-aware can still benefit from the improved high availability and disaster recovery capabilities provided by Global Data Services.

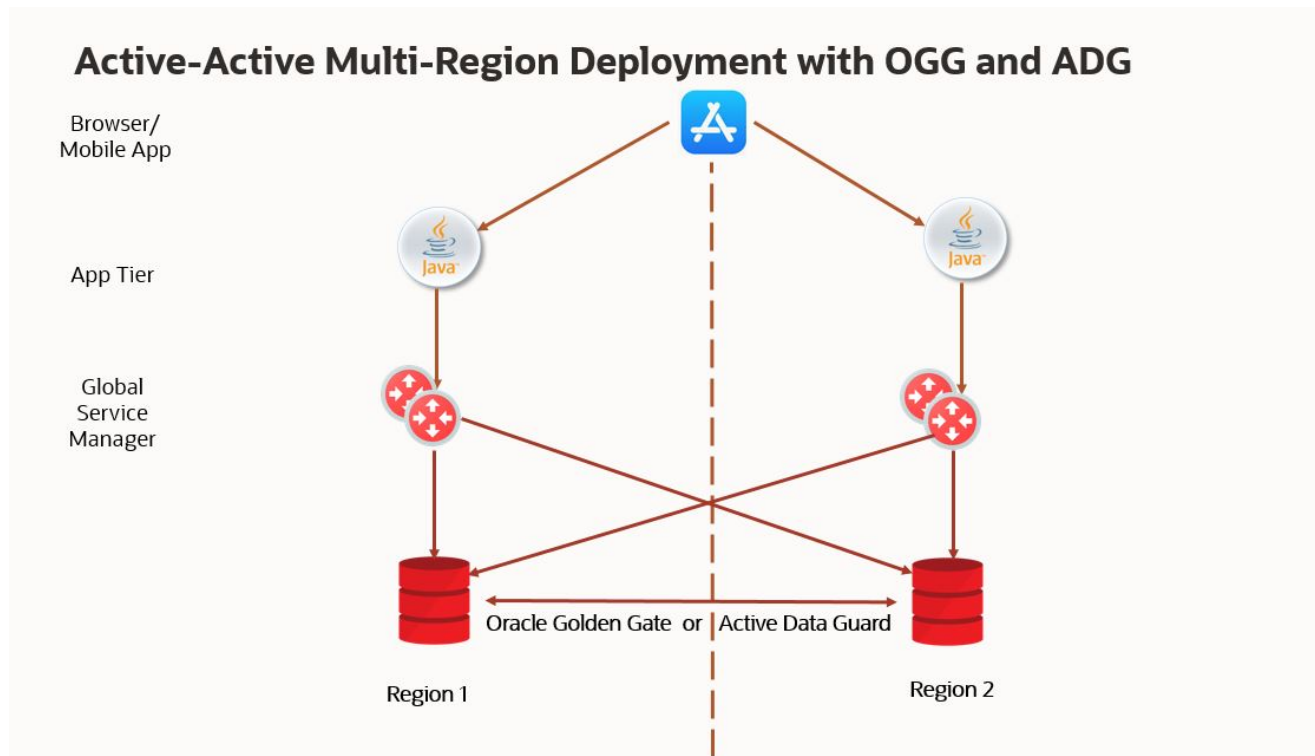
 **See Also:**

- [Global Data Services Control Utility \(GDSCTL\) Command Reference](#), for more information about GDSCTL
- *Oracle Database Administrator's Guide* for more info about database services

Global Data Services Architecture

The following figure shows an example of a Global Data Services (GDS) configuration and the GDS components that are present in every GDS configuration.

Figure 1-1 Global Data Services Components



The sections that follow describe these components.

Global Data Services Pool

A **Global Data Services pool** is a named subset of databases within a GDS configuration that provides a unique set of global services and belongs to the same administrative domain. Partitioning of GDS configuration databases into pools simplifies service management and provides higher security by allowing each pool to be administered by a different administrator.

A database can only belong to a single Global Data Services pool. All databases in a pool need not provide the same set of global services. However, all databases that provide the same global service must belong to the same pool.

Global Data Services Region

A **Global Data Services region** is a named subset of databases in a GDS configuration and database clients that share network proximity such that the network latency between members of a region is typically lower than between members of different regions. A region usually corresponds to a local area or metropolitan area network (LAN or MAN). For example, a data center hosting one or more GDS configuration databases and database clients in geographical proximity to the data center might belong to the same region.

A region can contain multiple Global Data Services pools, and these pools can span multiple regions.

For high availability purposes, each region in a GDS configuration should have a designated **buddy region**, which is a region that contains **global service managers** that can provide

continued access to a GDS configuration if the global services managers in the local region become unavailable.

Global Service Manager

A **global service manager** is the central software component of Global Data Services, providing service-level load balancing, failover, and centralized management of services in the GDS configuration. Global Data Service clients use a global service manager to perform all GDS configuration operations.

A global service manager is analogous to the remote listener in an Oracle RAC database, except that a global service manager serves multiple databases. A global service manager does the following:

- Acts as a regional listener that clients use to connect to global services
- Provides connect-time load balancing for clients
- Manages global services across the regions of a GDS configuration
- Collects performance metrics from databases in the GDS configuration and measures network latency between regions of a configuration
- Creates a run-time load balancing advisory and publishes it to client connection pools
- Monitors availability of database instances and global services and notifies clients if they fail.

A global service manager is associated with one and only one GDS configuration. Each region in the GDS configuration must have at least one global service manager. It is recommended that multiple global service managers be configured in each region to improve availability and performance. Every global service manager in a GDS configuration manages all global services supported by the configuration.

Global Data Services Catalog

A [Global Data Services catalog](#) is a repository that stores configuration data for a Global Data Services configuration and all global services provided by that configuration.

A catalog is associated with one and only one GDS configuration. A catalog must reside in an Oracle Database 12c or later database, and that database may be inside or outside the associated GDS configuration. For large-scale GDS configurations, it is recommended that the GDS catalog be hosted outside the databases in the GDS configuration. GDS catalog may be co-hosted along with catalogs of RMAN or Oracle Enterprise Manager.

Oracle strongly recommends that high availability technologies such as Oracle RAC, Oracle Data Guard, and Oracle Clusterware be used to enhance the availability of the database where the Global Data Services catalog resides.

Oracle Notification Service Servers

GDS clients use **Oracle Notification Service (ONS)** to receive run-time load balancing advisory and high availability events from global service managers.

An Oracle Notification Service (ONS) server is co-located with each global service manager. All such ONS servers within a region are interconnected. Clients of global services subscribe to the ONS server network within their region and its buddy region, and receive FAN notifications from those ONS server networks.

 **Note:**

An Oracle RAC database in a Global Data Service configuration may also contain ONS servers running on the cluster nodes. These ONS servers generate FAN notifications related to local services and are not connected to ONS server networks in the GDS regions.

Overview of Global Services

For database clients, a Global Data Services configuration is represented by a set of global services. A global service manager serving a Global Data Services configuration is aware of all global services that the configuration provides and acts as a mediator for the database clients and databases in the configuration. A client program connects to a regional global service manager and requests a connection to a global service. The client does not need to specify which database or instance it requires. The global service manager forwards the client's request to the optimal instance in the configuration that offers the global service. Database clients that share a global service must have the same service level requirements.

The functionality of *local* services, defined as traditional database services provided by a single database, is not changed by global services. Oracle Database 12c and later can provide local and global services, simultaneously. A client application can also work with global and local services, simultaneously.

 **Note:**

Database versions earlier than Oracle Database 12c can provide local services, but only Oracle Database 12c, and later, can provide global services.

The configuration and run-time status of global services are stored in the Global Data Services catalog. Each database that offers global services also maintains information about those services in a local repository (such as a system dictionary or Oracle Cluster Registry), with data on local services. Global services that are stored in a local repository have a flag to distinguish those global services from traditional local services.

If you are locally connected to a particular database, then you can query data on global services provided by that database. You can configure, modify, start, or stop a global service using the Global Data Services Control utility (GDSCTL) when you are connected to the Global Data Services catalog. This ensures centralized coordinated management of global services. You cannot configure, modify, start, or stop a global service using either the Server Control utility (SRVCTL) or the Oracle Clusterware Control utility (CRSCTL).

 **Note:**

Under certain circumstances (such as patching a database or clusterware software), you can stop or start global services using SRVCTL with the `-force` option with the appropriate command. You must have the appropriate system privileges.

After you configure global services for a Global Data Services configuration, the global service manager manages the global services on GDS configuration databases according to service properties that you specify when you create the services.

When a database joins the configuration or restarts after a shutdown, the database registers with all global service managers in the configuration. After receiving the registration request, one of the global service managers queries the GDS catalog and checks whether all global services which this database is supposed to provide are created there and have the correct attributes. If there is a discrepancy between the information in the catalog and database, the global service manager may create, delete, or modify some global services in the database, or change their attributes to synchronize them with the information in catalog. Then the global service manager determines which global services need to be running on the database and starts them if necessary.

The global service manager can start or stop a global service in a database. However, if it is an Oracle RAC database, the global service manager does not control which particular instances within the database offer the service. This is controlled by the clusterware and the administrator of the Oracle RAC database.

When a database instance in a Global Data Services configuration fails, all global service managers in the configuration get notified about the failure and stop forwarding requests to the instance. If this instance belongs to a noncluster database, or the instance is the last instance that was available in an Oracle RAC database, then, depending on the configuration, a global service manager can automatically start the service on another database in the Global Data Services pool where the service is enabled. If you decide to manually move a global service from one database to another using the appropriate GDSCTL command, then the global service manager stops and starts the service on the corresponding databases.

Global Service Attributes

Global services have a set of attributes associated with them that control starting the global services, load balancing connections to the global services, failing over those connections, and more. Attributes applicable to local services, including those services specific to Oracle RAC and Oracle Data Guard broker environments, are also applicable to global services.

The following attributes are unique to global services:

- Preferred or available databases
- Replication lag
- Region affinity

You can modify global services just as you can modify local services. You can enable and disable a global service, you can move the global service to a different database, and you can change the properties of the global service.

**Note:**

You cannot upgrade a local service to a global service.

Global Services in an Oracle RAC Database

Some properties of a global service are only applicable to Oracle RAC databases and are unique for each Oracle RAC database included in a Global Data Services configuration. These properties are related to the placement of global services with instances within an Oracle RAC database, including server pools and service cardinality.

All other existing global service attributes, such as load balancing, role, transparent application failover parameters, and database edition must be the same for all databases offering a global service. Specify these attributes at the global service level.

**Note:**

By default, in an Oracle RAC environment, a SQL statement executed in parallel can run across all of the nodes in the cluster. The cross-node parallel execution is not intended to be used with GDS load balancing. For an Oracle RAC database in a GDS configuration Oracle recommends that you restrict the scope of the parallel execution to an Oracle RAC node by setting the `PARALLEL_FORCE_LOCAL` initialization parameter to `TRUE`.

Global Services in an Oracle Data Guard Broker Configuration

Oracle Data Guard broker permits a primary database to have up to 30 standby databases. Oracle Data Guard broker logically groups primary and standby databases into a *broker configuration* that enables the broker to manage and monitor the databases as an integrated unit. When you include a broker configuration in a GDS configuration, you manage the broker configuration as one unit. Only an entire broker configuration can be included in (or deleted from) a GDS pool, and a broker configuration cannot span multiple pools.

If you attempt to add or remove a database that belongs to a broker configuration to or from a GDS pool, then an error occurs. You can only add a database to the GDS pool by adding the database to the broker configuration using the Oracle Data Guard command-line interface, DGMGRL. After you add a database to the broker configuration, you must run the `gdsctl sync brokerconfig` command to synchronize GDS and Oracle Data Guard.

A GDS pool can consist of a set of databases of a given Data Guard broker configuration. Databases that belong to different Data Guard broker configurations must be mapped to different GDS pools. A pool that contains a Data Guard configuration cannot have databases that do not belong to the configuration

 **See Also:**

[sync brokerconfig \(synchronize brokerconfig\)](#) for information about this command

Conversely, when you remove a database from a broker configuration, the database is removed from the GDS pool to which this broker configuration belongs. The `gdsctl sync brokerconfig` command must be executed after removing a database. This is the only way to remove a database from a pool that contains a broker configuration.

You can configure global services with a role attribute to be active in a specific database role, such as the role of primary or physical standby database. If you enable fast-start failover, then the Oracle Data Guard broker automatically fails over to a standby database if the primary database fails. The global service managers that you have configured to work with Oracle Data Guard broker ensure that the appropriate database services are active and that the appropriate Fast Application Notification (FAN) events are published after a role change. These FAN events enable Fast Connection Failover (FCF) of client connections to an appropriate database instance within the GDS configuration.

The Global Data Services framework supports the following Oracle Data Guard broker configurations:

- The set of databases in a Global Data Services pool can be either the set of databases that belong to a single broker configuration or a set of databases that do not belong to a broker configuration. You can add a broker configuration only to an empty Global Data Services pool. If a GDS pool already contains a broker configuration, then, to add a database to the pool, you must add the database to the broker configuration contained in the pool.
- Role-based global services are supported only for database pools that contain a broker configuration.

 **See Also:**

Oracle Data Guard Broker for more information about broker configuration

Database Placement of a Global Service

You can specify which databases support a global service. These databases are referred to as **preferred databases**. The global service manager ensures that a global service runs on all preferred databases for which the service has been specified. The number of preferred databases is referred to as the **database cardinality** of a global service. You must specify at least one preferred database for a global service.

 **Note:**

If you set `preferred_all` for which databases support a service, then you do not have to explicitly specify preferred or available databases. The `preferred_all` setting implies that all databases in the pool are preferred.

When you add or modify a global service, you can specify a list of available databases for this global service. If one of the preferred databases fails to provide a global service, then the global service manager relocates the service to an available database to maintain the specified database cardinality for that service.

In a Global Data Services pool that contains an Oracle Data Guard broker configuration, a role-based global service can be started on a database only if the database is listed as preferred or available for the service and the `role` attribute of the database corresponds to the `role` attribute specified for the service. For example, a global service that can run on any database in a broker configuration (as long as the `role` attribute of the database is set to `primary`) must have `primary` specified for its `role` attribute and have all other databases in the broker configuration with `role` attributes set to `preferred`.

 **Note:**

Do not confuse *database cardinality* of global services with their *instance cardinality*. Instance cardinality is specified and maintained separately for each Oracle RAC database and is not maintained across databases of a Global Data Services pool.

For example, consider a case when there are few instances of an Oracle RAC database offering a global service and one of the instances fails, but there are no available instances on which to start the global service. In this case, the service is not started on an available database instance of another Oracle RAC database. However, if the instance that failed was the last instance in the database offering the service, then the service may start on another database that is listed as available for this service.

Replication Lag and Global Services

For performance reasons, distributed database systems often use asynchronous replication of data between databases, which means that there is the possibility of a delay between the time an update is made to data on a primary database and the time this update appears on a replica. When this happens, the replica database lags behind its primary database.

Global Data Services enables applications to differentiate between global services that provide real-time data from services that can return out-of-date data because of replication lag. For applications that can tolerate a certain degree of lag, you can configure a maximum acceptable lag value.

For applications that cannot tolerate replication lag, you can set the lag time for global services to zero. Requests for this global service are forwarded only to a primary database, or to a replica database that is synchronized with the primary database.



See Also:

[Administering Global Data Services Configurations](#) for more information



Note:

Only an Oracle Data Guard standby database can be synchronized with the primary database, if necessary, so that you can enable real-time, read-only services on both the primary and synchronized standby databases.

For many applications it is acceptable to read stale data as long as the data is consistent. Such applications can use global services running on any database replica without regard to the length of the replication lag time.

If you configure the lag time for a service to a value other than zero, then a client request can be forwarded only to a replica database that is not lagging behind the primary database by longer than the configured lag time for the service. If there is no such database, then the connection request fails.



Note:

Specification of the maximum replication lag is only supported for Active Data Guard configurations.

If a service that is currently running starts to exceed the specified maximum lag, then the service is brought down after all current requests are completed. New requests for the service are routed to a database that meets the configured lag value for the service.



Note:

If you use Oracle Data Guard with the Oracle Active Data Guard option, then you can use the `STANDBY_MAX_DATA_DELAY` session parameter to specify a session-specific lag tolerance. When you set this parameter to a nonzero value, a query issued to a physical standby database is executed only if the application lag time is less than or equal to the `STANDBY_MAX_DATA_DELAY` parameter value. Otherwise, the database returns an error to alert the client that the lag time is too long. If both session-level and service-level parameters are set and the `STANDBY_MAX_DATA_DELAY` parameter value is less than the service level parameter value, the client remains connected to the database, but queries would return an error.

 **Caution:**

If a database fails, then the global service manager can route a client connection to another database that meets the maximum lag value specified for the service, even if it lags behind the failed database. This solution, however, can create data consistency problems for some applications.

 **See Also:**

Oracle Data Guard Concepts and Administration for more information about the `STANDBY_MAX_DATA_DELAY` session parameter

Global Services and Distributed Transaction Processing

When performing XA (distributed) transactions against a GDS configuration, tightly-coupled branches of an XA transaction must be directed to the same database. This can be accomplished by ensuring that each global service used in a distributed transaction is, that is runs exactly on one database at a time. It is responsibility of the GDS pool administrator to configure and maintain the property of global services used in distributed transactions.

If a global service used in an XA transaction is running on a multi-instance Oracle RAC database, tightly-coupled branches of the XA transaction by default can be sent to separate database instances. This may result in sub-optimal performance. To avoid the performance degradation in this case, it is highly recommended to set the DTP parameter (`-dtp`) to `TRUE` for any service used by an XA transaction that can run in an Oracle RAC database. Setting the parameter guarantees that all distributed transactions performed through the service have their tightly-coupled branches running on a single Oracle RAC instance.

 **Note:**

The DTP parameter of a global service only affects processing of XA transactions on an Oracle RAC database that provides this service. It has no effect on the database cardinality of the global service. A global service used by a distributed transaction must be manually configured to be active on a single database at a time.

Global Services in Multitenant Architecture

Multitenant container databases (CDB) can be part of the GDS configuration. A global service can be defined at root level of the CDB (CDB\$ROOT) or at the pluggable database (PDB) level. A global service that runs on a PDB can be created by setting the PDB property to the PDB name at creation time. The PDB property of a global service cannot be modified once the service is created. If the PDB property is not set, the service is created at the CDB root level (CDB\$ROOT).

When the service is created at the PDB-level all of the service's preferred and available databases should contain the PDB it belongs to. Global services defined at the PDB level are managed by the same operations and rules as the services defined at any other level.

Global Connection Load Balancing

When a client connects to an Oracle RAC database using a service, the client can use the Oracle Net connection load balancing feature to spread user connections across all the instances that support that service. Similarly, in a Global Data Services configuration, clients connecting to a global service are load balanced, as necessary, across different databases and regions.

Client-Side Load Balancing

Client-side load balancing balances the connection requests across global service managers. Connection failover is also supported. With connection failover, if an error is returned from the chosen global service manager, then Oracle Net Services tries the next address in the address list until either a successful connection is made or Oracle Net Services has exhausted all the addresses in the list.

Client-side load balancing and failover in a Global Data Services configuration is similar to that for an Oracle RAC database. In a Global Data Services configuration, however, a client in a Global Data Services region first tries to connect to any of the global service managers in its local region. If a global service manager from the local region does not respond, then the client tries a global service manager in another region.

To enable client-side load balancing and failover across multiple regions in a Global Data Services configuration, clients must use an Oracle Net connect descriptor that contains one list of addresses of local regional global service managers for load balancing and intraregion failover, and one (or more) list of addresses of remote regional global service managers for interregion failover. If a region is not specified, it defaults to the region name of the global service manager to which the client is connected. You can also configure timeout and retry attempts for each list to enable multiple connection attempts to the current global service manager before moving to another global service manager in the list.



See Also:

[Database Client Configuration](#) for more information

Server-Side Load Balancing

Server-side load balancing for an Oracle RAC database has the listener directing connection requests to the best Oracle RAC database instance. Some applications have long-lived connections, while other applications have short-lived connections.

For global services, server-side load balancing works similarly, except that, instead of being limited to a single database, workloads are balanced across multiple databases in the Global Data Services configuration. In most cases, a global service manager directs a client request for a global service to a database server in the same region,

unless all local servers are overloaded and a remote server can provide significantly better quality of service.

In some cases, to take advantage of data caching on a local server, you can direct requests to the local region. Global Data Services enables you to specify a desired level of client/server affinity for a global service.

Region Affinity for Global Services

You can configure global services to operate within specific regions or in any region in the Global Data Services configuration. This is called **region affinity**.

Any-Region Affinity

Any-region affinity (the default) for a global service routes a client connection request to the best database in the Global Data Services configuration, regardless of region, that can meet the connection load balancing goal for the requested service. The choice of the database is based on its performance and network latency between the regions where the client and database reside. If databases in different regions are equally loaded, then this policy gives preference to a local region. An interregion connection is made only if the difference in database performance between regions outweighs network latency.

If you specify preferred, available databases for a global service with any-region affinity, then service cardinality is maintained at the Global Data Services pool level. If a service fails on a preferred database, then the service is started on any available database in the Global Data Services pool, and the number of service providers in the pool remains the same. When starting the service on an available database, databases in the region where the service failed have preference. If there is no available database for this service in the local region, then an available database is chosen from a nearby region.

Affinity to a Local Region

Affinity to a local Global Data Services region for a global service routes a client connection request to the best database in the client's region that can meet the connection load balancing goal for the requested service. The global service manager chooses the database based only on the database's performance. A global service with affinity to a local region can be provided in multiple Global Data Services regions at the same time, but a client from one region is never connected to a database in another region.

If you specify preferred or available databases for a global service with local region affinity, then service cardinality is maintained at the regional level. If a service fails on a preferred database, then the service starts on an available database in the same region, so the number of service providers in the region remains the same. If there is no available database for this global service in the local region, then no action is taken and the service cardinality decreases. If there is no database in the local region offering the global service, then the client connection request is denied. For global services that have affinity to a local region, database cardinality of the service is maintained independently in each region.

Affinity to a Local Region with Interregion Failover

This type of affinity is similar to that of affinity to a local Global Data Services region, except that, if there are no databases in the local region offering a global service, then, instead of denying a client request, the request is forwarded to the best database in another region where the requested global service is running. This service failover does not trigger the service to be started on an available database in another region because, with affinity to a

local region, database cardinality is maintained independently in each region, and should not change because of service failure in another region. If regional databases become overloaded because of interregion failover, then you can manually add a preferred database for the service.

Global Run-Time Connection Load Balancing

Global run-time connection load balancing distributes client work requests across persistent connections that span the instances of an Oracle RAC database, based on load-balancing information from the database. This feature is supported with connection pools that can receive load balancing recommendations. The database uses the run-time connection load balancing goal for a service and the relative performance of the database instances to generate a recommendation about where to direct service requests.

There are two types of service-level goals for run-time connection load balancing:

- **SERVICE_TIME**: Attempts to direct work requests to instances according to response time. Load-balancing data is based on elapsed time for work done in the service plus available bandwidth to the service.
- **THROUGHPUT**: Attempts to direct work requests according to throughput. The load-balancing data is based on the rate that work is completed in the service plus available bandwidth to the service.

The Global Data Services framework also supports the balancing of work requests at run time to a global service. In the Global Data Services framework the requests are spread across connections to instances in multiple databases. Work is routed to provide the best service times globally and routing responds gracefully to changing system conditions.

To provide global run-time connection load balancing, a global service manager receives performance data for each service from all database instances in the Global Data Services configuration. The global service manager also measures interregion network latency by periodically exchanging messages with global service managers in other Global Data Services regions.

If the load-balancing goal for a global service is set to `SERVICE_TIME`, then a global service manager considers interregion network latency and instance performance data when deciding how to distribute work requests. For example, clients in Region A have run-time load balancing metrics that are weighted toward Region A, and clients in Region B have metrics that are weighted toward Region B. This implies that, even though the service may be the same, clients in different regions receive different run-time load balancing metrics.

If the load-balancing goal for a global service is set to `THROUGHPUT`, then run-time load balancing metrics are calculated only based on the performance of database instances.

In addition to calculating run-time load balancing metrics for local clients, a global service manager may also need to calculate run-time load balancing metrics of remote regions and publish the metrics for clients residing in a region where all global service managers are not available.

Affinity

The global service manager provides advice about how to direct incoming work to the databases in a Global Data Services configuration that provide the optimal quality of service for that work. The load balancing advisory also sends additional affinity information that indicates to the clients, subscribers to the Oracle Net Service events (as Universal Connection Pool (UCP) and ODP.NET Connection Pool, and so on), whether it should re-connect to the same database. Reconnecting a session to the same database can improve buffer cache efficiency and lower CPU usage and transaction latency. The affinity information indicates whether affinity is active or inactive for a particular database. It is always inactive for a single instance database and can be either active or inactive for a particular instance if the database is an Oracle RAC.

Disk I/O and CPU Thresholds

Disk I/O and CPU thresholds are resource consumption limits that a database administrator can set. The disk I/O threshold is configured by setting the single-block read latency limit, and the CPU threshold is set by setting the CPU usage limit. The global service manager monitors the resource usage and checks whether the I/O and CPU thresholds have been reached or not. When a database reaches one of the thresholds, the load balancing advisory readjusts its advice about how to direct incoming work to this database.

Global Services Failover

When a global service or database fails, a global service that was running on the database fails over to another database where the global service is enabled, but not yet running (if the database role matches the service role). The global service manager considers preferred databases as the failover target before available databases.

If you stop a global service using GDSCTL, then the service does not fail over to another database. However, the database where the service was stopped remains a failover target for this service. If the service fails on another database, then the service can start on that database.

When a global service fails over to an available database, the Global Data Services framework does not move the service back to the preferred database when the preferred database restarts because of the following:

- The service has the desired cardinality
- Maintaining the service on the current instance or database provides a higher level of service availability
- Not moving the service back to the initial preferred instance or database prevents a second outage

If necessary, you can manually relocate the global service back to the preferred database after the service has restarted, without terminating active sessions.

In a Global Data Services pool that contains an Oracle Data Guard broker configuration, the Global Data Services framework supports role-based global services. Valid roles are `PRIMARY`, `PHYSICAL_STANDBY`, `LOGICAL_STANDBY`, and `SNAPSHOT_STANDBY`. The Global Data Services framework automatically starts a global service only when the database role matches the role specified for the service.

If a database switches roles or fails, then the Oracle Data Guard broker notifies the Global Data Services framework about the role change or failure, and the global service manager ensures that the services start according to the new database roles.

 **Note:**

A global service cannot fail over from a database in one Global Data Services region to a database in another region if the locality parameter is set to `LOCAL_ONLY`, and interregion failover is not enabled.

When a global service fails over, fast connection failover, if enabled on Oracle clients, provides rapid failover of the connections to that global service. The Global Data Services framework, similar to Oracle RAC, uses Fast Application Notification (FAN) to notify applications about service outages. Instead of waiting for the application to poll the database and detect problems, clients receive FAN events and react, immediately. Sessions to the failed instance or node are terminated, and new connections are directed to available instances providing the global service.

All global service managers monitor service availability on all databases in a Global Data Services configuration. A global service is started on an available database when a global service manager detects that a global service becomes unavailable due to a failure.

 **Note:**

A global service manager cannot automatically fail over a service if it cannot connect to the Global Data Services catalog.

 **Note:**

When the target database for failover is under Oracle Clusterware control, the global service manager failover request to start a service on this database may lead to a Oracle Clusterware "pull up" event that starts another instance. This will occur when at least one instance is registered with the global service manager listener, allowing the global service manager to note that the database is running, and is therefore a valid target for failover, but that the service is not already running on the database. This can occur when the registered instance is not a primary or valid instance for the service, either because it is an available instance on an administrator-managed Oracle RAC database, or because it is an instance of a policy-managed Oracle RAC database that is not in the same server pool as the service.

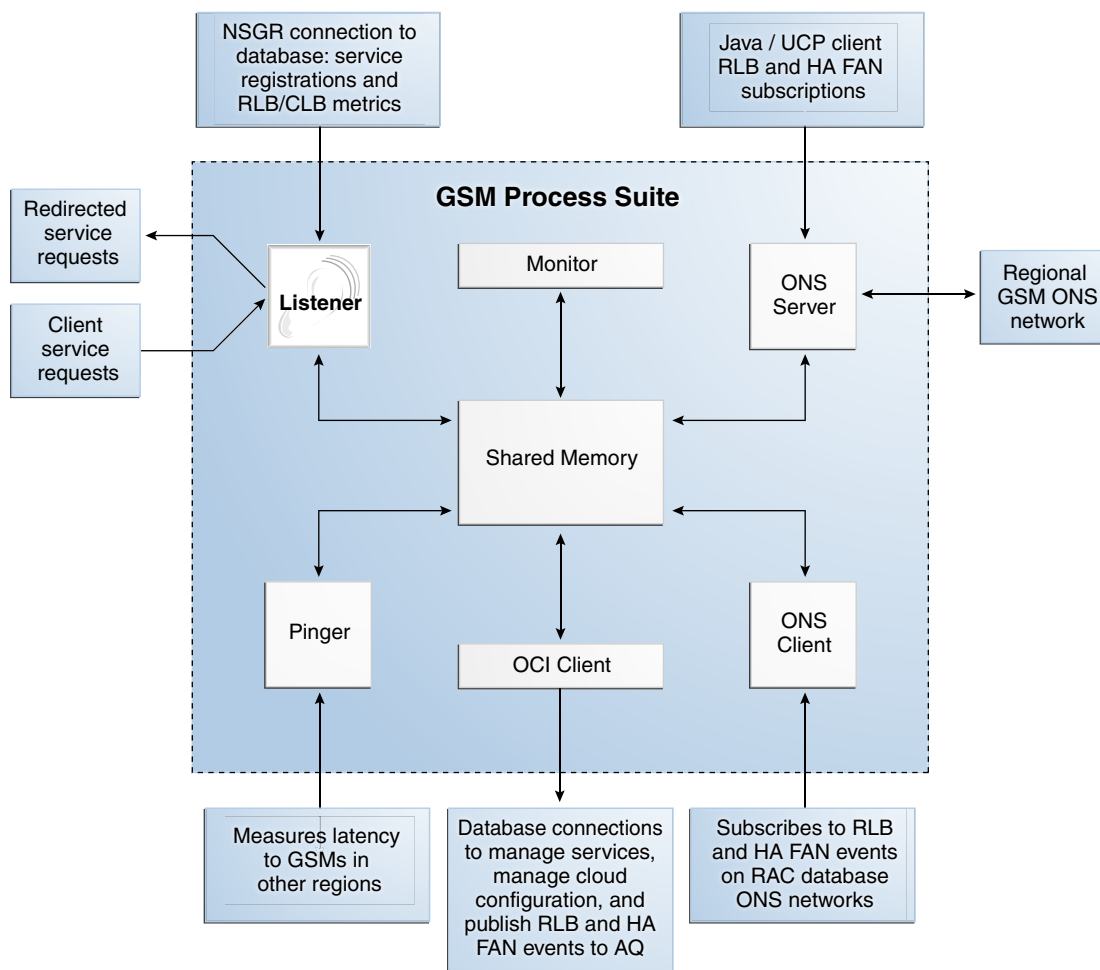
Global Service Manager Process Suite Architecture

The global service manager process suite architecture is a process suite containing multiple processes that communicate using shared memory. Global service manager contains the following processes:

- Monitor - Monitors the health of all other processes in the process suite. Responsible for starting the other global service manager processes, creating the shared memory area and re-spawning a global service manager process after detecting its death.
- Listener - Listens for incoming client connection requests and forwards requests to appropriate database instances. Each instance in Global Data Services configuration establishes a communication channel to all global service manager listeners. This channel is used for registration of global services and sending performance metrics to the listener. These metrics are later used for connection-time and run-time load balancing. Detects when a database instance goes down. The listener process provides the full functionality of the regular database listener. Therefore a global service manager plays the role of a remote listener for all databases in a Global Data Services configuration. A global service manager listener routes connection requests directly to local database listeners, bypassing SCAN listeners on an Oracle RAC database.
- Pinger - Measures network latency between regions by exchanging data with pinger processes of global service managers in the other regions. Measurements are required for connection-time and run-time load balancing in a globally distributed database cloud. Monitors other global service managers and tells its own global service manager when it has to become the regional or master
- OCI Process - Responsible for establishing SQLNET connections to Global Data Services databases for managing services, and to Global Data Services catalog for managing the Global Data Services configuration.
- ONS Server - Publishes RLB and HA fan events to regional ONS network so that clients can subscribe to the events.
- ONS Client - Responsible for subscribing to HA events that are published by Oracle RAC databases in Global Data Services configuration.

Figure 1-2 shows the internal process architecture and the information flow in and out of the process suite.

Figure 1-2 Global Service Manager Process Suite Architecture



Global Data Services Use Cases

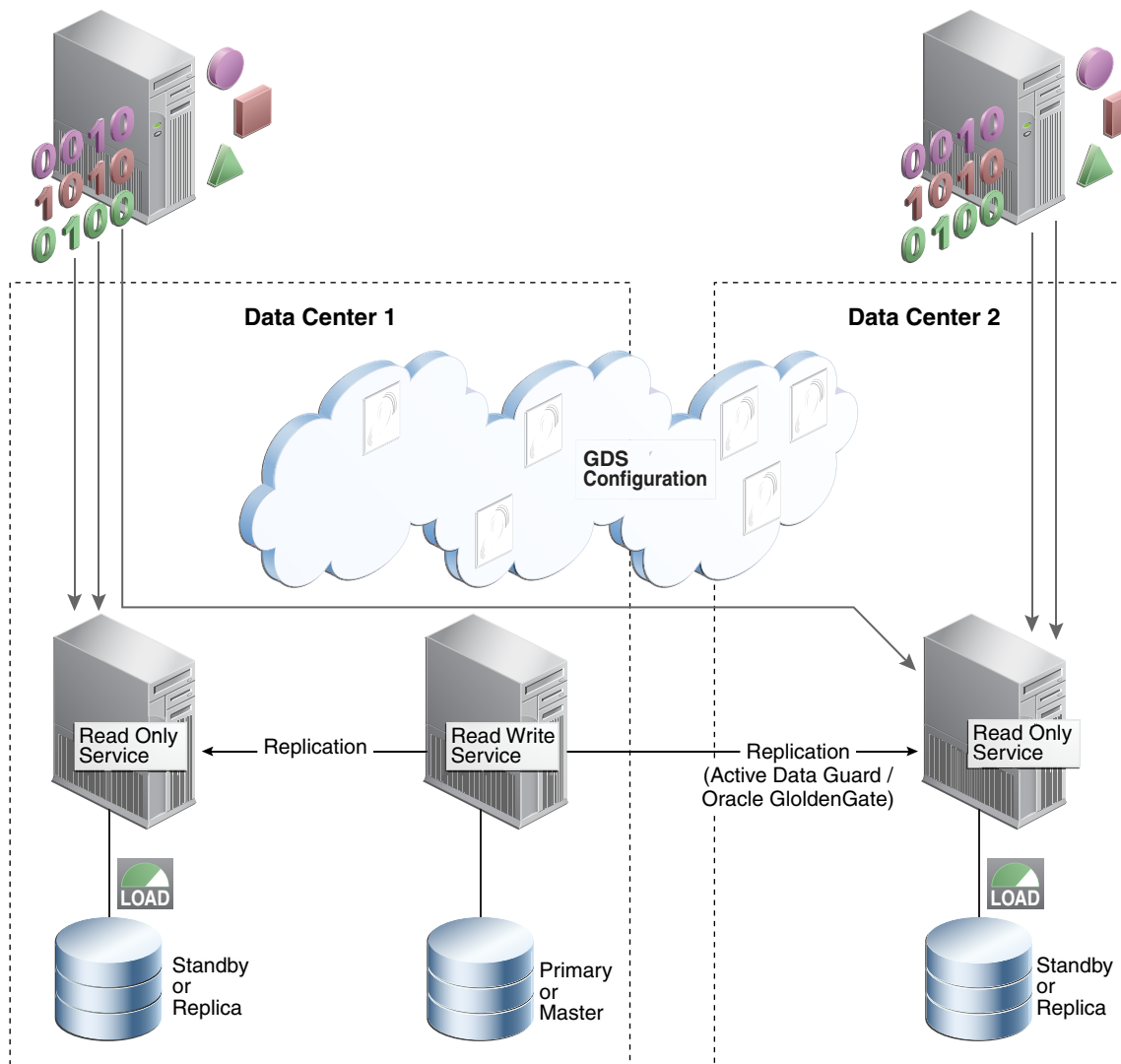
The following are some use cases for Oracle Global Data Services:

Load Balancing for Replicated Databases

Global Data Services extends Oracle RAC-style connect-time and run-time load balancing capabilities (within and across data centers) to a set of replicated databases. The algorithm takes into account load metrics, region affinity, network latency, and load balancing goals. It maximizes performance and achieves efficient resource utilization by enabling Global Data Services on Active Data Guard.

Figure 1-3 shows load balancing across replicated databases, both local and remote, in a Global Data Services configuration. The Read Write Service runs on the Primary (or Master) database. Read Only Services run on the two Standby (or Replica) databases. Client connections are load balanced between the read-only services running on the Standby databases (across the two data centers).

Figure 1-3 Load Balancing for Replicated Databases



Service Failover for Replicated Databases

Global Data Services extends Oracle RAC-style service failover (within and across data centers) and management capabilities to replicated databases, and takes into account service placement policies. It achieves higher availability and improved manageability by enabling Global Data Services with Active Data Guard and Oracle GoldenGate.

Figure 1-4 shows databases replicated across two data centers, in a Global Data Services Configuration. A Read Write Service runs on the Primary (or Master) database. Read Only Services are load balanced across the two Standby (or Replica) databases.

Figure 1-4 depicts the failure of the Standby in Data Center 1. In Figure 1-5 Global Data Services fails over the Read Only Service to another available database (in this case the Primary) and load balances it with the Read Only Service running on the remote Standby in Data Center 2.

Figure 1-4 Read Only Service Failure

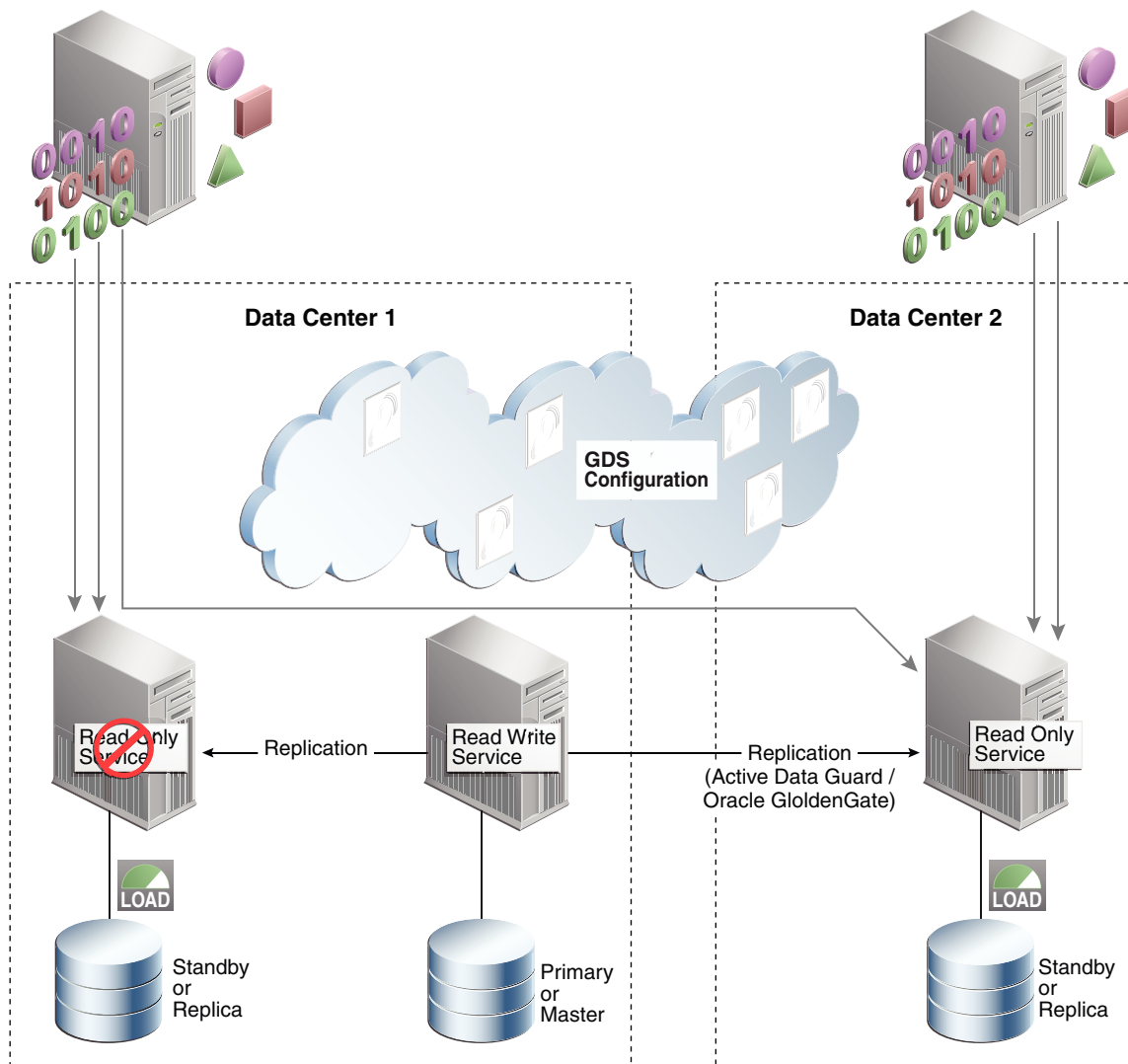
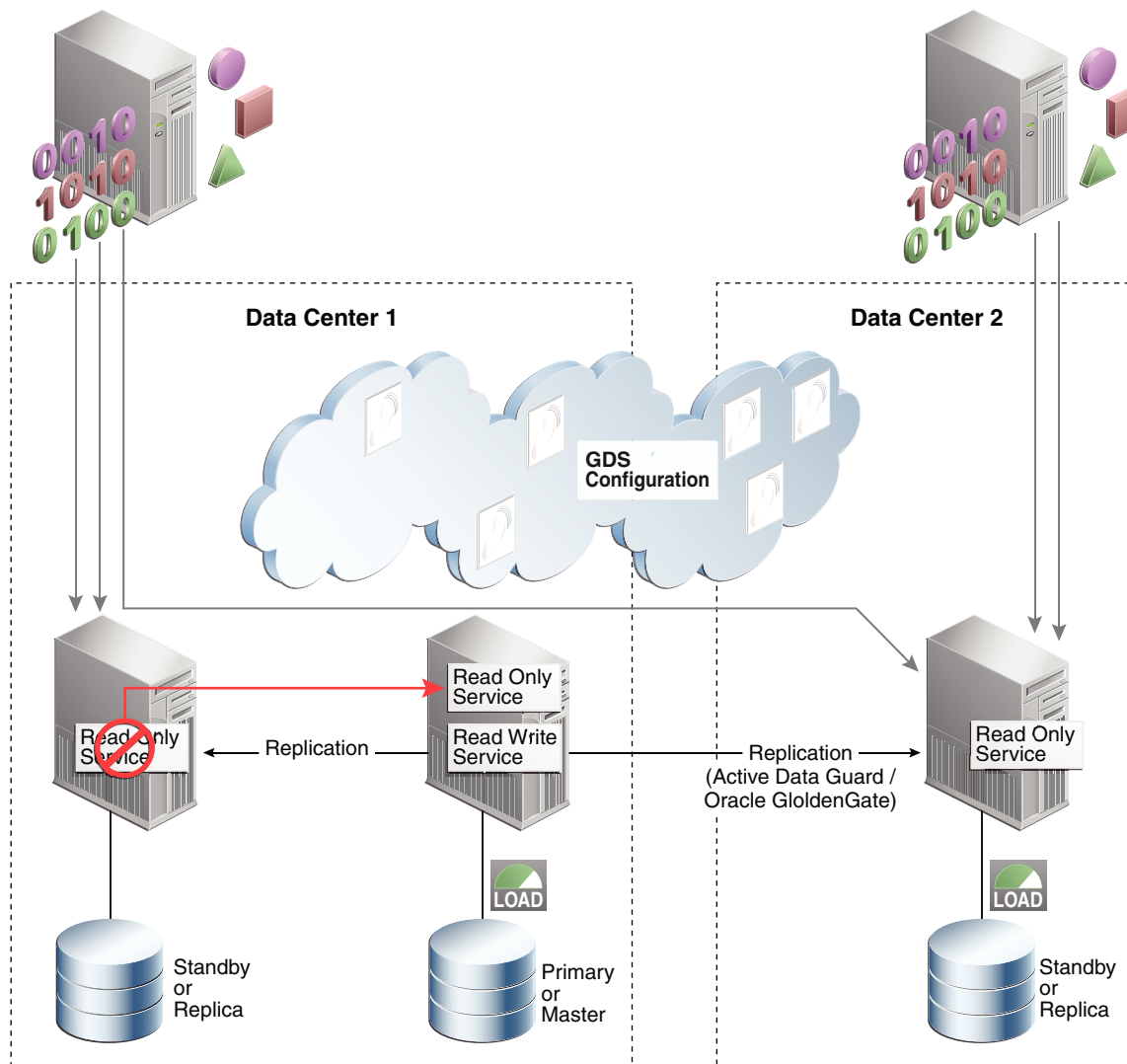


Figure 1-5 Read Only Service Failover



Region Affinity in Oracle GoldenGate Multi-Master

Figure 1-6 depicts Oracle GoldenGate multi-mastered replicas in a Global Data Services configuration. In this use case the application needs to avoid multi-master read/write conflicts, and Global Data Services can route all of the workloads to nearest and best replica in the client's region.

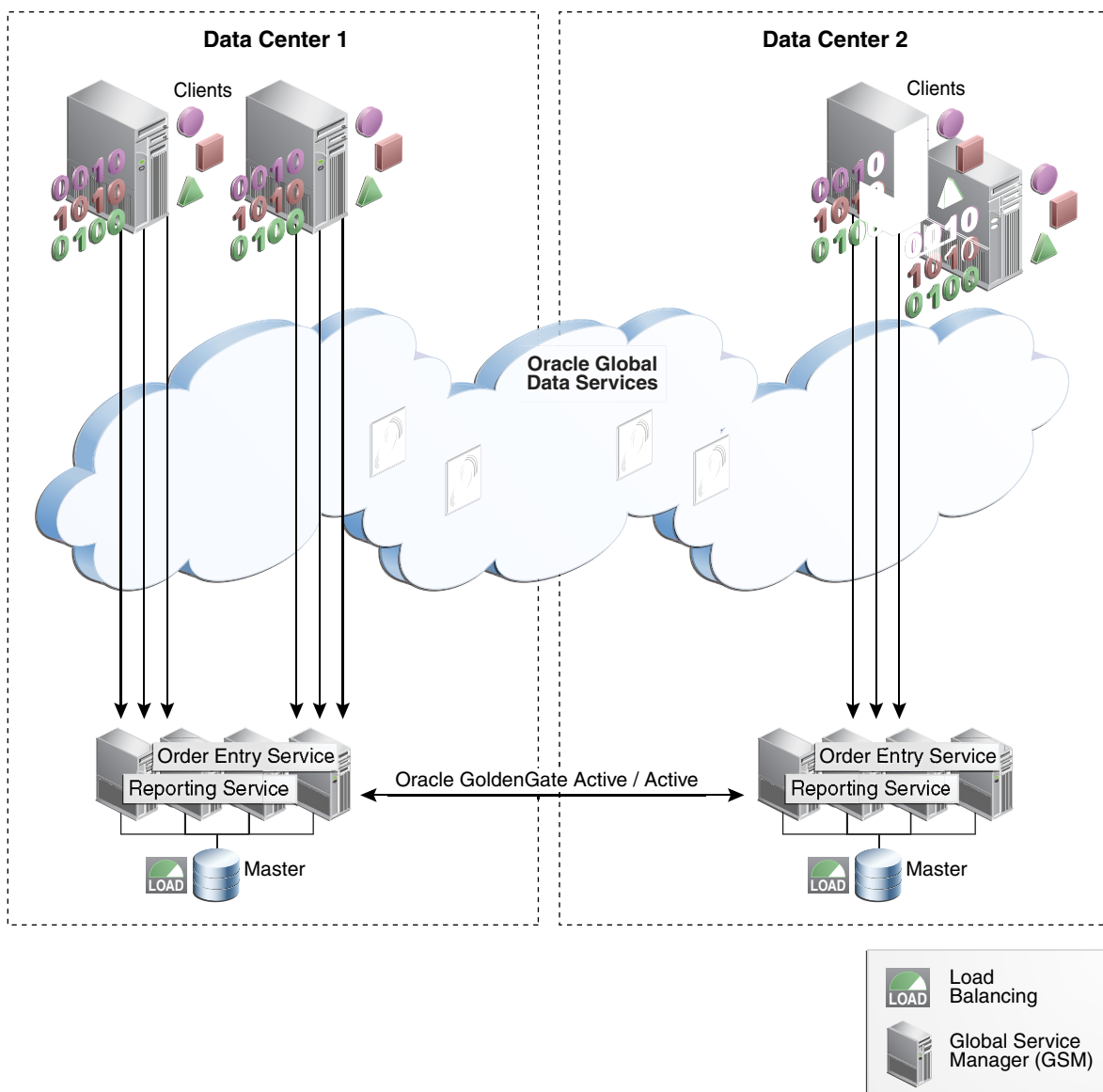
```
GDSCTL>add service -service reporting_srvc -gdspool sales
      -preferred_all -locality LOCAL_ONLY -region_failover
GDSCTL>add service -service order_entry_srvc -gdspool sales
      -preferred_all -locality LOCAL_ONLY -region_failover
```

This use case features the following:

- The application handles multi-master conflict resolutions.

- Read/write and read-only global services run on both replicas.
- Client connections are routed to the nearest and the best replica database in the client's region that meets the connection load balancing goal for the requested Order Entry and Reporting global services.
- Takes into account load metrics, region affinity, and load balancing goals.
- Global Data Services provides workload routing for read-only and read/write services for Oracle GoldenGate multi-master configuration.
- Routes all database workloads to the nearest datacenter by enabling Global Data Services for Oracle GoldenGate multi-master configuration.

Figure 1-6 Region Affinity in Oracle GoldenGate Multi-Master



Load Balancing in Oracle GoldenGate Multi-Master

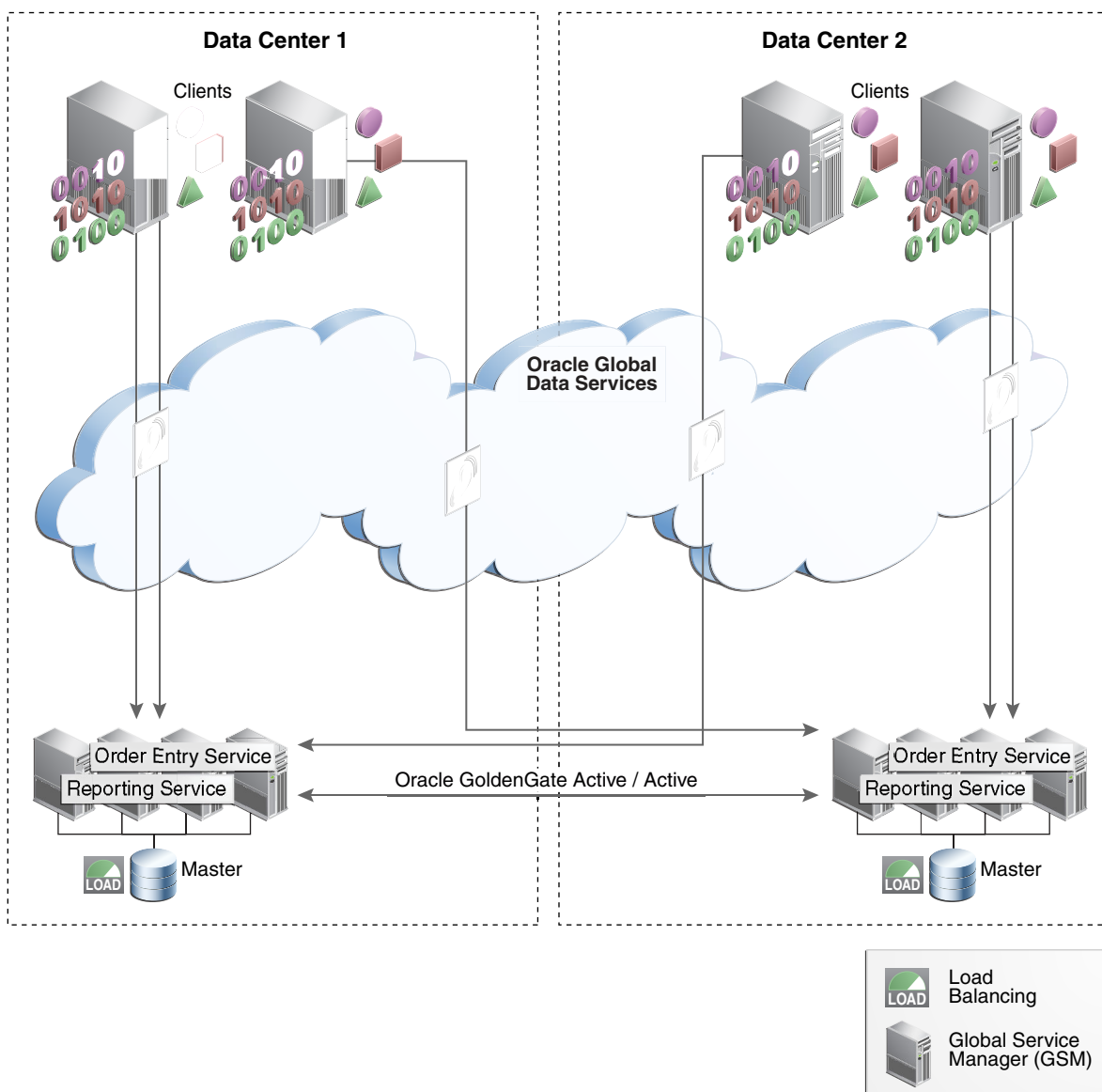
Figure 1-7 depicts Oracle GoldenGate multi-mastered replicas in a Global Data Services configuration. In this use case the application needs to handle multi-master conflict resolutions, and Global Data Services provides connect-time and run-time load balancing (within and across data centers) for all work requests.

```
GDSCTL>add service -service order_entry_srvc -gdspool sales
                -preferred_all -clbgoal LONG
GDSCTL>add service -service reporting_srvc -gdspool sales
                -preferred_all -clbgoal LONG
```

This use case features the following:

- Application handles multi-master conflict resolutions.
- Read Write and Read Only global services run on both replicas.
- Client connections are load balanced between the READ WRITE global services running on both the masters (across data centers). Same is the case for the Read only service.
 - Takes into account load metrics, region affinity, network latency and load balancing goals.
 - Load balance all workloads by enabling GDS on Oracle GoldenGate multi-master configuration.

Figure 1-7 Load Balancing in Oracle GoldenGate Multi-Master



Balancing Oracle Active Data Guard and Oracle GoldenGate Reader Farms

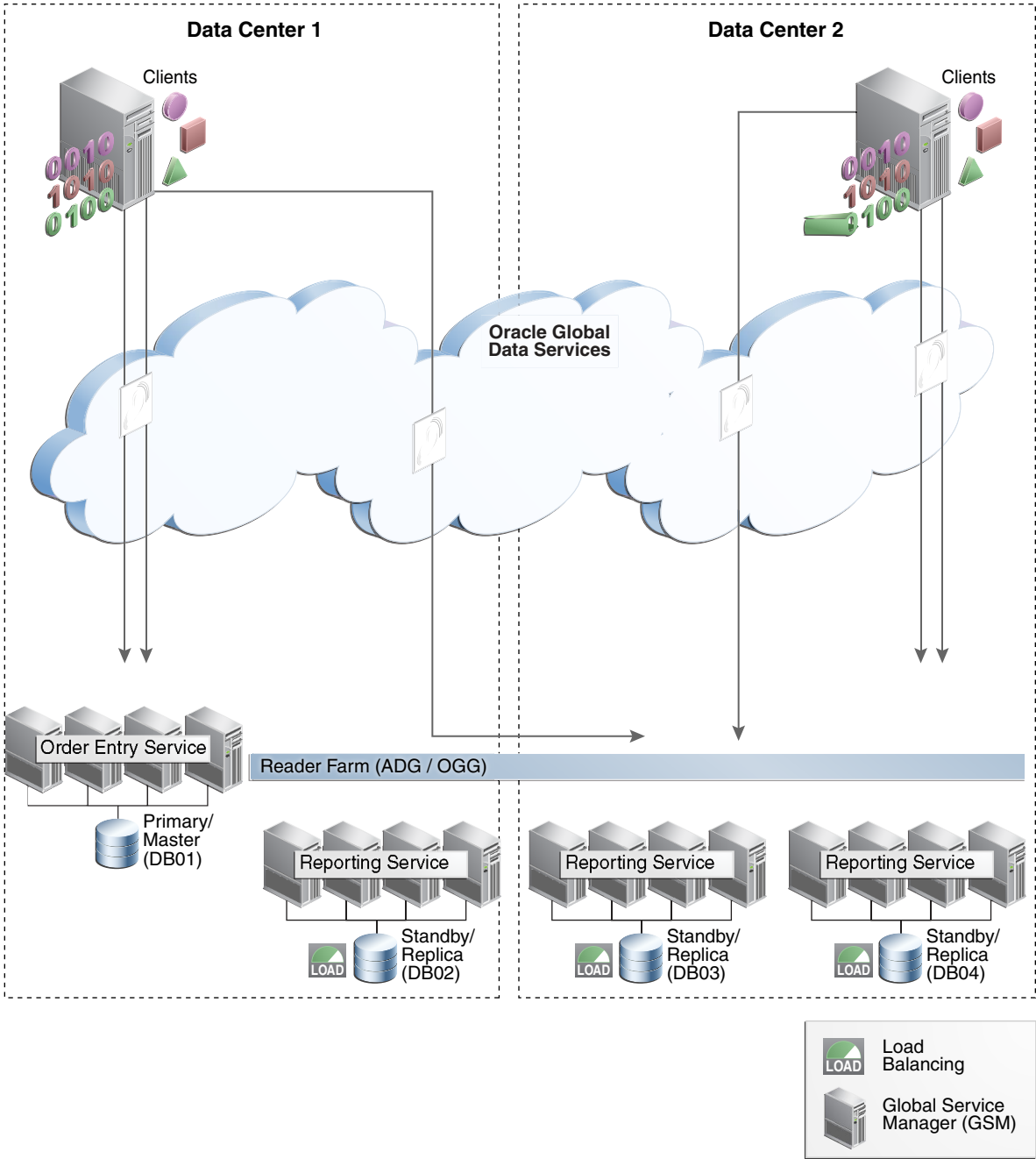
Figure 1-8 depicts Global Data Services enabled for an Oracle Active Data Guard or Oracle GoldenGate reader farm with physical standby/replicas located in both local and remote data centers. In this use case with Global Data Services, route read/write workload to primary/master, balance read-only workload on the reader farm, and you can expect improved resource utilization and higher scalability for read workloads.

This use case features the following:

- Read/write global service runs on the Primary/Master database.
- Read-only global services run on the reader farm.

- Client connections are load balanced between the read-only global services running on the reader farm (across data centers).
- Global Data Services provides connect-time and run-time load balancing (within and across data centers) on a reader farm and better resource utilization and higher scalability by balancing read-only workload on an Oracle Active Data Guard or Oracle GoldenGate reader farm.

Figure 1-8 Balancing Oracle Active Data Guard and Oracle GoldenGate Reader Farms



Example 1-1 Oracle GoldenGate

```
GDSCTL>add service -service reporting_srvc -gdspool sales  
-preferred DB02, DB03, DB04 -clbgoal LONG -rlbgoal SERVICE_TIME
```

2

Configuring the Global Data Services Framework

The Global Data Services framework consists of at least one global service manager, a Global Data Services catalog, and the GDS configuration databases. Some components of the framework are installed when you install Oracle Database. Other components require that you perform certain tasks using the Global Data Services control utility (GDSCTL).

This chapter includes the following topics:

Planning an Installation

Before you install any software, review these hardware, network, operating system, and other software requirements for Linux.

- Each and every GDS pool database must be able to reach (in both directions) each and every global service manager's Listener and ONS ports. The global service manager Listener ports and the ONS ports must also be opened to the Application/Client tier, all the GDS pool databases, the GDS catalog and all other global service managers.
- The TNS Listener port (Default: 1521) of each GDS pool database must be opened (in both directions) to global service managers and the GDS catalog.
- If GDSCTL is run from a separate machine, you also must have a port opened (in both directions) from that machine directly to each GDS pool database that is added.

For detailed information about memory, physical storage, kernel versions and packages required by Global Data Services see: Database Installation Guide for Linux

What You Need to Know About Installing a Global Service Manager

The global service manager is the central component of the Global Data Services framework, and you must install the global service manager using separate media. No other Oracle software is required to install and run the global service manager.

You can install the global service manager on a system where you have other Oracle products installed, but you must install the global service manager in a separate Oracle home directory. You can install more than one global service manager on a single system, but each global service manager must have a separate Oracle home directory. For performance reasons, depending on the number of databases in your Global Data Services configuration, you may want to deploy the global service manager on a dedicated host.

You must install at least one global service manager for each Global Data Services region. Global service managers can be hosted on physical or virtual environments. For high availability, Oracle recommends installing multiple (typically 3) global service managers in each region running on separate hosts.

 **Note:**

Oracle Universal Installer does not currently support installing software on multiple hosts. You must install each global service manager on its respective host.

The Global Data Services administrator installs the global service manager. The Global Data Services administrator's responsibilities include:

- Administering global service managers
- Administering the Global Data Services catalog
- Administering regions and database pools

 **Note:**

The Global Data Services administrator must have an operating system user account on all hosts where global service managers are deployed, and you must run the installation under that user account. The installation must *not* be run by a `root` user.

 **Note:**

When you install the global service manager on Windows platforms, Oracle Universal Installer provides you with the option to use a Windows Built-in Account or to specify a standard Windows user account as the Oracle home user for the Oracle home. This account is used for running the Windows Services for the Oracle home and it cannot be an administrator account.

Note that LocalService account is used for running the services if Windows Built-in Account option is chosen. For administrative tasks, including global service manager installation, upgrade, and patching, you should log on using a Windows account that has administrative privileges.

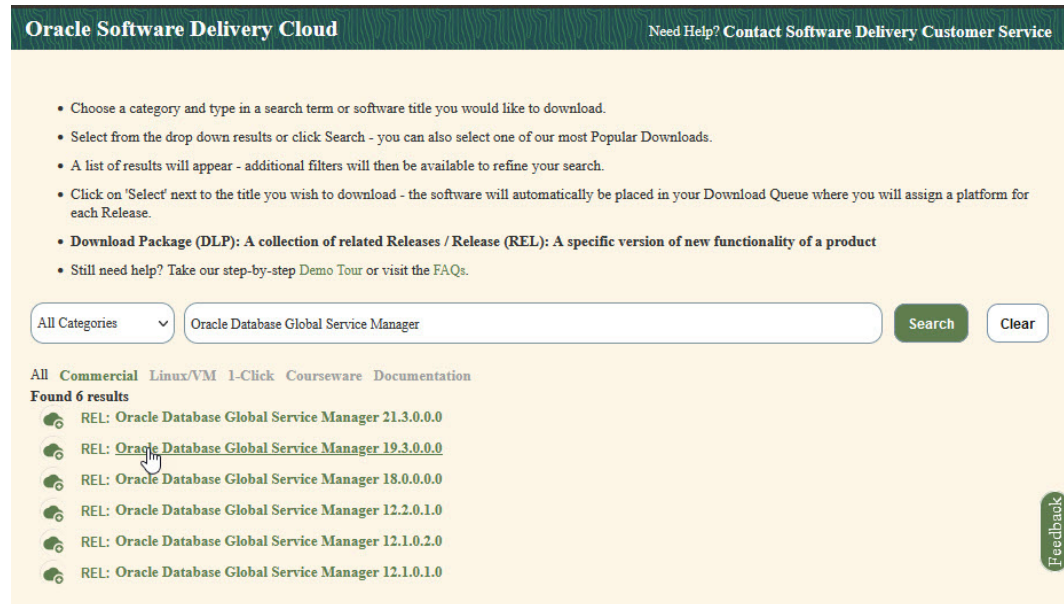
See Chapter 3 of the *Oracle Database Administrator's Reference for Microsoft Windows* for details and recommendations regarding the use of Oracle home user.

Installing a Global Service Manager

To install a global service manager:

1. Download the global service manager software from edelivery.oracle.com and unzip.

Figure 2-1 Oracle Software Delivery Cloud



2. Start Oracle Universal Installer from the root directory of the software media and follow the prompts.

When the installation completes, the global service manager home directory contains binaries required to run the global service manager and the Global Service Manager Control utility (GDSCTL).

3. Set the `ORACLE_HOME` environment variable to the directory you specified during installation.
4. Add the `$ORACLE_HOME/bin` directory created for the global service manager to the `PATH` environment variable.
5. Set the `TNS_ADMIN` environment variable set to `$ORACLE_HOME/network/admin`.

 **Note:**

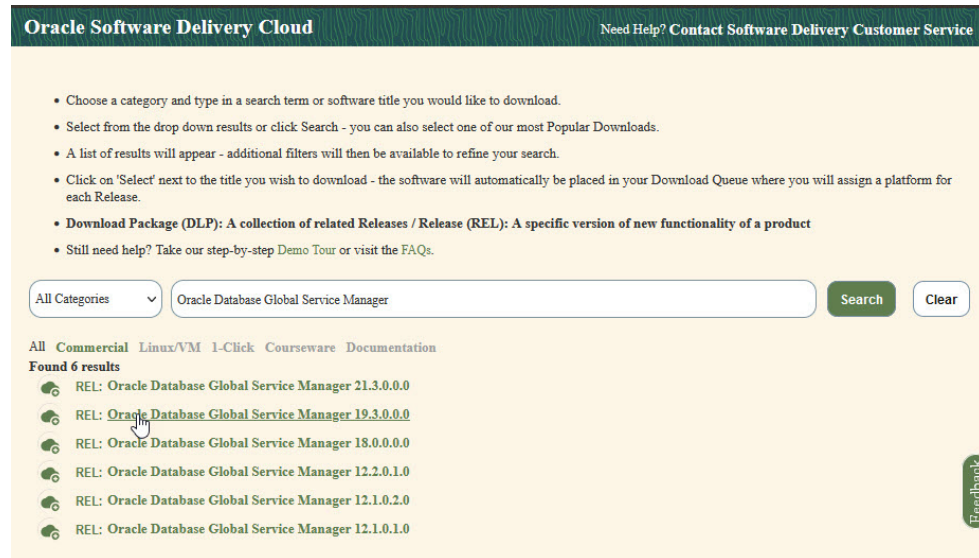
After installing the Global Data Services software, it is recommended that the installation be updated to the latest Oracle Database release. For more information regarding Oracle Database 19c release updates, see Oracle Support (MOS) Note [19202301.9](https://support.oracle.com/knowledge/Database/19202301.9)

Performing a Silent Install of Global Service Manager

You can run the global service manager installation in the command line.

1. Download the global service manager software from edelivery.oracle.com and unzip.

Figure 2-2 Oracle Software Delivery Cloud



2. Edit the `response/gsm_install.rsp` file to suit your environment.

You might want to edit the following variables:

- `ORACLE_HOME`
- `ORACLE_BASE`
- `INVENTORY_LOCATION`
- `UNIX_GROUP_NAME`

3. Run the installer in the silent mode.

```
./runinstaller -silent -responseFile response/gsm_install.rsp -
showProgress
-ignorePrereq
```

4. As a root user, execute the following scripts.

```
/scratch/oraInventory/orainstRoot.sh
/scratch/user/product/19.3.0/gsmhome_1/root.sh
```

What You Need to Know About Upgrading Global Data Services

There are four components that comprise the distributed Global Data Services infrastructure, and each component may reside in a separate installation and may be upgraded independently using the usual upgrade procedure; however, there are certain rules about component versioning that should be followed. The components and rules are as follows:

- **Catalog database:** The catalog database is the central repository for the GDS metadata; it is a standard Oracle Database installation. The version of the catalog database must always be greater than or equal to the version of any GDSCTL

session that connects to it, and exactly equal to the version of any global service manager server that connects to it, with one exception: to ease migration, the catalog may temporarily have a version greater than some global service manager servers that are connected to it, until any change is made to the catalog, at which time any connected global service manager that is not at the same version will be disconnected, and any stopped global service manager that is not at the same version will not be allowed to connect.

 **Note:**

GDSCTL sessions running release 12.1.0.1 cannot make changes to a later versioned catalog; when running commands that will update the catalog, the GDSCTL client should be at a minimum version of 12.1.0.2.

- **GDSCTL client:** This component is run in an ad-hoc manner from a terminal session on any system that contains a global service manager installation. The version of the GDSCTL client is the version of the global service manager installation that it runs from.

 **Note:**

When connecting a 12.1.0.1 GDSCTL client to a later versioned catalog only a limited set of commands are allowed, and any command that may cause catalog changes will result in a compatibility error. Commands that update the catalog metadata in a catalog at version 12.1.0.2 or later should be executed from a GDSCTL client running at least release 12.1.0.2.

- **Global service manager server:** The version of the global service manager server is the version of the global service manager installation from which the server runs. A global service manager server cannot connect to any catalog database that is at a lower version than itself. A global service manager server cannot connect to any catalog database that is at a higher version than itself in which changes have already been made to the catalog at that higher version. A global service manager can connect to a pool database running any version of Oracle Database 12c or later.
- **Pool database:** A pool database is any database added to a GDS pool which runs a global service. A pool database may be at any version of Oracle 12c or later, including versions later than the catalog version. You may upgrade or downgrade pool databases at any time.

Given these rules, it is possible to perform a rolling upgrade of the distributed GDS infrastructure with zero service downtime.

Upgrading Global Data Services

The advised order of upgrade is:

1. Upgrade the catalog database. For best results this should be done using a rolling database upgrade; however, global services will remain available during the upgrade if the catalog is unavailable, although service failover will not occur.

 **Note:**

When upgrading Oracle Database 19c with a patch release, you must execute the following command after upgrading the catalog database:

```
modify catalog -newkeys
```

This command generates encryption keys and encrypt existing GSMUSR passwords stored in the GDS catalog.

2. Upgrade global service manager installations that are used to run GDSCTL clients, which do not also run a global service manager server (if any).

 **Note:**

Global service manager upgrades should be done in-place; however, an in-place upgrade will cause erroneous error messages unless permissions on the following files for the following platforms are updated to 755:

On Linux/Solaris64/Solaris Sparc64:

```
$ORACLE_HOME/QOpatch/qopiprep.bat
```

```
$ORACLE_HOME/jdk/bin/jcontrol
```

```
$ORACLE_HOME/jdk/jre/bin/jcontrol
```

On AIX:

```
$ORACLE_HOME/QOpatch/qopiprep.bat
```

```
$ORACLE_HOME/jdk/jre/bin/classic/libjvm.a
```

```
$ORACLE_HOME/jdk/bin/policytool
```

On HPI:


```
$ORACLE_HOME/jdk/jre/lib/IA64N/server/Xusage.txt
```


```
$ORACLE_HOME/jdk/jre/bin/jcontrol
```

```
$ORACLE_HOME/QOpatch/qopiprep.bat
```

On Windows, no error messages are expected.

3. Stop, upgrade, and restart all global service manager servers one-at-a-time. In order to ensure zero downtime, at least one global service manager server should always be running. Global service manager servers at an earlier version than the catalog will continue to operate fully until catalog changes are made.
4. Upgrade pool databases in any order, either before or after the global service manager and catalog database upgrades, at the discretion of the pool database administrator.

 **Note:**
For general information regarding upgrading and patching see: Oracle Database Upgrade Guide.

 **Note:**
For Oracle Database 19c Proactive Patch Information, see Oracle Support (MOS) Note [2521164.1](#)

GSM Out-of-Place Update/Patching Examples

This example shows the steps required to apply the 19.18 Database Release Update (DBRU) to a 19.3 Oracle environment. The following assumptions are made:

- A 19.3.0.0.0 GDS Catalog database exists on Host A
- Two 19.3.0.0.0 GSMs (GSM1 and GSM2) exist on Host B. GSM1 is installed in ORACLE_HOME1 and GSM2 is installed in ORACLE_HOME2
- A pair of 19.3.0.0.0 Oracle pool databases exist on Host C and Host D respectively.

19.18.0.0.0 DBRU on GDS catalog, two GSMs & two Pool Databases

1. Stop the GDS catalog database and apply 19.18.0.0.0DBRU then start the GDS catalog database.

```
$ORACLE_HOME/OPatch/opatch lspatches
34765931;DATABASE RELEASE UPDATE : 19.18.0.0.230117 (REL-JAN230131)
(34765931)
29585399;OCW RELEASE UPDATE 19.3.0.0.0 (29585399)
```

OPatch succeeded.

```
SQL> select PATCH_ID, PATCH_UID, INSTALL_ID, STATUS, ACTION, DESCRIPTION
from DBA_REGISTRY_SQLPATCH;
```

PATCH_ID	PATCH_UID	INSTALL_ID	STATUS	ACTION	DESCRIPTION
29517242	22862832	1	SUCCESS	APPLY	Database Release Update : 19.3.0.0.190416 (29517242)
34765931	25098466	2	SUCCESS	APPLY	DATABASE RELEASE UPDATE : 19.18.0.0.230117 (REL-JAN230131) (34765931)

SQL>

2. Verify that the GSM setup is working properly before proceeding (now that the GDS catalog is at 19.18 and the GSMs and pool databases are at version 19.3).
3. Next, stop GSM1, making GSM2 the new master. Apply the 19.18.0.0.0 DBRU on GSM1.

```
$ORACLE_HOME1/OPatch/opatch lspatches
34765931;DATABASE RELEASE UPDATE : 19.18.0.0.230117 (REL-JAN230131)
(34765931)
```

OPatch succeeded.

```
GDSCTL> config
```

Regions

```
-----
east
west
```

GSMs

```
-----
gsm1
gsm2
```

GDS pools

```
-----
sdbpool
```

Databases

```
-----
cloud
clouddb
```

Services

```
-----
srv1
```

GDSCTL pending requests

```
-----
```

Command	Object
Status	
-----	-----

Global properties

```
-----
Name: orasampl
Master GSM: gsm2
DDL sequence #: 0
```

```
GDSCTL>
```

- Next, start GSM1 and stop GSM2 (making GSM1 the new master) and apply the 19.18.0.0 DBRU on GSM2 and then start GSM2.

```
$ORACLE_HOME2/OPatch/opatch lspatches
34765931;DATABASE RELEASE UPDATE : 19.18.0.0.230117 (REL-JAN230131)
(34765931)
```

OPatch succeeded.

```
GDSCTL> config
```

Regions

```
-----
```

```
east
west
```

GSMs

```
-----
```

```
gsm1
gsm2
```

GDS pools

```
-----
```

```
sdbpool
```

Databases

```
-----
```

```
cloud
clouddb
```

Services

```
-----
```

```
srv1
```

GDSCTL pending requests

```
-----
```

Command	Object
Status	

```
-----
```

```
-----
```

Global properties

```
-----
```

```
Name: orasamp1
Master GSM: gsm1
DDL sequence #: 0
```

```
GDSCTL>
```

- Verify that the GSM environment works properly, now that the GDS catalog & GSM versions are at 19.18 and pool database versions are still at 19.3).

6. Stop the first pool database and apply the 19.18.0.0.0 DBRU. When complete, start the database.

```
$ORACLE_HOME/OPatch/opatch lspatches
34765931;DATABASE RELEASE UPDATE : 19.18.0.0.230117 (REL-JAN230131)
(34765931)
29585399;OCW RELEASE UPDATE 19.3.0.0.0 (29585399)
```

OPatch succeeded.

```
SQL> select PATCH_ID, PATCH_UID, INSTALL_ID, STATUS, ACTION,
DESCRIPTION from DBA_REGISTRY_SQLPATCH;
```

PATCH_ID	PATCH_UID	INSTALL_ID	STATUS	ACTION

DESCRIPTION				

29517242	22862832	1	SUCCESS	APPLY
Database Release Update : 19.3.0.0.190416 (29517242)				
34765931	25098466	2	SUCCESS	APPLY
DATABASE RELEASE UPDATE : 19.18.0.0.230117 (REL-JAN230131)				
(34765931)				

SQL>

7. Stop the second pool database and apply the 19.18.0.0.0 DBRU then restart the database.

```
$ORACLE_HOME/OPatch/opatch lspatches
34765931;DATABASE RELEASE UPDATE : 19.18.0.0.230117 (REL-JAN230131)
(34765931)
29585399;OCW RELEASE UPDATE 19.3.0.0.0 (29585399)
```

OPatch succeeded.

```
SQL> select PATCH_ID, PATCH_UID, INSTALL_ID, STATUS, ACTION,
DESCRIPTION from DBA_REGISTRY_SQLPATCH;
```

PATCH_ID	PATCH_UID	INSTALL_ID	STATUS	ACTION

DESCRIPTION				

29517242	22862832	1	SUCCESS	APPLY
Database Release Update : 19.3.0.0.190416 (29517242)				
34765931	25098466	2	SUCCESS	APPLY
DATABASE RELEASE UPDATE : 19.18.0.0.230117 (REL-JAN230131)				
(34765931)				

SQL>

8. Verify that the GSM setup works correctly now that the GDS catalog, GSMs & pool database versions are at 19.18.0.0.0

GDSCTL> config

Regions

```
-----
east
west
```

GSMs

```
-----
gsm1
gsm2
```

GDS pools

```
-----
sdbpool
```

Databases

```
-----
cloud
clouddb
```

Services

```
-----
srv1
```

GDSCTL pending requests

```
-----
Command          Object
Status
-----          -----
-----
```

Global properties

```
-----
Name: orasampl
Master GSM: gsm1
DDL sequence #: 0
```

GDSCTL>

GDSCTL> databases;

```
Database: "cloud" Registered: Y State: Ok ONS: N. Role: PRIMARY
Instances: 1 Region: east
  Service: "srv1" Globally started: Y Started: Y
           Scan: N Enabled: Y Preferred: Y
Registered instances:
  sdbpool%1
```

```
Database: "clouddb" Registered: Y State: Ok ONS: N. Role: PRIMARY
Instances: 1 Region: west
  Service: "srv1" Globally started: Y Started: N
           Scan: N Enabled: Y Preferred: N
Registered instances:
  sdbpool%11
```

GDSCTL>

```
GDSCTL> status database;
Database: "cloud" Registered: Y State: Ok ONS: N. Role: PRIMARY
Instances: 1 Region: east
  Service: "srv1" Globally started: Y Started: Y
           Scan: N Enabled: Y Preferred: Y
Registered instances:
  sdbpool%1
Database: "clouddb" Registered: Y State: Ok ONS: N. Role: PRIMARY
Instances: 1 Region: west
  Service: "srv1" Globally started: Y Started: N
           Scan: N Enabled: Y Preferred: N
Registered instances:
  sdbpool%11
```

GDSCTL>

```
GDSCTL> status service;
Service "srv1.sdbpool.orasamp1" has 1 instance(s). Affinity:
ANYWHERE
  Instance "sdbpool%1", name: "cloud", db: "cloud", region:
"east", status: ready.
```

GDSCTL>

GSM_HOME to 19.18.0.0.0 DBRU, Move Existing GSM to New Home on Same Host

1. Install 19.3.0.0.0 GSM (GSM3) in ORACLE_HOME3 and apply 19.18.0.0.0 DBRU.

```
$ORACLE_HOME3/OPatch/opatch lspatches
34765931;DATABASE RELEASE UPDATE : 19.18.0.0.230117 (REL-JAN230131)
(34765931)
```

OPatch succeeded.

2. Copy gsm.ora, tnsnames.ora and the gsmwallet directory from the old \$TNS_ADMIN folder to the new one.

3. Stop GSM1 from the old GSM1 home.

```
GDSCTL> stop gsm -gsm gsm1;  
GSM is stopped successfully  
GDSCTL>
```

4. Change the WALLET_LOCATION directory to point the new GSM_HOME under gsm.ora.

5. Start GSM3 from new GSM3 home

```
GDSCTL> start gsm -gsm gsm1;  
GSM is started successfully  
GDSCTL>
```

6. Execute modify gsm -gsm <gsm name> from the new home.

```
GDSCTL> modify gsm -gsm gsm1  
GSM modified  
GDSCTL>
```

7. Install 19.3.0.0.0 GSM4 on ORACLE_HOME4 and apply 19.18.0.0.0 DBRU.

```
$ORACLE_HOME4/OPatch/opatch lspatches  
34765931;DATABASE RELEASE UPDATE : 19.18.0.0.230117 (REL-JAN230131)  
(34765931)  
OPatch succeeded.
```

8. Copy gsm.ora, tnsnames.ora and the gsmwallet directory from the old \$TNS_ADMIN folder to new one.

9. Stop GSM2 from the old GSM2 home.

```
GDSCTL> stop gsm -gsm gsm2;  
GSM is stopped successfully  
GDSCTL>
```

10. Change the WALLET_LOCATION directory to point to the new GSM_HOME under gsm.ora.

11. Start GSM4 from the new GSM4 home.

```
GDSCTL> start gsm -gsm gsm2;  
GSM is started successfully  
GDSCTL>
```

12. Execute modify gsm -gsm <gsm name> from the new home.

```
GDSCTL> modify gsm -gsm gsm2  
GSM modified  
GDSCTL>
```

13. Verify the new GSM environment.

```
GDSCTL> config  
  
Regions
```

```

-----
east
west

GSMs
-----
gsm1
gsm2

GDS pools
-----
sdbpool

Databases
-----
cloud
clouddb

Services
-----
srv1

GDSCTL pending requests
-----
Command                Object
Status                 -----
-----

Global properties
-----
Name: orasubbu
Master GSM: gsm1
DDL sequence #: 0

GDSCTL>

GDSCTL> databases;
Database: "cloud" Registered: Y State: Ok ONS: N. Role: PRIMARY
Instances: 1 Region: east
  Service: "srv1" Globally started: Y Started: Y
           Scan: N Enabled: Y Preferred: Y
Registered instances:
  sdbpool%1
Database: "clouddb" Registered: Y State: Ok ONS: N. Role: PRIMARY
Instances: 1 Region: west
  Service: "srv1" Globally started: Y Started: N
           Scan: N Enabled: Y Preferred: N
Registered instances:
  sdbpool%11

GDSCTL>

```

```
GDSCTL> status database;
Database: "cloud" Registered: Y State: Ok ONS: N. Role: PRIMARY
Instances: 1 Region: east
  Service: "srv1" Globally started: Y Started: Y
           Scan: N Enabled: Y Preferred: Y
Registered instances:
  sdbpool%1
Database: "clouddb" Registered: Y State: Ok ONS: N. Role: PRIMARY
Instances: 1 Region: west
  Service: "srv1" Globally started: Y Started: N
           Scan: N Enabled: Y Preferred: N
Registered instances:
  sdbpool%11

GDSCTL>

GDSCTL> status service;
Service "srv1.sdbpool.orasubbu" has 1 instance(s). Affinity: ANYWHERE
  Instance "sdbpool%1", name: "cloud", db: "cloud", region: "east",
  status: ready.

GDSCTL>
```

GSM_HOME to 19.18.0.0.0 DBRU on a Different Host

1. Install 19.3.0.0.0 GSM1 on ORACLE_HOME1 and apply 19.18.0.0.0 DBRU.

```
$ORACLE_HOME1/OPatch/opatch lspatches
34765931;DATABASE RELEASE UPDATE : 19.18.0.0.230117 (REL-JAN230131)
(34765931)
```

```
OPatch succeeded.
```

2. Copy `gsm.ora`, `tnsnames.ora` and the `gsmwallet` directory from source host to target host (`$GSM_HOME/network/admin`).
3. Stop GSM1 on the source host.
4. Modify the `gsm.ora` file with target host and target host wallet directory and modify the target host in the `tnsnames.ora` file for the GSM1 entry.
5. Modify the GSM with endpoint entry and verify that `config gsm` points to the correct target host details. For example:

```
GDSCTL> modify gsm -gsm gsm1 -endpoint (ADDRESS=(HOST=myhost.example.com)
(PORT=1587) (PROTOCOL=tcp))
GSM modified
GDSCTL>
```

```
GDSCTL> config gsm
Name      Region
ENDPOINT
```

```

-----
-----
gsm1      east      (address=(host=myhost.example.com) (port=1587)
(protocol=tcp))
gsm2      west      (ADDRESS=(HOST=myhost.example.com) (PORT=1787)
(PROTOCOL=tcp))

GDSCTL>

```

6. Start GSM1 from the new GSM1 home.

```

GDSCTL> start gsm -gsm gsm1;
GSM is started successfully
GDSCTL>

```

7. Execute the `modify gsm -gsm <gsm name> -pwd <GSMCATUSER password>` command like the example below:

```

GDSCTL> modify gsm -gsm gsm1 -pwd <GSMCATUSER secret_password>
GSM modified
GDSCTL>

```

Perform the following steps on GSM2.

1. Install 19.3.0.0.0 GSM2 on ORACLE_HOME2 and apply the 19.18.0.0.0 DBRU.

```

/$ORACLE_HOME2/OPatch/opatch lspatches
34765931;DATABASE RELEASE UPDATE : 19.18.0.0.230117 (REL-JAN230131)
(34765931)

OPatch succeeded.

```

2. Copy `gsm.ora`, `tnsnames.ora` and the `gsmwalletdirectory` from the source host to the target host (`$GSM_HOME/network/admin`).

3. Stop GSM2 on the source host.

4. Modify the `gsm.ora` file with target host and target host wallet directory and modify the target host in `tnsnames.ora` file for the GSM2 entry.

5. Modify the GSM configuration with the endpoint entry and verify using `config gsm` that it contains the correct target host details.

```

GDSCTL> modify gsm -gsm gsm2 -endpoint
(ADDRESS=(HOST=myhost.example.com) (PORT=1787) (PROTOCOL=tcp))
GSM modified
GDSCTL>

```

```

GDSCTL> config gsm
Name      Region
ENDPOINT
-----
-----
gsm1      east      (address=(host=myhost.example.com) (port=1587)
(protocol=tcp))

```

```
gsm2      west      (address=(host=myhost.example.com) (port=1787)
(protocol=tcp))
```

```
GDSCTL>
```

6. Start GSM2 on the target host.

```
GDSCTL> start gsm -gsm gsm2;
```

```
GSM is started successfully
```

```
GDSCTL>
```

7. Execute the `modify gsm -gsm <gsm name> -pwd <GSMCATUSER password>` command:

```
GDSCTL> modify gsm -gsm gsm2 -pwd <GSMCATUSER secret_password>
```

```
GSM modified
```

```
GDSCTL>
```

8. Verify the new GSM environment.

```
GDSCTL> config database;
```

Name	Pool	Status	State	Region
cloud	sdbpool	Ok	none	east
clouddb	sdbpool	Ok	none	west

```
GDSCTL>
```

```
GDSCTL> config
```

```
Regions
```

```
-----
east
west
```

```
GSMs
```

```
-----
gsm1
gsm2
```

```
GDS pools
```

```
-----
sdbpool
```

```
Databases
```

```
-----
cloud
clouddb
```



```

Services
-----
srv1

GDSCTL pending requests
-----
Command                Object
Status                 -----
-----

Global properties
-----
Name: orasubbu
Master GSM: gsm1
DDL sequence #: 0

GDSCTL

GDSCTL> status database;
Database: "cloud" Registered: Y State: Ok ONS: N. Role: PRIMARY
Instances: 1 Region: east
  Service: "srv1" Globally started: Y Started: Y
           Scan: Y Enabled: Y Preferred: Y
Registered instances:
  sdbpool%1
Database: "clouddb" Registered: Y State: Ok ONS: N. Role: PRIMARY
Instances: 1 Region: west
  Service: "srv1" Globally started: Y Started: N
           Scan: Y Enabled: Y Preferred: N
Registered instances:
  sdbpool%11

GDSCTL>

GDSCTL> databases;
Database: "cloud" Registered: Y State: Ok ONS: N. Role: PRIMARY
Instances: 1 Region: east
  Service: "srv1" Globally started: Y Started: Y
           Scan: Y Enabled: Y Preferred: Y
Registered instances:
  sdbpool%1
Database: "clouddb" Registered: Y State: Ok ONS: N. Role: PRIMARY
Instances: 1 Region: west
  Service: "srv1" Globally started: Y Started: N
           Scan: Y Enabled: Y Preferred: N
Registered instances:
  sdbpool%11

GDSCTL>

```

```

GDSCTL> status service;
Service "srv1.sdbpool.orasubbu" has 1 instance(s). Affinity: ANYWHERE
  Instance "sdbpool%1", name: "cloud", db: "cloud", region: "east",
  status: ready.

GDSCTL>

GDSCTL> config gsm
Name      Region
ENDPOINT
-----
-----
gsm1     east      (address=(host=myhost.example.com) (port=1587)
(protocol=tcp))
gsm2     west      (address=(host=myhost.example.com) (port=1787)
(protocol=tcp))

GDSCTL>

```

Using GDSCTL

The GDSCTL utility is a command-line interface for configuring and managing the Global Data Services framework. To run some commands, GDSCTL must establish a connection to a global service manager, a Global Data Services catalog database, or a database in the Global Data Services configuration.

Operational Notes



Note:

Unless specified, GDSCTL resolves connect strings with the current name resolution methods (such as TNSNAMES). The exception is the global service manager name. GDSCTL queries the `gsm.ora` file to resolve the global service manager name.

To start GDSCTL, enter the following command at the operating system prompt:

```
$ gdsctl
```

The preceding command starts GDSCTL and displays the GDSCTL command prompt. You can enter GDSCTL commands at either the operating system prompt or the GDSCTL command prompt, as shown in the following examples:

```
$ gdsctl add gsm -gsm gsm1 -catalog 127.0.0.1:1521:db1
```

```
GDSCTL> add gsm -gsm gsm1 -catalog 127.0.0.1:1521:db1
```

Both of the preceding commands achieve the same result. The first command is run at the operating system command prompt while the second command is run at the GDSCTL

command prompt. The command syntax examples in this document use the GDSCTL command prompt.

 **Note:**

- Many GDSCTL commands require you to first connect to the Global Data Services catalog before running the command.

If you run commands from the GDSCTL prompt, then you must execute the `connect` command before the first GDSCTL command that requires connection to the Global Data Services catalog. The `connect` command needs only to be run once in a GDSCTL session.
- A net service name may be specified instead of a connect descriptor when adding a database or broker configuration to a GDS configuration. If a net service name is specified, it must be resolvable at each global service manager in the GDS configuration to a connect descriptor that allows connectivity to the entity that is being added.

Alternatively, you can gather all the GDSCTL commands in one file and run them as a batch with GDSCTL, as follows:

```
$ gdsctl @script_file_name
```

The preceding command starts GDSCTL and runs the commands contained in the specified script file.

Using GDSCTL Help

You can display help for GDSCTL, as follows:

- GDSCTL> `help`: The `help` command displays a summary of all GDSCTL commands. If you specify a command name after `help`, then the help text for that command displays.
- GDSCTL> `command -h`: This syntax displays help text for the specified command, where `command` is the command name.

The following examples display identical help text for the `start` command:

```
GDSCTL> help start  
GDSCTL> start -h
```

Privileges and Security

Only users with the proper privileges can run GDSCTL commands.

 **See Also:**

[Overview of Global Data Services Administration](#) for more information about GDSCTL privileges and security

GDSCTL Command Syntax and Objects

GDSCTL Command Syntax and Options

GDSCTL commands, objects, and options; database names, instance names, Global Data Services region names, Global Data Services pool names, and service names are all case insensitive. Passwords and server pool names are also case sensitive. GDSCTL uses the following command syntax:

```
$ gdsctl command [object] [options] [argument]
or
GDSCTL> command [object] [options] [argument]
```

In GDSCTL syntax:

- *command*: A verb such as `add`, `start`, `stop`, or `remove`
- *object* (also known as a noun): The target or object on which GDSCTL performs the command, such as `service` or `database`. You can find a list of objects in [Table 2-1](#).
- *options*: Optional flags that extend the use of a preceding command combination to include additional parameters for the command. For example, the `-gdspool` option indicates that the name of a specific Global Data Services pool follows. If a comma-delimited list follows an option, then do not use spaces between the items in the list.
- *argument*: Additional variables for the GDSCTL command to specify actions for an object, or to specify actions for GDSCTL without an object.

GDSCTL Objects Summary

[Table 2-1](#) lists the keywords that you can use for the *object* portion of GDSCTL commands. You can use either the full name or the abbreviation for each object keyword. The **Purpose** column describes the object and the actions that can be performed on that object.

Table 2-1 Object Keywords and Abbreviations for GDSCTL

Object	Keyword (Abbreviations)	Purpose
Global Data Services catalog	<code>autovncr</code>	Enables or disables valid node checking for registration (VNCR) list for database registration
Oracle Data Guard broker configuration	<code>brokerconfig</code>	To add, modify, and manage the Oracle Data Guard broker configuration. The Oracle Data Guard broker logically groups primary and standby databases into a broker configuration that enables the broker to manage and monitor them together as an integrated unit.
Global Data Services catalog	<code>catalog</code>	To manage the Global Data Services catalog stored in an Oracle database.
Database	<code>database</code>	To add, modify, and remove database configuration information about databases.

Table 2-1 (Cont.) Object Keywords and Abbreviations for GDSCTL

Object	Keyword (Abbreviations)	Purpose
Global Data Services pool	gdspool	To add, modify, and manage a Global Data Services pool. A Global Data Services pool is a set of databases within a GDS configuration that provides a unique set of global services and belongs to a certain administrative domain.
Global service manager	gsm	To add, modify, and manage a global service manager. A global service manager is a software component that provides service-level load balancing and centralized management of services within the Global Data Services configuration.
Global Data Services catalog	invitednode	Adds host address information to the valid node checking for registration (VNCR) list in the Global Data Services catalog.
Global Data Services catalog	invitedsubnet	Adds subnet information to the valid node checking for registration (VNCR) list in the Global Data Services catalog
Global Data Services Region	region	To add, modify, and manage a Global Data Services Region, which is a logical boundary that contains database clients and servers that are considered to be geographically close to each other.
Service	service	To add, modify, list the configuration of, enable, disable, start, stop, obtain the status of, relocate, and remove global services.

GDSCTL Connections

For certain operations, GDSCTL must connect to a global service manager. To connect to a global service manager, GDSCTL must be running on the same host as the global service manager. When connecting to a global service manager, GDSCTL looks for the `gsm.ora` file associated with the local global service manager.

The following are the GDSCTL operations that require a connection to a global service manager.

- `add gsm` adds a global service manager.
- `start gsm` starts the global service manager.
- `stop gsm` stops the global service manager.
- `modify gsm` modifies the configuration parameters of the global service manager.
- `status gsm` returns the status of a global service manager.
- `set inbound_connect_level` sets the `INBOUND_CONNECT_LEVEL` listener parameter.
- `set trace_level` sets the trace level for the listener associated with the specified global service manager.

- `set outbound_connect_level` sets the timeout value for the outbound connections for the listener associated with a specific global service manager.
- `set log_level` sets the log level for the listener associated with a specific global service manager.

For all other operations, GDSCTL uses Oracle Net Services to connect to the Global Data Services catalog database or another database in the Global Data Services configuration. For these connections you can run GDSCTL from any client or host that has the necessary network configuration.

What You Need to Know About Creating the Global Data Services Catalog

Every Global Data Services configuration must have a Global Data Services catalog. The Global Data Services catalog can reside on the same host as a GDS configuration database, but Oracle does not recommend this scenario for large configurations. Oracle recommends that you use Oracle high availability features such as Oracle Real Application Clusters (Oracle RAC) and Oracle Data Guard to protect the Global Data Services catalog against outages.

Global Data Services Catalog Requirements

- The Global Data Services catalog must reside on an Oracle Database 12c (or later) database that uses a server parameter file (SPFILE).
If you create the Global Data Services catalog in an Oracle RAC database, then Oracle recommends that you set up Single Client Access Name (SCAN) for that database.
- The Global Data Services administrator who creates the Global Data Services catalog must have a user account on the [catalog database](#), and must have `GSMADMIN_ROLE` privileges and an account password. For example, the following SQL statements can be executed on the catalog database.

```
CREATE USER gsm_admin IDENTIFIED BY ****;  
GRANT gsmadmin_role TO gsm_admin;
```

- The Global Data Services catalog must be protected for high availability and disaster recovery.

Note:

Oracle recommends that the Global Data Services administrator does not directly connect to the catalog database, despite having a user account on the catalog database. Global Data Services administrators can use the GDSCTL utility to manage Global Data Services. GDSCTL connects to the Global Data Services catalog with the credentials that the Global Data Services administrator provides when running GDSCTL commands.

For example:

```
GDSCTL> create gds catalog -database serv1:1521:catdb.example.com  
-user gsm_admin
```

In the preceding example, `serv1:1521:catdb.example.com` is an Easy Connect string that contains the host name and port number of the listener that is used to connect to the database, and `catdb.example.com` is the service name for the Global Data Services catalog database.

You designate one database as the primary repository for the Global Data Services catalog. You can use existing high availability technologies, such as Oracle RAC, Oracle Data Guard, and Oracle Clusterware, to protect the Global Data Services catalog.

If you use Oracle GoldenGate, then ensure that the Global Data Services catalog gets replicated to a secondary database.



See Also:

[create gds catalog](#) for complete usage information

Creating the Global Data Services Catalog

Use GDSCTL on any host where GDSCTL is installed and configured to create a Global Data Services catalog, as follows:

```
GDSCTL> create gds catalog -database db_name -user user_name
```



See Also:

[create gds catalog](#)

Adding a Global Service Manager to the Global Data Services Catalog

Before a global service manager can be started, the global service manager should be registered in the Global Data Services catalog.

To add a global service manager:

1. Alter the `GSMCATUSER` account.

Every global service manager in a Global Data Services configuration maintains a direct Oracle Net Services connection to the catalog database under the `GSMCATUSER` account, which is created by default during Oracle Database installation. The database administrator (DBA) of the catalog database must unlock the account and give the account password to the Global Data Services administrator.

```
ALTER USER gsmcatuser ACCOUNT UNLOCK;  
ALTER USER gsmcatuser IDENTIFIED BY password;
```

2. Run the following command on the host where you want the global service manager to run:

```
GDSCCTL> add gsm -gsm gsm_name -listener listener_port -catalog catalogdb_name
```

For example:

```
GDSCCTL> add gsm -gsm east_gsm1 -listener 1523 -catalog  
serv1:1521:catdb.example.com
```

In the preceding example, `serv1:1521:catdb.example.com` is the connect identifier of the catalog database. The Global Data Services administrator is prompted for the `GSMCATUSER` password during the execution of the command.

 **See Also:**

[add gsm](#) for complete usage information

3. After you add the global service manager to the Global Data Services framework, start the global service manager, as follows:

```
GDSCCTL> start gsm -gsm gsm_name
```

During startup, the global service manager creates or modifies the `ONS.CONFIG` file and populates the file with configuration data from the Oracle Notification Service server that belongs to the global service manager.

By default, the file is created in the `$ORACLE_HOME/opmn/conf` directory. The location can be changed to `$ORACLE_CONFIG_HOME/opmn/conf` if the environment variable `ORACLE_CONFIG_HOME` is set.

 **Note:**

The `ONS.CONFIG` file cannot be shared, and there must be a unique `ONS.CONFIG` file for each global service manager installation.

Connecting to the Global Data Services Catalog

Many `GDSCCTL` commands require a connection to the Global Data Services catalog. You can connect to the Global Data Services catalog using one of the following two methods:

Method 1

Connect to the Global Data Services catalog, as follows:

```
GDSCCTL> connect [user_name]@connect_identifier
```

If you run the preceding command but do not specify a password, then `GDSCCTL` prompts you for a password, as shown in the following example:

```
GDSCCTL> connect gsm_admin@catalog  
Enter password: *****  
Catalog connection is established  
GDSCCTL>
```


In the preceding example, `catalog` is a connect identifier that resolves to one or more global service manager endpoints. For high availability, Oracle recommends that the connect identifier resolves to the list of all global service managers in the configuration. For example, if there are two global service managers in the Global Data Services configuration and you use `TNSNAMES` for name resolution, then the `tnsnames.ora` file must contain an entry similar to the following:

```
catalog = (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp) (HOST=xyz) (PORT=1523))
          (ADDRESS=(PROTOCOL=tcp) (HOST=abc) (PORT=1523))
          )
```

In the preceding example, the first global service manager runs on the host named `xyz`, and the second global service manager runs on the host named `abc`.



See Also:

[connect](#) for complete usage information

Method 2

To run `GDSCTL` commands from the operating system prompt, append the `-catalog` parameter to any of the commands that require you to be connected to the Global Data Services catalog.

For example:

```
$ gdsctl add gdspool -gdspool hr -catalog mygdscatlog
```

```
username:Robert
```

```
password:
```

`GDSCTL` must use a global service manager as a listener to connect to the Global Data Services catalog because the location of the Global Data Services catalog can change. The global service manager records location changes and can route connection requests.

What You Need to Know About Adding a Global Data Services Pool

Note:

- If you require only one Global Data Services pool, then you do not need to add one using these instructions. A default Global Data Services pool, `DBPOOLORA`, is created for you when you create the Global Data Services catalog.
- The Global Data Services administrator has permissions to run `GDSCTL` commands to manage a Global Data Services pool and, if there is only a single pool, then the Global Data Services administrator also administers the pool.
- If you specify a user when you run the `gdsctl add gdspool` command, then the local DBA where the Global Data Services catalog resides must first add the user to the catalog database.

Large database clouds can require multiple Global Data Services pools that are managed by different administrators.

For example:

```
GDSCTL> add gdspool -gdspool hr -users rjones
```

The preceding example adds a Global Data Services pool called `hr`, and adds the user `rjones`, who is assigned the privileges to administer the `hr` pool. The privileges enable the pool administrator to add databases to the pool and manage global services on the databases in the pool.

A Global Data Services pool must have a unique name within its GDS configuration. If you do not specify a name for the pool when you create it, then the name defaults to `oradbpool`. The pool name can be up to 30 bytes long and can be any valid identifier (an alphabetical character followed by zero or more alphanumeric ASCII characters or the underscore (`_`)).

Adding a Global Data Services Pool

Ensure that you are connected to the Global Data Services catalog and add a pool, administered by a specific user, as follows:

```
GDSCTL> add gdspool -gdspool database_pool_list [-users user_list]
```

What You Need to Know About Adding a Global Data Services Region

Note:

If you require only one Global Data Services region, then you do not need to add a region using these instructions. A default Global Data Services region, `REGIONORA`, is created for you when you create the Global Data Services catalog.

For example:

```
GDSTCL> add region -region west,east
```

The preceding example adds two regions, `east` and `west`, to the Global Data Services framework.

A Global Data Services region should have a name that is unique within the corresponding Global Data Services configuration. If no name is specified at the first region creation time, the default name, `oraregion`, is given to the region. The region name can be up to 30 characters long and can be any valid identifier - an alphabetical character followed by zero or more alphanumeric ASCII characters or `'_'`.

Adding a Global Data Services Region

Ensure that you are connected to the Global Data Services catalog and add a Global Data Services region, as follows:

```
GDSTCL> add region -region region_list
```

Adding a Database to a Global Data Services Pool

To provide global services, a database must be added to a Global Data Services pool.

Before adding a database to a pool, the database administrator should unlock the `GSMUSER` account and give the password to the Global Data Services pool administrator, as shown in the following example:

```
ALTER USER gsmuser ACCOUNT UNLOCK;  
ALTER USER gsmuser IDENTIFIED BY password;
```

To be part of a Global Data Services pool, a database must use a server parameter file (SPFILE). An Oracle RAC database should also have `SCAN` set up.

To add a database:

1. Connect to the Global Data Services catalog using the Global Data Services pool or Global Data Services administrator credentials, for example:

```
GDSTCL>connect rjones@catalog
```

2. Run the `gdsctl add database` command:

```
GDSCTL>add database -connect edc007:1521/db14.east.example.com -region east  
-gdspool hr
```

In this example `edc007:1521/db14.east.example.com` is the connect identifier of the database, and then you are prompted for the `GSMUSER` account password on this database.

Note:

If the pool already contains databases and there are global services associated with the pool, then the services are automatically created on the new database.

Valid Node Checking for Registration

The valid node checking for registration (VNCR) feature provides the ability to configure and dynamically update a set of IP addresses, host names, or subnets from which registration requests are allowed by the global service manager. Database instance registration with a global service manager succeeds only when the request originates from a valid node.

By default, the Global Data Services framework automatically adds a VNCR entry for the host on which a remote database is running each time the `gdsctl add database` command is run. The automation (called auto-VNCR) requires that the host name entry exists in either the local hosts file or in the name server. If the remote host is identified by a different name on any of the nodes on which the global service manager runs, then the Global Data Services administrator must manually add VNCR entry to the Global Data Services catalog by running the `gdsctl add invitednode` command.

See Also:

[add invitednode \(add invitedsubnet\)](#) for complete usage information

Adding a Service to a Global Data Services Pool

The `gdsctl add service` command is used to create a service on the Global Data Services pool databases. A simple example of the command is as follows:

```
GDSCTL> add service -gdspool hr -service emp_report1 -preferred_all
```

In this example `emp_report1` is the service name and the `-preferred_all` option means that the service should normally run on all of the databases in the pool.

The service name specified in the 'add service' command can be domain qualified (for example, `sales.example.com`) or not (for example, `sales`). If the specified name is not domain qualified, the service is created with the default domain name "`<GDS_pool_name>.<GDS_configuration_name>`", however the shorter non-domain qualified name can be used with subsequent `GDSCTL` commands to manage the service. If the specified name is domain qualified, the full domain qualified service name must be used in all subsequent `GDSCTL` commands used to manage the service.

For Oracle RAC-enabled pool databases, after the service has been added, run `GDSCtl modify service` to specify which Oracle RAC instance a given global service should run on, as shown in the following example.

```
GDSCtl> modify service -service emp_report1 -gdspool hr - database db14
      -modify_instances -preferred db14_n1, db14_n2
```

A global service name must be unique within a GDS pool and when qualified by domain, must also be unique within a GDS configuration. A global service cannot be created at a database if a local or global service with the same name already exists at that database.

A global service name can contain alphanumeric characters, `'_'` and `'.'`. The first character must be alphanumeric. The maximum length of a global service name is 64 characters. The maximum length of a domain qualified global service name is 250 characters.

An Oracle Net connect descriptor used to connect to a global service must contain a domain qualified service name



See Also:

[add service](#) and [modify service](#) for complete usage information

Starting a Global Service

The `gdscctl start service` command is used to start an existing service on the Global Data Services pool databases.

```
GDSCtl>start service -service emp_report1 -gdspool hr
```

If the `-role` parameter is specified for the service, the service only starts on the databases in which the role matches the specified value. If the `-lag` parameter is specified for the service, the service only starts on the databases for which replication lag does not exceed the specified value. Unless `-preferred_all` is specified for the service, the service only starts on the databases that are listed as preferred for the service.



Note:

Before starting services which run on administrator-managed databases, they must be modified for those databases to stipulate which instances should run the service. Please refer to the `-modify_instances` parameter of the `modify service` command.

**See Also:**[start service](#)[modify service](#)

Database Client Configuration

Database clients connect to database services using an Oracle Net connect string. The connect string used for a global service differs from the connect string used for a local service in the following ways:

- The service name parameter in the connection data section specifies a global service
- Multiple connection endpoints are specified, and these endpoints are global service managers rather than local, remote, or single client access name (SCAN) listeners
- The database client's region may be specified in the connection data section

Consider the following connect string:

```
(DESCRIPTION=
  (FAILOVER=on)
  (ADDRESS_LIST=
    (LOAD_BALANCE=ON)
    (ADDRESS=(host=sales-east1) (port=1522))
    (ADDRESS=(host=sales-east2) (port=1522))
    (ADDRESS=(host=sales-east3) (port=1522)))
  (ADDRESS_LIST=
    (LOAD_BALANCE=ON)
    (ADDRESS=(host=sales-west1) (port=1522))
    (ADDRESS=(host=sales-west2) (port=1522))
    (ADDRESS=(host=sales-west3) (port=1522)))
  (CONNECT_DATA=
    (SERVICE_NAME=sales)
    (REGION=east)))
```

This connect string contains three global service managers (sales-east1, sales-east2, and sales-east3) in the client's local region (east), and three global service managers (sales-west1, sales-west2, and sales-west3) in the client's buddy region (west).

Client-side load balancing is enabled across the global service managers within each region by setting the `LOAD_BALANCE` parameter to `ON` in the address list for each region. Connect-time failover between regions is enabled by setting the `FAILOVER` parameter to `ON`.

It is a best practice to have three global service managers in each region, for each region to have a buddy region, and for client-side load balancing and connect-time failover to be configured as shown in the example connect string.

The `REGION` parameter is optional if only global service managers from the local region are specified in the connect string. This is the case when there is only one region in the GDS configuration, or could be the case when there are multiple regions, but it is not feasible to change the connect string of an existing client designed to work with a single database. If the `REGION` parameter is not specified, the client's region is assumed to be the region of the global service manager used to connect to the global service.

 **Note:**

The pre-12c Thin JDBC client can only be used with a GDS configuration that has a single region, unless the region parameter is specified in the connect string.

The Oracle Database 12c and later integrated clients use Oracle Notification Services (ONS) to receive the Fast Application Notification (FAN) events that support load balancing and Fast Connection Failover (FCF). The integrated clients automatically subscribe, without user-configuration, to up to 3 of the ONS servers co-located with the global service managers in each of the client's local and buddy regions.

 **Note:**

Automatic ONS configuration is not supported if connections to an ONS server have to be secured using SSL. You must configure ONS manually to enable SSL. See client-specific guides for information on how to configure ONS manually.

Note for Pre-12c Clients:

The pre-12c OCI and ODP.NET clients do not support global services.

The pre-12c JDBC client supports global services, but you must manually configure it to subscribe to the ONS servers co-located with the global service managers in the client's local and buddy regions. It is a best practice to subscribe to three ONS servers in each of the client's local and buddy regions.

See the *Oracle Database JDBC Developer's Guide* for information about how to configure ONS subscriptions.

Configuring Integrated Clients for FAN and FCF

Load balancing and Fast Connection Failover (FCF) of client connections across the databases in a GDS configuration is supported by the Oracle database integrated clients, and requires that those clients be configured for FAN and FCF.

See one of these client-specific Programmer's Guides for information about that client's FAN and FCF configuration requirements:

- *Oracle Database JDBC Developer's Guide*
- *Oracle Universal Connection Pool Developer's Guide*
- *Oracle Database Development Guide*
- *Oracle Data Provider for .NET Developer's Guide for Microsoft Windows*

What You Need to Know About Exporting the GDS Catalog Data for Logical Backups

Because the GDS catalog stores metadata for the entire GDS configuration, loss or corruption of the catalog data may require significant efforts to manually recreate it. While unavailability of the GDS catalog does not impact core GDS functionality such as load balancing, service failover, and application notification, no changes can be made to the GDS configuration until the catalog is restored. Therefore it is important to develop a strategy for protecting the GDS catalog.

Even when the GDS catalog is protected by HA technologies such as Oracle RAC and Data Guard, it is highly recommended that you regularly back up the GDS catalog. You can create a logical backup of the catalog by exporting the catalog data to a file. This backup of the GDS catalog can help you in a disaster recovery scenario, and when there is need to undo changes made to the catalog since the last backup was made. You can also use the backup when moving the GDS catalog to a new database.

The catalog configuration will be saved to the specified file on the system where GDSCTL is running. Access to the file should be limited to Global Data Services administrators since it may contain sensitive information such as connection strings for the pool databases.

Note:

- It is strongly recommended that the catalog be validated before exporting it to ensure that there are no inconsistencies in the catalog data. Any errors reported by the `validate catalog` command should be corrected before exporting the catalog data.
- You must not make any change to the file with exported catalog data. Any changes to the file may prevent using of this file for the catalog restore, or may cause catalog corruption after restore. It is recommended to store the file checksum along with the backup file. Do not try to restore the catalog configuration if the file has been modified.

Exporting the GDS Catalog Data for Logical Backups

To export the GDS catalog data to a file, ensure that you are connected to the catalog and execute the following command:

```
GDSCTL> export catalog file_name_with_full_path
```

Restoring Logical Backup of the GDS Catalog into the Same Catalog Database

To restore GDS catalog data from a backup file:

Connect to the catalog database and issue the following command:

```
GDSCTL> import catalog file_name_with_full_path
```


After the import of the catalog data is finished, pool databases will be automatically synchronized (see the `sync database` command description in [sync database \(synchronize database\)](#).) If there are no global service managers available, this action will be deferred until a global service manager registers with the catalog.

It is recommended that you validate the catalog after the import is done and all the databases are synchronized.

 **Note:**

Trying to restore the GDS catalog from the file that has been modified may result in a corrupt catalog. It is the responsibility of the GDS administrator to check consistency of the backup file (for example, by using the checksum.)

Restoring Logical Backup of the GDS Catalog into a new Catalog Database

When moving a catalog to a new database, you must first create an empty catalog on the database (see [What You Need to Know About Creating the Global Data Services Catalog](#).) After that the `import catalog` command may be executed as described in the previous section.

If the new catalog database has a different connection string, it is the administrator's responsibility to change the connection string on global service manager systems. It is also required to restart all global service managers in this case. The synchronization procedure will not be completed, and thus the restore procedure will not be finished, until at least one global service manager registers with the catalog.

Changing the GSMCATUSER Password

To change GSMCATUSER password:

1. Run the following command in SQL*Plus while connected to the GDS catalog:

```
ALTER USER gsmcatuser IDENTIFIED BY ****
```

2. Then in GDSCTL run the following command:

```
GDSCTL> modify catalog -oldpwd oldpassword -newpwd newpassword
```

3

Administering Global Data Services Configurations

The `GDSCTL` utility is used to create, manage and monitor a Global Data Services configuration and all of its components. This utility is very similar to the `SRVCTL` utility used to manage an Oracle Real Application Cluster (Oracle RAC). The following topics explain how to administer your GDS configurations.

Overview of Global Data Services Administration

Global Data Services is managed by the Global Data Services administrator whose responsibilities include the following tasks:

- Installing and upgrading the global service manager software
- Creation and maintenance of the Global Data Services catalog
- Starting, stopping, and configuring global service managers
- Creation and administration of Global Data Services regions and pools
- Management of global services
- Monitoring of the Global Data Services framework components

Each Global Data Services configuration requires at least one Global Data Services administrator. A small configuration can be administered by a single person who performs all the administrative duties. For a large configuration with many regions and pools it may be necessary to have a group of Global Data Services administrators who share responsibilities. All Global Data Services administrators have privileges to perform all the listed administrative tasks for a given Global Data Services configuration.

An operating system account should exist for the Global Data Services administrator on all computers where global service managers are expected to run. The account user should have privileges to install and run global service manager software. Only Global Data Services administrators should be granted these privileges.

A Global Data Services administrator must also be added as a user to the Global Data Services catalog database and granted the `GSMADMIN_ROLE` role. The database account for a Global Data Services administrator should be created by a database administrator of the catalog database. The Global Data Services administrator might create this account by himself if he happens to have local database administrator privileges on this database.

If a Global Data Services configuration contains multiple pools, then in addition to Global Data Services administrators who manage the entire configuration, each pool can have one or more Global Data Services pool administrators. Responsibilities of a pool administrator are limited to the administration of a particular pool and include the following tasks:

- Adding and removing databases in the pool
- Management of global services in the pool

To perform these tasks a Global Data Services pool administrator must be a user of the Global Data Services catalog database with the appropriate privileges. Creation of the database user for a pool administrator and granting of the privileges is performed automatically when a Global Data Services pool is created with the `-USER` option. A pool administrator can also be added to a pool after its creation using `gdsctl modify gdspool` command. A Global Data Services administrator always has privileges to administer any pool in the database configuration.

All administrative operations should be performed using the appropriate GDSCTL commands. Execution of the most GDSCTL commands requires access to the Global Data Services catalog. For such commands, credentials for the catalog database must be specified using the appropriate command options.

Many administrative operations, such as adding a database to a Global Data Services pool, or enabling a global service, require making changes not only to the Global Data Services catalog, but also to databases in the Global Data Services configuration. The generic workflow for such commands is as follows:

- GDSCTL connects to the catalog database with credentials provided by the administrator and makes appropriate changes to the catalog.
- The catalog database notifies all global service managers in the Global Data Services configuration about the changes. The notification is sent using an Oracle Net Services connection that each global service manager maintains with the catalog database.
- After receiving the notification one of the global service managers connects to the configuration databases that need to be configured and makes the appropriate changes.

To support this workflow a global service manager should be able to connect to the catalog and configuration databases. The connection to the catalog database is established using `GSMCATUSER` account, which is created by default on any Oracle database during database installation. The account must be unlocked by the database administrator of the catalog database and its password given to the Global Data Services administrator. Whenever a new global service manager is added to the GDS configuration, the Global Data Services administrator has to specify the password for the `GSMCATUSER` account. The password is then encrypted and stored in the global service manager wallet for future use by the global service manager.

The global service manager connects to the pool databases using the `GSMUSER` account, which also exists by default on any Oracle database. The account is locked after the database installation. It should be unlocked by the local database administrator before the database can be added to a Global Data Services pool. The password for the `GSMUSER` account is given to the pool or Global Data Services administrator who adds the database to a Global Data Services pool and must be specified in the `gdsctl add database` command. The password is stored in the Global Data Services catalog for future use by all global service managers.

The `GSMUSR` passwords are stored the GDS catalog in an encrypted form using the PKCS 1 encryption/decryption schema. You can encrypt `GSMUSR` passwords stored in the GDS catalog with a newly generated keys by executing the `modify catalog` command. For example:

```
GDSCTL> modify catalog -newkeys
```

`GSMCATUSER` and `GSMUSER` accounts are shared by all global service managers in the Global Data Services framework and used for all management operations

performed by global service managers, including automatic operations such as service failover. Human users should never connect to databases using these accounts.

In addition to the GSMCATUSER and GSMUSER accounts, the GSMADMIN_INTERNAL account is also used in a GDS configurations, both in the catalog and pool databases. This account's only purpose is to own the tables, packages, and other objects needed to support a GDS installation. It should never be unlocked, assigned a password, or used for interactive logins.

Managing Database Pools

This section describes the administration tasks associated with managing database pools in the global data services framework. It contains the following topics:

Adding Oracle Data Guard Broker Managed Databases to a Database Pool

When you include an Oracle Data Guard broker configuration in a Global Data Services configuration, you manage the broker configuration as one unit. Only an entire Oracle Data Guard broker configuration can be added to (or deleted from) a database pool. A configuration cannot span multiple pools. An attempt to add or remove an individual database to or from a pool that belongs to a broker configuration results in an error.

The only way to add a database to the pool is to add the database to the broker configuration (using the `DGMGRL` utility). Adding a database to the broker configuration causes its automatic addition to the database pool to which this configuration belongs. Removing a database from a broker configuration causes its removal from the pool that contains the configuration. This is the only way to remove a database from a pool that contains a broker configuration.

Also, note the following limitations:

- The set of databases in a database pool can be either:
 - The set of databases that belong to a single broker configuration
 - A set of databases that belong to no broker configuration

You can add a broker configuration only to an empty database pool and, if a pool already contains a broker configuration, then, to add a database to a database pool, you must add the database to the broker configuration contained in the database pool.

- Role-based global services are supported only for database pools that contain a broker configuration.



See Also:

Oracle Data Guard Broker for more information about the `DGMGRL` utility

Managing Global Services

This section describes the administration tasks associated with global services. It contains the following topics:

Creating a Global Service

A global service is created by execution of the `add service` command. This command associates the global service with a Global Data Services pool and stores attributes of the service in the Global Data Services catalog. If databases are specified using the `-preferred` or `-available` options, the service is created on those specified databases. If the `-preferred_all` option is used, the service is created on all databases in the Global Data Services pool.

A service that already exists in a Global Data Services pool is also automatically created on a database in the following cases:

- The service is modified to add a database that is part of the pool.
- The service has the `-preferred_all` attribute and a new database is added to the pool.



See Also:

[add service](#)

Starting a Global Service

A global service is automatically enabled immediately after it has been created. The term *enabled* means that the service can be started on a database if the database is eligible for running the service, namely, when the following conditions are met:

- The database is open and registered with a global service manager.
- The service has not been disabled on that database.
- The database role matches the role attribute of the service.
- The replication lag on the database does not exceed the maximum value specified for the service.
- The service has not reached its cardinality defined by the number of preferred databases.
- No other database in the pool is a better candidate for starting the service, for example, the service can be started on an available database only if there is no eligible preferred database.

A newly created global service never gets started automatically until the `start service` command is executed by the user. This gives the pool administrator control over the initial service startup which may be important in the case when multiple services are being added to the pool and a certain sequence of service startups is required.

A service with the automatic management policy (the default option) must be initially started by executing the `start service` command without the `-database` option. This command not only starts the service on all eligible databases in the pool, but also enables the automatic service startup in the following cases:

- After the service is automatically created on a database that is eligible to run it. (The two cases of automatic service creation are listed in the previous section.)
- A database that was down gets restarted and is eligible for the service.
- A database becomes eligible to run the service. This can happen, for example, because the replication lag on a database has decreased to an acceptable value, or the service cardinality has been increased by the user.

The `start service` command with `-database` option can be used to start a service with the automatic management policy on particular databases if the service was shut down there by the `stop service` command described in [Stopping a Global Service](#).

A service with the manual policy must be started manually on each individual database, including when a database gets restarted or becomes eligible to run the service. When executed against a service with the manual policy, the `start service` command without the `-database` option starts the service on all eligible databases that are currently present in the pool. If used with the `-database` option, the command starts the service only on the specified databases, if they are eligible to run it.

 **Note:**

The cases of automatic service startup listed in this section only describe what happens when the `start service` command is executed against a service with the automatic management policy. They do not include cases when a service is started automatically on a database because of a failover from another database. Service failover is not associated with the `start service` command, and its behavior is the same for services with automatic and manual management policy.

 **See Also:**

[start service](#)

Stopping a Global Service

A global service running on databases in a Global Data Services pool can be shut down by the `stop service` command. If the `stop service` command is executed with the `-database` option, then the service is stopped on the specified databases; otherwise it is stopped on all databases in the pool.


 **Note:**

A stopped service with the automatic management policy is restarted if the database where it was running gets restarted and is eligible to run the service. Also, stopping a service with the automatic management policy on all databases in a Global Data Services pool does not prevent the automatic service startup on a new database when the service is created there. To completely disable the automatic startup of a service, its management policy should be changed to manual.

When the service is stopped by the user, the Global Data Services framework considers that database to be temporarily unavailable for this service. Stopping a global service does not cause a service failover event; the service cardinality is temporarily decreased and the global service manager does not attempt to start the service on another database in the pool.

However, a database with a stopped service is still considered a failover target for this service; when the service fails on another database, it can be started on this database if it is eligible to run the service. After the service failover to a database, the service on that database is no longer considered to be stopped by the user.


A stopped service can be manually restarted by executing the `start service` command.

 **See Also:**
[stop service](#)

Disabling a Global Service


A global service can be disabled on a database or a set of databases by executing the `disable service` command. A disabled service cannot be started until it is reenabled. This includes the service failover from another database; a database with the disabled service is never considered a failover target.

A service has to be stopped to be disabled. An error is returned if `disable service` is executed against a database where the service is running.

 **See Also:**
[enable service](#)

Enabling a Global Service


A disabled global service can be reenabled on a database by executing `enable service` command. If the service management policy is `AUTOMATIC` and the database is eligible for running the service, it is started automatically after being enabled. A service with the `MANUAL` management policy must be started manually. A database can become a failover target after a service is enabled there.

 **See Also:**
[add service](#)

Modifying Global Service Attributes


The `modify service` command is used to modify global service attributes. In addition to specifying service properties (such as role, maximum lag, load balancing method, and so on) service attributes define on which databases the service should run. Therefore `modify service` can be used to add a database to a service, remove it from a service, or move a service from one database to another. As the result of the command execution, a service may be created, deleted, started, or stopped on one or more databases in a Global Data Services pool.

Most global service attributes are specified at the service creation time in the `add service` command and only need to be modified when some changes have to be made. However, a few service attributes related to Oracle RAC databases, must be set by executing the `modify service` command right after the `add service` command has been executed. These attributes include the name of the server pool, instance cardinality (UNIFORM/) and some other parameters that are specific to particular Oracle RAC databases. Such attributes cannot be set by the `add service` command because this command is only used to specify attributes that have the same values for all databases in a Global Data Services pool.

 **See Also:**
[modify service](#)

Deleting a Global Service

The `remove service` command deletes a global service from the Global Data Services pool by removing it from the Global Data Services catalog and all databases where it was created. A service should be stopped before being deleted.

 **See Also:**
[remove service](#)

Managing the GDS Stack

This section describes the startup and shutdown of components in the global data services framework. It contains the following topics:

Starting Up the GDS Stack

The following is the recommended startup sequence of the GDS stack:

- Start the global data services catalog database and local listener.
- Start the global service managers.
- Start the GDS pool databases and local listeners.

- Start the global services.
- Start the application tier and the clients.

Shutting Down the GDS Stack

The following is the recommended shutdown sequence of the GDS stack:

- Shut down the application tier and the clients.
- Stop the global services.
- Shut down the GDS pool databases and local listeners.
- Stop the global service managers.
- Stop the global data services catalog database and the local listener.

4

Troubleshooting Global Data Services

The following topics provide information about tools and solutions for troubleshooting the GDS issues you might encounter.

Troubleshooting Oracle Error Codes

This section contains information for troubleshooting specific Oracle error messages you might encounter, such as:

ORA-01045: user GSMADMIN_INTERNAL lacks CREATE SESSION privilege; logon denied

The user `GSMADMIN_INTERNAL` is an internal only user, it should never be unlocked or used for any database login. No direct modifications should be made on the Global Data Services schema objects unless directed by Oracle Technical Support.

ORA-12514: TNS:listener does not currently know of service requested in connect descriptor

The global service may be down. Verify that the pool databases are up and the service is started.

The global service may be disabled. Ensure that the pool databases are up and the service is enabled and started.

The GDS pool database may be down. Ensure that the GDS pool databases are up and the service is enabled and started.

ORA-12516: TNS:listener could not find available handler with matching protocol stack

The GDS pool database's local listener may be down. Ensure that the GDS pool database local listener is running.

ORA-12541: TNS:no listener

All global service managers may be down. Verify that the global service managers are running.

GSM-45034: Connection to GDS catalog is not established

The GDS catalog database or its listener may be down. Verify that the GDS catalog database and its local listener are running.

GSM-45054: GSM error or NET-40006: unable to start GSM

The GDS catalog database or its listener may be down. Verify that the GDS catalog database and its local listener are running.

Solutions for General Issues

This section contains information for solving some general issues you might encounter, such as:

Connecting to GDS Configuration Databases When No Global Service Managers Are Running

You need multiple address lists; the first list should be exclusively regional global service manager listeners, the second list contains global service manager listeners of the buddy region and the third list contains local listeners.

You can always connect through a global service manager while it is up, and only fail over to local listeners when all global service manager listeners are down.

Template:

```
(DESCRIPTION=
  (FAILOVER=on)
  (ADDRESS_LIST=
    (LOAD_BALANCE=ON)
    (ADDRESS=(global_protocol_address_information))
    (ADDRESS=(global_protocol_address_information))
    (ADDRESS=(global_protocol_address_information))
  )
  (ADDRESS_LIST=
    (LOAD_BALANCE=ON)
    (ADDRESS=(global_protocol_address_information))
    (ADDRESS=(global_protocol_address_information))
    (ADDRESS=(global_protocol_address_information))
  )
  (ADDRESS_LIST=
    (LOAD_BALANCE=ON)
    (ADDRESS=(local_protocol_address_information))
    (ADDRESS=(local_protocol_address_information))
  )
  (CONNECT_DATA=
    (SERVICE_NAME=global_service_name)
    (REGION=region_name)))
```

Example:

```
(DESCRIPTION=
  (FAILOVER=on)
  (ADDRESS_LIST=
    (LOAD_BALANCE=ON)
    (ADDRESS=(HOST=gsmhost1) (PORT=1523) (PROTOCOL=TCP))
    (ADDRESS=(HOST=gsmhost2) (PORT=1523) (PROTOCOL=TCP))
    (ADDRESS=(HOST=gsmhost3) (PORT=1523) (PROTOCOL=TCP))
  )
  (ADDRESS_LIST=
```

```
(LOAD_BALANCE=ON)
(ADDRESS=(HOST=gsmhost4) (PORT=1523) (PROTOCOL=TCP))
(ADDRESS=(HOST=gsmhost5) (PORT=1523) (PROTOCOL=TCP))
(ADDRESS=(HOST=gsmhost6) (PORT=1523) (PROTOCOL=TCP))
)
)
(ADDRESS_LIST=
(LOAD_BALANCE=ON)
(ADDRESS=(HOST=server1) (PORT=1521) (PROTOCOL=TCP))
)
)
(CONNECT_DATA=
(SERVICE_NAME=sales_read_service.dbpoolora.oradbccloud)
(REGION=WEST))
```

**Note:**

In the case of an Oracle RAC enabled GDS database, the third address list contains the local Oracle RAC database's SCAN listeners.

Connecting to Catalog Databases When No Global Service Managers Are Running

Local listener enables access to the GDS catalog database even when global service managers are down.

This access may be needed for any DB Administration/maintenance activities on the catalog database when global service managers are not running.

Obtaining the Running Status of Global Data Services Components

The `status` command can be used to obtain the running status of the GDS components.

```
GDSCCTL>status gsm
GDSCCTL>status service
GDSCCTL>status database
```

Viewing Static Configuration Information for Global Data Services Components

The `gdscctl config` command can be used to obtain the static configuration information of various GDS components.

```
GDSCCTL>config
GDSCCTL>config gsm
GDSCCTL>config region
GDSCCTL>config gdspool
```

```

GDSCTL>config database

GDSCTL>config service

GDSCTL>config invitednode

```

Enabling and Disabling Tracing on a Global Service Manager

You can enable tracing using the `set trace_level` command.

```
GDSCTL>set trace_level -gsm gsm_name SUPPORT
```

The `SUPPORT` option provides trace with troubleshooting information for Oracle Support Services. The other options are `ADMIN` and `USER`.

To disable tracing:

```
GDSCTL>set trace_level -gsm gsm_name OFF
```

Using Global Service Manager Log and Trace Files

The exact location of a given global service manager's log and trace files can be obtained using the `status gsm` command as shown in the following example.

```

GDSCTL>status gsm

Alias                MYGSM
Version              12.1.0.0.2
Start Date           13-OCT-2012 12:20:16
Trace Level          support
Listener Log File    /scratch/oracle/diag/gsm/myhost/mygsm/alert/log.xml
Listener Trace File  /scratch/oracle/diag/gsm/myhost/mygsm/trace/ora_1829_
47542149303936.trc
Endpoint summary     (ADDRESS=(HOST=myhost.com) (PORT=1571) (PROTOCOL=tcp))
GSMOCI Version       0.1.7
Mastership           N
Connected to GDS catalog Y
Process Id           1833
Number of reconnections 0
Pending tasks.      Total 0
Tasks in process.   Total 0
Regional Mastership TRUE
Total messages published 34261
Time Zone            -07:00
Orphaned Buddy Regions: None
GDS region           east
Network metrics:
  Region: euro RTT:34 Bandwidth:40

```

In this example `myhost` is the global service manager host name and `mygsm` is the name of the global service manager.

The text based listener log can be found in `/scratch/oracle/diag/gsm/hostname/gsm_name/trace` directory. The file is called `alert_gsm*.log` (for example, `alert_gsml.log`)

Using SYS_CONTEXT Parameters in a GDS Environment

For a session established using a connection to a global service, some parameters of namespace USERENV have values that are different from values set when connecting to a local service on the same database. The different values for a global service are set to make the database pool appear to clients as a single database with many instances. This was done to provide backward compatibility with pre-12c clients which expect multiple instances of a service to exist only on an Oracle RAC database.

When a client connects to a global service, GDS sets the following in the session context differently.

- DB_UNIQUE_NAME and DB_DOMAIN are set to <gdspool_name>.<config_name>
- INSTANCE is set to a system generated number <inst_num> which is unique within a GDS configuration
- INSTANCE_NAME is set to <gdspool_name>%<virtual_instance_num>
- SERVICE_NAME is set to <region_name>%<service_name>

A

GDSCTL Commands Used For Oracle Sharding

A subset of GDSCTL commands are applicable to an Oracle Sharding configuration.

The following GDSCTL commands are commonly used in an Oracle Sharding configuration:

- `add cdb`
- `add credential`
- `add file`
- `add gsm`
- `add invitednode (add invitedsubnet)`
- `add region`
- `add service`
- `add shard`
- `add shardgroup`
- `add shardspace`
- `config`
- `config cdb`
- `config chunks`
- `config credential`
- `config file`
- `config gsm`
- `config region`
- `config sdb`
- `config service`
- `config shard`
- `config shardgroup`
- `config shardspace`
- `config table family`
- `config vncr`
- `configure`
- `connect`
- `create shard`
- `create shardcatalog`

- delete catalog
- deploy
- disable service
- enable service
- modify catalog
- modify cdb
- modify credential
- modify file
- modify gsm
- modify region
- modify service
- modify shard
- modify shardgroup
- modify shardspace
- move chunk
- relocate service
- remove cdb
- remove credential
- remove file
- remove gsm
- remove invitednode (remove invitedsubnet)
- remove region
- remove service
- remove shard
- remove shardgroup
- remove shardspace
- services
- set gsm
- set inbound_connect_level
- set log_level
- set outbound_connect_level
- set trace_level
- split chunk
- sql
- start gsm
- start service
- status gsm

- [status service](#)
- [stop gsm](#)
- [stop service](#)
- [validate catalog](#)

**See Also:**

Using Oracle Sharding for information about Oracle Sharding

B

GDSCTL Commands Used For Global Data Services

A subset of GDSCTL commands are applicable to a Global Data Services (GDS) configuration.

The following GDSCTL commands are commonly used in a GDS configuration:

- `add brokerconfig`
- `add database`
- `add gdspool`
- `add gsm`
- `add invitednode (add invitedsubnet)`
- `add region`
- `add service`
- `config`
- `config database`
- `config gdspool`
- `config gsm`
- `config region`
- `config service`
- `config vncr`
- `configure`
- `connect`
- `create gdscatalog`
- `delete catalog`
- `disable service`
- `enable service`
- `export catalog`
- `import catalog`
- `modify catalog`
- `modify database`
- `modify gdspool`
- `modify gsm`
- `modify region`
- `modify service`

- relocate service
- remove brokerconfig
- remove database
- remove gdspool
- remove gsm
- remove invitednode (remove invitedsubnet)
- remove region
- remove service
- services
- set gsm
- set inbound_connect_level
- set log_level
- set outbound_connect_level
- set trace_level
- start gsm
- start service
- status database
- status gsm
- status service
- stop gsm
- stop service
- sync brokerconfig (synchronize brokerconfig)
- sync database (synchronize database)
- validate catalog

C

Global Data Services Control Utility (GDSCTL) Command Reference

This appendix includes a complete reference of the Global Data Services utility (GDSCTL) commands for use with a Global Data Services or Oracle Sharding configuration.

add brokerconfig

Adds an Oracle Data Guard broker configuration to a Global Data Services pool.

Syntax

```
add brokerconfig -connect connect_identifier
                 [-pwd password]
                 [-gdspool gdspool_name]
                 [-region region_name]
                 [-savename]
                 [-force]
```

Options

Table C-1 GDSCTL add brokerconfig Options

Option	Description
<code>-connect <i>connect_identifier</i></code>	Specify an Oracle Net connect descriptor or net service name that resolves to a connect descriptor for a database in the broker configuration.
<code>-force</code>	If specified, the existing GDS configuration is deleted. Deletes an existing, running SDB, and should only be used if you want to get rid of the entire SDB.
<code>-gdspool <i>gdspool_name</i></code>	The pool to which the databases of the Oracle Data Guard broker configuration are to be added. If the specified Global Data Services pool already contains databases or another configuration, GDSCTL returns an error.
<code>-pwd <i>password</i></code>	The password for the GSMUSER. If <code>-pwd</code> is not specified, then you are prompted for the password.
<code>-region <i>region_name</i></code>	The Global Data Services region to which the databases belong. If you specify a region, then all the databases are added to that region. If you do not specify a region, then all databases are added with a region of UNASSIGNED. If the region is UNASSIGNED, then you must use the modify database command to change the region.

Table C-1 (Cont.) GDSCTL add brokerconfig Options

Option	Description
-savename	Specify this option to store a net service name specified with the <code>-connect</code> option in the Global Data Services catalog, rather than the connect descriptor mapped to that net service name.

Usage Notes

- You must connect to the Global Data Services catalog database as a user with the pool administrator privileges, the GSMUSER database account, using the `connect` command before running the `add brokerconfig` command. You should use the `CONNECT` command to connect to the GSMUSER for the database that you are adding the broker configuration for.
- If a GDS pool already has databases or another configuration, an error is returned. If `-region` is specified, it defines only the region of primary database. If there is more than one region in catalog, GDS region property of standbys will be unassigned. The user will have to use `modify database` to specify GDS region.

Examples

Add the Oracle Data Guard broker configuration for the DB1 database to the Global Data Services pool MYREADERFARM and the WEST region.

```
GDSCTL> add brokerconfig -connect 192.168.1.1:1521:sid -region west -
gdspool myreaderfarm
```

Exceptions or Error Codes

GDSCTL returns the errors listed below if you use this command incorrectly.

Table C-2 GDSCTL add brokerconfig Exceptions or Error Codes

Exception	Description
ERROR-44866	A pool can only contain one Data Guard broker configuration. If a Global Data Services pool already contains an Oracle Data Guard broker configuration, then GDSCTL returns error 44866 because a database must be added using Oracle Data Guard in this case.

add cdb

Add a cdb to the shard catalog.

Syntax

```
add cdb -connect connect_identifier
        [-pwd gsmrootuser_pwd]
        [-savename]
        [-cpu_threshold cpu]
        [-disk_threshold disk]
```

```
[-rack rack_id]
[-force]
```

Options

Table C-3 GDSCTL add cdb Options

Option	Description
<code>-connect connect_identifier</code>	Specify an Oracle Net connect descriptor or net service name that resolves to a connect descriptor for the database being added as the shard.
<code>-pwd gsmrootuser_pwd</code>	Enter the GSMROOTUSER password. If not specified, the user is prompted for the password.
<code>-savename</code>	Store in the shard catalog a net service name specified with the <code>-connect</code> option rather than the connect descriptor mapped to that net service name.
<code>-force</code>	If specified, the existing GDS and sharding configuration on the shard and in the shard catalog with information about this shard will be rewritten.
<code>-cpu_threshold cpu</code>	Specify the CPU Utilization percentage threshold.
<code>-disk_threshold disk</code>	Specify the average latency in milliseconds of a synchronous single-block read.
<code>-rack rack_id</code>	Specify an identifier of a rack (hardware cabinet), or another physical grouping of nodes with similar availability characteristics. If specified, GDS will enforce that databases that contain replicated data are not placed in the same rack. If this is not possible an error is raised.

Usage Notes

ADD CDB adds metadata about a CDB to a sharding catalog. This command is only necessary if you intend to deploy a PDB as a shard with the `-cdb` option in the ADD SHARD command. CDBs can support multiple PDB shards from different sharded databases; however, this support is limited to only one PDB shard from a given sharded database for each CDB.

Examples

Adds a CDB called db11 to the shard catalog.

```
GDSCTL> add cdb -connect db11 -pwd gsmrootuser_pwd
```

add credential

Adds a credential which can be used by the remote scheduler agent to execute shard jobs.

Syntax

```
add credential -credential credential_name
               -osaccount account_name
               -ospassword password
               [-windows_domain domain_name]
```

Options

Table C-4 GDSCTL add credential Options

Option	Description
<code>-credential <i>credential_name</i></code>	Specify the name of the credential to add.
<code>-osaccount <i>account_name</i></code>	Specify the operating system account which will be used for remote jobs.
<code>-ospassword <i>password</i></code>	Specify the corresponding password for the account.
<code>-windows_domain <i>domain_name</i></code>	If a Windows account has been specified, specify the corresponding domain name for that account.

Usage Notes

This command adds a credential which will be used to execute jobs on sharded hosts in response to administrative commands. The operating system account may be any valid account on the remote host which is in the OSDBA group; the account does not need to be enabled for interactive login unless it is used for other purposes. A specific non-interactive account may be created for use with the remote scheduler, if desired. The OS password must be a valid and current password for the specified account.

If the specified credential already exists, the command returns an error.

Examples

Add a credential named `east_region_cred`.

```
GDSCTL> add credential -credential east_region_cred -osaccount
agent_user
-ospassword password
```

add database

Adds databases to a Global Data Services region and Global Data Services pool.

Syntax

```
add database -connect connect_identifier
               [-region region_name]
               [-gdspool gdspool_name]
               [-pwd password]
               [-savename]
               [-cpu_threshold cpu]
               [-disk_threshold disk]
```

Options

Table C-5 GDSCTL add database Options

Option	Description
<code>-connect connect_identifier</code>	Specify an Oracle Net connect descriptor or net service name that resolves to a connect descriptor for the database being added.
<code>-cpu_threshold cpu</code>	Specifies CPU Utilization percentage threshold.
<code>-disk_threshold disk</code>	Specifies the average latency in milliseconds of a synchronous single-block read.
<code>-gdspool gdspool_name</code>	The Global Data Services pool to which the database belongs.
<code>-pwd password</code>	The password for the GSMUSER. If <code>-pwd</code> is not specified, then you are prompted for the password.
<code>-region region_name</code>	The Global Data Services region to which the database belongs.
<code>-savename</code>	Specify this option to store a net service name specified with the <code>-connect</code> option in the Global Data Services catalog, rather than the connect descriptor mapped to that net service name.

Usage Notes

- You must connect to the Global Data Services catalog database as a user with the pool administrator privileges, using the `connect` command before running this command.
- If `-savename` is *not* specified, then GDSCTL replaces what you specify for the net service name with the full connection string before saving the configuration to the catalog.
- The default for GDSCTL is for `autovncr` to be enabled for the catalog. If `autovncr` has been disabled for the catalog, before configuring Global Data Services pools and adding databases to the Global Data Services configuration, the nodes where those databases run must be part of the valid node checking for registration (VNCR) list for database registration. Use the `add invitednode (add invitedsubnet)` command to define the valid nodes.

Example

Adds database DB1 to the WEST region and Global Data Services pool MYREADERFARM.

```
GDSCTL> add database -connect 127.0.0.1:1521:db1 -region west -gdspool
myreaderfarm
```

Adds a database using `myalias` instead of the IP address connection string.

```
GDSCTL> add database -connect myalias -gdspool myreaderfarm
```

Exceptions or Error Codes

GDSCTL returns the errors listed below if you use this command incorrectly.

Table C-6 GDSCTL add database Exceptions or Error Codes

Exception	Description
ERROR-44866	If a pool already contains an Oracle Data Guard broker configuration, then GDSCTL returns an error; you must add databases using Oracle Data Guard in this case. That is, if a pool contains an Oracle Data Guard broker configuration, then additional databases can only be added to the pool by adding them to that Data Guard broker configuration.
ERROR-44868	If the database being added is part of a Oracle Data Guard broker configuration, then GDSCTL returns an error; you must use the add brokerconfig command in this case.

add file

Adds the contents of a file to the catalog which can be used by subsequent GDSCTL commands.

Syntax

```
add file -file file_name
        -source local_filename
```

Options

Table C-7 GDSCTL add file Options

Option	Description
-file <i>file_name</i>	Specify the name of the file object to add.
-source <i>local_filename</i>	Specify an operating system file name specifying a file local to the machine running GDSCTL.

Usage Notes

This command creates a named file object in the catalog and associates the contents of an operating system file with that object by opening the file and storing its contents in the catalog. If the contents of the operating system file change, the MODIFY FILE command can be used to reload the contents into the catalog.

If the specified file object already exists, the command returns an error.

Examples

Add a file named `east_region_db_params` from the local source file `/tmp/dbca_params.txt`

```
GDSCTL> add file -file east_region_db_params -source /tmp/
dbca_params.txt
```

add gdsPOOL

Adds a Global Data Services pool to the Global Data Services framework.

Syntax

```
add gdsPOOL -gdsPOOL gdsPOOL_name_list
            [-users user_list]
```

Options

Table C-8 GDSCTL add gdsPOOL Options

Option	Description
<code>-gdsPOOL <i>gdsPOOL_name_list</i></code>	A comma-delimited list of Global Data Services pool names. A Global Data Services pool must have a unique name within its GDS configuration. If you do not specify a name for the pool when you create it, then the name defaults to <code>oradbPOOL</code> . The pool name can be up to 30 bytes long and can be any valid identifier (an alphabetical character followed by zero or more alphanumeric ASCII characters or the underscore (<code>_</code>)).
<code>-users <i>user_list</i></code>	A comma-delimited list of users that are granted the pool administrator role.

Usage Notes

- A default GDS pool, `DBPOOLORA`, will be created automatically when a GDS catalog is created using [create gdsCatalog](#).
- You must connect to the Global Data Services catalog database as a user with the Global Data Services administrator privileges, using the [connect](#) command before running this command.
- The default for GDSCTL is for `autoVnCr` to be enabled for the catalog. If `autoVnCr` has been disabled for the catalog, then before configuring Global Data Services pools and adding databases to the Global Data Services configuration, the nodes where those databases run must be part of the valid node checking for registration (VNCR) list for database registration. Use the [add invitednode \(add invitedsubnet\)](#) command to define the valid nodes.

Example

Add a Global Data Services pool named `MYREADERFARM` to the configuration:

```
GDSCTL> add gdsPOOL -gdsPOOL myreaderfarm
```

add gsm

Adds a global service manager to the Global Data Services framework.

Syntax

```
add gsm -gsm gsm_name
        -catalog connect_id
        [-pwd password]
        [-wpwd password]
        [-region region_name]
        [-localons ons_port]
        [-remoteons ons_port]
        [-listener listener_port]
        [-endpoint gsmendpoint]
        [-remote_endpoint remote_endpoint]
        [-trace_level level]
```

Options

Table C-9 GDSCTL add gsm Options

Option	Description
-catalog <i>connect_id</i>	Specify the connect identifier for the Global Data Services catalog database. If a network service name is specified, it must be resolvable by the local naming method to a connect descriptor that allows the global service manager being added to connect to the catalog database.
-endpoint <i>gsmendpoint</i>	Specifies the protocol address that the global services manager listens on for client connection requests. If you use this option, the value that you specify overrides the default endpoint.
-gsm <i>gsm_name</i>	Specify the name of the global service manager that you want to add. If you do not specify a name, then GDSCTL uses the current global service manager name for the session (specified with the command set gsm).
-listener <i>listener_port</i>	Specify the listener port. The default port is 1522.
-localons <i>ons_port</i>	Specify the local ONS port. If you do not specify this option, then GDSCTL uses the default ONS port (which is 6123 on most platforms).
-pwd <i>password</i>	Specify the password for the <code>GSMCATUSER</code> . If you do not specify a password, then you are prompted to enter one.

Table C-9 (Cont.) GDSCTL add gsm Options

Option	Description
<code>-region <i>region_name</i></code>	Specify the region to which the global service manager belongs. The value for <i>region_name</i> must match the name of an existing Global Data Services region. If you do not specify a region, then GDSCTL adds the global service manager without assigning a region.
<code>-remote_endpoint <i>remote_endpoint</i></code>	Specifies the protocol address that is used by the global service manager to receive database registration requests and communicate with other global service managers in the configuration. If you use this option, the value that you specify overrides the default endpoint.
<code>-remoteons <i>ons_port</i></code>	Specify the remote ONS port. If you do not specify this option, then GDSCTL uses the default ONS port (which is 6234 on most platforms).
<code>-trace_level <i>level</i></code>	Specify the global service manager trace level (to be used as directed by Oracle Support Services).
<code>-wpwd <i>password</i></code>	Specify a password to protect the global service manager wallet. If a wallet password is not specified, a system-generated password is used instead. Note that if a password is specified with this option, the wallet cannot be modified without supplying that password.

Usage Notes

- You must specify the Global Data Services catalog database when using this command.
- You must run this command, locally, on the computer where you want to add the global service manager.
- You must have operating system privileges on the computer where you want to add the global service manager to run this command.
- When you run this command, GDSCTL connects to the Global Data Services catalog as the GSMCATUSER user and prompts you for the GSMCATUSER password.

Example

Add a global service manager named gsm1, specifying the location of the Global Data Services catalog database, DB1.

```
GDSCTL> add gsm -gsm gsm1 -catalog 127.0.0.1:1521:db1
```

add invitednode (add invitedsubnet)

Adds host address or subnet information to the valid node checking for registration (VNCR) list in the catalog, before starting the first global service manager, by establishing a direct connection to the Global Data Services catalog database.

Syntax

```
add {invitednode | invitedsubnet}
    [-group group_name]
    [-catalog catalog_dbname [-user user_name/password]]
    vncr_id
```

Options

Table C-10 GDSCTL add invitednode (add invitedsubnet) Options

Option	Description
<code>-catalog <i>catalog_dbname</i></code>	Specify the Global Data Services catalog database net alias or connect string. If you enter an invalid address or connect string, then GDSCTL uses the pre-established connection created with the connect command.
<code>-group <i>group_name</i></code>	Specify an alias which defines a group of invited nodes. This alias can be referenced in other commands related to invited nodes.
<code>-user <i>user_name[/password]</i></code>	Specify the user credentials for the Global Data Services administrator in the catalog database. If you do not specify a user or a password, then GDSCTL prompts you this information.
<code><i>vncr_id</i></code>	Specify the list of nodes that can register with the global service manager. The list can include host names or CIDR notation for IPv4 and IPv6 addresses. The wildcard format (*) is supported for IPv4 addresses. The presence of a host name in the list results in the inclusion of all IP addresses mapped to the host name. The host name should be consistent with the public network interface.

Usage Notes

- You must connect to the Global Data Services catalog database as a user with the pool administrator privileges, using the [connect](#) command before running this command.
- The default for GDSCTL is that `autovncr` is enabled for the catalog. If `autovncr` has been disabled for the catalog, before configuring Global Data Services pools and adding databases to the Global Data Services configuration, then the nodes where those databases run must be part of the valid node checking for registration (VNCR) list for database registration. Use the [add invitednode \(add invitedsubnet\)](#) command to define the valid nodes.
- VNCR enables or denies access from specified IP addresses to Oracle services. See *Oracle Database Net Services Administrator's Guide* for more information about VNCR.

Examples

Add the netmask 255.255.255.248 to the catalog.

```
GDSCCTL> add invitednode 255.255.255.248
```

Add the server east1.example.com to the catalog in the alias group EAST_SRV.

```
GDSCCTL> add invitednode east1.example.com
```

Add the server east2.example.com to the catalog in the alias group EAST_SRV.

```
GDSCCTL> add invitednode east2.example.com
```

add region

Adds a region to a Global Data Services framework or an Oracle Sharding configuration.

Syntax

```
add region -region region_list
           [-buddy region_name]
```

Options

Table C-11 GDSCCTL add region Options

Option	Description
-buddy <i>region_name</i>	Specify the name of the buddy region.
-region <i>region_list</i>	Specify a comma-delimited list of Global Data Services region names. A Global Data Services region should have a name that is unique within the corresponding Global Data Services configuration. If no name is specified at the first region creation time, the default name, oraregion, is given to the region. The region name can be up to 30 characters long and can be any valid identifier - an alphabetical character followed by zero or more alphanumeric ASCII characters or underscore (_).

Usage Notes

- When the Global Data Services catalog is created using the [create gdscatalog](#) command, the default REGIONORA region is created automatically.
- You must connect to the Global Data Services catalog database as a user with the Global Data Services administrator privileges, using the command [connect](#) before running this command

Example

Add two Global Data Services regions, EAST and WEST to the current configuration:

```
GDSCTL> add region -region east,west
```

add service

Adds a global service to a Global Data Services pool.

Syntax

```
add service
    [-gdspool gdspool_name]
    -service service_name
    (-preferred_all | (-preferred dbname_list [-available
dbname_list]))
    [-locality {ANYWHERE | LOCAL_ONLY [-region_failover]}]
    [-role {PRIMARY | PHYSICAL_STANDBY [-failover_primary] |
    LOGICAL_STANDBY | SNAPSHOT_STANDBY}]
    [-lag {lag_value | ANY}]
    [-notification {TRUE | FALSE}]
    [-rlbgoal {SERVICE_TIME | THROUGHPUT}]
    [-dtp {TRUE | FALSE}]
    [-sql_translation_profile stp_name]
    [-clbgoal {SHORT | LONG}]
    [-tafpolicy {BASIC | NONE | PRECONNECT}]
    [-policy policy]
    [-failovertype {NONE | SESSION | SELECT | TRANSACTION |
    AUTO}]
    [-failovermethod {NONE | BASIC}]
    [-failoverretry failover_retries]
    [-failoverdelay failover_delay]
    [-edition edition_name]
    [-commit_outcome {TRUE | FALSE}]
    [-retention retention_seconds]
    [-session_state {DYNAMIC | STATIC | AUTO}]
    [-replay_init_time replay_init_time]
    [-pdbname pdbname]
    [-drain_timeout]
    [-stop_option {NONE,IMMEDIATE, TRANSACTIONAL}]
    [-failover_restore {NONE|LEVEL1|AUTO}]
    [-table_family family]
```

Options

Table C-12 GDSCTL add service Options

Option	Description
<code>-available dbname_list</code>	Specify a comma-delimited list of available databases on which the service runs if the preferred databases are not available. You <i>cannot</i> specify a list of available instances, only databases. You can use the modify service command with the <code>-server_pool</code> parameter to specify instance-level preferences. The list of available databases must be mutually exclusive with the list of preferred databases. You <i>cannot</i> use this option with the <code>-preferred_all</code> option.
<code>-clbgoal {SHORT LONG}</code>	Connection Load Balancing Goal. Use a value of <code>SHORT</code> for this parameter for run-time load balancing, or if using an integrated connection pool. Use a value of <code>LONG</code> for this parameter for long running connections, such as batch jobs, that you want balanced by the number of sessions for each node for the service. The default value for this option, if not specified, is <code>SHORT</code> .
<code>-commit_outcome {TRUE FALSE}</code>	Enable Transaction Guard; when set to <code>TRUE</code> , the commit outcome for a transaction is accessible after the transaction's session fails due to a recoverable outage.
<code>-drain_timeout</code>	Set drain time in seconds.
<code>-dtp {TRUE FALSE}</code>	Indicates whether Distributed Transaction Processing should be enabled for this service. This service can either be a service in a policy-managed database or a preferred service on a single node in an administrator-managed database.
<code>-edition edition_name</code>	Specify the initial session edition of the service. When an edition is specified for a service, all subsequent connections that specify the service use this edition as the initial session edition. However, if a session connection specifies a different edition, then the edition specified in the session connection is used for the initial session edition. GDSCTL does not validate the specified edition name. During connection, the connect user must have <code>USE</code> privilege on the specified edition. If the edition does not exist or if the connect user does not have <code>USE</code> privilege on the specified edition, then an error is raised.
<code>-failover_primary</code>	If you set the <code>-role</code> option to <code>PHYSICAL_STANDBY</code> , then you can use this option to enable the service for failover to the primary database.

Table C-12 (Cont.) GDSCTL add service Options

Option	Description
<code>-failoverdelay <i>failover_delay</i></code>	For Application Continuity and TAF, this parameter specifies the time delay (in seconds) between reconnect attempts for each incident at failover.
<code>-failovermethod {NONE BASIC}</code>	TAF failover method (for backward compatibility only). If the failover type (<code>-failovertype</code>) is set to a value other than <code>NONE</code> , then you should choose <code>BASIC</code> for this parameter.
<code>-failoverretry <i>failover_retries</i></code>	For Application Continuity and TAF, this parameter determines the number of attempts to connect after an incident.
<code>-failovertype {NONE SESSION SELECT TRANSACTION}</code>	Specify the failover type. To enable Application Continuity for Java, set this parameter to <code>TRANSACTION</code> . To enable Transparent Application Failover (TAF) for OCI, set this parameter to <code>SELECT</code> or <code>SESSION</code> .
<code>-gdspool <i>gdspool_name</i></code>	Specify the name of the Global Data Services pool to which you want to add a service. If the pool name is not specified and there is only one <code>gdspool</code> with access granted to the user, then this the <code>gdspool</code> with access granted is used as the default <code>gdspool</code> .
<code>-lag {<i>lag_value</i> ANY}</code>	Specify the lag for the service in seconds. You can use the keyword <code>ANY</code> to indicate that there is no upper threshold on the lag time. This parameter specifies the maximum lag that a provider of this service may have. The service cannot be provided by a database whose lag exceeds this value. The default value for <code>lag</code> , if not specified, is <code>ANY</code> .
<code>-locality {ANYWHERE LOCAL_ONLY}</code>	Specify the service region locality. If you do not specify this option, then GDSCTL uses the default value of <code>ANYWHERE</code> for the service.
<code>-notification {TRUE FALSE}</code>	Enable Fast Application Notification (FAN) for OCI connections.
<code>-pdbname <i>pdb_name</i></code>	Specify the pluggable database name.
<code>-policy {AUTOMATIC MANUAL}</code>	Specify the management policy for the service. If you specify <code>AUTOMATIC</code> (the default), then the service automatically starts when the database restarts, either by a planned restart or after a failure. Automatic restart is also subject to the service role. If you specify <code>MANUAL</code> , then the service is never automatically restarted upon planned restart of the database. A <code>MANUAL</code> setting does not prevent the global service manager from monitoring the service when it is running and restarting it if a failure occurs.

Table C-12 (Cont.) GDSCTL add service Options

Option	Description
<code>-preferred dbname_list</code>	Specify a comma-delimited list of preferred databases on which the service runs. You <i>cannot</i> specify preferred instances, only databases. You can use the modify service command to specify instance-level preferences. The list of preferred databases must be mutually exclusive with the list of available databases. You <i>cannot</i> use this option with the <code>-preferred_all</code> option.
<code>-preferred_all</code>	Specifies that all the databases in the Global Data Services pool are preferred databases. Any databases you later add to the pool are configured as preferred databases for this service. You <i>cannot</i> use this option with the <code>-preferred</code> and <code>-available</code> options.
<code>-region_failover</code>	Indicates that the service is enabled for region failover. You can only use this option when you specify <code>LOCAL_ONLY</code> for the <code>-locality</code> option.
<code>-replay_init_time replay_init_time</code>	For Application Continuity, this parameter specifies the time (in seconds) after which replay cannot be initiated. The default value is 300 seconds.
<code>-retention retention_seconds</code>	If <code>commit_outcome</code> is set to <code>TRUE</code> , then this parameter determines the amount of time (in seconds) that the commit outcome is retained in the database.
<code>-rlbgoal {SERVICE_TIME THROUGHPUT}</code>	Run-time Load Balancing Goal (for the Load Balancing Advisory). Set this parameter to <code>SERVICE_TIME</code> to balance connections by response time. Set this parameter to <code>THROUGHPUT</code> to balance connections by throughput. If you do not use this option, then the value defaults to <code>SERVICE_TIME</code> for the run-time load balancing goal.
<code>-role {[PRIMARY] [PHYSICAL_STANDBY] [-failover_primary] [LOGICAL_STANDBY] [SNAPSHOT_STANDBY]}</code>	Specify the database role that the database must be for this service to start on that database. This applies only to Global Data Services pools that contain an Oracle Data Guard broker configuration. See Also: <i>Oracle Data Guard Concepts and Administration</i> for more information about database roles

Table C-12 (Cont.) GDSCTL add service Options

Option	Description
<code>-service service_name</code>	<p>Specify the name of the global service.</p> <p>The service name specified in the <code>add service</code> command can be domain qualified (for example, <code>sales.example.com</code>) or not (for example, <code>sales</code>). If the specified name is not domain qualified, the service is created with the default domain name <code><GDS_pool_name>.<GDS_configuration_name></code>, however the shorter non-domain qualified name can be used with subsequent <code>gdsctl</code> commands to manage the service. If the specified name is domain qualified, the full domain qualified service name must be used in all subsequent <code>gdsctl</code> commands used to manage the service.</p> <p>A global service name must be unique within a GDS pool and when qualified by domain, must also be unique within a GDS configuration. A global service cannot be created at a database if a local or global service with the same name already exists at that database.</p> <p>A global service name can contain alphanumeric characters, underscore (<code>_</code>), and period (<code>.</code>). The first character must be alphanumeric. The maximum length of a global service name is 64 characters. The maximum length of a domain qualified global service name is 250 characters.</p> <p>An Oracle Net connect descriptor used to connect to a global service must contain a domain qualified service name.</p>
<code>-session_state {DYNAMIC STATIC}</code>	<p>For Application Continuity, this parameter specifies whether the session state that is not transactional is changed by the application. A setting of <code>DYNAMIC</code> is recommended for most applications.</p>
<code>-sql_translation_profile stp_name</code>	<p>Use this option to specify a SQL translation profile for a service that you are adding after you have migrated applications from a non-Oracle database to an Oracle database.</p> <p>This option corresponds to the SQL translation profile parameter in the <code>DBMS_SERVICE</code> service attribute.</p> <p>Notes:</p> <ul style="list-style-type: none"> • Before using the SQL translation feature, you must migrate all server-side application objects and data to the Oracle database. • Use the command config service to display the SQL translation profile. <p>See Also: <i>Oracle Database SQL Translation and Migration Guide</i> for more information about SQL translation</p>
<code>-stop_option</code>	<p>Set the default stop option to <code>NONE</code>, <code>IMMEDIATE</code>, or <code>TRANSACTIONAL</code></p>

Table C-12 (Cont.) GDSCTL add service Options

Option	Description
<code>-table_familyfamily</code>	<p>Specifies the name of the table family as a property of the service. This parameter takes one of the table family values (root table schema.name) as shown in the CONFIG TABLE FAMILY output.</p> <p>If the schema name or the table name is case-sensitive, use two-level quotes (single quotes outside, double quotes inside) around the whole string, for example, <code>'"TESTUSER1.Customers6"'</code>. No quotes are needed if neither name is case sensitive.</p> <p>If this parameter is not specified, but there is currently only one table family, the service created with the <code>add service</code> command is automatically associated with that table family.</p>
<code>-tafpolicy {BASIC NONE }</code>	TAF policy specification (for administrator-managed databases only).

Usage Notes

Database-specific options cannot be set at this level. The `modify service` command must be used to set database-specific options.

One of `-preferred_all` or `-preferred` must be specified. If `-preferred_all` is specified, then all databases in the pool are preferred for this global service (databases inserted into the pool will also have this global service added).

In Oracle Sharding, note that when there is no `table_family` parameter specified, the service is not associated with any table family, and the value of the property is set to NULL. This is the case for user-defined and composite sharding, where there is always only one table family, and can also be the case when there is only one table family in system-managed sharding. When a table family is deleted (that is, the root table of the table family is dropped) the `table_family` property of the service is reset to NULL.

Examples

Add a service named `sales_report` to the Global Data Services pool MYREADERFARM with a value of ANYWHERE for the locality.

```
GDSCTL> add service -gdspool myreaderfarm -service sales_report -locality
ANYWHERE
```

Add a service named `daily_sales_rept` to the Global Data Services pool MYDGPOOL with preferred instance set to DB1 and the available instances set to DB3 and DB4. The service should use the basic transaction failover policy.

```
GDSCTL> add service -gdspool mydgpool -s daily_sales_rept -preferred db1
-avaialable db3,db4 -tafpolicy BASIC
```

In a system-managed sharded database, the table family ID parameter is specified as a property of the service.

```
GDSCTL> add service -gdspool shdpool -table_family sales.customer -
service sales -preferred_all -locality ANYWHERE
```



See Also:

[Creating a Global Service](#)

add shard

Add a shard to the shard catalog.

Syntax

```
add shard -connect connect_identifier
          [-pwd password]
          [-savename]
          [-region region_name]
          [-force]
          [-cdb cdb_name]
          [-cpu_threshold cpu]
          [-disk_threshold disk]
          [(-shardgroup shardgroup_name | -shardspace shardspace_name)]
          [-deploy_as {PRIMARY | STANDBY | ACTIVE_STANDBY}]
          [-rack rack_id]
          [-replace old_db_name]
          [-gg_service (http|https):ogg_host:sm_port/GGHOME_directory]
```

Options

Table C-13 GDSCTL add shard Options

Option	Description
<code>-connect <i>connect_identifier</i></code>	Specify an Oracle Net connect descriptor or net service name that resolves to a connect descriptor for the database being added as the shard.
<code>-pwd <i>password</i></code>	Enter the GSMUSER password. If not specified, the user is prompted for the password.
<code>-savename</code>	Store in the shard catalog a net service name specified with the <code>-connect</code> option rather than the connect descriptor mapped to that net service name.
<code>-region <i>region_name</i></code>	Specify the GDS region that this shard belongs to. This parameter is only valid for user-defined sharding. For other sharing methods it is specified per shardgroup.

Table C-13 (Cont.) GDSCTL add shard Options

Option	Description
<code>-force</code>	If specified, the existing GDS and sharding configuration on the shard and in the shard catalog with information about this shard will be rewritten.
<code>-cdb <i>cdb_name</i></code>	If this parameter is used, the shard must be a PDB and the CDB must already exist in the catalog.
<code>-cpu_threshold <i>cpu</i></code>	Specify the CPU Utilization percentage threshold.
<code>-disk_threshold <i>disk</i></code>	Specify the average latency in milliseconds of a synchronous single-block read.
<code>{-shardgroup <i>shardgroup_name</i> -shardspace <i>shardspace_name</i>}</code>	Specify the name of the shardgroup or shardspace that this shard is being added to. Use <code>-shardspace</code> when using this command in a user-defined sharding configuration. Use <code>-shardgroup</code> with system-managed and composite sharding configurations.
<code>-deploy_as {PRIMARY STANDBY ACTIVE_STANDBY}</code>	Specify the role that is assigned to a shard added to the shardgroup after deployment. This parameter is only used with Data Guard replication. The specified role will be assigned to the shard database after deployment. The valid values are: <ul style="list-style-type: none"> PRIMARY – the shard should be deployed as the primary database STANDBY – the shard should be deployed as a Data Guard standby (mounted) ACTIVE_STANDBY – the shard should be deployed as an Active Data Guard standby If the parameter is not specified, the default value is STANDBY
<code>-rack <i>rack_id</i></code>	Specify an identifier of a rack (hardware cabinet), or another physical grouping of nodes with similar availability characteristics. If specified, GDS will enforce that databases that contain replicated data are not placed in the same rack. If this is not possible an error is raised.
<code>-replace <i>old_db_name</i></code>	This parameter specifies <code>db_unique_name</code> of the old shard when replacing it. The existing parameters of the <code>ADD SHARD</code> command (such as <code>connect</code>) must refer to attributes for the new (replacement) shard. This parameter is not supported in an Oracle GoldenGate environment.
<code>-gg_service (<i>http https</i>):<i>ogg_host:sm_port</i>/ <i>deployment</i></code>	This parameter is mandatory for Oracle GoldenGate replication and specifies the URI for the GoldenGate Admin Server that will manage the GoldenGate replication at this shard. The format will be as follows Example: <code>shard1.example.com:9005/shard1.</code>

Usage Notes

Before running `add shard`, you must validate the shard by running the `validateShard` procedure as described in *Using Oracle Sharding*

- The shard will become part of the sharded database after a `DEPLOY` command is executed, except in the case of `-replace`.
- Any databases added to a sharding configuration using `ADD SHARD` must not have ever been deployed as a shard in another configuration (unless `-replace` is specified). Re-adding a previously deployed shard will cause the `ADD SHARD` command to succeed, but the shard will be unable to successfully deploy and register with the shard director (GSM) when the `DEPLOY` command is eventually run.
- `ADD SHARD` only registers the database (shard) with GDS. Replication is not configured on a newly added database and data from other databases is not distributed to it until `DEPLOY` is run.
- With Data Guard replication, a shard can be added as a standby to a preexisting Data Guard configuration. There is no need to re-shard the data. It is expected that the shard being added is in a the correct state for configuration; the standbys should be cloned from the primary and have the same DBID. When you run `DEPLOY`, the existing primary and standby databases are matched with each other, using the DBID to form a broker configuration. If the broker has not been configured, it is configured, otherwise it is validated that it has been configured correctly. Once the broker is configured, Data Guard does its work, and it should be able to perform catch-up on the standbys if needed before bringing them online.
- See *Working with Oracle GoldenGate Sharding in the Fusion Middleware Using the Oracle GoldenGate Microservices Architecture* guide for more information about using Oracle GoldenGate with Oracle Sharding.
- When using the `-replace` parameter, see *Using Oracle Sharding* for more information about its usage.

Examples

Add the shard to shardgroup GROUP1 of the DB11 database.

```
GDSCTL> add shard -connect db11 -shardgroup group1
```

Replace shard SH1 with database DB11.

```
GDSCTL> add shard -replace sh1 -connect db11
```

add shardgroup

Add a shardgroup to a shardspace.

Syntax

```
add shardgroup -shardgroup shardgroup_name  
                [-region region_name]
```

```
[-shardspace shardspace_name]
[-deploy_as {PRIMARY | STANDBY | ACTIVE_STANDBY}]
[-repfactor number]
```

Options

Table C-14 GDSCTL add shardgroup Options

Option	Description
<code>-shardgroup <i>shardgroup_name</i></code>	Specify the name of the shardgroup. The name must be unique across all shardspaces. The shardgroup name can be up to 30 characters long and can be an alphabetical character followed by zero or more alphanumeric ASCII characters or an underscore (_).
<code>-region <i>region_name</i></code>	Specify the name of the region. If not specified, a default region will be used.
<code>-shardspace <i>shardspace_name</i></code>	Specify the name of the shardspace to which to add the shardgroup.
<code>-deploy_as {PRIMARY STANDBY ACTIVE_STANDBY}</code>	Specify the role that is assigned to a shard added to the shardgroup after deployment. This parameter is only used with Data Guard replication. The valid values are: If the parameter is not specified, the default value is STANDBY <ul style="list-style-type: none"> PRIMARY – the shard should be deployed as the primary database STANDBY – the shard should be deployed as a Data Guard standby (mounted) ACTIVE_STANDBY – the shard should be deployed as an Active Data Guard standby
<code>-repfactor <i>number</i></code>	Specify the replication factor - the number of replicas for each piece of data stored in this shardgroup. This parameter can only be used with Oracle GoldenGate replication and is mandatory unless the default value of replication factor was specified in CREATE SHARDCATALOG command. This parameter does not apply to user-defined sharding because GoldenGate does not support that sharding methods.

Usage Notes

This command can only be used with system-managed or composite sharding, not user-defined sharding.

See *Working with Oracle GoldenGate Sharding in the Fusion Middleware Using the Oracle GoldenGate Microservices Architecture* guide for more information about using Oracle GoldenGate with Oracle Sharding.

Examples

Add the GROUP1 shardgroup in the WEST region within the GOLD shardspace.

```
GDSCTL> add shardgroup -shardgroup group1 -region west -shardspace gold
```


add shardspace

Add a shardspace to the shard catalog.

Syntax

```
add shardspace -shardspace shardspace_name
                [-chunks number]
                [-protectmode dg_protection_mode]
```

Options

Table C-15 GDSCTL add shardspace Options

Option	Description
<code>-shardspace <i>shardspace_name</i></code>	Specify the name of the shardspace. The shardspace name can be up to 30 characters long and can be an alphabetical character followed by zero or more alphanumeric ASCII characters or an underscore (_).
<code>-chunks <i>number</i></code>	Specify the number of unique chunks in the shardspace. The value of <code>-chunks</code> must be greater than 2 times the size of the largest shardgroup in any shardspace. This parameter does not apply to user-defined sharding. All shardgroups in a shardspace have the same number of chunks. If this parameter is not specified, the default number of chunks is determined at the time of execution of the first <code>DEPLOY</code> command and is 120 per database of the shardgroup with the biggest number of databases.
<code>-protectmode <i>dg_protection_mode</i></code>	Specify the Data Guard protection mode: <code>MAXPROTECTION</code> , <code>MAXAVAILABILITY</code> , or <code>MAXPERFORMANCE</code> (default). This parameter does not apply to Oracle GoldenGate replication.

Usage Notes

The command is applicable to user-defined sharding, composite sharding that assumes multiple shardspaces, and system managed sharding when there are no other shardspaces present in the current configuration.

Examples

Add the GOLD shardspace with Data Guard MAXAVAILABILITY protection mode.

```
GDSCTL> add shardspace -shardspace gold -protectmode maxavailability
```

config

Displays the configuration data for all components defined for the configuration.

Syntax

```
config [-support]
```

Options

Table C-16 GDSCTL config Options

Option	Description
-support	If specified, GDSCTL output displays additional information for support purposes.

Usage Notes

When using the command, it does not matter if the components (except for the catalog database) are started. The configuration data displayed is retrieved from the catalog database.

Example

Display the configuration data for all components defined for the configuration.

```
GDSCTL> config
```

config cdb

Displays properties of a specified CDB.

Syntax

```
config cdb [-cdb cdb_name]
```

Options

Table C-17 GDSCTL config cdb Options

Option	Description
-cdb <i>cdb_name</i>	Specify the name of the cdb.

Examples

Display information about CDB called cdb1.

```
GDSCTL> config cdb -cdb cdb1
```

```
Name: tstdsbyb
Connection string: (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=cdb1host)
(PORT=1521))
(CONNECT_DATA=(SERVICE_NAME=cdb1.example.com)))
SCAN address:
ONS remote port: 0
Disk Threshold, ms: 20
CPU Threshold, %: 75
Version: 18.0.0.0
Rack:
```

config chunks

Displays properties of a specified chunk.

Syntax

```
config chunks [-support]
              ( [-shard shd] | [-shardgroup sh] | [-show_reshard] | [-
cross_shard] )
              ( [-chunk chunk_id] | [-key key [-superkey superkey]] )
```

Options

Table C-18 GDSCTL config chunks Options

Option	Description
-chunk <i>chunk_id</i>	Specify a numeric chunk ID.
-cross_shard	Show cross-shard placement.
-key <i>key</i>	Sharding key
-shard <i>shd</i>	The name of the shard.
-shardgroup <i>sh</i>	The name of the shardgroup.
-show_reshard	Display information about ongoing chunk management operations.
-superkey <i>superkey</i>	Sharding super key. This is only needed for the composite sharding method.
-support	If specified, GDSCTL output displays additional information.

Usage Notes

The `config chunks` command lists all of the database shards and the chunks that they contain. Some chunks are listed more than once if there are standbys that contain replicated chunks.

Examples

The output from `config chunks` is shown below.

```
GDSCCTL> config chunks
```

```

Chunks
-----
Database          From      To
-----
sh1a              1        10
sh1b              1        10

```

config credential

Displays remote credentials currently available for shard jobs.

Syntax

```
config credential [-support]
```

Options

Table C-19 GDSCCTL config credential Options

Option	Description
-support	If specified, GDSCCTL output displays additional information.

Usage Notes

This command displays all existing remote credentials that can be used to execute sharding jobs.

Examples

Display credentials.

```
GDSCCTL> config credential
```

```

Name          Username Windows domain
-----
CREDENTIAL_ONE OraUser
CREDENTIAL_TWO OraUser2

```

config database

Displays the static configuration data stored in the catalog for the specified database.

Syntax

```
config database [-support]
                [-database db_name]
```

Options

Table C-20 GDSCTL config database Options

Syntax	Description
-database <i>db_name</i>	Specify the name of a database. If you do not specify a database name, then GDSCTL displays the configuration data for all databases in the Global Data Services configuration.
-support	GDSCTL output displays additional support information.

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the [connect](#) command before running this command

Examples

Display the static configuration data stored in the catalog for all the databases in the Global Data Services configuration.

```
GDSCTL> config database
```

The `gdsctl config database` command returns information similar to the following:

```
Name      Pool      Status    Region
----      -
dbcat     sales     Ok        east
dbcat1    sales     Ok        west
dbcat3    sales     Ok        west
```

config file

Displays file objects currently available that can be specified in GDSCTL commands.

Syntax

```
config file [-support]
            [-file file_name]
```

Options

Table C-21 GDSCTL config file Options

Option	Description
<code>-file <i>file_name</i></code>	The name of the file object.
<code>-support</code>	If specified, GDSCTL output displays additional information.

Usage Notes

If the specified file object does not exist, the command returns an error.

Example

Display the list of files defined in the catalog database.

```
GDSCTL> config file
Name
-----
dbcfg1
```

config gdspool

Displays the static configuration data that is stored in the catalog for the specified database pool.

Syntax

```
config gdspool [-support]
                [-gdspool gdspool_name]
```

Options

Table C-22 GDSCTL config gdspool Options

Syntax	Description
<code>-gdspool <i>gdspool_name</i></code>	Specify the name of a database pool. If you do not specify a database pool name, then GDSCTL displays the configuration data for all database pools.
<code>-support</code>	GDSCTL output displays additional support information.

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the [connect](#) command before running this command

Example

Display the static configuration data stored in the catalog for all Global Data Services pools.

```
GDSCTL> config gdspool
```

The `gdsctl config gdspool` command returns output similar to the following:

Name	Broker
dbpoolora	No
mkt	No
sales	No
marketing	No

The following command shows the configuration detail of Global Data Services pool `marketing`.

```
GDSCTL> config gdspool -gdspool marketing
```

The above example returns output similar to the following:

```
GDS Pool administrators
-----

Databases
-----
dbcat2
dbcat1
dbcat3

Services
-----
sales_report
sales_analysis
sales_estimation
sales_peragent
sales_global
```

config gsm

Displays the static configuration data stored in the catalog for the specified global service manager.

Syntax

```
config gsm [-gsm gsm_name]
           [-support]
```

Options

Table C-23 GDSCTL config gsm Options

Syntax	Description
<code>-gsm <i>gsm_name</i></code>	Specify the name of a global service manager. If you do not specify a global service manager name, then GDSCTL displays the static configuration data for all global service managers in the cloud.
<code>-support</code>	If specified, GDSCTL output displays additional information.

Usage Notes

You must connect to the catalog database as a user with the Global Data Services administrator privileges, using the [connect](#) command before running this command

Example

Display the static configuration data stored in the catalog for the global service manager `mygsm`:

```
GDSCTL> config gsm -gsm mygsm
```

The `gdsctl config gsm` command returns output similar to the following:

```
Name: mygsm
Endpoint 1: (ADDRESS=(HOST=stcal.us.hq.com) (PORT=1523) (PROTOCOL=tcp))
Endpoint 2: (ADDRESS=(HOST=stcal.us.hq.com) (PORT=1523) (PROTOCOL=tcp))
Local ONS port: 6123
Remote ONS port: 6234
Region: east
Buddy
-----
```

config region

Displays the static configuration data for the specified region.

Syntax

```
config region [-region region_name]
              [-support]
```


Options

Table C-24 GDSCTL config region Options

Syntax	Description
<code>-region <i>gsm_name</i></code>	Specify the name of a global service manager.
<code>-support</code>	If specified, GDSCTL output displays additional information.

Example

Displays the static configuration data for the specified region.

```
GDSCTL> config region -region east
```

Displays the following output:

```
Name                Buddy
----                -
east
```

config sdb

Displays the static configuration data stored in the catalog for the sharded database.

Syntax

```
config sdb [-support]
```

Options

Table C-25 GDSCTL config sdb Options

Option	Description
<code>-support</code>	If specified, GDSCTL output displays additional information.

Examples

The output for `config sdb` is similar to the following.

```
GDSCTL> config sdb

GDS Pool administrators
-----

Replication Type
-----
Data Guard

Shard type
-----
```

```

System-managed

Shard spaces
-----
shardspaceora

Services
-----
oltp_ro_srvc
oltp_rw_srvc

```

config service

Displays the static configuration data stored in the Global Data Services catalog for the specified services that are located in a database pool.

Syntax

```

config service [-gdspool gdspool_name]
               [-service service_name]
               [-support]

```

Options

Table C-26 GDSCTL config service Options

Syntax	Description
<code>-gdspool <i>gdspool_name</i></code>	Specify the name of the database pool that contains the services. If the name is not specified, and there is only one <i>gdspool</i> with access granted to the user, it is used as the default <i>gdspool</i> .
<code>-service <i>service_name</i></code>	Specify a comma-delimited list of service names. If you do not use this option, then GDSCTL displays the configuration data for all services in the specified database pool.
<code>-support</code>	If specified, GDSCTL output displays additional information.

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the [connect](#) command before running this command

Examples

Show all the services in the user's Global Data Services pool:

```
GDSCTL> config service
```

The `gdsctl config service` command returns information similar to the following:

```

Name          Network name          Pool          Started Preferred all
-----

```

```

sales_svc1 sales_svc1.sales.oradbcloud sales Yes
Yes
sales_svc2 sales_svc2.sales.oradbcloud sales NO
Yes
sales_svc3 sales_svc3.sales.oradbcloud sales Yes
Yes
mkt_svc1 mkt_svc1.mkt.oradbcloud mkt NO Yes

```

Display the static configuration data stored in the Global Data Services catalog for sales:

```
GDSCTL> config service -service sales
```

```

Name: sales
Network name: sales.shdpool.oradbcloud
Pool: shdpool
Started: Yes
Preferred all: Yes
Locality: ANYWHERE
Region Failover: No
Role: NONE
Primary Failover: No
Lag: ANY
Runtime Balance: SERVICE_TIME
Connection Balance: LONG
Notification: Yes
TAF Policy: NONE
Policy: AUTOMATIC
DTP: No
Failover Method: NONE
Failover Type: NONE
Failover Retries:
Failover Delay:
Edition:
PDB:
Commit Outcome:
Retention Timeout:
Replay Initiation Timeout:
Session State Consistency:
SQL Translation Profile:
Stop option: NONE
Drain timeout:
Table Family: sales.customer

```

Supported services

```

-----
Database          Preferred Status
-----
shdb              Yes          Enabled
shdc              Yes          Enabled

```

config shard

Displays properties of a specified shard.

Syntax

```
config shard -shard shard_name
              [-support]
```

Options

Table C-27 GDSCTL config shard Options

Option	Description
-shard <i>shard_name</i>	Specify the name of the shard.
-support	GDSCTL output displays additional support information.

Examples

```
GDSCTL> config shard
```

Name	Shard Group	Status	State	Region	Availability
den17b	db1	Ok	Deployed	east	ONLINE
den17c	db2	Ok	Deployed	east	READ ONLY

The State column in the results can have the following values:

- **Created:** Indicates that `add shard` or `create shard` was run, but `deploy` has not yet been run for that shard.
- **Replicated:** Indicates that `deploy` was run and the Data Guard broker configuration was created. No other metadata (chunks, for example) are on the shard and the shard has not yet registered with the shard director
- **Sharded:** Indicates that the database has successfully registered with the shard director. Creates chunk metadata for new shards, but does not start any automatic rebalancing. To manually get from Replicated to Sharded and beyond, run `GDSCTL sync -database <shard_name>`. This is what is happening internally in this step.
- **Deployed:** Indicates that all DDL catchup is completed and the shard is ready for operations. At this point, any scheduled chunk moves are begun in the background. A shard can be Deployed without having been rebalanced because rebalancing is a background operation.

config shardgroup

Displays properties of a specified shardgroup.

Syntax

```
config shardgroup [-shardgroup shardgroup_name]
                  [-support]
```

Options

Table C-28 GDSCTL config shardgroup Options

Option	Description
<code>-shardgroup <i>shardgroup_name</i></code>	Specify the name of the shardgroup.
<code>-support</code>	GDSCTL output displays additional support information.

Examples

The `config shardgroup` command generates the following output.

```
GDSCTL> config shardgroup -shardgroup northeast
```

```
Shard Group Chunks Region Shard space
-----
dbs1         10     east   shd1
dbs2         10     east   shd1
```

By specifying a shardgroup, you get the following output.

```
GDSCTL> config shardgroup -shardgroup dbs1
```

```
Shard Group: dbs1
Chunks: 10
Replicas:
Region: east
Shard space: shd1
Shards
-----
Shard Chunks
-----
den17b 10
```

config shardspace

Displays properties of a specified shardspace.

Syntax

```
config shardspace [-shardspace shardspace_name]
                  [-support]
```

Options

Table C-29 GDSCTL config shardspace Options

Option	Description
<code>-shardspace <i>shardspace_name</i></code>	Specify the name of the shardspace. Optional for system-managed sharding.
<code>-support</code>	GDSCTL output displays additional support information.

Usage Notes

The output varies depending on whether the command is issued on a shardspace configured in a user-defined SDB.

Examples

The `config shardspace` command generates the following output

```
GDSCTL> config shardspace
```

```
Shard space Chunks
-----
shd1          10
```

When a shardspace is specified, the output is returned in the following format.

```
GDSCTL> config shardspace -shardspace silver
```

```
Shard Group Region Role
-----
dbs1          east  Primary
dbs2          east  Standby
PROTECTION_MODE Chunks
-----
MaxProtection 10
```

config table family

Displays information about all table families in the sharded database.

Syntax

```
config table family
```

Examples

The `config table family` command generates the following output

```
GDSCTL> config table family
```

```
Schema  Name      ID      Shard Type
-----  -
```

```
sales    customer    1      System
hr       department  25     System
```

config vncr

Displays the static configuration data stored in the catalog for valid node checking for registration (VNCR).

Syntax

```
config vncr [-group group_name]
            [-support]
```

Options

Table C-30 GDSCTL config vncr Options

Syntax	Description
-group <i>group_name</i>	A group alias that defines a group of VNCRs. The same alias can be used in multiple ADD calls.
-support	GDSCTL output displays additional support information.

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the [connect](#) command before running this command

Example

The `config vncr` command returns information similar to the following:

```
GDSCTL> config vncr

Name          Group ID
----          -
192.0.2.1    group_name
```

configure

Sets the GDSCTL parameters.

Syntax

```
configure [-gsmpport port]
          [-timeout seconds]
          [-show]
          [-driver {THIN | OCI}]
          [-resolve {IP | HOSTNAME | QUAL_HOSTNAME}]
          [-log {ALL|OFF|INFO|FINE|FINER|FINEST|SEVERE|WARNING}]
          [-log_file log_file]
```

```

[-gsm gsm_name]
[-showtime ON|OFF]
[-verbose ON|OFF]
[-save_config]
[-gsmdebug (1|0)]
[-spool]
[-width]

```

Options

Table C-31 GDSCTL configure Options

Syntax	Description
<code>-driver THIN OCI</code>	Oracle JDBC driver.
<code>-gsm <i>gsm_name</i></code>	Set current global service manager.
<code>-gsmdebug (1 0)</code>	Global service manager debug mode.
<code>-gsmport <i>port</i></code>	Default global service manager port.
<code>-log {ALL OFF INFO FINE FINER FINEST SEVERE WARNING}</code>	Set the logging level. The default is OFF.
<code>-log_file <i>log_file</i></code>	Set the location of the log file. The default is <code>\$TNS_ADMIN/GDSTL.log</code> .
<code>-resolve IP HOSTNAME QUAL_HOSTNAME</code>	Default host resolution for global service manager endpoint.
<code>-save_config</code>	Store configuration changes to GSM.ORA.
<code>-show</code>	Show the configuration.
<code>-showtime ON OFF</code>	Print time stamps.
<code>-spool</code>	Enable spooling. Warning: prints security-sensitive information to log file.
<code>-timeout <i>seconds</i></code>	Global service manager requests timeout in seconds.
<code>-verbose ON OFF</code>	Enable or disable verbose output. The default value is ON.
<code>-width</code>	Console width in number of characters (default 80).

Example

Set the `mygsm` driver to OCI:

```
configure -driver OCI mygsm
```

connect

Specifies the credentials to administer a global service management environment. Credentials must be specified to perform certain operations using GDSCTL.

Syntax

```
connect [user_name[/password]]@connect_identifier
```


Options

Table C-32 GDSCTL connect Options

Syntax	Description
<i>connect_identifier</i>	Specify an Oracle Net connect descriptor or a net service name that maps to a connect descriptor (for example, a list of global service managers).
<i>password</i>	Specify the password for the specified user. If you do not specify a password, then you are prompted to enter a password. The password is obscured when entered.
<i>user_name</i>	Specify the name of the user to connect as. The user that you specify must have either the Global Data Services administrator or the pool administrator role. If you specify no user name, then you are prompted for a user name.

Usage Notes

You must connect to the catalog database as a user with either the Global Data Services administrator or the pool administrator privileges, depending on which command you want to run after you connect

 **WARNING:**

Specifying a password as a connect command option is a security risk. You can avoid this risk by omitting the password, and entering it only when the system prompts for it.

Examples

Connect as the `gsmadmin` user to the private cloud:

```
GDSCTL> connect gsmadmin@mycloud
Enter password:
```

Connect using a connect descriptor, without specifying a user name and password:

```
GDSCTL> connect (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp) (HOST=myhost)
(PORT=1521)))
Enter username:
```

create catalog

The `create catalog` command is deprecated. Use `create gds catalog` or `create shard catalog` instead. Creates a Global Data Services catalog for global service management in a specific database.

Syntax

```
create catalog -database db_name
               [-user user_name[/password]]
               [-region region_name_list]
               [-gdspool gdspool_name_list]
               [-configname confname]
               [-autovncr {ON | OFF}]
               [-force]
```

Options

Table C-33 GDSCTL create catalog Options

Options	Description
<code>-autovncr {ON OFF}</code>	This option enables (ON) or disables (OFF) autovncr mode. The default value is ON. See the Usage Notes below for important information about this option.
<code>-configname <i>confname</i></code>	Specify the name of the GDS configuration. The default configuration name is ORADBCLOUD. The configuration name can be up to 32 bytes long and can contain an alphabetical character followed by zero or more alphanumeric ASCII characters, '_', or '#' and possibly separated by periods if there are multiple identifiers.
<code>-database <i>db_name</i></code>	Specify the connect identifier for the database in which you want to create catalog.
<code>-force</code>	Rewrites existing global service manager configuration on catalog database.
<code>-gdspool <i>gdspool_name_list</i></code>	Specify a comma-delimited list of database pool names. When you use this option, the specified database pools are created as part of the catalog creation. If you do not specify this option, then GDSCTL creates a default database pool named DBPOOLORA.
<code>-region <i>region_name_list</i></code>	Specify a comma-delimited list of region names. This command creates each region and adds the regions to the catalog. If you do not specify a region, then a default region named REGIONORA is created.

Table C-33 (Cont.) GDSCTL create catalog Options

Options	Description
<code>-user user_name[/password]</code>	Specify a user (and optionally, the password) that has the Global Data Services administrator privileges on the catalog database. If you do not use this option, then GDSCTL prompts you for the name and the password of a user with Global Data Services administrator privileges. If you specify a user name but not the password for the user, then GDSCTL prompts you for the password.

Usage Notes

This command is deprecated in Oracle Database 12c Release 2. Use [create gds catalog](#) or [create shard catalog](#) for your specific environment.

The `create catalog` command generates a pair of PKI public and private keys and stores them in the catalog, along with a fixed string "GSM" that is encrypted with a private key. It uses the `GSMCATUSER` password.

You must have the Global Data Services administrator privileges on the computer where you want to create the Global Data Services catalog.

Auto VNCR is best used in environments with simple private networks where ease of configuration is the most important consideration. To have the highest level of control over which hosts may participate in a GDS configuration, disable Auto VNCR and explicitly add the IP addresses of each database host to the VNCR configuration.

When `-autovncr` is enabled, Oracle attempts to find the host name of the target database when it is validated during `add shard` or `add database` execution. This host is then automatically added to the VNCR list in the catalog as an invited node. This mechanism is not compatible with all network configurations and may not work in the following cases:

- The catalog or global service manager host does not know how to translate the host name discovered on the target database host to an real IP address. This can happen if they have different names in the hosts file or DNS on the catalog or global service manager host, or if they just do not exist on those hosts.
- The target database host has multiple public network addresses, and Oracle chooses an address that is different than the address used when the database registers with the global service manager. This can happen if the host has multiple network cards or has configured virtual network interfaces.
- The database is running Oracle RAC, and other Oracle RAC instances run on a different subnet. This is not a recommended configuration for Oracle RAC. Refer to the Oracle RAC documentation for recommended configurations. With Oracle RAC, Oracle Database connects to a single database host to validate the target, and returns a subnet mask that includes the entire subnet that the host is on. If other instances are on a different subnet, they have no valid VNCR entry, and registration is rejected.

When `-autoVNCR` is not enabled, or, if any of the above cases apply, new hosts should be added manually using [add invitednode](#) ([add invitedsubnet](#)).

Example

Create a Global Data Services catalog for global service management in the database named DB1. Also create the regions EAST and WEST, and the database pool READERFARM.

```
GDSCTL> create catalog -database db1 -region west,east -gdspool readerfarm
```

create gds catalog

Creates a Global Data Services catalog for global service management in a specific database.

Syntax

```
create gds catalog -database db_name
                  [-user user_name[/password]]
                  [-region region_name_list]
                  [-gdspool gdspool_name_list]
                  [-configname confname]
                  [-autovncr {ON | OFF}]
                  [-force]
```

Options

Table C-34 GDSCTL create gds catalog Options

Syntax	Description
-autovncr {ON OFF}	This option enables (ON) or disables (OFF) autovncr mode. The default value is ON. See the Usage Notes below for important information about this option.
-configname <i>confname</i>	Specify the name of the GDS configuration. The default configuration name is ORADBCLOUD. The configuration name can be up to 32 bytes long and can contain an alphabetical character followed by zero or more alphanumeric ASCII characters, '_', or '#' and possibly separated by periods if there are multiple identifiers.
-database <i>db_name</i>	Specify the connect identifier for the database in which you want to create catalog.
-force	Rewrites existing global service manager configuration on catalog database.
-gdspool <i>gdspool_name_list</i>	Specify a comma-delimited list of database pool names. When you use this option, the specified database pools are created as part of the catalog creation. If you do not specify this option, then GDSCTL creates a default database pool named DBPOOLORA.

Table C-34 (Cont.) GDSCTL create gds catalog Options

Syntax	Description
<code>-region <i>region_name_list</i></code>	Specify a comma-delimited list of region names. This command creates each region and adds the regions to the catalog. If you do not specify a region, then a default region named REGIONORA is created.
<code>-user <i>user_name</i>[/<i>password</i>]</code>	Specify a user (and optionally, the password) that has the Global Data Services administrator privileges on the catalog database. If you do not use this option, then GDSCTL prompts you for the name and the password of a user with Global Data Services administrator privileges. If you specify a user name but not the password for the user, then GDSCTL prompts you for the password.

Usage Notes

The `create gds catalog` command generates a pair of PKI public and private keys and stores them in the catalog, along with a fixed string "GSM" that is encrypted with a private key. It uses the `GSMCATUSER` password.

You must have the Global Data Services administrator privileges on the computer where you want to create the Global Data Services catalog.

Auto VNCR is best used in environments with simple private networks where ease of configuration is the most important consideration. To have the highest level of control over which hosts can participate in a GDS configuration, disable Auto VNCR and explicitly add the IP addresses of each database host to the VNCR configuration.

When `-autovncr` is enabled, Oracle attempts to find the host name of the target database when it is validated during `add shard` or `add database` execution. This host is then automatically added to the VNCR list in the catalog as an invited node. This mechanism is not compatible with all network configurations and may not work in the following cases:

- The catalog or global service manager host does not know how to translate the host name discovered on the target database host to an real IP address. This can happen if they have different names in the hosts file or DNS on the catalog or global service manager host, or if they just do not exist on those hosts.
- The target database host has multiple public network addresses, and Oracle chooses an address that is different than the address used when the database registers with the global service manager. This can happen if the host has multiple network cards or has configured virtual network interfaces.
- The database is running Oracle RAC, and other Oracle RAC instances run on a different subnet. This is not a recommended configuration for Oracle RAC. Refer to the Oracle RAC documentation for recommended configurations. With Oracle RAC, Oracle Database connects to a single database host to validate the target, and returns a subnet mask that includes the entire subnet that the host is on. If other instances are on a different subnet, they have no valid VNCR entry, and registration is rejected.

When `-autoVNCR` is not enabled, or, if any of the above cases apply, new hosts should be added manually using `add invitednode` (`add invitedsubnet`).

Example

Create a Global Data Services catalog for global service management in the database named DB1. Also create the regions EAST and WEST, and the database pool READERFARM.

```
GDSTCL> create gds catalog -database db1 -region west,east -gdspool readerfarm
```

create shard

Create a new shard and add it to a shardspace or shardgroup.

Syntax

```
create shard [{-shardgroup shardgroup_name | -shardspace shardspace_name}]
            [-region region_name]
            [-deploy_as {primary | standby | active_standby}]
            [-rack rack_id]
            [-gg_service (http|https):ogg_host:sm_port/deployment
             -gg_password gg_user_password]
            -destination destination_name
            {-credential credential_name |
             -osaccount account_name
             -ospassword password
             [-windows_domain domain_name]}
            [-dbparam db_parameter_file |
             -dbparamfile db_parameter_file]
            [-dbtemplate db_template_file |
             -dbtemplatefile db_template_file]
            [-netparam net_parameter_file |
             -netparamfile net_parameter_file]
            [-serviceuserpassword pwd]
            [-sys_password]
            [-system_password]
```

Options

Table C-35 GDSTCL create shard Options

Option	Description
-credential <i>credential_name</i>	The name of the credential object.
-dbparam <i>db_parameter_file</i>	Specify a file object containing Database Configuration Assistant (DBCA) parameters to use during database creation on the remote machine.
-dbparamfile <i>db_parameter_file</i>	Specify an operating system file name containing Database Configuration Assistant (DBCA) parameters to use during database creation on the remote machine. GDSTCL uses this to open a local file on the machine on which it is running.

Table C-35 (Cont.) GDSCTL create shard Options

Option	Description
<code>-dbtemplate db_template_file</code>	Specify a file object containing Database Configuration Assistant (DBCA) database template information to use during database creation on the remote machine.
<code>-dbtemplatefile db_template_file</code>	Specify an operating system file name containing Database Configuration Assistant (DBCA) database template information to use during database creation on the remote machine. GDSCTL uses this to open a local file on the machine on which it is running.
<code>-deploy_as {PRIMARY STANDBY ACTIVE_STANDBY}</code>	Specify the role that is assigned to a shard added to the shardgroup after deployment. This parameter is only used with Data Guard replication. The specified role will be assigned to the shard database after deployment. The valid values are: <ul style="list-style-type: none"> PRIMARY – the shard should be deployed as the primary database STANDBY – the shard should be deployed as a Data Guard standby (mounted) ACTIVE_STANDBY – the shard should be deployed as an Active Data Guard standby If the parameter is not specified, the default value is STANDBY
<code>-destination destination_name</code>	The name of the remote executable agent through which the database will be created as listed in the <code>ALL_SCHEDULER_EXTERNAL_DESTS</code> view.
<code>-gg_password gg_user_password</code>	Specifies the password for the user account that will be created on the shard database that the Oracle GoldenGate replication processes will use. This parameter is mandatory for Oracle GoldenGate replication.
<code>-gg_service hostname:port/ GGHOME_directory</code>	This parameter is mandatory for Oracle GoldenGate replication and specifies the URI for the GoldenGate Admin Server that will manage the GoldenGate replication at this shard. The format will be as follows Example: <code>https:shard1.example.com:9005/shard1.</code>
<code>-netparam net_parameter_file</code>	Specify a file object containing Net Configuration Assistant (NETCA) parameters to use during network listener setup on the remote machine.
<code>-netparamfile net_parameter_file</code>	Specify an operating system file name containing Net Configuration Assistant (NETCA) parameters to use during network listener setup on the remote machine. GDSCTL uses this to open a local file on the machine on which it is running.
<code>-osaccount account_name</code>	Specify the operating system account which will be used for remote jobs
<code>-ospassword password</code>	Specify the corresponding password for the account name specified in the <code>-osaccount</code> parameter.

Table C-35 (Cont.) GDSCTL create shard Options

Option	Description
<code>-rack rack_id</code>	Specify an identifier of a rack (hardware cabinet), or another physical grouping of nodes with similar availability characteristics. If specified, GDS will enforce that databases that contain replicated data are not placed in the same rack. If this is not possible an error is raised.
<code>-region region_name</code>	Specify the GDS region database, catalog, shard, shardgroup, or shard director (global service manager) that this shard belongs to. This parameter is only valid for user-defined sharding. For other sharing methods it is specified per shardgroup.
<code>-serviceuserpassword pwd</code>	The password for the operating system account under which Windows RDBMS and TNSLSNR services run.
<code>{-shardgroup shardgroup_name -shardspace shardspace_name}</code>	Specify the name of the shardgroup or shardspace that this shard is being added to. Use <code>-shardspace</code> when using this command in a user-defined sharding configuration. Use <code>-shardgroup</code> with system-managed and composite sharding configurations.
<code>-sys_password</code>	SYS account password.
<code>-system_password</code>	SYSTEM account password.
<code>-windows_domain domain_name</code>	Specify a corresponding domain name for an account if a Windows account has been specified in the <code>-osaccount</code> parameter.

Usage Notes

`CREATE SHARD` causes a new database to be created using DBCA and NETCA.

`CREATE SHARD` only registers the database with GDS. The database is not created and replication is not configured on a newly create database, and data from other databases is not distributed to it until `DEPLOY` is executed.

The `-DEPLOY_AS` option can only be specified if a shard is being added to a shardspace. If a shard is being added to a shardgroup, its role will determined by the role of the shardgroup. The role is assigned when the shard gets deployed.

If the `-CREDENTIAL` option is specified, the credential name must have previously been created with the `ADD CREDENTIAL` command. If `-CREDENTIAL` is not specified, then `OSACCOUNT` and `OSPASSWORD` (and optionally `WINDOWS_DOMAIN`) must be specified.

The value for `DBPARAM`, `DBTEMPLATE`, or `NETPARAM` must be the name of a file object as previously specified in the `ADD FILE` command. The file content specified by the `-netparam` or `-netparamfile` parameter in `CREATE SHARD` is a NETCA response file. Examples can be found in `$ORACLE_HOME/assistants/netca`. By default, port 1521 is used with a listener name of `LISTENER_shard_name`. These values can be changed by modifying a sample response file and specifying it using the `-netparam` or `-netparamfile` parameters. The file content specified by the `-dbtemplate` or `-dbtemplatefile` parameter in `CREATE SHARD` is a DBCA template file. Examples of template files can be found in `$ORACLE_HOME/assistants/dbca/templates`. If not specified, the `General_Purpose.dbc` file located in the above directory on the shard destination host will be used. The format of the file contents specified in the `-dbparam` or `-dbparamfile` command is a space-delimited list of DBCA command line

parameters and values on one or more lines. For example, the following is an example of a valid parameter file: `-gdbName shard1.example.com -initParams sort_area_size=200000`. For a complete list of possible DBCA parameters, run `dbca -help -createDatabase`.

See *Working with Oracle GoldenGate Sharding in the Fusion Middleware Using the Oracle GoldenGate Microservices Architecture* guide for more information about using Oracle GoldenGate with Oracle Sharding.

Example

Create a shard.

```
GDSCTL> create shard -shardgroup group1 -destination dbdest -
credential group1_cred
-dbparam group1_db_params
```

create shardcatalog

Creates a shard catalog for the sharded database.

Syntax

```
create shardcatalog -database connect_identifier
[-user username[/password]]
[-region region_name_list]
[-configname config_name]
[-autovncr {ON | OFF}]
[-force]
[-sdb sdb_name]
[-shardspace shardspace_name_list]
[-agent_password password]
[-repl DG | OGG}]
[-repfactor number]
[-sharding {system | composite | user}]
[-chunks number]
[-protectmode dg_protection_mode]
[-agent_port port]
```

Options

Table C-36 GDSCTL create shardcatalog options

Command Option	Description
<code>-agent_password password</code>	Specify the password to be used for remote scheduler agent registration with the catalog database.
<code>-agent_port port</code>	Port number for XDB to use. If NULL and no current value is set, then it will default to 8080. Execute on catalog as well.

Table C-36 (Cont.) GDSCTL create shardcatalog options

Command Option	Description
<code>-autovncr {ON OFF}</code>	<p>This option enables (ON) or disables (OFF) auto VNCR mode. The default value is ON.</p> <p>See the Usage Notes below for important information about this option.</p>
<code>-chunks <i>number</i></code>	<p>Specify the default number of unique chunks in a shardspace. The value of <code>-chunks</code> must be greater than 2 times the size of the largest shardgroup in any shardspace.</p> <p>This parameter does not apply to user-defined sharding. It will apply to all shardspaces created for composite sharding if the number of chunks is not specified in the <code>ADD SHARDSPACE</code> command.</p> <p>All shardgroups in a shardspace have the same number of chunks. If this parameter is not specified, the default number of chunks is determined at the time of execution of the first <code>DEPLOY</code> command and is 120 per database of the shardgroup with the biggest number of shards</p>
<code>-configname <i>config_name</i></code>	<p>Specify the name of the GDS configuration. This name is used as the virtual <code>DB_DOMAIN</code> of the sharded database. The default configuration name is <code>ORADBCLLOUD</code>.</p> <p>The configuration name can be up to 32 bytes long and can contain an alphabetical character followed by zero or more alphanumeric ASCII characters, <code>'_'</code>, or <code>'#'</code> and possibly separated by periods if there are multiple identifiers.</p>
<code>-database <i>connect_identifier</i></code>	Specify the connect identifier for the database in which you want to create the catalog.
<code>-force</code>	Before creating the new catalog, delete an existing shard or GDS catalog on this database.
<code>-protectmode <i>dg_protection_mode</i></code>	Specify the Data Guard protection mode: <code>MAXPROTECTION</code> , <code>MAXAVAILABILITY</code> , or <code>MAXPERFORMANCE</code> (default). This parameter does not apply to Oracle GoldenGate replication. For Data Guard replication this parameter applies to any shardspace created without specification of protection mode in the <code>ADD SHARDSPACE</code> command.
<code>-region <i>region_name_list</i></code>	Specify a comma-delimited list of region names. This command creates each region and adds the regions to the catalog. If you do not specify a region, then a default region named <code>REGIONORA</code> is created.

Table C-36 (Cont.) GDSCTL create shardcatalog options

Command Option	Description
-repl DG OGG	Specify the technology used to replicate data in the sharded database. Only one value can be specified for this parameter: DG for Data Guard and OGG for Oracle GoldenGate. The default value is DG. This parameter cannot be modified after the catalog has been created.
-refactor <i>number</i>	Specify the default replication factor (the number of replicas for each piece of data stored in a shardgroup). This parameter can only be used with Oracle GoldenGate replication and system-managed or composite sharding, and this parameter is mandatory in these cases. It does not apply to user-defined sharding because there are no shardgroups in user-defined sharding. The default replication factor specified by this parameter can be overridden for a particular shardgroup by specifying the replication factor in the corresponding <code>ADD SHARDGROUP</code> command. The parameter cannot be overridden for automatically created default shardgroups. A non-default shardgroup must be created to customize the replication factor.
-sdb <i>sdb_name</i>	Specify the virtual DB_UNIQUE_NAME for the sharded database. The default name is ORASDB. The sharded database (SDB) name can be up to 30 characters long and can be an alphabetical character followed by zero or more alphanumeric ASCII characters or an underscore (_).
-sharding {system composite user}	Specify the sharding type: SYSTEM for system-managed (default), USER for user-defined, or COMPOSITE. This parameter cannot be modified after the catalog has been created. Oracle GoldenGate does not support the user-defined sharding method.
-shardspace <i>shardspace_name_list</i>	Specify a comma-delimited list of shardspace names. This option creates specified shardspaces and adds them to the catalog. If you do not specify a shardspace, then a default shardspace named SHARDSPACEORA is created.
-user <i>username[/password]</i>	Specify a user (and optionally, the password) that has the administrator privileges on the catalog database. If you do not use this option, then GDSCTL prompts you for the name and the password of a user with administrator privileges. If you specify a user name but not the password for the user, then GDSCTL prompts you for the password.

Usage Notes

The `create shardcatalog` command creates a GDS catalog specifically designed for a sharded database (SDB). This command cannot be used to create a conventional GDS catalog. Execution of this command is the first step required to create an SDB. The command is executed by the user with the GDS administrator or SYSDBA privileges.

Keep the following points in mind when using `create shardcatalog`:

- Only a single sharded database can be created using a shard catalog. A shard catalog cannot be used for a regular GDS configuration.
- Any arbitrary password can be specified for remote agent registration. If one is stipulated and an agent registration password already exists, it will be overridden with the new password. In order to successfully execute the `GDSCTL CREATE SHARD` command, an agent password must be set using `CREATE SHARDCATALOG` or `MODIFY CATALOG`.
- `CHUNKS` defines the default number of unique chunks in a shardspace. It is applied to all shardspaces created for composite sharding if the number of chunks is not specified in the `ADD SHARDSPACE` command.
- This command creates each region and adds the regions to the catalog. If you do not specify a region, then a default region named `REGIONORA` is created.
- The default replication factor specified by `REPFACOR` can be overridden for a particular shardgroup by specifying the replication factor in the corresponding `ADD SHARDGROUP` command. For automatically created default shardgroups the parameter cannot be overridden. A non-default shardgroup must be created to customize the replication factor.
- The `SHARDSPACE` option creates specified shardspaces and adds them to the catalog. If you do not specify a shardspace, then a default shardspace named `SHARDSPACEORA` is created.
- For Data Guard replication the `PROTECTMODE` parameter applies to any shardspace created without specification of protection mode in the `ADD SHARDSPACE` command.
- Auto VNCR is best used in environments with simple private networks where ease of configuration is the most important consideration. To have the highest level of control over which hosts may participate in a GDS configuration, disable Auto VNCR and explicitly add the IP addresses of each database host to the VNCR configuration.

When `-autovncr` is enabled, Oracle attempts to find the host name of the target shard when it is validated during `add shard` execution. This host is then automatically added to the VNCR list in the catalog as an invited node. This mechanism is not compatible with all network configurations and may not work in the following cases:

- The catalog or global service manager host does not know how to translate the host name discovered on the target shard host to an real IP address. This can happen if they have different names in the hosts file or DNS on the catalog or global service manager host, or if they just do not exist on those hosts.
- The target shard host has multiple public network addresses, and Oracle chooses an address that is different than the address used when the shard registers with the global service manager. This can happen if the host has multiple network cards or has configured virtual network interfaces.
- The shard is running Oracle RAC, and other Oracle RAC instances run on a different subnet. This is not a recommended configuration for Oracle RAC. Refer to the Oracle RAC documentation for recommended configurations. With Oracle RAC, Oracle Database connects to a single shard host to validate the target, and returns a subnet

mask that includes the entire subnet that the host is on. If other instances are on a different subnet, they have no valid VNCR entry, and registration is rejected.

- See Working with Oracle GoldenGate Sharding in the Fusion Middleware *Using the Oracle GoldenGate Microservices Architecture* guide for more information about using Oracle GoldenGate with Oracle Sharding.

When `-autoVNCR` is not enabled, or, if any of the above cases apply, new hosts should be added manually using [add invitednode](#) ([add invitedsubnet](#)).

Examples

The following example creates a shard catalog on the mydb database.

```
GDSCTL> CREATE SHARDCATALOG -DATABASE mydb
```

databases

Displays the status of all databases.

Syntax

```
{status database | databases} [-gsm gsm_name]  
                               [-database db_name]  
                               [-gdspool gdspool_name]  
                               [-raw|-support|-verbose]
```

Options

Table C-37 GDSCTL databases Options

Option	Description
<code>-database <i>db_name</i></code>	Specify the name of the database on which you want to start the service. If you do not specify this option, then GDSCTL starts the services on all preferred databases.
<code>-gdspool <i>gdspool_name</i></code>	Specify the name of the database pool in which the services you want to start are located. If not specified and there is only one <i>gdspool</i> with access granted to the user, it is used as the default <i>gdspool</i> .
<code>-gsm <i>gsm_name</i></code>	Specify the name of a global service manager to check. If the name is not specified, then GDSCTL uses the current global service manager name for the session (specified with the GDSCTL <code>set gsm</code> command).
<code>-raw</code>	If specified, GDSCTL output is presented in a raw non-parsed format.
<code>-support</code>	If specified, GDSCTL output displays additional information.
<code>-verbose</code>	Enable verbose mode.

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the command `connect` before running this command.

Example

Display the status of all databases:

```
GDSCTL> databases
```

The `databases` command returns output similar to the following:

```
Database: "dbcat1" Registered: Y State: Ok ONS: N. Role: PRIMARY Instances: 1
Region: east

Service: "sales_svc2" Globally started: N Started: N
Scan: Y Enabled: Y Preferred: Y
Service: "sales_svc1" Globally started: Y Started: Y
Scan: N Enabled: Y Preferred: Y
Registered instances:
sales%11
Database: "dbcat2" Registered: Y State: Ok ONS: N. Role: PRIMARY Instances: 1
Region: east
Service: "sales_svc2" Globally started: N Started: N
Scan: Y Enabled: Y Preferred: Y
Service: "sales_svc1" Globally started: Y Started: Y
Scan: N Enabled: Y Preferred: Y
Registered instances:
sales%1
```

delete catalog

Deletes the specified catalog.

Syntax

```
delete catalog [-connect [user/[password]@]conn_str]
               [-force]
```

Options

Table C-38 GDSCTL delete catalog Options

Syntax	Description
<code>-connect</code>	Specify an Oracle Net connect descriptor or a net service name that maps to a connect descriptor for the database (or shard). If you do not use this option, then GDSCTL deletes the Global Data Services catalog that is used by the global service manager associated with the current session.
<code>-force</code>	Silently remove GDS metadata. No warnings are shown.

Usage Notes

You must have the Global Data Services administrator privileges on the computer where the database resides from which you want to delete the Global Data Services catalog

If `-connect` is not specified, the catalog that belongs to currently connected database (if any) is deleted.

Example

Delete the Global Data Services catalog located in the database named DB1.

```
GDSCTL> delete catalog -connect db1
```

deploy

Deploys the sharded database.

Syntax

```
deploy [-no_rebalance]
```

Options

Table C-39 GDSCTL deploy Options

Option	Description
<code>-no_rebalance</code>	Skip automatic chunk migration during incremental deploy.

Usage Notes

This command is executed after one or more shards have been added to the shard catalog. As the result of the command execution, a certain range of data is associated with a newly added database. If a database is part of a Data Guard Broker configuration, a role (primary or standby) is assigned to it. Then replication and/or migration of data to from other databases to newly deployed databases are initiated.

- Deploy runs almost entirely in parallel, and mostly in the background, and will not deploy any shards which do not have all their counterparts in other shardgroups. All undeployed shards that can be deployed are deployed as the result of execution of this command.
- Before configuring replication, this command cross-checks parameters of all databases included into the replication configuration. An error is returned if the cross-check finds inconsistency or ambiguity, for example, no primary shardgroup in a shardspace with Data Guard replication.
- If a `CREATE SHARD` command had previously been issued, these new shards will be created during deployment and added to the shard catalog. If a shard needs to be created, `DEPLOY` runs a job for each database which requires a remote credential (see [add credential](#) and [create shard](#)). This credential must be valid at the time of deployment.

- The `NO_REBALANCE` option allows to skip automatic rebalancing of chunks across shards during incremental `DEPLOY`. Use the `move chunk` command to perform manual chunk migration.

Examples

Deploy the sharded database.

```
GDSCTL> deploy
```

disable service

Disables specified global services.

Syntax

```
disable service [-gdspool gdspool_name]
                [-service service_name_list]
                [-database db_name | [-override -connect conn_str [-pwd
password]]]
```

Options

Table C-40 GDSCTL disable service Options

Syntax	Description
<code>-connect <i>conn_str</i></code>	An Oracle Net connect descriptor or net service name that resolves to a connect descriptor for the database (or shard).
<code>-database <i>db_name</i></code>	Specify the name of the database on which the service is located. If you do not specify this option, then the service is disabled, globally.
<code>-gdspool <i>gdspool_name</i></code>	Specify the database pool in which the services are located. If not specified and there is only one <code>gdspool</code> with access granted to the user, then this one is used as the default <code>gdspool</code> .
<code>-override</code>	Skip the GDS catalog (used when the GDS catalog is not available).
<code>-pwd</code>	The GSMUSER password.
<code>-service <i>service_name_list</i></code>	Specify a comma-delimited list of global service names. If you do not use <code>-service</code> to specify an individual global service or to specify a list of global services, then all the services in the database pool are disabled.

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the `connect` command before running this command

A running service cannot be disabled. If `-override` is specified, the command is executed without going to the GDS catalog. Use this option when the GDS catalog is unavailable. It is not recommended for use under normal operation.

Example

Disable and stop the service G_SALES_REPORT on all databases in the database pool READERFARM.

```
GDSCTL> disable service -gdspool readerfarm -service g_sales_report -database db1
```



See Also:

[Disabling a Global Service](#)

enable service

Enables the specified global services.

Syntax

```
enable service [-gdspool gdspool_name]  
              [-service service_name_list]  
              [-database db_name | [-override -connect conn_str [-pwd password]]]
```

Options

Table C-41 GDSCTL enable service Options

Syntax	Description
<code>-connect <i>conn_str</i></code>	An Oracle Net connect descriptor or net service name that resolves to a connect descriptor for the database (or shard).
<code>-database <i>db_name</i></code>	Specify the name of the database on which the service is located. If you do not specify this option, then the service is enabled globally.
<code>-gdspool <i>gdspool_name</i></code>	Specify the GDS pool in which the services are located. If not specified and there is only one gdspool with access granted to the user, it is used as the default gdspool.
<code>-override</code>	Skip the GDS catalog (used when the GDS catalog is not available).
<code>-pwd</code>	The GSMUSER password.
<code>-service <i>service_name</i></code>	Specify a comma-delimited list of global service names. If you do not use <code>-service</code> to specify an individual global service or to specify a list of global services, then all the services in the database pool are disabled.

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the [connect](#) command before running this command

ENABLE SERVICE will start the global service if it is preferred_all or cardinality is not reached.

If -override is specified, the command is executed without going to the GDS catalog. Use this option when the GDS catalog is unavailable. It is not recommended for use under normal operation.

Example

Enable the service G_SALES_REPORT on the database DB1 in the database pool READERFARM.

```
GDSCTL> enable service -gdspool readerfarm -service g_sales_report -database db1
```



See Also:

[Enabling a Global Service](#)

exit

Quit GDSCTL utility.

Syntax

```
quit | exit
```

export catalog

Saves the current catalog configuration to a local file.

Syntax

```
export catalog [-force] source
```

Options

Table C-42 GDSCTL export catalog Options

Syntax	Description
-force	If not specified, export will be cancelled if there are ongoing GDS operations.
source	Name of a file on the same computer where the command is being executed. The configuration will be saved to this file. If the file already exists, it will be overwritten without a prompt. If the file is not writable (for example the path does not exist), you will get an error.

Usage Notes

You must connect to the catalog database as a user with GDS Administrator privileges before running this command.

It is recommended that you validate the catalog, using the [validate catalog](#) command before exporting it.

Example

Save the catalog backup to your home directory.

```
GDSCTL> export catalog /home/user/cat-201307.backup
```

help

Provides a list of the GDSCTL commands supported in the current release. When followed by a command name, it returns the help page associated with the command.

Syntax

```
help [gdsctl_command]
```

Options

Table C-43 GDSCTL help Options

Option	Description
<i>gdsctl_command</i>	Enter any GDSCTL command name to return a help page with syntax, options, usage notes and examples.

import catalog

Restores the catalog configuration from the specified file, created using export catalog command.

Syntax

```
import catalog [-database catalog_db_name]  
               [-catpwd gsmcatusrpwd]  
               [-user gsmadminname[/password]]  
               source
```

Options

Table C-44 GDSCTL import catalog Options

Syntax	Description
<i>-catpwd gsmcatusrpwd</i>	GSMCATUSER password.
<i>-database catalog_db_name</i>	The connect identifier for the database in which to create catalog.
<i>source</i>	Name of a file on the same computer where the command is being executed. The configuration will be restored from this file. If the file is not readable, you will get an error.

Table C-44 (Cont.) GDSCTL import catalog Options

Syntax	Description
<code>-user gsmadminname[/password]</code>	Credentials of the user that has the GDS administrator privileges on the catalog database.

Usage Notes

If `-database` is not specified, the GDS catalog that the current global service manager is associated with will be used. The `-catpwd` option should be specified in case you need to perform cleanup of databases in the existing catalog that are not found in imported file.

When restoring to a new catalog database, catalog must be created first, using the [create gds catalog](#) command.

You must connect to the catalog database as a user with GDS Administrator privileges before running this command.

The import procedure can be considered finished only when there are no pending requests after import. Use the [config](#) command to get the list of pending requests.

Example

Load the catalog backup from your home directory.

```
GDSCTL> import catalog /home/user/cat-201307.backup
```

modify catalog

Modifies the properties of the GDS catalog or shard catalog.

Syntax

```
modify catalog [-autovncr {ON | OFF}]
               [-oldpwd oldpassword -newpwd newpassword]
               [-pwd password -newkeys]
               [-agent_password password]
               [-agent_port port]
               [-region region]
               [-recover]
```

Options**Table C-45 GDSCTL modify catalog Options**

Syntax	Description
<code>-agent_password password</code>	Specify the agent registration password in the catalog for the remote Scheduler agent.
<code>-agent_port port</code>	Port number for XDB to use. If it is NULL and no current value is set, then it will default to 8080. Execute on catalog as well.

Table C-45 (Cont.) GDSCTL modify catalog Options

Syntax	Description
<code>-autovncr {ON OFF}</code>	This option enables (ON) or disables (OFF) autovncr mode. The default value is ON.
<code>-newkeys</code>	Generates a new PKI key pair.
<code>-newpwd <i>newpassword</i></code>	Used along with <code>-oldpwd</code> , sets the GSMCATUSER password after changing it on the catalog database.
<code>-oldpwd <i>oldpassword</i></code>	Used along with <code>-newpwd</code> , sets the GSMCATUSER password after changing it on the catalog database.
<code>-pwd <i>password</i></code>	Provides the GSMCATUSER password to generate the PKI keys when using <code>-newkeys</code> .
<code>-recover</code>	Perform catalog recovery to the last consistent state.
<code>-region <i>region</i></code>	Region that the database, catalog, shard, shardgroup, or global service manager belongs to.

Usage Notes

To use this command, there must be at least one global service manager running and a connection with the catalog database must have already been established (see the [connect](#) command).

You must connect to the catalog database as a user with the Global Data Services administrator privileges, using the [connect](#) command before running this command.

Auto VNCR is best used in environments with simple private networks where ease of configuration is the most important consideration. To have the highest level of control over which hosts may participate in a GDS configuration, disable Auto VNCR and explicitly add the IP address(es) of each database host to the VNCR configuration.

The GSMCATUSER password should be updated regularly for security reasons. Use the following command to perform this operation.

```
modify catalog -oldpwd oldpassword -newpwd newpassword
```

This command fetches the encrypted private key and encryption string, decrypts them using the old password, re-encrypts them with the new password and stores them again.

If the GSMCATUSER password is changed, you must execute `MODIFY CATALOG` to update catalog security scheme, with `-newpwd` and `-oldpwd` specified.

The PKI keys must be updated regularly, which is done using `modify catalog -oldpwd oldpassword -newkeys`. This command generates a new PKI key pair and replaces the corresponding fields in the database.

If you decide to replace the PKI keys, or just after a patchset upgrade on the catalog database, run this command:

```
modify catalog -pwd ** -newkeys
```

An arbitrary password can be stipulated for remote agent registration. If an agent registration password already exists, it will be overridden with the new password. In

order to successfully execute the `GDSCTL CREATE SHARD` command, an agent password must be set using the `CREATE SHARDCATALOG` or `MODIFY CATALOG` command.

Examples

Turn off `autovncr` mode for the catalog database.

```
connect gsmadmin@mycloud
GDSCTL> modify catalog -autovncr off
```

Specify the remote Scheduler agent registration password.

```
connect gsmadmin@mycloud
GDSCTL> modify catalog -agent_password mypass
```

Update catalog security scheme.

```
GDSCTL> modify catalog -autovncr OFF -oldpwd opwd -newpwd npwd -pwd pwd -
newkeys
```

modify cdb

Modify cdb attributes.

Syntax

```
modify cdb -shard cdbname_list
           [-connect connect_identifier]
           [-pwd gsmrootuser_pwd]
           [-scan scan_address [-ons port]]
           [-savename]
```

Options

Table C-46 GDSCTL modify cdb Options

Option	Description
<code>-shard <i>cdbname_list</i></code>	Specify a comma-delimited list of cdb names.
<code>-connect <i>connect_identifier</i></code>	Specify an Oracle Net connect descriptor or a net service name that maps to a connect descriptor for the database that is being modified.
<code>-ons <i>port</i></code>	Specify the ONS port.
<code>-pwd <i>gsmrootuser_pwd</i></code>	Specify the password for GSMROOTUSER.
<code>-savename</code>	Specify this option to store a net service name specified with the <code>-connect</code> option in the Global Data Services catalog, rather than the connect descriptor mapped to that net service name.
<code>-scan <i>scan_address</i></code>	Specify the SCAN address for a cluster.

Usage Notes

Some parameters are not supported after the CDB contains shards or contains shards that have been deployed.

Examples

Change a password on a CDB.

```
GDSCTL> modify cdb -shard cdb1 -pwd new_gsmrootuser_password
```

modify credential

Modifies an existing credential which will be used by the remote Scheduler agent to execute shard jobs.

Syntax

```
modify credential -credential credential_name
                  -osaccount account_name
                  -ospassword password
                  [-windows_domain domain_name]
```

Options

Table C-47 GDSCTL modify credential Options

Option	Description
-credential <i>credential_name</i>	Specify the name of the credential to modify.
-osaccount <i>account_name</i>	Specify the operating system account which will be used for remote jobs.
-ospassword <i>password</i>	Specify the corresponding password for the account.
-windows_domain <i>domain_name</i>	If a Windows account has been specified, specify the corresponding domain name for that account.

Usage Notes

This command modifies credentials which will be used to execute jobs on sharded hosts in response to administrative commands.

If the specified credential does not exist, the command returns an error.

Examples

Modify a credential named east_region_cred.

```
GDSCTL> modify credential -credential east_region_cred -osaccount
agent_user
-ospassword newpass
```

modify database

Modifies the configuration parameters of the databases in a GDS pool, such as region, connect identifier, global service manager password, SCAN address, and ONS port.

Syntax

```
modify database -database db_name_list
                [-gdspool gdspool_name]
                [-shard shard_name]
                [-deploy_as PRIMARY|STANDBY]
                [-region region_name]
                [-pwd password]
                [-connect connect_identifier]
                [-scan scan_address]
                [-ons port]
                [-savename]
                [-cpu_threshold cpu]
                [-disk_threshold disk]
                [-rack rack_id]
                [-NETPARAM net_parameter_file | -NETPARAMFILE
net_parameter_file]
                [-DBPARAM db_parameter | -DBPARAMFILE db_parameter_file]
                [-DBTEMPLATE db_template | -DBTEMPLATEFILE db_template_file]
```

Options

Table C-48 GDSCTL modify database Options

Option	Description
<code>-connect <i>connect_identifier</i></code>	Specify an Oracle Net connect descriptor or a net service name that maps to a connect descriptor for the database that is being modified.
<code>-cpu_threshold <i>cpu</i></code>	Specifies CPU Utilization percentage threshold.
<code>-database <i>dbname_list</i></code>	Specify a comma-delimited list of database names.
<code>-disk_threshold <i>disk</i></code>	Specifies the average latency in milliseconds of a synchronous single-block read.
<code>-gdspool <i>gdspool_name</i></code>	Specify the database pool to which the databases belong.
<code>-ons <i>port</i></code>	Specify the ONS port.
<code>-pwd <i>password</i></code>	Specify the password for the GSMUSER.
<code>-region <i>region_name</i></code>	Specify the region to which the databases belong.
<code>-savename</code>	Specify this option to store a net service name specified with the <code>-connect</code> option in the Global Data Services catalog, rather than the connect descriptor mapped to that net service name.
<code>-scan <i>scan_address</i></code>	Specify the SCAN address for a cluster.

Usage Notes

You can multiple databases if the `region` property is specified.

For all parameters except for the GDS region, first the appropriate changes must be done by the database administrator and then the `modify database` command must be run to update the modified parameters in the GDS catalog. Alternatively, you can use the [sync database \(synchronize database\)](#) command for this purpose.

You must connect to the catalog database as a user with the pool administrator privileges, using the [connect](#) command before running this command

Example

Change the region of databases DB1 and DB3 to EAST.

```
GDSCTL> modify database -database db1,db3 -region east
```

modify file

Updates the contents of a file in the catalog which can be used by subsequent GDSCTL commands.

Syntax

```
modify file -file file_name
           -source local_filename
```

Options

Table C-49 GDSCTL modify file Options

Option	Description
<code>-file <i>file_name</i></code>	Specify the name of the file object to update.
<code>-source <i>local_filename</i></code>	Specify an operating system file name specifying a file local to the machine running GDSCTL.

Usage Notes

This command updates a named file object in the catalog by reloading the contents of an operating system file into the catalog.

Examples

Update a file named `east_region_db_params` with content from the local source file `/tmp/dbca_params.txt`

```
GDSCTL> modify file -file east_region_db_params -source /tmp/
dbca_params.txt
```

modify gdspool

Modifies the configuration parameters of GDS pools.

Syntax

```
modify gdspool -gdspool gdspool_name_list
                [-removeuser user_name | -adduser user_name]
```

Options

Table C-50 GDSCTL modify gdspool Options

Option	Description
-adduser <i>user_name</i>	Specify the user to add to the list of GDS pool administrators. This option grants the pool administrator role to the specified user.
-gdspool <i>database_pool_list</i>	Specify a comma-delimited list of GDS pool names.
-removeuser <i>user_name</i>	Specify the user to remove from the list of GDS pool administrators. This option revokes the pool administrator role from the specified user.

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the [connect](#) command before running this command

Example

Add PETER to the list of database pool administrators for the pool MYREADERFARM:

```
GDSCTL> modify gdspool -gdspool myreaderfarm -adduser peter
```

modify gsm

Modifies the configuration parameters of the global service manager. The changes take effect after the global service manager is restarted.

Syntax

```
modify gsm -gsm gsm_name
            [-catalog connect_id [-pwd password]]
            [-region region_name]
            [-localons ons_port]
            [-remoteons ons_port]
            [-endpoint gmsendpoint [-remote_endpoint remote_endpoint]]
            [-listener listener_port]
            [-wpwd wallet_password]
```

Options

Table C-51 GDSCTL modify gsm Options

Option	Description
<code>-catalog connect_id</code>	Specify the connect identifier for the Global Data Services catalog database. If a network service name is specified, it must be resolvable by the local naming method to a connect descriptor that allows the global service manager being modified to connect to the catalog database.
<code>-endpoint gsmendpoint</code>	Specify the protocol address that the global services manager listens on for client connection requests. If you use this option, the value you specify overrides the default endpoint.
<code>-gsm gsm_name</code>	Enter the name of the global service manager that you want to modify. If you do not specify a name, then the current global service manager name for the session (specified with the set gsm command) is used.
<code>-listener listener_port</code>	Specify the new listener port.
<code>-localons ons_port</code>	Specify the new local ONS port.
<code>-pwd password</code>	Specify the password for the <code>GSMCATUSER</code> account. If you do not specify a password, then you are prompted to enter one.
<code>-region region_name</code>	Specify the region to which the global service manager belongs.
<code>-remote_endpoint remote_endpoint</code>	Specify the protocol address that is used by the global service manager to receive database registration requests and communicate with other global service managers in the configuration. If you use this option, the value you specify overrides the default endpoint.
<code>-remoteons ons_port</code>	Specify the new remote ONS port.
<code>-wpwd</code>	Specify the password for the global service manager wallet.

Usage Notes

- You must run this command locally on the computer where you want to modify the global service manager.
- This command can be run only by the operating system user who started the global service manager.
- When you run this command, GDSCTL connects to the Global Data Services catalog as the `GSMCATUSER` user and prompts you for the `GSMCATUSER` password.

Example

Modify the global service manager named `gsm1` so that it is in the EAST region.

```
GDSCTL> modify gsm -gsm gsm1 -region east
```

modify region

Modifies the configuration parameters for a region.

Syntax

```
modify region -region region_name_list
              [-buddy region_name]
              [-weights weight]
```

Options**Table C-52 GDSCTL modify region Options**

Option	Description
<code>-buddy <i>region_name</i></code>	Specify the name of the buddy region
<code>-region <i>region_list</i></code>	Specify a comma-delimited list of region names
<code>-weights <i>weight</i></code>	Used for static RLB distribution. format: name = value,...name = value

Usage Notes

You must connect to the catalog database as a user with the Global Data Services administrator privileges, using the [connect](#) command before running this command.

To clear buddy region or weight, call `MODIFY REGION` and specify empty quotes as the value. If `WEIGHTS` is specified, dynamic load balancing is replaced by static (not recommended).

Example

Modify two regions, EAST and WEST, as follows:

```
GDSCTL> modify region -region west -buddy east
```

modify service

Modifies the service attributes.

Syntax

To add more preferred or available databases to a global service:

```
modify service [-gdspool gdspool_name]
               -service service_name
               {-preferred db_name_list | -available db_name_list}
```

To modify the attributes of a global service:

```
modify service [-gdspool gdspool_name]
               -service service_name
               [-locality {ANYWHERE | LOCAL_ONLY}]
               [-region_failover]
               [-role {PRIMARY | PHYSICAL_STANDBY [-failover_primary]
                     |
                     LOGICAL_STANDBY | SNAPSHOT_STANDBY}]
               [-lag {lag_value | ANY}]
               [-notification {TRUE | FALSE}]
               [-rlbgoal {SERVICE_TIME | THROUGHPUT}]
               [-clbgoal {SHORT | LONG}]
               [-tafpolicy {BASIC | NONE | PRECONNECT}]
               [-policy policy]
               [-failovertype {NONE | SESSION | SELECT | TRANSACTION}]
               [-failovermethod {NONE | BASIC}]
               [-dtp {TRUE | FALSE}]
               [-sql_translation_profile stp_name]
               [-failoverretry failover_retries]
               [-failoverdelay failover_delay]
               [-edition edition_name]
               [-commit_outcome {TRUE | FALSE}]
               [-retention retention_seconds]
               [-session_state {DYNAMIC | STATIC}]
               [-replay_init_time replay_init_time]
               [-table_family family]
```

To move a global service from one database to another database:

```
modify service [-gdspool gdspool_name]
               -service service_name
               -old_db db_name
               -new_db db_name
               [-force]
```

To change an available database to a preferred database for a service:

```
MODIFY SERVICE [-gdspool gdspool_name]
               -service service_name
               -available db_name_list
               -preferred
```

To change databases between preferred and available status:

```
modify service [-gdspool gdspool_name]
              -service service_name
              {-preferred_all |
               {-modifyconfig -preferred db_name_list [-available
               db_name_list]]}}
```

To modify properties for a global service that are specific to an Oracle RAC database:

```
modify service [-gdspool gdspool_name]
              -service service_name
              -database db_name
              [-server_pool server_pool_name |
               {-add_instances|-modify_instances} -preferred inst_list
               -available inst_list |
               -drop_instances inst_list
               -cardinality {UNIFORM | SINGLETON}
```

Options

Table C-53 GDSCTL modify service Options

Option	Description
<code>-add_instances [-preferred <i>comma-delimited-list</i>] [-available <i>comma-delimited-list</i>]</code>	Provides a list of preferred and available instances for the given service on the given database. The provided list over-rides existing assigned instances, if any. Using the <code>-preferred</code> and <code>-available</code> options is optional, but at least one of these must be provided.
<code>-available <i>db_name_list</i></code>	Specify a comma-delimited list of available databases on which the service runs, if the preferred databases are not available. The list of available instances must be mutually exclusive with the list of preferred instances. If you attempt to add a preferred or available database to a service that was configured with <code>-preferred_all</code> , then GDSCTL returns an error.
<code>-cardinality {UNIFORM SINGLETON}</code>	Specify the cardinality option for a service running on a policy-managed Oracle RAC database. Services with cardinality set to <code>UNIFORM</code> are offered on all database instances. Services with cardinality set to <code>SINGLETON</code> are offered on only one database instance.

Table C-53 (Cont.) GDSCTL modify service Options

Option	Description
<code>-clbgoal {SHORT LONG}</code>	For connection load balancing goal: set to <code>SHORT</code> if using run-time load balancing, set to <code>LONG</code> for long running connections such as batch jobs or older SQL*Forms style. The default value for this option is <code>SHORT</code> .
<code>-commit_outcome {TRUE FALSE}</code>	Enable Transaction Guard; when set to <code>TRUE</code> , the commit outcome for a transaction is accessible after the transaction's session fails due to a recoverable outage.
<code>-database db_name</code>	Specify the name of the database on which you want to modify the service. When <code>-database</code> is specified, you must specify exactly one of: <ul style="list-style-type: none"> <code>-server_pool</code> and/or <code>-cardinality</code>. Either is optional, but at least one must appear, both can be used at once. <code>-add_instances</code>, <code>-modify_instances</code>, or <code>-drop_instances</code>. Exactly one of these three options must be used.
<code>-dtp {TRUE FALSE}</code>	Indicates whether Distributed Transaction Processing should be enabled for this service. This ensures that the service is offered at exactly one instance at a time for XA affinity.
<code>-drop_instances inst_list</code>	Provide a list of instances to be removed from the existing assigned instances for a given service on a given database. The provided list of instances will be removed from the existing assigned list.
<code>-edition edition_name</code>	Specify the initial session edition of the service. When an edition is specified for a service, all subsequent connections that specify the service use this edition as the initial session edition. However, if a session connection specifies a different edition, then the edition specified in the session connection is used for the initial session edition. GDSCTL does not validate the specified edition name. During connection, the connect user must have <code>USE</code> privilege on the specified edition. If the edition does not exist or if the connect user does not have <code>USE</code> privilege on the specified edition, then an error is raised.
<code>-failover_primary</code>	If you set the <code>-role</code> option to <code>PHYSICAL_STANDBY</code> , then you can use this option to enable the service for failover to the primary database.
<code>-failoverdelay failover_delay</code>	For Application Continuity and TAF, the time delay (in seconds) between reconnect attempts for each incident at failover.

Table C-53 (Cont.) GDSCTL modify service Options

Option	Description
<code>-failovermethod {NONE BASIC}</code>	TAF failover method (for backward compatibility only). If <code>failovertype</code> is set to either <code>SESSION</code> or <code>SELECT</code> , then choose <code>BASIC</code> for this option.
<code>-failoverretry failover_retries</code>	For Application Continuity and TAF, the number of attempts to connect after an incident.
<code>-failovertype {NONE SESSION SELECT TRANSACTION}</code>	Specify the failover type. To enable Application Continuity for Java, set this parameter to <code>TRANSACTION</code> . To enable Transparent Application Failover (TAF) for OCI, set this parameter to <code>SELECT</code> or <code>SESSION</code> .
<code>-force</code>	If you use this option, then all sessions are disconnected when the service is moved, requiring the sessions using the service to reconnect (potentially to a different instance). If you do not use this option, then the sessions that are connected to a database using this service stay connected, but all new sessions cannot be established to the service.
<code>-gdspool gdspool_name</code>	Specify the name of the database pool to which the service belongs. If not specified and there is only one <code>gdspool</code> with access granted to user, it is used as the default <code>gdspool</code> .
<code>-lag {lag_value ANY}</code>	Specify the lag for the service in seconds. You can use the keyword <code>ANY</code> to indicate that there is no upper threshold on the lag time. The default value for the <code>-lag</code> option is <code>ANY</code> .
<code>-locality {ANYWHERE LOCAL_ONLY}</code>	The service region locality. If you do not use this option, then the default value of <code>ANYWHERE</code> is used for the service.
<code>-modifyconfig</code>	Use this option to indicate that you are changing the current list of preferred and available databases for the service. If you use this option, then any databases that are not specified in either the preferred or available list, but were previously assigned, are removed from the list of databases on which the service can run.

Table C-53 (Cont.) GDSCTL modify service Options

Option	Description
<code>-modify_instances [-preferred comma-delimited-list] [-available comma-delimited-list]</code>	<p>The provided <i>comma-delimited-list</i> of preferred and available instances is merged with the existing list currently stored in the catalog.</p> <p>If you specify an instance in the <i>comma-delimited-list</i> that is not already in the stored list, it is added to the stored list in its correct mode (preferred or available.)</p> <p>If you specify in <i>comma-delimited-list</i> an instance that is already in the stored list, then the mode of the instance in the stored list is modified to the provided mode (preferred or available). If the user provided mode is the same as the stored mode, then the mode of the instance will not be changed.</p> <p>Any instances already in the stored list that are not in the provided list remain unchanged in the stored list.</p> <p>Note that an instance cannot be both preferred and available, it can be in one mode only.</p> <p><code>-preferred</code> and <code>-available</code> are optional but at least one list must be provided.</p>
<code>-new_db database_name</code>	<p>Specify the name of the new database on which the service runs.</p> <p>If you attempt to move a service that was configured with <code>-preferred_all</code>, then GDSCTL returns an error.</p>
<code>-notification {TRUE FALSE}</code>	<p>Enable Fast Application Notification (FAN) for OCI connections.</p>
<code>-old_db database_name</code>	<p>Specify the name of the old database on which the service runs.</p> <p>If you attempt to move a service that was configured with <code>-preferred_all</code>, then GDSCTL returns an error.</p>
<code>-policy {AUTOMATIC MANUAL}</code>	<p>Specify the management policy for the service.</p> <p>If you specify <code>AUTOMATIC</code> (the default), then the service automatically starts when the database restarts, either by a planned restart or after a failure. Automatic restart is also subject to the service role.</p> <p>If you specify <code>MANUAL</code>, then the service is never automatically restarted upon planned restart of the database. A <code>MANUAL</code> setting does not prevent the global service manager from monitoring the service when it is running and restarting it if a failure occurs.</p>
<code>-pdbname pdb_name</code>	<p>Specify the pluggable database name.</p>

Table C-53 (Cont.) GDSCTL modify service Options

Option	Description
<code>-preferred db_name_list</code>	Specify a comma-delimited list of preferred databases on which the service runs. When changing a database from available to preferred, you do not specify a value for the <code>-preferred</code> option. The list of preferred instances must be mutually exclusive with the list of available instances. If you attempt to add a preferred or available database to a service that was configured with <code>-preferred_all</code> , then GDSCTL returns an error.
<code>-preferred_all</code>	Specifies that all the databases in the database pool are preferred databases. Any new databases added to the pool are configured as preferred databases for this service. This option cannot be used with the <code>-preferred</code> and <code>-available</code> options.
<code>-region_failover</code>	Indicates that the service is enabled for region failover. You can only use this option when you specify <code>LOCAL_ONLY</code> for the <code>-locality</code> option.
<code>-replay_init_time replay_init_time</code>	For Application Continuity, this parameter specifies the time (in seconds) after which replay is not initiated. Default value is 300 seconds.
<code>-retention retention_seconds</code>	For Transaction Guard (<code>commit_outcome</code> set to <code>TRUE</code>), this parameter determines the amount of time (in seconds) that the commit outcome is retained in the database.
<code>-rlbgoal {SERVICE_TIME THROUGHPUT}</code>	Run-time Load Balancing Goal. Set this parameter to <code>SERVICE_TIME</code> to balance connections by response time. Set this parameter to <code>THROUGHPUT</code> to balance connections by throughput. If you do not use this option, then the value defaults to <code>SERVICE_TIME</code> for the run-time load balancing goal.
<code>-role {[PRIMARY] [PHYSICAL_STANDBY] [-failover_primary] [LOGICAL_STANDBY] [SNAPSHOT_STANDBY]}</code>	Specify the database role that the database must be for this service to start on that database. This applies only to database pools that contain an Oracle Data Guard broker configuration. See Also: <i>Oracle Data Guard Concepts and Administration</i> for more information about database roles
<code>-server_pool server_pool_name</code>	Specify the name of the Oracle RAC server pool in the GDS pool database to which the service belongs (for a policy-managed Oracle RAC database).
<code>-service service_name</code>	Specify the name of the global service.
<code>-session_state {DYNAMIC STATIC}</code>	For Application Continuity, this parameter specifies whether the session state that is not transactional is changed by the application. A value of <code>DYNAMIC</code> is recommended for most applications.

Table C-53 (Cont.) GDSCTL modify service Options

Option	Description
<code>-sql_translation_profile stp_name</code>	<p>Use this option to specify a SQL translation profile for a service that you are adding after you have migrated applications from a non-Oracle database to an Oracle database.</p> <p>This option corresponds to the SQL translation profile parameter in the <code>DBMS_SERVICE</code> service attribute.</p> <p>Notes:</p> <ul style="list-style-type: none"> • Before using the SQL translation feature, you must migrate all server-side application objects and data to the Oracle database. • Use the config service command to display the SQL translation profile. <p>See Also: <i>Oracle Database SQL Translation and Migration Guide</i> for more information about SQL translation</p>
<code>-table_family family</code>	<p>Specifies the table family name as a property of the service. This parameter takes one of the table family values (root table schema.name) as shown in the <code>CONFIG TABLE FAMILY</code> output.</p> <p>If the schema name or the table name is case-sensitive, use two-level quotes (single quotes outside, double quotes inside) around the whole string, for example, <code>'"TESTUSER1.Customers6"'</code>. No quotes are needed if neither name is case sensitive.</p>
<code>-tafpolicy {BASIC NONE }</code>	TAF policy specification (for administrator-managed databases only).

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the [connect](#) command before running this command.

Use this command to:

- Add databases to the preferred or available lists for the service
- Move a service from one database to another database
- Change an available database to a preferred database or a preferred database to an available database
- Modify the high availability attributes of the service

If you want to temporarily move a service from one database to a different database, then use the [relocate service](#) command.

Examples

Add the database DB3 as a preferred database for the service G_SALES_REPORT in the database pool MYREADERFARM.

```
GDSCTL> modify service -gdspool myreaderfarm -service g_sales_report -
preferred db3
```

Modify the service G_DAILY_SALES_REPT in the database pool MYREADERFARM to change the run-time load balancing goal to THROUGHPUT.

```
GDSCTL> modify service -gdspool myreaderfarm -service g_daily_sales_rept
-rlbgoal THROUGHPUT
```

Move the service G_SALES_REPORT in the database pool MYREADERFARM from the database DB1 to DB4.

```
GDSCTL> modify service -gdspool myreaderfarm -service g_sales_report
-old_db db1 -new_db db4
```

Upgrade the DB3 database from an available database to a preferred database for the service G_SALES_REPORT in the database pool READFARM.

```
GDSCTL> modify service -gdspool readfarm -service g_sales_report
-available db3 -preferred
```

Assume the service G_SALES_REPORT currently has the databases DB1 and DB2 assigned as preferred databases, and the database DB3 assigned as an available database. Exchange the preferred and available databases DB1 and DB3, and remove the DB2 database for the service SALES_REPORT in the database pool READFARM.

```
GDSCTL> modify service -gdspool readfarm -service g_sales_report -
modifyconfig
-available db3 -preferred db1
```

Modify the properties of the service G_SALES_REPORT in the database pool READFARM to specify that it should run only in the server pool named SALESPool for the policy-managed Oracle RAC database DB1.

```
GDSCTL> modify service -gdspool readfarm -service g_sales_report -database
db1
-server_pool salespool
```

Supply the preferred and available instances for the given service on the given database.

```
GDSCTL> modify service -gdspool mypool -service mysvc -database mydb -
add_instances
-preferred inst1,inst2 -available inst3,inst4
```

In a system-managed sharded database, the table family ID parameter is specified as a property of the service.

```
GDSCTL> modify service -GDSPool shdpool -table_family sales.customer -
service sales
```



See Also:

[Modifying Global Service Attributes](#)

modify shard

Modify shard attributes.

Syntax

```
modify shard -shard shname_list
            [-region region_name]
            [-connect connect_identifier]
            [-pwd password]
            [-scan scan_address [-ons port]]
            [-savename]
            [-cpu_threshold cpu]
            [-disk_threshold disk]
            [-destination destination_name]
            [-credential credential_name |
            [[-osaccount account_name]
            [-ospassword password]
            [-windows_domain domain_name]]]
```

Options

Table C-54 GDSCTL modify shard Options

Option	Description
<code>-connect <i>connect_identifier</i></code>	Specify an Oracle Net connect descriptor or a net service name that maps to a connect descriptor for the database that is being modified.
<code>-cpu_threshold <i>cpu</i></code>	Specifies CPU Utilization percentage threshold.
<code>-credential <i>credential_name</i></code>	Specify the credential to use on the remote machine which specifies the user name and password under which database creation will occur.
<code>-destination <i>destination_name</i></code>	Specify the name of the remote executable agent through which the database will be created.
<code>-disk_threshold <i>disk</i></code>	Specifies the average latency in milliseconds of a synchronous single-block read.
<code>-ons <i>port</i></code>	Specify the ONS port.

Table C-54 (Cont.) GDSCTL modify shard Options

Option	Description
<code>-osaccount <i>account_name</i></code>	Specify the operating system account which will be used for remote jobs.
<code>-ospassword <i>password</i></code>	Specify the corresponding password for the account specified in <code>-osaccount</code> .
<code>-pwd <i>password</i></code>	Specify the password for the GSMUSER.
<code>-region <i>region_name</i></code>	Specify the region to which the databases belong.
<code>-savename</code>	Specify this option to store a net service name specified with the <code>-connect</code> option in the Global Data Services catalog, rather than the connect descriptor mapped to that net service name.
<code>-scan <i>scan_address</i></code>	Specify the SCAN address for a cluster.
<code>-shard <i>shname_list</i></code>	Specify a comma-delimited list of shard names.
<code>-windows_domain <i>domain_name</i></code>	Specify the corresponding domain name if a Windows account has been specified in <code>-osaccount</code> .

Usage Notes

The `REGION` parameter cannot be modified for a shard that belongs to a shardgroup. The modification has to be done at the shardgroup level.

The `DESTINATION` and `CREDENTIAL` parameters are only modifiable when the shard has not yet been deployed, since these parameters only have meaning for the deployment process and are no longer referenced once deployment has completed successfully.

Examples

```
GDSCTL> modify shard -shard shard1 -ons 23222
```

modify shardgroup

Modify shardgroup attributes.

Syntax

```
modify shardgroup -shardgroup shardgroup_name
                  [-region region_name]
                  [-shardspace shardspace_name]
                  [-repfactor number]
                  [-deploy_as {PRIMARY | STANDBY | ACTIVE_STANDBY}]
```

Options

Table C-55 GDSCTL modify shardgroup Options

Option	Description
<code>-shardgroup <i>shardgroup_name</i></code>	Specify the name of the shardgroup to be modified.
<code>-region <i>region_name</i></code>	Specify the region the shardgroup resides in.
<code>-shardspace <i>shardspace_name</i></code>	Specify the shardspace this shardgroup belongs to.
<code>-repfactor <i>number</i></code>	Specify the number of replicas for each piece of data stored in this shardgroup.
<code>-deploy_as {PRIMARY STANDBY ACTIVE_STANDBY}</code>	Specify the initial role for a newly deployed database: PRIMARY, STANDBY, or ACTIVE_STANDBY.

Usage Notes

All shardgroup attributes, except for `DEPLOY_AS`, can only be modified when the shardgroup does not contain any deployed shards. `DEPLOY_AS` can be modified at any time because it does not affect shards that were already added to the shardgroup.

Examples

Modify the GROUP1 shardgroup to have a replication factor of 3.

```
GDSCTL> modify SHARDGROUP -SHARDGROUP group1 -REPFACOR 3
```

modify shardspace

Modify shardspace parameters.

Syntax

```
modify shardspace -shardspace shardspace_name
                  [-chunks number]
                  [-protectmode dg_protection_mode]
```

Options

Table C-56 GDSCTL modify shardspace Options

Option	Description
<code>-shardspace <i>shardspace_name</i></code>	Specify the name of the shardspace to be modified.
<code>-chunks <i>number</i></code>	Specify the number of chunks in the shardspace.
<code>-protectmode <i>dg_protection_mode</i></code>	Specify the Data Guard Protection mode: MAXPROTECTION, MAXAVAILABILITY, or MAXPERFORMANCE. This option can only be executed where Data Guard replication technology is used.

Usage Notes

The number of chunks can only be modified if a shardspace does not contain deployed shards.

Examples

Modify the GOLD shardspace to have 6000 chunks.

```
GDSCTL> modify shardspace -shardspace gold -chunks 6000
```

move chunk

Moves a listed set of chunks from one shard to another shard or multiple shards.

Syntax

```
move chunk -chunk {chunk_id_list | ALL}
           -source shard_name
           [-target shard_name]
           [-timeout]
           [-verbose]
```

Options

Table C-57 GDSCTL move chunk Options

Option	Description
-chunk { <i>chunk_id_list</i> ALL}	Specify a comma-separated list of chunk IDs. If -chunk ALL is specified without the -target option, all of the chunks are removed from the source shard and distributed to all of the remaining shards in a round-robin manner.
-source <i>shard_name</i>	Specify the name of the source shard.
-target <i>shard_name</i>	Specify the name of the target shard.
-timeout	Specify a connection retention time-out for the interval between when FAN is sent to the clients and a chunk going into read-only mode or down.
-verbose	Enable verbose output mode.

Usage Notes

This command can only be used with the system-managed sharding method.

Chunks cannot be moved between shards that belong to different shardgroups.

If -chunk ALL is specified without the -target option, all of the chunks are removed from the source shard and distributed to all of the remaining shards in a round-robin manner.

Examples

Move chunks 3 and 4 from SALE1 to SALE3.

```
GDSCTL> move chunk -chunk 3,4 -source sale1 -target sale3
```


Move all chunks from sale1 and distribute evenly among the remaining shards.

```
GDSCTL> move chunk -chunk ALL -source sale1
```

quit

Quit GDSCTL utility.

Syntax

```
quit | exit
```

recover shard

Executes all DDL statements on the specified shard (database), starting from the one that was previously executed with errors. The command is intended to perform all skipped DDL changes after database administrator fixes shard issues.

Syntax

```
recover shard -gdspool pool
              -shard shard_name
              [-skip_first|-ignore_first]
              [-full]
```

Options

Table C-58 GDSCTL recover shard Options

Option	Description
-full	Full recovery mode.
-gdspool <i>pool</i>	Specify the GDS pool. If not specified and there is only one GDS pool with access granted to user, it will be used by default.
-ignore_first	Make first failed DDL statement obsolete.
-shard <i>shard_name</i>	The name of the shard.
-skip_first	Skip the first failed DDL statement.

Usage Notes

Use `SKIP_FIRST` to skip first DDL. This is typically required after manual fix done by database administrator. For example, if `CREATE TABLE` statement fails because of a lack of space, the database administrator fixes the issue and re-executes `CREATE TABLE`. To avoid `ORA-39151 (table exists)` in `RECOVER SHARD` the database administrator must specify `-SKIP_FIRST`.

Use `IGNORE_FIRST` to mark the first DDL as obsolete. This is required when the wrong DDL statement was specified and failed on all shards. In this case, you need to mark it down as obsolete. `FULL` mode performs a complete recovery, including DDL operations, failed chunk migration, tablespace sets reconstruction, and database parameters.

Examples

Recover shard shd1.

```
GDSCTL> recover shard -shard shd1
```

relocate service

Stops a service on one database and starts the service on a different database.

Syntax

```
relocate service [-gdspool gdspool_name]  
                -service service_name  
                -old_db db_name  
                -new_db db_name  
                [-force]  
                [-override [-oldpwd oldpassword] [-newpwd newpassword]]
```

Options

Table C-59 GDSCTL relocate service Options

Option	Description
-force	If you use this option, then all sessions are disconnected when the service is moved, requiring the sessions using the service to reconnect (potentially to a different instance). If you do not use this option, then the sessions that are connected to a database using this service stay connected, but new sessions cannot be established to the service.
-gdspool <i>gdspool_name</i>	Specify the name of the database pool where the service is located. If not specified and there is only one <i>gdspool</i> with access granted to user, it is used as the default <i>gdspool</i> .
-new_db <i>db_name</i>	Specify the name of the database to which you want to move the service.
-newpwd <i>newpassword</i>	Specify the password for the GSMUSER in the database to which the service is being relocated (the target database).
-old_db <i>db_name</i>	Specify the name of the database where the service is currently located.
-oldpwd <i>oldpassword</i>	Specify the password for the GSMUSER in the source database, or the database where the service is currently located.
-override	This option causes the command to execute without updating the global service manager catalog. You can use this option when the catalog database is unavailable. During normal operation, you should not use this option.
-service <i>service_name</i>	Specify the name of the global service you are relocating.

Usage Notes

Unlike using the [modify service](#) command to change the location of a service, this command does not change the underlying configuration. This command temporarily relocates a service to run on another database.

If `-force` is not specified, then the global service must have been started on the old database and not running on the new database prior to command execution. If `-force` is not specified, then sessions already connected to this global service stay connected, but new sessions cannot be established.

If `-override` is specified the command will be executed without going to the GDS catalog. Use this option when the GDS catalog is unavailable. It is not recommended for use under normal operation.

If you attempt to use this command on a service that was previously configured with the `-preferred_all` option, then GDSCTL returns an error.

You must connect to the catalog database as a user with the pool administrator privileges, using the command [connect](#) before running this command

Example

Relocate the service SALES_REPORT in the READFARM database pool from the DB2 database to the DB3 database.

```
GDSCTL> relocate service -gdspool readfarm -service sales_report -
old_db db1
-new_db db3
```

remove brokerconfig

Removes an Oracle Data Guard broker configuration from a GDS pool.

Syntax

```
remove brokerconfig [-gdspool gdspool_name]
```

Options

Table C-60 GDSCTL remove brokerconfig Options

Syntax	Description
<code>-gdspool <i>gdspool_name</i></code>	Specify the GDS pool from which you want to remove the Oracle Data Guard broker configuration (not required, however, if not specified and there is only one GDS pool with access granted to the user, and it is used by default).

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the command [connect](#) before running this command.

If a GDS pool does not contain a Data Guard Broker configuration, an error is returned.

Example

Remove the Oracle Data Guard broker configuration from the database pool MYDGPOOL.

```
GDSTL> remove brokerconfig -gdspool myreaderfarm
```

remove cdb

Removes one or more CDBs from the shard catalog, but does not destroy it.

Syntax

```
remove cdb -cdb {cdb_name_list | ALL}
           [-force]
```

Options

Table C-61 GDSCTL remove cdb Options

Option	Description
-cdb {cdb_name_list ALL}	Specify a comma-delimited list of CDB names to remove, or specify ALL to remove all CDBs from the catalog.
-force	Remove one or more specified CDBs, even if they are inaccessible and/or contain PDB shards which may contain chunks. It might result in a lower number of replicas or total unavailability for a certain range of data.

WARNING:

No chunks are moved before removing the CDB which may result in data loss.

WARNING:

Forced removal of a CDB will also cause the removal of all CDBs that are replicas of the CDB being forcibly removed.

Examples

Remove the cdb named cdb1.

```
GDSTL> remove cdb -cdb cdb1
```

remove credential

Removes an existing credential.

Syntax

```
remove credential -credential credential_name
```

Options

Table C-62 GDSCTL remove credential Options

Option	Description
<code>-credential <i>credential_name</i></code>	Specify the name of the credential to remove.

Usage Notes

This command removes an existing credential. When the credential is removed, the catalog may no longer be able to execute jobs on sharded hosts in response to administrative commands.

If the specified credential does not exist, the command returns an error.

Examples

Remove a credential named `east_region_cred`.

```
GDSCTL> remove credential -credential east_region_cred
```

remove database

Removes databases from a GDS pool.

Syntax

```
remove database [-gdspool gdspool_name]  
                {-all | -database db_name_list}  
                [-force]
```

Options

Table C-63 GDSCTL remove database Options

Option	Description
<code>-all</code>	Removes all databases in the database pool.

Table C-63 (Cont.) GDSCTL remove database Options

Option	Description
<code>-database db_name_list</code>	Specify a comma-delimited list of database names that you want to remove from the database pool. You cannot specify a database that was added through an Oracle Data Guard broker configuration; you must use Oracle Data Guard to remove these databases.
<code>-gdspool gdspool_name</code>	Specify the GDS pool name. If not specified, and there is only one GDS pool with access granted to the user, it will be used by default.
<code>-force</code>	Removes the database from the catalog even if the database is not available. Using this option can result in global services not being removed from the database.

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the command [connect](#) before running this command.

If a pool already contains a Data Guard Broker configuration, an error is returned because a database must be removed through DGMGRL in this case.

With Oracle Sharding, only an undeployed database can be removed. If a database is offline or inaccessible, it has to be first undeployed with the `-force` option and then removed with the `-force` option.

Example

Remove the database DB1 from the global service management configuration.

```
GDSCTL> remove database -database db1 -gdspool pool1
```

remove file

Removes an existing file object from the catalog.

Syntax

```
remove file -file file_name
```

Options**Table C-64 GDSCTL remove file Options**

Option	Description
<code>-file file_name</code>	Specify the name of the file object to remove from the catalog.

Usage Notes

If the specified file object does not exist, the command returns an error.

Examples

Remove a file named `east_region_db_params`.

```
GDSCCTL> remove file -file east_region_db_params
```

remove gdspool

Removes a GDS pool from the current configuration.

Syntax

```
remove gdspool -gdspool gdspool_name_list
```

Options**Table C-65 GDSCCTL remove gdspool Options**

Option	Description
<code>-gdspool <i>gdspool_name_list</i></code>	Specify a comma-delimited list of GDS pool names.

Usage Notes

You must connect to the catalog database as a user with the Global Data Services administrator privileges, using the command [connect](#) before running this command.

Example

Remove the GDS pools `tempreaders` and `myfarm` from the Global Data Services framework.

```
GDSCCTL> remove gdspool -gdspool tempreaders,myfarm
```

remove gsm

Removes a global service manager from the configuration.

Syntax

```
remove gsm [-gsm gsm_name]
```

Options

Table C-66 GDSCTL remove gsm Options

Syntax	Description
<code>-gsm gsm_name</code>	Specify the name of the global service manager that you want to remove. If the name is not specified, then the current global service manager is removed.

Usage Notes

The removal of a global service manager requires at least one global service manager to be running to perform cleanup of Global Data Services databases. If there is only one global service manager in the Global Data Services configuration, then it has to be running to be removed.

You must connect to the catalog database as a user with the Global Data Services administrator privileges, using the command [connect](#) before running this command.

Example

Remove the global service manager named gsm5 from the configuration.

```
GDSCTL> remove gsm -gsm gsm5
```

remove invitednode (remove invitedsubnet)

Remove host address information from the valid node checking for registration (VNCR) list in the Global Data Services catalog. This command removes either the specified invitednode or all invitednodes that correspond to an alias.

Syntax

```
remove invitednode {[-group group_name]|vncr_id}
```

Options

Table C-67 GDSCTL remove invitednode (remove invitedsubnet) Options

Option	Description
<code>-group group_name</code>	Specify an alias which defines a group of VNCRs. This alias can be referenced in other commands related to invited nodes.
<code>vncr_id</code>	The host address information, which can be an IPv4 or IPv6 address, a host name, a netmask, or other identifier for a server. The host address information cannot contain any spaces.

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the command [connect](#) before running this command

Examples

Remove the invitednode 198.51.100.22 from the catalog.

```
GDSCCTL> remove invitednode 198.51.100.22
```

Remove the VNCR alias group EAST_SRV from the catalog.

```
GDSCCTL> remove invitednode -group east_srv
```

remove region

Removes the specified regions from the global service management framework.

Syntax

```
remove region -region region_list
```

Options

Table C-68 GDSCCTL remove region Options

Option	Description
-region <i>region_list</i>	Specify a comma-delimited list of region names

Usage Notes

You must connect to the catalog database as a user with the Global Data Services administrator privileges, using the [connect](#) command before running this command.

Example

Remove the region named SOUTH from the configuration.

```
GDSCCTL> remove region -region south
```

remove service

Removes a service from a database pool.

Syntax

```
remove service [-gdspool gdspool_name]  
               -service service_name
```

Options

Table C-69 GDSCTL remove service Options

Option	Description
<code>-gdspool <i>gdspool_name</i></code>	Specify the name of the GDS pool from which you want to remove the service. If not specified and there is only one <i>gdspool</i> with access granted to user, then it is used as the default <i>gdspool</i> .
<code>-service <i>service_name</i></code>	Specify the name of the service that you want to remove.

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the command [connect](#) before running this command

Example

Remove the service `sales_report` from the database pool MYREADERFARM.

```
GDSCTL> remove service -gdspool myreaderfarm -service sales_report
```



See Also:

[Deleting a Global Service](#)

remove shard

Removes one or more shards from the sharded database.

Syntax

```
remove shard {-shard {shard_name_list | ALL} |
             -shardspace shardspace_list |
             -shardgroup shardgroup_list}
[-force]
```

Options

Table C-70 GDSCTL remove shard Options

Option	Description
<code>-shard {<i>shard_name_list</i> ALL}</code>	Specify a comma-delimited list of shard names to remove, or specify <code>ALL</code> to remove all shards from the catalog.
<code>-shardspace <i>shardspace_list</i></code>	Specify a comma-delimited list of names of shardspaces from which to remove all shards.
<code>-shardgroup <i>shardgroup_list</i></code>	Specify a comma-delimited list of names of shardgroups from which to remove all shards.

Table C-70 (Cont.) GDSCTL remove shard Options

Option	Description
-force	Remove one or more specified shards, even if they are inaccessible and/or contain chunks. It might result in a lower number of replicas or total unavailability for a certain range of data.

⚠ WARNING:

No chunks are moved before removing the shard which may result in data loss.

⚠ WARNING:

Forced removal of a shard will also cause the removal of all shards that are replicas of the shard being forcibly removed.

Examples

Remove the shards from shardgroup GROUP1.

```
GDSCTL> remove shard -shardgroup group1
```

remove shardgroup

Removes a shardgroup from the shard catalog.

Syntax

```
remove shardgroup -shardgroup shardgroup_name
```

Options

Table C-71 GDSCTL remove shardgroup Options

Option	Description
-shardgroup <i>shardgroup_name</i>	Specify the name of the shardgroup to be removed.

Usage Notes

Only a shardgroup that does not contain any shards can be removed.

Examples

Remove the GROUP1 shardgroup.

```
GDSCCTL> remove shardgroup -shardgroup group1
```

remove shardspace

Removes a shardspace from the shard catalog.

Syntax

```
remove shardspace -shardspace shardspace_name
```

Options

Table C-72 GDSCCTL remove shardspace Options

Option	Description
<code>-shardspace <i>shardspace_name</i></code>	Specify the name of the shardspace to be removed.

Usage Notes

Only a shardspace that does not contain any shards or shardgroups can be removed.

Examples

Remove the GOLD shardspace.

```
GDSCCTL> remove shardspace -shardspace gold
```

services

Retrieves information about the services that are registered with the specified global service manager.

Syntax

```
[status service | services] [-gsm gsm_name]  
                             [-service service_name]  
                             [-raw | -verbose | -support]
```

Options

Table C-73 GDSCCTL services Options

Option	Description
<code>-gsm <i>gsm_name</i></code>	Specify the name of a global service manager. If the name is not specified, then GDSCCTL uses the current global service manager name for the session (specified with the GDSCCTL <code>set gsm</code> command).

Table C-73 (Cont.) GDSCTL services Options

Option	Description
-raw	If specified, GDSCTL output is presented in a raw, non-parsed format.
-service <i>service_name</i>	Specify a fully qualified service name. If the service name is not specified, then the information about all the services registered with the global service manager is retrieved.
-support	If specified, GDSCTL output displays additional information.
-verbose	Enables verbose output mode.

Usage Notes

You must run this command on the host where the global service manager for which you want to retrieve service information resides.

You must have the privileges of the user who started the global service manager to run this command.

If `-service` is not specified, then information for all global services is displayed.

Example

Display information about the services registered with global service manager `mygsm`:

```
GDSCTL> services -gsm mygsm
```

The `gdsctl services` command returns output similar to the following:

```
GDSCTL>services -gsm mygsm
Service "localsvc.dbpoolora.oradbcloud" has 2 instance(s). Affinity: LOCALPREF
  Instance "dbpoolora%1", name: "gdscat", db: "gdscat", region: "regionora",
  status: ready.
  Instance "dbpoolora%11", name: "gdscat2", db: "gdscat2", region: "regionora",
  status: ready.
Service "sales_report1.dbpoolora.oradbcloud" has 2 instance(s). Affinity:
LOCALONLY
  Instance "dbpoolora%1", name: "gdscat", db: "gdscat", region: "regionora",
  status: ready.
  Instance "dbpoolora%11", name: "gdscat2", db: "gdscat2", region: "regionora",
  status: ready.
Service "sales_report2.dbpoolora.oradbcloud" has 2 instance(s). Affinity:
ANYWHERE
  Instance "dbpoolora%1", name: "gdscat", db: "gdscat", region: "regionora",
  status: ready.
  Instance "dbpoolora%11", name: "gdscat2", db: "gdscat2", region: "regionora",
  status: ready.
```

 **Note:**

Affinity values can be `LOCALONLY` when the service locality is defined as `local_only`, `LOCALPREF` when the service locality is defined as `local_only` with the `region_failover` option enabled, and `ANYWHERE` when the service locality is defined as `anywhere`.

Display the status of `mtgly_report` service:

```
GDSCTL>services -service mtgly_report.sales.oradbcloud
```

Returns output similar to the following:

```
Service "mtgly_report.sales.oradbcloud" has 1 instance(s). Affinity:
ANYWHERE
Instance "sales%1", name: "debug", db: "debug", region: "eastcoast",
status: ready.
```

set gsm

Sets the global service manager for the current session.

This command establishes to which global service manager the successive commands apply. The specified global service manager name is resolved in the `gsm.ora` configuration file.

Syntax

```
set gsm -gsm gsm_name
```

Options

Table C-74 GDSCTL set gsm Options

Syntax	Description
<code>-gsm <i>gsm_name</i></code>	Specify the name of the global service manager to work with in the current session. If you do not specify a specific global service manager, then GDSCTL uses the default global service manager name of <code>GSMORA</code> .

Usage Notes

You must run this command on the host where the global service manager that you want to set for the current session resides.

You must have the privileges of the user who started the global service manager to run this command.

Example

Set the global service manager for the current session to `gsm1`.

```
GDSCTL> set gsm -gsm gsm1
```

set inbound_connect_level

Sets the `INBOUND_CONNECT_LEVEL` listener parameter.

Syntax

```
set inbound_connect_level [-gsm gsm_name]  
                           timeout_value
```

Options

Table C-75 GDSCTL set inbound_connect_level Options

Option	Description
<code>-gsm <i>gsm_name</i></code>	Specify the name of the global service manager that you want to start. If you do not specify a specific global service manager, then GDSCTL uses the current global service manager name for the session (specified with the command set gsm).
<code><i>timeout_value</i></code>	Specify in seconds the connection timeout value.

Usage Notes

- You must run this command on the host where the global service manager for which you want to set the `INBOUND_CONNECT_LEVEL` listener parameter resides
- You must have the privileges of the user who started the global service manager to run this command

Example

Set the `INBOUND_CONNECT_LEVEL` listener parameter for `mygsm` to time out in 60 seconds:

```
GDSCLTL> set inbound_connect_level -gsm mygsm 60
```

set inbound_connect_timeout

Sets the `INBOUND_CONNECT_TIMEOUT` listener parameter.

Syntax

```
set inbound_connect_timeout timeout_value  
                             [-gsm gsm_name]  
                             [-save_config | -config_only]
```

Options

Table C-76 GDSCTL set inbound_connect_timeout Options

Option	Description
-config_only	Update GSM.ORA only, without trying to connect to a running global service manager instance.
-gsm <i>gsm_name</i>	Specify the name of the global service manager that you want to start. If you do not specify a specific global service manager, then GDSCTL uses the current global service manager name for the session (specified with the command set gsm).
-save_config	Store configuration changes to GSM.ORA.
<i>timeout_value</i>	Specify in seconds the connection timeout value.

Usage Notes

- You must run this command on the host where the global service manager for which you want to set the `INBOUND_CONNECT_TIMEOUT` listener parameter resides
- You must have the privileges of the user who started the global service manager to run this command
- By default, parameter values changes remain in effect until the global service manager is shut down.

Example

Set the `INBOUND_CONNECT_TIMEOUT` listener parameter for `mygsm` to time out in 60 seconds:

```
GDSCLTL> set inbound_connect_timeout -gsm mygsm 60
```

set log_level

Sets the log level for the listener associated with a specific global service manager.

Syntax

```
set log_level [-gsm gsm_name]
              log_level
```

Options

Table C-77 GDSCTL set log_level Options

Option	Description
-gsm <i>gsm_name</i>	Specify the name of the global service manager.
<i>log_level</i>	Specify the level of detail to write to the log. Valid values are ON or OFF.

Usage Notes

- You must run this command on the host where the global service manager for which you want to set the listener log level resides
- You must have the privileges of the user who started the global service manager to run this command

Example

Set logging on for global service manager `mygsm`:

```
GDSCCTL> set log_level -gsm mygsm ON
```

set log_status

Sets the `LOG_STATUS` listener parameter.

Syntax

```
set log_status ON|OFF
           [-gsm gsm_name]
           [-save_config | -config_only]
```

Options

Table C-78 GDSCCTL set log_status Options

Option	Description
ON OFF	Turns listener logging on or off.
-config_only	Update GSM.ORA only, without trying to connect to a running global service manager instance.
-gsm <i>gsm_name</i>	Specify the name of the global service manager that you want to start. If you do not specify a specific global service manager, then GDSCCTL uses the current global service manager name for the session (specified with the command set gsm).
-save_config	Store configuration changes to GSM.ORA.

Usage Notes

- You must run this command on the host where the global service manager for which you want to set the `LOG_STATUS` listener parameter resides
- You must have the privileges of the user who started the global service manager to run this command
- By default, parameter values changes remain in effect until the global service manager is shut down.

Example

Set the `LOG_STATUS` listener parameter to ON.

```
GDSCCTL> set log_status on -save_config
```

set outbound_connect_level

Sets the timeout value for the outbound connections for the listener associated with a specific global service manager.

Syntax

```
set outbound_connect_level [-gsm gsm_name]  
                           timeout_value
```

Options

Table C-79 GDSCTL set outbound_connect_level Options

Option	Description
<i>-gsm gsm_name</i>	Specify the name of the global service manager
<i>timeout_value</i>	Specify the connection timeout value

Usage Notes

- You must run this command on the host where the global service manager for which you want to set the timeout value of outbound connections for the listener resides.
- You must have the privileges of the user who started the global service manager to run this command.

Example

Set timeout value for all outbound connections:

```
GDSCTL> set outbound_connect_level 60
```

set outbound_connect_timeout

Sets the `OUTBOUND_CONNECT_TIMEOUT` listener parameter.

Syntax

```
set outbound_connect_timeout timeout_value  
                             [-gsm gsm_name]  
                             [-save_config | -config_only]
```

Options

Table C-80 GDSCTL set outbound_connect_timeout Options

Option	Description
<i>timeout_value</i>	Specify in seconds the connection timeout value.
<i>-config_only</i>	Update GSM.ORA only, without trying to connect to a running global service manager instance.

Table C-80 (Cont.) GDSCTL set outbound_connect_timeout Options

Option	Description
-gsm <i>gsm_name</i>	Specify the name of the global service manager that you want to start. If you do not specify a specific global service manager, then GDSCTL uses the current global service manager name for the session (specified with the command set gsm).
-save_config	Store configuration changes to GSM.ORA.

Usage Notes

- You must run this command on the host where the global service manager for which you want to set the `OUTBOUND_CONNECT_TIMEOUT` listener parameter resides
- You must have the privileges of the user who started the global service manager to run this command
- By default, parameter values changes remain in effect until the global service manager is shut down.

Example

Set the `OUTBOUND_CONNECT_TIMEOUT` listener parameter for `mygsm` to time out in 60 seconds:

```
GDSCLTL> set outbound_connect_timeout -gsm mygsm 60
```

set trace_level

Sets the trace level for the listener associated with the specified global service manager.

Syntax

```
set trace_level [-gsm gsm_name]
               trace_level
```

Options**Table C-81 GDSCTL set trace_level Options**

Option	Description
-gsm <i>gsm_name</i>	Specify the name of the global service manager. If the name is not specified, then GDSCTL uses the current global service manager name for the session (specified with the GDSCTL <code>set gsm</code> command).

Table C-81 (Cont.) GDSCTL set trace_level Options

Option	Description
<i>trace_level</i>	Specify the trace level for the global service manager listener. Valid values are USER - provides traces to identify user-induced error conditions ADMIN - provides traces to identify installation-specific problems SUPPORT - provides trace with troubleshooting information for Oracle Support Services OFF - provides no tracing

Usage Notes

- You must run this command on the host where the global service manager for which you want to set the listener trace level resides.
- You must have the privileges of the user who started the global service manager to run this command.

Example

Set the trace level for all listeners associated with `mygsm` to ADMIN

```
GDSCTL> set trace_level -gsm mygsm ADMIN
```

set trc_level

Sets the `TRC_LEVEL` listener parameter.

Syntax

```
set trc_level trace_level
           [-gsm gsm_name]
           [-save_config | -config_only]
```

Options**Table C-82 GDSCTL set trc_level Options**

Option	Description
<i>trace_level</i>	Specify the trace level for the global service manager listener. Valid values are USER provides traces to identify user-induced error conditions ADMIN provides traces to identify installation-specific problems SUPPORT provides trace with troubleshooting information for Oracle Support Services OFF provides no tracing

Table C-82 (Cont.) GDSCTL set trc_level Options

Option	Description
-config_only	Update GSM.ORA only, without trying to connect to a running global service manager instance.
-gsm <i>gsm_name</i>	Specify the name of the global service manager that you want to start. If you do not specify a specific global service manager, then GDSCTL uses the current global service manager name for the session (specified with the command set gsm).
-save_config	Store configuration changes to GSM.ORA.

Usage Notes

- You must run this command on the host where the global service manager for which you want to set the LOG_STATUS listener parameter resides
- You must have the privileges of the user who started the global service manager to run this command
- By default, parameter values changes remain in effect until the global service manager is shut down.

Example

Set the TRC_LEVEL listener parameter to SUPPORT.

```
GDSCLTL> set trc_level support
```

show ddl

Shows DDL statements execution status.

Syntax

```
show ddl {[ddl ddl_id] [-count cnt] | [-failed_only]}
        [-support]
```

Options**Table C-83 GDSCTL show ddl Options**

Option	Description
-count <i>cnt</i>	The maximum number of entries to display.
-ddl <i>ddl_id</i>	DDL numeric identifier.
-failed_only	Use this option to display only errored out statements.
-support	If specified, GDSCTL output displays additional support information.

Usage Notes

If -DDL and -COUNT are both unspecified, the command returns only the last 10 DDL statements.

If `-DDL` is specified but `-COUNT` is not, the command returns detailed information about the DDL statement. The `-COUNT` option defines the maximum number of DDLs to display.

Examples

```
GDSCCTL> show ddl -count 20
```



Note:

The `show ddl` command output might be truncated. You can run `SELECT ddl_text FROM gsmadmin_internal.ddl_requests` on the catalog to see the full text of the statements.

split chunk

Splits each of the specified chunks into two chunks with an equal number of records. After the split, the chunks remain in the same shard.

Syntax

```
split chunk -chunk chunk_id_list
            [-shardspace shard_space_list]
```

Options

Table C-84 GDSCCTL split chunk Options

Option	Description
<code>-chunk <i>chunk_id_list</i></code>	Specify a comma-separated list of numeric chunk identifiers.
<code>-shardspace <i>shard_space_list</i></code>	Specify a list of shardspace names in which to split the specified chunks.

Usage Notes

This command can only be used with system-managed sharding. For user-defined sharding, `ALTER TABLE` is used to split a partition of the root (parent) table.

Merging of chunks is not supported.

Examples

Split chunks 3, 4, and 5.

```
GDSCCTL> split chunk -chunk 3,4,5
```

sql

Executes a SQL statement or a PL/SQL stored procedure against a sharded database.

Syntax

```
sql "sql_statement"
```

Options

Table C-85 GDSCTL sql Options

Option	Description
<i>sql_statement</i>	Enter the SQL statement or PL/SQL stored procedure to be executed. Do not include a semi-colon (;) after the SQL statement to be executed.

Usage Notes

This command can only be executed against a sharded GDS pool. The statements are executed in the GDS catalog database and are then broadcast to all shards in the pool. Since there can be only one sharded pool in a GDS configuration, all SQL statements executed on the catalog database are applied to this pool, if it exists.

Database objects created by this command in the catalog database are used as a schema of the sharded database and are not intended to store user data. The only exception is tables duplicated on all shards (reference tables) – they are populated with data in the catalog database.

SELECT statements are not allowed as the parameter.

The SQL statement or PL/SQL stored procedure to be executed must be enclosed in double quotation marks.

If the string that GDSCTL passes to PL/SQL contains a filename, then the filename must be enclosed in single quotation marks.

Do not include a semi-colon (;) after the SQL statement to be executed.

Examples

Using the `gdsctl sql` command.

```
GDSCTL> sql "CREATE TABLESPACE SET ts1 IN SHARDGROUP sgr1"
```

start gsm

Starts a specific global service manager.

Syntax

```
start gsm [-gsm gsm_name]
```

Options

Table C-86 GDSCTL start gsm Options

Option	Description
<code>-gsm <i>gsm_name</i></code>	Specify the name of the global service manager that you want to start. If you do not specify a specific global service manager, then GDSCTL uses the current global service manager name for the session (specified with the command set gsm).

Usage Notes

- You must run GDSCTL on the same host where the global service manager you want to start is located.
- You must have operating system privileges on the computer where you want to start the global service manager to run this command.

Example

Start the global service manager `gsm1` on the local host.

```
GDSCTL> start gsm -gsm gsm1
```

start observer

Starts specific services.

Syntax

```
start observer -database db_name
               [-timeout seconds]
```

Options

Table C-87 GDSCTL start observer Options

Option	Description
<code>-database <i>db_name</i></code>	The name of the database.
<code>-timeout <i>seconds</i></code>	The global service manager requests timeout in seconds.

Usage Notes

`TIMEOUT` (default 15) represents the time between when the shard director/global service manager receives requests and starts the observer. See *Using Oracle Sharding* for the automatic rules for choosing the right region for the shard director (global service manager) server to start the observer. If shard director servers are not running in this region, the observer is not started.

Example

```
GDSCTL> start observer -database mydb
```

start service

Starts specific services.

Syntax

```
start service [-gdspool gdspool_name]  
              -service service_name  
              [{-database db_name |  
               -override [-pwd password] -connect  
               connect_identifier}]
```

Options

Table C-88 GDSCTL start service Options

Option	Description
<code>-database <i>db_name</i></code>	Specify the name of the database on which you want to start the service. If you do not specify this option, then GDSCTL starts the services on all preferred databases.
<code>-connect <i>connect_identifier</i></code>	Specify an Oracle Net connect descriptor or net service name that resolves to a connect descriptor.
<code>-gdspool <i>gdspool_name</i></code>	Specify the name of the database pool in which the services that you want to start are located. If not specified and there is only one <code>gdspool</code> with access granted to the user, it is used as the default <code>gdspool</code> .
<code>-override</code>	This option causes the command to run without updating the global service manager catalog. You can use this option when the catalog database is unavailable. During normal operation, you should not use this option.
<code>-pwd <i>password</i></code>	Specify the password of the GSMUSER in the specified database.
<code>-service <i>service_name</i></code>	Specify a comma-delimited list of global service names. If you do not use this option, then GDSCTL starts all the services in the database pool.

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the command [connect](#) before running this command.

Before starting services which run on administrator-managed databases, they must be modified for those databases to stipulate which instances should run the service. Please refer to the `-modify_instances` parameter of the `modify service` command.

Example

Start the service SALES_REPORT, located in the READERFARM database pool.

```
GDSCTL> start service -gdspool readerfarm -service sales_report
```

**See Also:**

[Starting a Global Service](#)

[modify service](#)

status

Displays the running status and runtime information for the global service manager.

Syntax

```
status [-gsm gsm_name] [-raw|-verbose|-support]
```

Options**Table C-89 GDSCTL status Options**

Option	Description
-gsm <i>gsm_name</i>	Specify the name of a global service manager to check. If the name is not specified, then GDSCTL uses the current global service manager name for the session (specified with the GDSCTL set gsm command).
-raw	If specified, GDSCTL output is presented in a raw non-parsed format.
-support	If specified, GDSCTL output displays additional information.
-verbose	Enable verbose mode.

Example

```
GDSCTL> status
```

The command returns output similar to the following.

```
Alias MYGSM
Version 19.3.0.0.0
Start Date 03-JUL-2020 16:48:54
Trace Level support
Listener Log File /u01/ORACLE/mygsm/alert/log.xml
```

```

Listener Trace File /u01/ORACLE/mygsm/trace/
ora_14816_47568108067776.trc
Endpoint summary (ADDRESS=(HOST=mymv.us.hq.com) (PORT=1523)
(PROTOCOL=tcp))
GSMOCI Version 0.1.8
Mastership Y
Connected to GDS catalog Y
Process Id 14818
Number of reconnections 0
Pending tasks. Total 0
Tasks in process. Total 0
Regional Mastership TRUE
Total messages published 28599
Time Zone -07:00
Orphaned Buddy Regions:
None
GDS region regionora

```

status database

Displays the runtime status of databases, such as registration information, services, and so on.

Syntax

```

{status database | databases} [-gsm gsm_name]
                               [-database db_name]
                               [-gdspool gdspool_name]
                               [-raw | -support | -verbose]

```

Options

Table C-90 GDSCTL status database Options

Option	Description
-database <i>db_name</i>	Specify the name of the database on which to check status.
-gdspool <i>gdspool_name</i>	Specify the name of the database pool. If not specified and there is only one gdspool with access granted to the user, it is used as the default gdspool.
-gsm <i>gsm_name</i>	Specify the name of a global service manager to check. If the name is not specified, then GDSCTL uses the current global service manager name for the session (specified with the GDSCTL <code>set gsm</code> command).
-raw	If specified, GDSCTL output is presented in a raw, non-parsed format.
-support	If specified, GDSCTL output displays additional support information.
-verbose	Enable verbose output mode.

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the command `connect` before running this command.

This command requires a locally started global service manager. If `-gsm` is not specified for `STATUS DATABASE`, then the currently connected global service manager name is used by default.

Example

Display the status of all databases:

```
GDSCTL> status database
```

The `gdsctl status database` command returns output similar to the following:

```
Database: "dbcat1" Registered: Y State: Ok ONS: N. Role: PRIMARY Instances: 1
Region: east

Service: "sales_svc2" Globally started: N Started: N
Scan: Y Enabled: Y Preferred: Y
Service: "sales_svc1" Globally started: Y Started: Y
Scan: N Enabled: Y Preferred: Y
Registered instances:
sales%11
Database: "dbcat2" Registered: Y State: Ok ONS: N. Role: PRIMARY Instances: 1
Region: east
Service: "sales_svc2" Globally started: N Started: N
Scan: Y Enabled: Y Preferred: Y
Service: "sales_svc1" Globally started: Y Started: Y
Scan: N Enabled: Y Preferred: Y
Registered instances:
sales%1
```

status gsm

Displays the status of a specific global service manager.

Syntax

```
status (gsm)? [-gsm gsm_name]
                [-raw | -verbose | -support]
```

Options

Table C-91 GDSCTL status gsm Options

Option	Description
<code>-gsm gsm_name</code>	Specify the name of a global service manager to check. If the name is not specified, then GDSCTL uses the current global service manager name for the session (specified with the GDSCTL <code>set gsm</code> command).
<code>-raw</code>	If specified, GDSCTL output is presented in a raw, non-parsed format.
<code>-support</code>	If specified, GDSCTL output displays additional support information.
<code>-verbose</code>	Enable verbose output mode.

Usage Notes

You must run GDSCTL on the same host where the global service manager for which you want to display the status is located.

You must have operating system privileges on the computer where you want to display the global service manager status to run this command.

Example

Display status of `mygsm`:

```
GDSCTL> status gsm -gsm mygsm
```

The `gdscctl status gsm` command returns output similar to the following:

```
Alias MYGSM
Version 19.3.0.0.0
Start Date 03-JUL-2020 16:48:54
Trace Level support
Listener Log File /u01/ORACLE/mygsm/alert/log.xml
Listener Trace File /u01/ORACLE/mygsm/trace/
ora_14816_47568108067776.trc
Endpoint summary (ADDRESS=(HOST=mymv.us.hq.com) (PORT=1523)
(PROTOCOL=tcp))
GSMOCI Version 0.1.8
Mastership Y
Connected to GDS catalog Y
Process Id 14818
Number of reconnections 0
Pending tasks. Total 0
Tasks in process. Total 0
Regional Mastership TRUE
Total messages published 28599
Time Zone -07:00
Orphaned Buddy Regions:
```

None
GDS region regionora

status service

Displays the status of a specific service.

Syntax

```
{status service | services} [-gsm gsm_name]
                             [-service service_name]
                             [{-raw|-verbose|-support}]
```

Options

Table C-92 GDSCTL status service Options

Option	Description
-gsm <i>gsm_name</i>	Specify the name of a global service manager. If the name is not specified, then GDSCTL uses the current global service manager name for the session (specified with the GDSCTL set <i>gsm</i> command).
-raw	Used by oracle internal components.
-service <i>service_name</i>	Specify a comma-delimited list of global service names. If you do not specify any services, then GDSCTL displays the status of all services in the database pool.
-support	Display more detailed information concerning load balancing.
-verbose	Display extra information related to load balancing.

Usage Notes

- You must connect to the catalog database as a user with the pool administrator privileges, using the command [connect](#) before running this command.
- This command is similar to [services](#).

Example

Display the status of service `sales_report1.sales.oradbcloud`:

```
GDSCTL> status service -service sales_report1.sales.oradbcloud
```

The `gdscctl status service` command returns output similar to the following:

```
Service "sales_report1.sales.oradbcloud" has 3 instance(s). Affinity:
ANYWHERE
  Instance "sales%1", name: "dbcat2", db: "dbcat2", region: "east",
status: ready.
  Instance "sales%11", name: "dbcat1", db: "dbcat1", region: "west",
status: ready.
```

```
Instance "sales%31", name: "dbcat3", db: "dbcat3", region: "east",
status: ready.
```

stop gsm

Stops a specific global service manager.

Syntax

```
stop gsm [-gsm gsm_name]
```

Options

Table C-93 GDSCTL stop gsm Options

Option	Description
<code>-gsm <i>gsm_name</i></code>	Specify the name of a global service manager you want to stop. If you do not specify a specific global service manager, then GDSCTL uses the current global service manager name for the session (specified with the command set gsm).

Usage Notes

- You must run GDSCTL on the same host where the global service manager that you want to stop is located.
- You must have operating system privileges on the computer where you want to start the global service manager to run this command.

Example

Stop the global service manager `gsm1` on the local host.

```
GDSCTL> stop gsm -gsm gsm1
```

stop service

Stops the specified global services.

Syntax

```
stop service [-gdspool gdspool_name
              [-service service_name_list
               [{-database db_name |
                 -override -connect connect_identifier [-pwd
                 password]}]}
              [-force]
              [-drain_timeout time]
              [-stop_option {NONE|IMMEDIATE|TRANSACTIONAL}]
```

Options

Table C-94 GDSCTL stop service Options

Option	Description
<code>-connect connect_identifier</code>	Specify an Oracle Net connect descriptor or net service name that resolves to a connect descriptor for the database (or shard).
<code>-database db_name</code>	Specify the name of the database on which you want to stop the service. If you do not specify this option, then GDSCTL stops the services on all databases on which the service is currently running.
<code>-drain_timeout</code>	Set drain time in seconds.
<code>-force</code>	If you use this option, then GDSCTL disconnects all sessions when the service is stopped, requiring the sessions using the service to reconnect (potentially to a different instance). If you do not use this option, then the sessions that are connected to a database using this service remain connected, but new sessions cannot be established to the service.
<code>-gdspool gdspool_name</code>	Specify the name of the GDS pool in which the service that you want to stop is located. If not specified and there is only one GDS pool with access granted to user, that GDS pool is used as the default GDS pool.
<code>-override</code>	This option causes the command to execute without updating the global service manager catalog. You can use this option when the catalog database is unavailable. During normal operation, you should not use this option.
<code>-pwd password</code>	Specify the password of the GSMUSER in the specified database.
<code>-service service_name</code>	Specify a comma-delimited list of global service names you want to stop. If you do not use this option, then GDSCTL stops all the services in the database pool.
<code>-stop_option</code>	Set the default stop option to NONE, IMMEDIATE, or TRANSACTIONAL

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the command [connect](#) before running this command.

If `-service` is not specified, all global services of GDS pool are stopped.

If `-database` is not specified, the global services are stopped on all of the databases.


If `-force` is specified, all sessions are disconnected, requiring the session using the global service to reconnect (potentially to another instance). If `-force` is not specified, then sessions already connected to this global service stay connected, but new sessions cannot be established to the global service.

If `-override` is specified, the command is executed without connecting to the GDS catalog. Use this option when the GDS catalog is unavailable. It is not recommended for use under normal operation.

Example

Stop the service `SALES_REPORT`, on all databases in the database pool `READERFARM`.

```
GDSCCTL> stop service -gdspool readerfarm -service sales_report
```

 **See Also:**
[Stopping a Global Service](#)

sync brokerconfig (synchronize brokerconfig)

Synchronizes the Oracle Data Guard broker configuration in the global service manager with the configuration in the database pool. The `synchronize brokerconfig` command has the same options and usage.

Syntax

```
sync[hronize] brokerconfig [-gdspool gdspool_name]  
                           [-database db_name]
```

Options

Table C-95 GDSCCTL sync brokerconfig Options

Option	Description
<code>-database <i>db_name</i></code>	Specify the name of a database in the database pool to use as a referential database , from which the configuration is queried. If you do not use this option, then GDSCCTL uses the primary database as the referential database. If a primary database does not exist in the Oracle Data Guard broker configuration, then GDSCCTL uses a random database from the pool as the referential database.
<code>-gdspool <i>gdspool_name</i></code>	Specify the database pool to which the Oracle Data Guard broker configuration belongs. If not specified and there is only one <code>gdspool</code> with access granted to user, that <code>gdspool</code> is used as the default <code>gdspool</code> . If the specified database pool does not contain an Oracle Data Guard broker configuration, then GDSCCTL returns an error.

Usage Notes

You must connect to the catalog database as a user with the pool administrator privileges, using the command `connect` before running this command.

Example

Synchronize the Oracle Data Guard broker configuration in the database pool `MYREADERFARM` with the configuration stored in the Global Data Services catalog.

```
GDSCTL> sync brokerconfig -gdspool myreaderfarm
```

sync database (synchronize database)

Synchronizes attributes of global services and GDS related parameters of a GDS pool database with the contents of the GDS catalog. The `synchronize database` command has the same options and usage.

Syntax

```
sync[hronize] database [-gdspool gdspool_name]  
                        [-database database_name]
```

Options**Table C-96 GDSCTL sync database Options**

Option	Description
<code>-database <i>database_name</i></code>	Specify the name of a database in the database pool to use as a <i>referential database</i> , from which the configuration is queried.
<code>-gdspool <i>gdspool_name</i></code>	Specify the GDS pool to which the database belongs. If not specified and there is only one GDS pool with access granted to user, it is used as the default GDS pool.

Usage Notes

- If database has no GDS region assigned, an error is returned.
- If a GDS pool is specified and the database option is not specified, then each database in the pool is synchronized.

Example

Synchronize a database in the default database pool with the database `mydb`.

```
GDSCTL> sync database -database mydb
```

validate catalog

Cross checks the Global Data Services catalog, global service manager run-time status, and pool databases, and reports inconsistencies and errors.

Syntax

```
validate [catalog]
        [-gsm gsm_name]
        [ {-config | -database db_name} ]
        [-catpwd cpwd]
        [-dbpwd dpwd]
```

Options

Table C-97 GDSCTL validate catalog Options

Option	Description
<code>-catpwd <i>cpwd</i></code>	Provides the GSMCATUSER password, otherwise it is read from the local wallet file by default.
<code>-config</code>	Indicates that the validation should be performed on the Global Data Services catalog configuration only.
<code>-database <i>db_name</i></code>	Indicates the name of the database for which the cross-check validation should be performed.
<code>-dbpwd <i>dpwd</i></code>	Provides the pool database password directly if there is only one database in the pool, or if multiple databases in the pool share the same password.
<code>-gsm <i>gsm_name</i></code>	Specify the global service manager name. If the name is not specified, then GDSCTL uses the current global service manager name for the session (specified with the command set gsm).

Usage Notes

Because the execution of this command involves accessing all databases in a Global Data Services configuration, the GSMCATUSER password is required run it. The password is stored in the wallet of any global service manager that is part of the Global Data Services configuration. Therefore, if you run the command from the ORACLE_HOME of any of the global service managers, the password is automatically extracted from the wallet and does not have to be provided. Otherwise, you must provide the GSMCATUSER password using the `-catpwd` command option. Alternatively, if all databases in the Global Data Services configuration have the same GSMUSER password, you can specify the password instead of the GSMCATUSER password by using the `-dbpwd` option.

Example

Validate the catalog:

```
GDSCTL> validate
```

The output should be similar to the following:

```
Validation results:
VLD2: Region "regionora" does not have buddy region
VLD11: GDS pool "marketing" does not contain any databases
VLD12: GDS pool "marketing" does not contain any global services
VLD11: GDS pool "sales" does not contain any databases
VLD12: GDS pool "sales" does not contain any global services
VLD11: GDS pool "mkt" does not contain any databases
VLD12: GDS pool "mkt" does not contain any global services
```

validate

Cross checks the GDS catalog, global service manager run-time status, and databases from the GDS pool and reports any inconsistencies and errors.

Syntax

```
validate [catalog] [-gsm gsm]
          [-config | -database db_name [-dbpwd sipwd]]
          [-catpwd cpwd]
```

Options

Table C-98 GDSCTL validate Options

Option	Description
-catpwd <i>cpwd</i>	GSMCATUSER password.
-config	If specified, performs validation of GDS catalog configuration only.
-database <i>db_name</i>	Performs cross-check validation of the specified database.
-dbpwd <i>sipwd</i>	GSMUSER password.
-gsm <i>gsm</i>	Global service manager name

Usage Notes

If no options are specified, cross-checks are performed on the GDS catalog, database, and local global service manager.

Example

```
GDSCTL> validate catalog -catpwd cpwd -dbpwd sipwd
```

Glossary

catalog database

The Oracle Database in which the Global Data Services catalog resides.

endpoint

The address or connection point to a global service manager or listener.

GDSCTL

Global Data Services command-line interface.

Global Data Services catalog

A repository that holds configuration and run-time status of a Global Data Services configuration, including data on global services, their attributes, and all logical and physical components of the configuration, such as Global Data Services pools, Global Data Services regions, global service managers, and database instances. The catalog may also contain data on replication and network topologies related to the configuration.

Global Data Services configuration

A set of databases that are integrated by the Global Data Services framework into a single virtual server that offers one or more global services, while ensuring high performance, availability, and optimal utilization of resources.

Global Data Services pool

A set of databases within a GDS configuration that provides a unique set of global services and belongs to a certain administrative domain.

Global Data Services region

A logical boundary that contains database clients and servers that are considered to be in proximity to each other.

global service

A database service that can be provided by multiple databases synchronized through data replication.

global service manager

A software component that provides service-level load balancing and centralized management of services within the Global Data Services configuration.

global service

A service that is offered on only one database of a Global Data Services pool at a time.

Oracle Notification Service (ONS)

A publish and subscribe service for communicating information about all FAN events.

valid node checking for registration list

See [VNCR](#).

VNCR

Valid node checking for registration. Allows or denies access from specified IP addresses to Oracle Global Data Services pool.

Index

A

- add cdb, [C-2](#)
- affinity
 - any-region, [1-13](#)
 - local region, [1-13](#)
 - local region with failover, [1-13](#)
- any-region affinity, [1-13](#)

B

- broker configuration, [1-7](#)
- brokerconfig
 - GDSCTL object name, [2-21](#)

C

- catalog
 - GDSCTL object name, [2-21](#)

D

- database
 - GDSCTL object name, [2-21](#)
- database cardinality, [1-8](#)
- databases, [C-50](#)

G

- gdsctl
 - scripting commands, [2-19](#)
- GDSCTL
 - command syntax, [2-21](#)
 - commands
 - add brokerconfig, [C-1](#)
 - add cdb, [C-2](#)
 - add credential, [C-3](#)
 - add database, [C-4](#)
 - add file, [C-6](#)
 - add gds pool, [C-7](#)
 - add gsm, [2-19](#), [C-8](#)
 - add invitednode (add invitedsubnet),
[C-10](#)
 - add region, [C-11](#)

GDSCTL (continued)

- commands (continued)
 - add service, [C-12](#)
 - add shard, [C-18](#)
 - add shardgroup, [C-20](#)
 - add shardspace, [C-22](#)
 - config, [C-23](#)
 - config cdb, [C-23](#)
 - config chunks, [C-24](#)
 - config credential, [C-25](#)
 - config database, [C-26](#)
 - config file, [C-26](#)
 - config gds pool, [C-27](#)
 - config gsm, [C-28](#)
 - config sdb, [C-30](#)
 - config service, [C-31](#)
 - config shard, [C-33](#)
 - config shardgroup, [C-33](#)
 - config shardspace, [C-34](#)
 - config table family, [C-35](#)
 - config vncr, [C-36](#)
 - connect, [C-37](#)
 - create catalog, [C-39](#), [C-41](#)
 - create shard, [C-43](#)
 - create shardcatalog, [C-46](#)
 - databases, [C-50](#), [C-104](#)
 - delete catalog, [C-51](#)
 - deploy, [C-52](#)
 - disable service, [C-53](#)
 - enable service, [C-54](#)
 - exit, [C-55](#), [C-78](#)
 - help, [C-56](#)
 - modify catalog, [C-57](#)
 - modify cdb, [C-59](#)
 - modify credential, [C-60](#)
 - modify database, [C-61](#)
 - modify file, [C-62](#)
 - modify gds pool, [C-63](#)
 - modify gsm, [C-63](#)
 - modify region, [C-65](#)
 - modify service, [C-66](#)
 - modify shard, [C-74](#)
 - modify shardgroup, [C-75](#)
 - modify shardspace, [C-76](#)
 - move chunk, [C-77](#)

GDSCTL (*continued*)

commands (*continued*)
 quit, [C-55](#), [C-78](#)
 recover shard, [C-78](#)
 relocate service, [C-79](#)
 remove brokerconfig, [C-80](#)
 remove cdb, [C-81](#)
 remove credential, [C-82](#)
 remove database, [C-82](#)
 remove file, [C-83](#)
 remove gds pool, [C-84](#)
 remove gsm, [C-84](#)
 remove region, [C-86](#)
 remove service, [C-86](#)
 remove shard, [C-87](#)
 remove shardgroup, [C-88](#)
 remove shardspace, [C-89](#)
 remove vncr, [C-85](#)
 services, [C-89](#)
 set gsm, [C-91](#)
 set inbound_connect_level, [C-92](#)
 set inbound_connect_timeout, [C-92](#)
 set log_level, [C-93](#)
 set log_status, [C-94](#)
 set outbound_connect_level, [C-95](#)
 set outbound_connect_timeout, [C-95](#)
 set trace_level, [C-96](#)
 set trc_level, [C-97](#)
 show ddl, [C-98](#)
 split chunk, [C-99](#)
 sql, [C-100](#)
 start gsm, [C-100](#)
 start observer, [C-101](#)
 start service, [C-102](#)
 status, [C-103](#), [C-105](#)
 status database, [C-104](#)
 status gsm, [C-105](#)
 status service, [C-89](#), [C-107](#)
 stop gsm, [C-108](#)
 stop service, [C-108](#)
 sync brokerconfig, [C-110](#)
 sync database, [C-111](#)
 synchronize brokerconfig, [C-110](#)
 synchronize database, [C-111](#)
 validate, [C-113](#)
 validate catalog, [C-112](#)

connections, [2-22](#)
 for Global Data Services, [B-1](#)
 for Oracle Sharding, [A-1](#)
 help, [2-19](#)
 keywords, [2-21](#)
 object name
 brokerconfig, [2-21](#)
 catalog, [2-21](#)
 database, [2-21](#)

GDSCTL (*continued*)

object name (*continued*)
 gds pool, [2-22](#)
 gsm, [2-22](#)
 region, [2-22](#)
 service, [2-22](#)
 VNCR, [2-21](#)

object names, [2-21](#)
 reference, [C-1](#)
 running commands in a batch file, [2-19](#)
 usage information, [2-19](#)

gds pool
 GDSCTL object name, [2-22](#)

Global Data Services
 administrator, [2-1](#)
 catalog
 creating, [2-23](#), [2-24](#)
 requirements, [2-23](#)
 catalog database
 connecting to, [2-25](#)
 logical components
 database pool, [1-3](#)
 region, [1-3](#)
 physical components
 catalog, [1-4](#)
 global service manager, [1-4](#)
 Oracle Notification Service server, [1-4](#)
 pool
 adding, [2-27](#)
 region
 adding, [2-28](#)
 global run-time connection load balancing, [1-14](#)
 global service manager
 installing, [2-2](#)
 global services
 attributes, [1-6](#)
 in an Oracle Data Guard broker
 configuration, [1-7](#)
 in an Oracle RAC database, [1-7](#)
 overview, [1-5](#)
 gsm
 GDSCTL object name, [2-22](#)

I

introduction, [1-1](#)

L

load balancing
 server-side, [1-12](#)
 local region affinity, [1-13](#)
 local region affinity with failover, [1-13](#)
 local services, [1-5](#)

M

modify gdspool, [C-63](#)
modify gsm, [C-63](#)
modify region, [C-65](#)
modify service, [C-66](#)
modify shardgroup, [C-75](#)

O

objects
 GDSCTL object names and abbreviations,
 [2-21](#)
Oracle Data Guard, [1-7](#)

P

preferred databases, [1-8](#)

R

region
 GDSCTL object name, [2-22](#)
region affinity, [1-13](#)
 any, [1-13](#)
 local, [1-13](#)
 local with failover, [1-13](#)
remove brokerconfig, [C-80](#)
remove database, [C-82](#)
remove gdspool, [C-84](#)
remove gsm, [C-84](#)
remove region, [C-86](#)

remove service, [C-86](#)
remove vncr, [C-85](#)
replication lag time, [1-9](#)
run-time connection load balancing, [1-14](#)
 service-level goals, [1-14](#)
 SERVICE_TIME, [1-14](#)
 THROUGHPUT, [1-14](#)

S

server-side load balancing, [1-12](#)
service
 GDSCTL object name, [2-22](#)
services
 managing, [C-66](#)
set gsm, [C-91](#)
silent install, [2-3](#)
start gsm, [C-100](#)
start service, [C-102](#)
status database, [C-104](#)
status service, [C-107](#)
stop gsm, [C-108](#)
sync database, [C-111](#)
SYS_CONTEXT, [4-5](#)

V

valid node checking for registration
 See VNCR
validate catalog, [C-112](#)
VNCR,
 GDSCTL object name, [2-21](#)