Oracle® Database

Global Data Services Concepts and Administration Guide

23ai
F46808-03
February 2025

ORACLE

Oracle Database Global Data Services Concepts and Administration Guide, 23ai

F46808-03

Primary Authors: Jim Womack, 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

Contents

3 Installation and Configuration

4 GDS Administration

Preface

Welcome to the Oracle Global Data Services Concepts and Administration Guide!

This guide provides a comprehensive resource for understanding, deploying and effectively utilizing Oracle Global Data Services (GDS) to maximize the availability, performance, and scalability of your Oracle databases.

Whether you are a database administrator, architect, application developer, or IT manager, this guide equips you with the knowledge and practical guidance to:

- Understand the core concepts and architecture of Oracle GDS.
- Configure and manage global services for various deployment scenarios.
- Administer and manage Oracle GDS setup.
- Optimize workload management and performance.
- Troubleshoot and resolve common issues.

This guide covers various topics, from basic GDS configuration to advanced features like dynamic load balancing and intelligent workload routing. It includes detailed explanations, step-by-step instructions, and real-world examples to help you effectively implement and manage GDS in your environment.

We hope this guide is a valuable resource in your journey to mastering Oracle Global Data Services and achieving your availability, performance, and scalability goals.

Audience

This document is intended for database administrators, database and system 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 Database Net Services Administrator's Guide*

- *Oracle Database Net Services Reference*
- *Oracle Globally Distributed Database Guide*
- *Oracle True Cache User's Guide* for more information.

Conventions

The following text conventions are used in this document:

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

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

This preface contains:

Changes in Oracle Database 23ai

The following are changes in *Oracle Database Global Data Services Concepts and Administration Guide* for Oracle Database 23ai.

Global Data Services Suport of True Cache

With the introduction of Oracle Database 23ai, Global Data Services now supports True Cache. Oracle True Cache is an in-memory, consistent, and automatically managed SQL cache for Oracle Database.

Oracle True Cache can be deployed with Oracle Database Global Data Services (GDS) to manage workload routing, dynamic load balancing, and service failover across multiple True Caches and other database replicas.

> **See:**
>
> *Oracle True Cache User's Guide* for more information.

GDSCTL Commands Supporting RAFT Replication

With the introduction of Oracle Database 23ai, Global Data Services now supports RAFT Replication.

New `GDSCTL` commands have been added to support RAFT Replication in a sharded environment. They include:

- `ALTER MOVE`
- `ALTER TASK`
- `CONFIG TASK`
- `COPY RU (REPLICATION_UNIT)`
- `MOVE RU`
- `RELOCATE CHUNK`
- `REMOVE RU`

- `STATUS RU`
- `START|STOP RU`
- `SWITCHOVER RU`
- `SYNC RU`

See:

Global Data Services Control Utility (GDSCTL) Command Reference

Oracle Globally Distributed Database Guide

Changes in Oracle Database 21c

The following are changes in *Oracle Database Global Data Services Concepts and Administration Guide* for Oracle Database 21c.

New Features

The following features are new in this release:

GDSCTL Commands Supporting Coordinated Backup and Restore of a Sharded Database

New `GDSCTL` commands have been included in this release to provide an automated and centralized management and monitoring infrastructure for scheduling coordinated backup and restore operations of a sharded database.

See

- Global Data Services Control Utility (GDSCTL) Command Reference
- *Oracle Globally Distributed Database Guide*

GDSCTL Commands Supporting Sharding of Existing Databases

When creating a sharded environment from existing databases, the common shared schemas across the existing databases are retreived. The new `GDSCTL SYNC SCHEMA` command has been added to support this.

There is a new `GDSCTL CREATE SHARDCATALOG` parameter, `-FOR_FEDERATED_DATABASE` to support federated sharding.

See

- Global Data Services Control Utility (GDSCTL) Command Reference
- *Oracle Globally Distributed Database Guide*

Deprecation and Desupport

The following features are deprecated or desupported in this release:

Deprecation of Service Attribute Values

Several session-related service attribute values have been deprecated in 21c. For more information see:

Deprecation of Service Attribute Value SESSION_STATE_CONSISTENCY = STATIC

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 Globally Distributed Database 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 Globally Distributed Database 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
- *Oracle Globally Distributed Database Guide*
- *Oracle Database SQL Language Reference*

GSMROOTUSER

A new user called GSMROOTUSER is used to log into CDB$ROOT for CDBs in a sharding configuration (this user is not used in GDS configurations). Any connections to CDB$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 Globally Distributed Database 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.

1

Introduction to Oracle Database Global Data Services

This chapter provides an introduction to Oracle Database Global Data Services (GDS). Oracle GDS is a holistic automated workload management feature of Oracle Database.

1.1 Challenges in Managing Distributed, Replicated, and Globally Deployed Databases

Oracle Global Data Services (GDS) is designed to address a critical set of challenges that enterprises face when managing distributed, replicated, and globally deployed databases. These challenges are tied to the growing complexity, scale, and performance demands of modern database environments that support mission-critical applications within data centers and across geographies.

Modern enterprises increasingly rely on replicated and distributed database architectures to achieve high availability, scalability, and fault tolerance. However, these benefits come with a set of inherent challenges, including:

- **Complexity of Distributed and Replicated Databases**:
  Managing multiple databases, including primary, standby, replicas, and caches, across diverse geographies introduces significant operational complexity.

  Ensuring that these databases remain synchronized, available, and responsive to application demands requires sophisticated coordination and management.
- **Lack of Unified Service Management**:
  Traditional database systems require applications to be aware of the underlying database topology. This forces developers to embed failover, load balancing, and connection management logic into applications, increasing development overhead.

  The absence of centralized service provisioning and management creates silos and increases the risk of misconfigurations and inefficiencies.
- **Challenges in Achieving High Availability**:
  Ensuring uninterrupted and continuous database availability during planned maintenance and unplanned outages, such as hardware failures, is difficult.

  Manual intervention is often required to handle failovers, switchovers, and disaster recovery, leading to risks and potential downtime and data loss.
- **Load Balancing and Performance Optimization**:
  Distributed systems often suffer from uneven resource utilization. Some database nodes are overloaded while others remain underutilized, leading to performance bottlenecks.

  Applications may connect to suboptimal database instances, increasing latency and degrading user experience.
- **Global Scale and Compliance Requirements**:
  Enterprises operating across multiple geographies need to comply with local data sovereignty regulations, which require specific data to remain within certain regions.

Coordinating cross-region traffic while minimizing latency and ensuring compliance adds another layer of complexity.

- **Operational Inefficiencies and High Costs**:
  Without automated mechanisms, enterprises face high operational costs due to the need for specialized personnel to manage and optimize distributed databases manually.

  The lack of intelligent workload management leads to resource wastage and underutilized infrastructure.

1.2 How Oracle GDS Addresses These Challenges

Oracle GDS is engineered to provide a unified, intelligent solution to the above challenges by introducing centralized service management, dynamic load balancing, and automated failover capabilities. It abstracts the complexity of managing replicated and distributed databases, enabling enterprises to focus on their business goals rather than operational overhead.

Oracle Global Data Services core capabilities and features:

- **Centralized Service Management**:
  Oracle GDS provides a single, unified interface for managing database services across replicated and distributed environments. Applications connect to a single Global Service Manager (GSM), which handles all underlying complexities, including failover, load balancing, and routing.
- **Automated High Availability**:
  Oracle GDS ensures continuous availability by automating service placement and failover processes. During planned or unplanned events, it transparently redirects connections to healthy and most optimal database instances, minimizing downtime and maintaining data availability.
- **Dynamic Workload Management**:
  Oracle GDS intelligently routes workloads to the most optimal database instance based on policies, node health, nature of workload, and performance metrics. This ensures even resource utilization, reduces latency, and enhances the user experience.
- **Global Scalability with Compliance**:
  Oracle GDS enables global deployment of database services, ensuring low-latency access for geographically dispersed users while complying with data sovereignty regulations through region-aware routing.
- **Simplified Operations**:
  Oracle GDS intelligently abstracting the complexities of replicated and distributed systems, Oracle GDS reduces the need for manual intervention. Its centralized control and policy-driven data services management significantly lower operational costs and reduce the risk of human error.
- **Multi-Cloud and Hybrid Compatibility**:
  Oracle GDS supports deployment across on-premises environments, Oracle Cloud Infrastructure (OCI), and third-party cloud providers such as AWS, GCP, and Azure, making it ideal for enterprises with hybrid or multi-cloud strategies.

1.3 Benefits of Oracle Global Data Services

Global Data Services (GDS) allows fault-tolerant database services to be deployed and centrally managed across a set of replicated databases. The GDS framework provides workload balancing across these databases. More specifically, GDS is an Oracle-integrated solution that renders the following benefits:

- *Higher availability and global scalability*: Support seamless inter-database service failover among replicated databases in any data center, yielding higher application availability.
- *GDS provides application scalability on demand*: Allows dynamic addition of databases. It enables replicated databases to be added to the GDS infrastructure dynamically and transparently to obtain additional resource capability to scale application workloads. GDS allows this with no change to the application configuration or client connectivity.
- *Better Performance and Elasticity*: With integrated load balancing across multiple databases, GDS addresses inter-region resource fragmentation. Under-utilized resources in one region can be put to work on the workload of another region's over-utilized resources, achieving optimal resource utilization. GDS sends work requests to less powerful databases in a GDS pool containing replicated databases running on database servers of different processor generations and various resources (CPU, memory, I/O). When more powerful databases are overloaded, the goal should be to equalize the response time.

Extended benefits of Oracle GDS include:

- *Faster processing for analytics*: All shards are presented to an application as a single logical database, speeding query response time on extremely large data sets.
- *Future-proof scalability for increased data volume and transaction processing*: Eliminate performance bottlenecks while enabling linearly scaled database performance.
- *RAFT Replication*: Enables rapid failover within seconds and zero data loss during node or data center outages, facilitating an Active-Active-Active symmetric distributed database architecture that enhances availability, simplifies management, and optimizes resource utilization globally.
- *Meet data compliance and residency demands*: Ensures that data stays in a given geographical location. Facilitates a single global database, with data distributed across multiple regions.
- *Deploy in the cloud or your data center*: On Oracle Base Database Service, Globally Distributed Autonomous Database, or a multicloud architecture across Microsoft Azure and Amazon Web Services.

You can use Oracle GDS to achieve these benefits without the need to integrate with multiple-point solutions or homegrown products. GDS provides optimized hardware and software utilization, better performance, scalability, and availability for application workloads running on replicated databases.

1.4 Oracle GDS vs Network Load Balancers Comparison

Requirement	Network Load Balancer	Oracle GDS
Locality-based routing	Yes	Yes
Connect-time database load balancing	Yes	Yes
Publish routing and failover intelligence to clients	No	Yes
Replication lag-based database workload routing	No	Yes
Inter-database global service failover	No	Yes
Automatic role-based global services	No	Yes

Requirement	Network Load Balancer	Oracle GDS
Centralized management of database services across replicas	No	Yes
Native integration for Oracle Active Data Guard	No	Yes
Cost effectiveness	Additional expenditure required	Included with Oracle Active Data Guard or Oracle GoldenGate license

1.5 Capabilities of Global Data Services

Oracle Global Data Services (GDS) technology provides the following principal capabilities:

- **Dynamic Load Balancing** ensures optimal distribution of workload across available database instances.
  - **Connect-time load balancing**: Global service managers use the load statistics from all databases in the GDS pool, inter-region network latency, and the configured connect-time load balancing goal to route the incoming connections to the best database in a GDS pool. This prevents any single database from becoming a bottleneck.
  - **Runtime load balancing**: GDS enables runtime load balancing across replicated databases by publishing a real-time load balancing advisory for connection pool-based clients (for example, OCI, JDBC, ODP.NET, WebLogic, and so on.). The connection pool-based clients subscribe to this load-balancing advisory and route database requests in real time across already-established connections.

    With GDS's runtime connection load balancing feature, application client work requests are dynamically routed to the database that offers the best performance. Through a process called *Gravitation*, GDS dynamically adjusts connections between databases in response to changing load conditions, ensuring even workload distribution as demands fluctuate.
- **Intelligent Workload Routing** routes incoming workload based on established criteria.
  - **Region-based workload routing**: Allows you to configure client connections to be routed among a set of replicated databases in a local region. This capability allows you to maximize application performance (avoiding the network latency overhead of accessing databases in remote areas).
  - **Lag-based workload routing**: With Oracle Active Data Guard, standbys may lag behind the primary database. A global service allows you to choose the acceptable lag tolerance for a given application. GDS routes requests to a standby database whose lag is below the limit. If the lag exceeds the lag limit, the service is relocated to another available standby database that lags below the threshold. New requests are routed to a standby database that satisfies the lag limit. The global service is shut down if there is no available database. When the lag is resolved or comes within the limit, GDS automatically brings up the service.
- **Inter-Database Service Failover** provides uninterupted availability.
  - In the event of a database failure, GDS automatically and transparently fails over services to a healthy replica. This automated and transparent response capability eliminates human error and delayed action.

  - Connection pools are instantly notified, ensuring minimal disruption for applications and end-users. This significantly reduces downtime and ensures business continuity.

- **Role-based Global Services** provide tailored service placement based on business needs.
  - GDS enables you to configure services based on database roles, such as primary, physical standby, logical standby, and snapshot standby.
  - This allows you to enforce specific service placement policies and optimize resource utilization.
  - GDS supports RAFT Replication, Oracle Distributed Database and Oracle True Cache.
- **Centralized Service Management** provided by GDS allows more straightforward configuration and management of the replicated databases' resources located anywhere with a single unified framework.
  - Create, configure, and manage services across multiple databases from a single interface using the GDSCTL command line utility or through the graphical interface provided by Oracle Enterprise Manager.
  - These GDS tools simplify administration, eliminate the need for manual interventions on individual databases, and accelerate service provisioning and adjustments.

2
Oracle GDS Architecture and Concepts

2.1 Overview

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 can provide local and global services, simultaneously. A client application can also work with global and local services, simultaneously.

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.

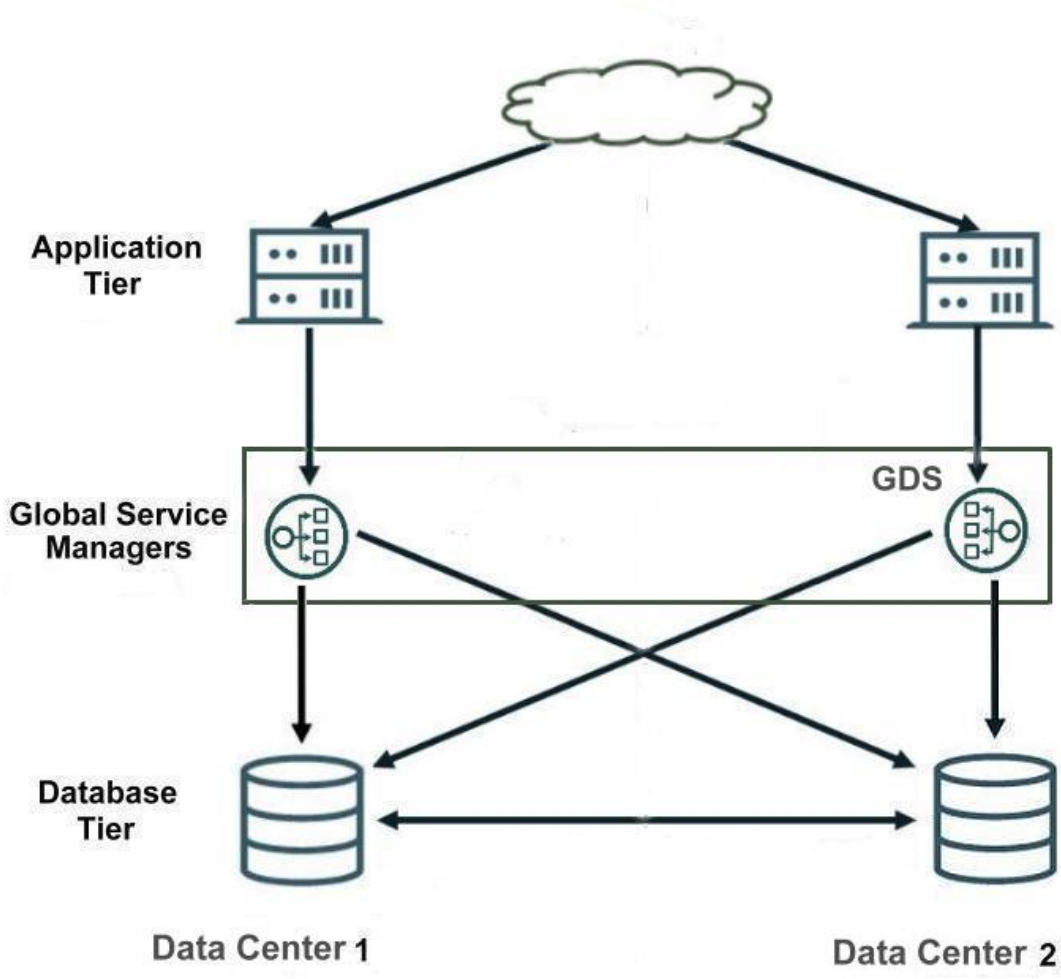
The global service manager can start or stop a global service in a database. However, if it is an Oracle RAC database, the global service manager does not control which particular instances within the database offer the service. This is controlled by the clusterware and the administrator of the Oracle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

2.2 Global Data Services Architecture

The following figure shows an example of a Global Data Services (GDS) configuration and common GDS components.

Figure 2-1 Global Data Services Components



2.3 GDS Concepts

Database services are logical abstractions for managing workloads in an Oracle database. Each service represents a workload with common attributes, service-level thresholds, and priorities. The grouping can be based on work characteristics, including the application function. For example, the Oracle E-Business Suite defines a service for each application module, such as general ledger, accounts receivable, and order entry. Services are built into Oracle Database, providing a single system image for workloads. Services enable administrators to configure a workload, administer it, enable and disable it, and measure the workload as a single entity. Clients connect using a database service name.

For replicated environments, GDS introduces the concept of a "global service". Global services are provided across a set of databases containing replicated data that belongs to a particular

administrative domain known as a GDS pool. Examples of a GDS pool are a SALES pool or an HR pool. A set of databases in a GDS configuration and the database clients are said to be in the same GDS region if they share the network proximity. Examples of GDS regions are the Asian region, European region, and so on.

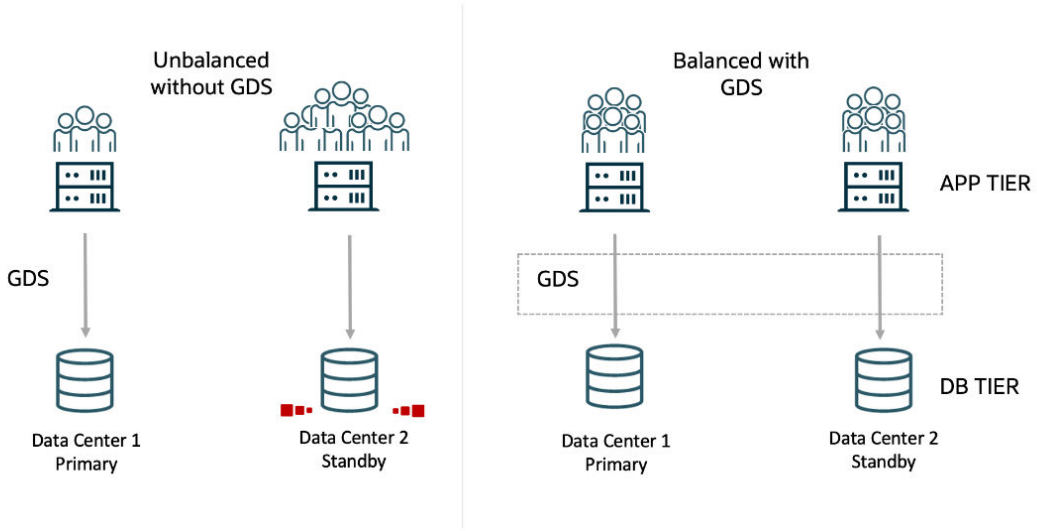
All of the characteristics of traditional database services are supported by global services. Global services extends traditional database services with additional attributes such as global service placement, replication lag (Oracle Active Data Guard and Oracle GoldenGate from 19c onwards), and region affinity.

Global service placement: When a global service is created, GDS allows the preferred and available databases for that service to be specified. The available databases support a global service if the preferred database fails. In addition, GDS allows you to configure a service to run on all the replicas of a given GDS pool.

Replication lag: Clients can be routed to the Oracle Active Data Guard standbys that are not lagging by the tolerance limit established with the lag attribute of a global service.

Region affinity: A global service allows you to set preferences to which region (for example Asia or Europe) your given applications should connect.

Figure 2-2 Workload Balancing with Global Data Services



2.3.1 Global Data Services Components

The sections that follow describe the Oracle GDS Configuration components.

2.3.1.1 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.

2.3.1.2 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 manager**s that can provide continued access to a GDS configuration if the global services managers in the local region become unavailable.

2.3.1.3 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.

> **Note:**
>
> A master GSM makes metadata changes in the catalog database, for instance when adding a new global data service or adding a new shard. If the master GSM goes down, another GSM will become the new master. Any other GSMs associated to a region is a region GSM.

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

2.3.1.5 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.

3
Installation and Configuration

Overview

The Oracle Global Data Services (GDS) framework consists of core, client-side, and database-side components that work together to setup, manage and orchestrate data services efficiently. At the core, the Global Service Manager (GSM) and GDS Catalog, handle intelligent routing, dynamic load balancing, service failover, and store configuration metadata. These components may use redundancy for high availability. On the client side, Oracle GDS integrates with an Oracle-enabled client and connection pool to support features like Fast Connection Failover (FCF) and intelligent connection and work request routing. On the database side, Oracle Notification Service (ONS) provides real-time event notifications to GSM, while the gsmuser schema enables local service orchestration. These components collectively ensure a scalable, resilient, and optimized global services management framework.

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).

3.1 Oracle GDS Capacities and Requirements

Capacity planning and sizing should be done before deployment, and periodically afterward, to ensure that there are sufficient system resources to meet application performance requirements. Capacity planning needs to accommodate growth or consolidation of databases, additional application workloads, additional processes, or anything that strains existing system resources. The table below lists GDS sizing capacities and requirements.

A Single GDS Manages	GDS Databases
• 5,000 GDS Pools • 10 GDS Regions • 5 global service managers per Region • 10,000 Database instances • 10,000 Global Services • 1,000 Mid-tier connection pools	• Can be a Single Instance or RAC • Can be CDB or Non-CDB • Can run on commodity or Engineered systems (Oracle Exadata, ODA) • Are managed with GDSCTL CLI or Enterprise Manager DB Plug-in

3.2 Planning GDS Deployment

Configure network connectivity for your GDS setup using the following guidelines:

- Global Service Manager (GSM) Listener and ONS Port Accessibility: All GDS pool databases must be able to communicate with every Global Service Manager's Listener (default port: 1522) and ONS ports (default: 6123 for local ONS and 6234 for remote ONS) in both directions. These ports must also be accessible from the Application/Client tier, all GDS pool databases, the GDS catalog database, other Global Service Managers in the deployment.
- TNS Listener Port Configuration for Pool Databases: The TNS Listener port (default: 1521) for each GDS pool database must be open in both directions to enable communication with all Global Service Managers, and the GDS catalog database.

- GDSCTL Connectivity: If the GDSCTL utility is run from a separate machine, ensure that this machine has direct, bidirectional port access to the TNS Listener port (default: 1521) of GDS catalog database.
- Default ONS Port Usage: On most platforms, the default ports for Oracle Notification Service (ONS) are: 6123 for local ONS, and 6234 for remote ONS.

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

3.3 GDS Software Installation

The global service manager (GSM) 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.

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

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.

3.3.1 Installing a Global Service Manager

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.

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

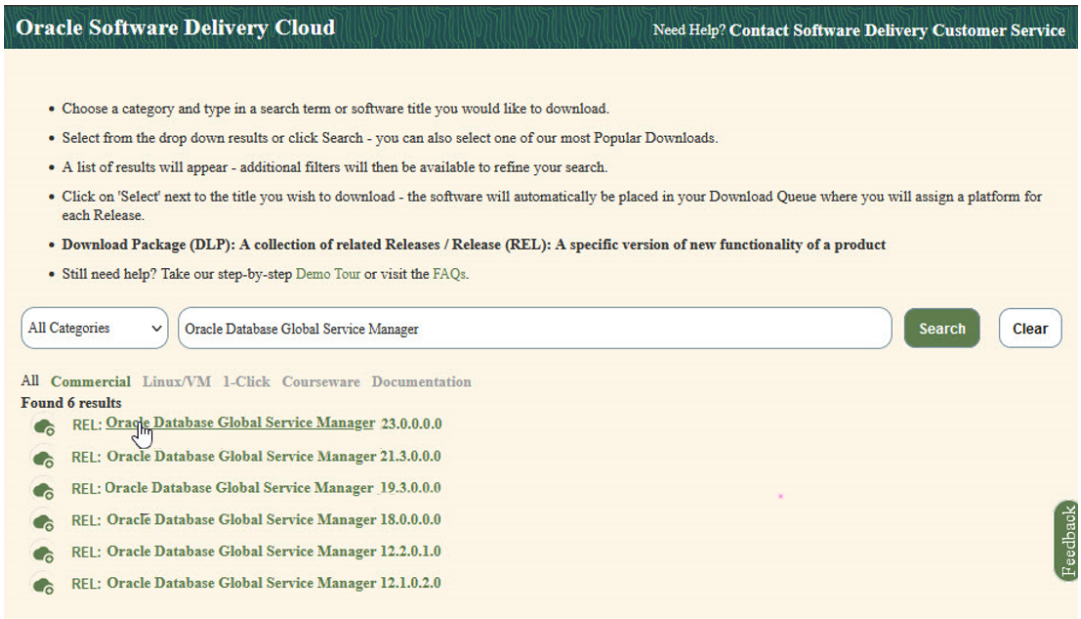
- Administering global service managers
- Administering the Global Data Services catalog
- Administering regions and database pools

> **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.

Installing a Global Service Manager:

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

Figure 3-1 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.

3.4 GDS Catalog Database and GDS Catalog Setup

What You Need to Know About Creating the GDS 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 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 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 gdscatalog -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 gdscatalog for complete usage information

3.5 Global Data Services Configuration

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

3.5.1 Oracle GDS Deployment Steps

The following are the basic steps you would take to implement Global Data Services.

1. Install Oracle GDS global service manager software on global service manager servers.
   - Minimum of 1 global service manager for each region
   - Recommended 3 global service managers for each region
2. Pre-create the Oracle GDS catalog database.
3. Set up Oracle GDS Administrator accounts and privileges.
4. Configure Oracle GDS.
   - Create the GDS catalog and standby databases.
   - Add global service managers, regions, pools, databases, and global services.
5. Set up client connectivity.

3.5.2 GDS Configuration Example

The following steps describe how to implement Global Data Services.

This example configuration of Global Data Services (GDS) uses an Administrator-managed Oracle RAC database. Administrator-managed deployment means that you configure database services to run on specific instances belonging to a particular database using a preferred and available designation.

Policy-managed deployment is based on server pools, where database services run within a server pool as singletons or uniformly across all of the servers in the server pool. Databases are deployed in one or more server pools, and the size of the server pools determines the number of database instances in the deployment.

1. Create and prepare a GDS catalog database.
   GDS uses a catalog database to store meta-data relating to the layout and status of the GDS configuration. For maximum availability, Oracle recommends that the GDS catalog database be deployed independently and that Oracle's high-availability features, such as Oracle Real Application Clusters (Oracle RAC) and Oracle Data Guard, be used to protect the catalog database against outages.
2. Create the `GSM_ADMIN` user and assign that user the `GSMADMIN_ROLE`.

Note that by default, the password for both `GSM_ADMIN`, `GSMUSER`, and `GSMCATUSER` expires after 180 days.

```
SQL> create user gsm_admin identified by password;

User created.

SQL> grant gsmadmin_role to gsm_admin;

Grant succeeded.

SQL> exit
```

3. With the environment configured for the global service manager home, use GDSCTL to create the GDS catalog database with Auto VNCR disabled (Auto VNCR can cause problems with Oracle RAC deployments).

```
GDSCTL> create gdscatalog -database gdscat -user gsm_admin -autovncr OFF
```

4. Connect to the catalog database, unlock the `GSMCATUSER` user, and set the password.

```
SQL> alter user gsmcatuser account unlock;

User altered.

SQL> alter user gsmcatuser identified by password;

User altered.
```

5. With the environment configured for the global service manager home, use GDSCTL to connect to, create, and start the global service manager listeners.
As a best practice, global service manager listeners should reside on hardware separate from that hosting the Oracle Databases in the GDS configuration. The resource requirements for hardware needed to run global service manager listeners are lightweight and can easily be accommodated using virtual machines.

```
GDSCTL> add gsm -gsm gsm1 -listener 1522 -catalog gdscat

"gsmcatuser" password:

Create credential oracle.security.client.connect_string1

GSM successfully added

GDSCTL>start gsm -gsm gsm1

GSM is started successfully

GDSCTL>status

Alias GSM1
Version 19.17.0.3.0
Start Date 13-APR-2023 09:40:59
Trace Level off
Listener Log File
```

```
 /u01/app/oracle/diag/gsm/hostname/gsm1/alert/log.xml
Listener Trace File
 /u01/app/oracle/diag/gsm/hostname/gsm1/trace/ora_64863_139739749930432.trc
Endpoint summary
(ADDRESS=(HOST=hostname.example.com)(PORT=1522)(PROTOCOL=tcp))
GSMOCI Version 0.6.11
Mastership Y
Connected to GDS catalog Y
Process Id 64883
Number of reconnections 0
Pending tasks. Total 0
Tasks in process. Total 0
Regional Mastership TRUE
Total messages published 0
Time Zone -04:00
Orphaned Buddy Regions: None
GDS region regionora
```

6. With the environment configured for the global service manager home, use GDSCTL to create a default GDS pool and default region.

```
GDSCTL> add gdspool -gdspool sales

GDSCTL> add region -region slc

GDSCTL> add region -region sca
```

7. With Auto VNCR disabled during GDS catalog creation to avoid issues, use GDSCTL to add hosts using the `add invitednode` command, using the host name or IP address appropriately.

```
GDSCTL> add invitednode 192.0.2.1

GDSCTL> add invitednode host1.example.com
```

8. Unlock the `GSMUSER` account.
   Before adding a database to a pool, the database administrator should unlock the `GSMUSER` account and give the password to the GDS pool administrator, as shown in the following example.

```
SQL> alter user gsmuser account unlock;

User altered.

SQL> alter user gsmuser identified by password;

User altered.
```

9. Add databases to the GDS pool.
   To be part of a GDS pool, a database must use a server parameter file (SPFILE). An Oracle RAC database should also have SCAN set up.

   To add a database, connect to the GDS catalog using the GDS pool or GDS administrator credentials. For example, without Data Guard, the following add database command can be used.

> **Note:**
>
> When using Oracle Active Data Guard with GDS, use `add brokerconfig` instead of `add database` , and then use `modify database` to configure the standby database (see add brokerconfig). The syntax for these commands would be like the following:
>
> ```
> GDSCTL> add brokerconfig -connect <primary_db> -gdspool <dbpool> -
> region <dc> -pwd <gsmuser_pwd>
>
> GDSCTL> modify database -database <standby_db> -connect <dc> -
> gdspool <dbpool> -region <dc> -pwd <gsmuser_pwd>
> ```

Database instance registration with a global service manager succeeds only when the request originates from a valid node. If a host on which a database resides contains multiple network interfaces, the auto-configuration could register the wrong set of IP addresses, leading to the database registration being rejected.

10. Correct any rejected registration and properly discover all database instances.
If a firewall exists between the global service managers, and the databases and the ports are not opened, the registration fails. From the global service manageralert log, you will see entries similar to the following.

```
Listener(VNCR option 1) rejected Registration request from destination

192.0.2.2

Listener(VNCR option 1) rejected Registration request from destination

192.0.2.3
```

You will find that the database object exists in the GDS catalog, but some or all instances associated with specific hosts are missing.

```
GDSCTL> databases

Database: "mts" Registered: Y State: Ok ONS: Y. Role: PRIMARY

Instances: 1 Region: slc

Registered instances:

sales%1
```

To correct the rejected registration and properly discover all database instances, run `add invitednode` using the rejected IP address listed in the global service manager alert log.

11. If there is a firewall between the global service managers and the database, then once the ports have been opened and verified using tnsping, issue the `add invitenode` command as shown here.

```
GDSCTL> add invitednode 192.0.2.3

GDSCTL>databases

Database: "mts" Registered: Y State: Ok ONS: Y. Role: PRIMARY

Instances: 2 Region: slc

Registered instances:

sales%1

sales%2
```

12. Create a service on the GDS pool databases.
The GDSCTL `add service` command creates a service on the GDS pool databases.

```
GDSCTL> add service -service sales_sb -preferred_all -gdspool sales -
notification TRUE
```

If this is an Oracle RAC database being added with multiple instances, then you must modify the service to add the database instances.

```
GDSCTL> modify service -gdspool sales -service sales_sb -database mts -
add_instances -preferred mts1,mts2

GDSCTL> modify service -gdspool sales -service sales_sb -database stm -
add_instances -preferred stm1,stm2

GDSCTL> start service -service sales_sb -gdspool sales
```

13. Verify that the global service is running.

```
GDSCTL> services

Service "sales_sb.sales.oradbcloud" has 2 instance(s). Affinity: ANYWHERE

Instance "sales%1", name: "mts1", db: "mts", region: "slc", status: ready.

Instance "sales%2", name: "mts2", db: "mts", region: "slc", status: ready.
```

3.5.3 GDS Configuration Best Practices

Oracle MAA recommends the following best practices for implementing Global Data Services:

- Each client communicates using an Oracle-integrated connection pool such as UCP, OCI, or ODP.NET. The connection pools will be notified about any service failovers and load balancing advisory notifications using Fast Application Notification Events.
- Run three global service managers in each region. Create three global service managers in each region so that if one global service manager goes down, you have two remaining

global service managers to provide redundancy. Each global service manager should reside on separate hardware. Global service managers enable connection routing among replicated databases. A global service manager is a stateless, lightweight, and intelligent listener that can repopulate its metadata from the GDS catalog.

- Protect the GDS catalog database with Oracle Data Guard.The GDS catalog is a small (less than 100 GB) repository that hosts the metadata of the GDS configuration, regions, global service managers, global services, databases, and so on. MAA recommends that you set up a local Data Guard standby database configured with Maximum Availability database protection mode, Data Guard Fast-Start failover, and a remote physical standby database. All GDS catalog standby databases should use Oracle Active Data Guard for the best data protection and reside on separate hardware and storage.

4
GDS Administration

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.

4.1 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.

4.2 Managing the GDS Stack

This section describes the startup and shutdown of components in the global data services framework.

4.2.1 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.

4.2.2 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.

4.3 Monitoring the GDS Environment

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

To monitor the global services configuration:

```
GDSCTL> services
Service "sales_sb.sales.oradbcloud" has 2 instance(s). Affinity: ANYWHERE
Instance "sales%1", name: "mts1", db: "mts", region: "slc", status: ready.
Instance "sales%2", name: "mts2", db: "mts", region: "slc", status: ready.
```

Monitoring member databases:

```
GDSCTL> databases
Database: "mts" Registered: Y State: Ok ONS: Y. Role: PRIMARY
Instances: 1 Region: slc
Registered instances:
sales%1
```

To monitor the global service manager configuration:

```
GDSCTL> status
Alias GSM1
Version 19.17.0.3.0
Start Date 13-APR-2023 09:40:59
Trace Level off
Listener Log File
/u01/app/oracle/diag/gsm/hostname/gsm1/alert/log.xml
Listener Trace File
/u01/app/oracle/diag/gsm/hostname/gsm1/trace/ora_64863_139739749930432.trc
Endpoint summary
(ADDRESS=(HOST=hostname.example.com)(PORT=1522)(PROTOCOL=tcp))
GSMOCI Version 0.6.11
Mastership Y
Connected to GDS catalog Y
Process Id 64883
Number of reconnections 0
Pending tasks. Total 0
Tasks in process. Total 0
Regional Mastership TRUE
```

```
Total messages published 0
Time Zone -04:00
Orphaned Buddy Regions: None
GDS region regionora
```

4.4 Managing Global Data Services

This section describes the administration tasks associated with Global Data Services.

4.4.1 Managing GDS Regions

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

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:

```
GDSCTL> 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 '_'.

To modify the configuration parameters for an existing region, use the `modify region` command:

```
GDSCTL> modify region -region west -buddy east
```

Where `-buddy` indicates a "buddy" region.

To remove a specified region(s) from the global service management framework, use the `remove region` command:

```
GDSCTL> GDSCTL> remove region -region south
```

4.4.2 Managing GDS Pools

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]
```

If you require only one Global Data Services pool, then you do not need to add one using this example. 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 (_)).

Use the `modify gdspool` command to modify the configuration parameters of GDS pools:

```
GDSCTL> modify gdspool -gdspool hr -adduser psmith
```

Use the `remove gdspool` command to remove a GDS pool(s) from the current configuration:

```
GDSCTL> remove gdspool -gdspool tempreaders,myfarm
```

4.4.3 Managing Member Databases

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:

   ```
   GDSCTL> 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.

Use the `modify database` command to modify the configuration parameters of the databases in a GDS pool, such as region, connect identifier, global service manager password, SCAN address, and ONS port:

```
GDSCTL> modify database -database db1,db3 -region east
```

Use the `remove database` command to remove databases from a GDS pool:

```
GDSCTL> remove database -database db1 -gdspool pool1
```

4.4.3.1 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

4.4.3.2 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

4.5 Managing Global Services

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

4.5.1 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. For example:

```
GDSCTL>add service -service sales_sb -preferred_all -gdspool sales -
notification TRUE
```

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.

If this is an Oracle RAC database being added with multiple instances, then you must modify the service to add the database instances.

```
GDSCTL>modify service -gdspool sales -service sales_sb -database mts -
add_instances -preferred mts1,mts2
GDSCTL>modify service -gdspool sales -service sales_sb -database stm -
add_instances -preferred stm1,stm2
GDSCTL>start service -service sales_sb -gdspool sales
```

> **See Also:**
>
> add service

4.5.2 Starting a Global Service

The `gdsctl 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.

> **See Also:**
>
> start service
>
> modify 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.

4.5.3 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. For example:

```
GDSCTL>stop service -gdspool readerfarm -service sales_report
```

> **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

4.5.4 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.

For example to 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:**
>
> add service

4.5.5 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.

To 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:**
>
> enable service

4.5.6 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

4.5.7 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

4.5.8 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

4.5.9 Global Data Services Failover Across Regions Flow

1. The administrator has failed over or switched the production database to the secondary site. This is automatic if you are using Data Guard fast-start failover.
2. The administrator starts the middle-tier application servers on the secondary site if they are not already running.
3. The wide-area traffic manager selection of the secondary site can be automatic in the case of an entire site failure. The wide-area traffic manager at the secondary site returns the virtual IP address of a load balancer at the secondary site, and clients are directed automatically on the subsequent reconnect. In this scenario, the site failover is accomplished by an automatic domain name system (DNS) failover.
4. Alternatively, a DNS administrator can manually change the wide-area traffic manager selection to the secondary site for the entire site or specific applications.
   The following is an example of a manual DNS failover:
   - Change the DNS to point to the secondary site load balancer: The primary (primary) DNS server is updated with the new zone information, and the change is announced with DNS NOTIFY.
   - The secondary DNS servers are notified of the zone update with a DNS NOTIFY announcement, and the secondary DNS servers pull the new zone information.
   - Clear affected records from caching DNS servers.
     A caching DNS server is used primarily for performance and fast response. The caching server obtains information from an authoritative DNS server in response to a host query and then saves (caches) the data locally. On a second or subsequent request for the same data, the caching DNS server responds with its locally stored data (the cache) until the response's time-to-live (TTL) value expires. At this time, the

server refreshes the data from the zone master. If the DNS record is changed on the primary DNS server, then the caching DNS server does not pick up the change for cached records until TTL expires. Flushing the cache forces the caching DNS server to go to an authoritative DNS server again for the updated DNS information.

- Flush the cache if the DNS server being used supports such a capability. The following is the flush capability of standard DNS BIND versions:
  - BIND 9.3.0: The command `rndc flushname name` flushes individual entries from the cache.
  - BIND 9.2.0 and 9.2.1: The cache can be flushed with the command `rndc flush`.
  - BIND 8 and BIND 9 up to 9.1.3: Restarting the named server clears the cache.

5. Refresh local DNS service caching.
   Some operating systems might cache DNS information locally in the local name service cache. If so, this cache must also be cleared to recognize DNS updates quickly.
6. The secondary site load balancer directs traffic to the secondary site middle-tier application server.

4.5.10 Graceful Application Switchover

Database services are used to manage workloads during the planned outage properly. Services must be properly created, and the application must obtain connections from a service.

These recommendations assume using a FAN-aware connection pool, such as Oracle Universal Connection Pool (UCP) to gracefully drain connections without application interruption from a service that is stopped. Your applications can use other connection types that don't support FAN-aware connection pools or have long-running transactions. Ideally, these applications will be disconnected before the maintenance window.

The recommendations below describe how to disconnect some sessions when their transaction ends in a timely manner or, ultimately, when the instance is shut down for maintenance.

The recommended and validated approach to understanding and optimizing your application's connection configuration is provided in the following sections; certain applications may have specific guidelines to follow.

Understanding Your Application's Use of Connections

Understanding how your application obtains and releases its connections is critical to determining whether it can gracefully switch to other instances in the cluster.

Find the following information about your application:

- What was the workload during the planned outage (OLTP/short or batch/long transactions)?
  - Short transactions using a connection pool such as UCP or ODP.NET can be quiesced rapidly.
  - Long transactions need additional time to quiesce or must have batch jobs stopped or queued at an appropriate time in advance.
- What type of connection was obtained: Java, OCI, ODP with C#, or ODP with OCI)?
  - UCP, ICC, ODP.NET, and OCI session pools use Fast Application Notification (FAN) to drain the pool quickly; other connections require waiting until the connection is closed (or termination if the application allows)

- What is the amount of time to wait for the connection pool to quiesce before stopping the service?
  - Useful to know the proper amount of time is given before disconnection is performed
- Can the application handle disconnection after the transaction completes (applies to batch workloads)?
  - If the application can't handle being disconnected gracefully, it must be stopped before the planned maintenance, or Application Continuity might be an option to avoid interruption.

Services and Application Configuration Best Practices

You must have properly configured services and application attributes to perform a graceful switchover successfully. See My Oracle Support Doc ID 1617163.1 for a matrix of validated drivers and applications clients.

> **Note:**
>
> You must test your configuration to ensure that it is set up and performs switchover properly before relying on it for a production system.

4.6 GDS Patching and Upgrading

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.
- **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.
- **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.
- **Pool database:** A pool database is any database added to a GDS pool which runs a global service. 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.

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

> **Note:**
>
> For Oracle Database Proactive Patch Information, see Oracle Support (MOS) Note 2521164.1

4.6.1 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.
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.

4.6.2 GSM Out-of-Place Update/Patching Examples

This example shows the steps required to apply the 19.18 Database Realease 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>
```

4. Next, start GSM1 and stop GSM2 (making GSM1 the new master) and apply the 19.18.0.0.0 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: orasampl
Master GSM: gsm1
DDL sequence #: 0

GDSCTL>
```

5. 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.orasampl" 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 `gsmwallet`directory 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 contais 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
Availability
----                ----                ------    -----       ------
------------
cloud               sdbpool             Ok        none        east
ONLINE
clouddb             sdbpool             Ok        none        west
ONLINE

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>
```

4.7 Password Management in a GDS Environment

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
   ```

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.

In a sharded invironment it is important to manage password rotation for the GSMUSER, GSMCATUSER, and GSMROOTUSER users. For a detailed instructions on performing this management task, navigate to Oracle Support and reference Doc ID 3052933.1.

5

Using Global Data Services (Architectures, Use Cases, Application Development)

5.1 Distributed Databases

In environments where databases are distributed for data residency, global scalability, performance, and availability, GDS acts as an intelligent traffic director and workload coordinator

- **Shard Director and Shard Catalog**: GDS acts as a "shard-aware" proxy, intercepting client requests and routing them to the appropriate shard containing the relevant data.
- **Data Locality**: GDS ensures that requests are directed to the shard where the data resides, minimizing data movement and optimizing performance.
- **Scalability and Parallelism**: GDS facilitates scaling out the database by adding more shards, and it can leverage parallel query capabilities to improve query performance.
- **Simplified Application Development**: GDS hides the complexity of the sharding scheme from applications, allowing them to interact with the database as a single logical unit.

Distributed Databases Example Use Cases

- **Global E-commerce Platforms**: Stores product data across multiple regions to provide low-latency access to users worldwide.
- **Social Media Networks**: Distributes user profiles, posts, and media across geographically dispersed nodes, improving scalability and response time.
- **Financial Services**: Manages transactions and customer data across multiple regions while ensuring consistency and high availability.
- **Content Delivery Networks** (CDNs): Distributes large datasets, like videos and images, across various data centers to optimize content delivery to end-users.
- **IoT Applications**: Processes real-time data from sensors distributed globally, requiring local processing and centralized analytics.

5.1.1 Using Global Data Services with Oracle Sharding

Oracle sharding enables distribution and replication of data across a pool of Oracle databases that share no hardware or software. The pool of databases is presented to the application as a single logical database. Applications elastically scale (data, transactions, and users) to any level, on any platform, simply by adding additional databases (shards) to the pool. Scaling up to 1000 shards is supported.

Oracle Sharding provides superior run-time performance and simpler life-cycle management compared to home-grown deployments that use a similar approach to scalability. It also provides the advantages of an enterprise DBMS, including relational schema, SQL, and other programmatic interfaces, support for complex data types, online schema changes, multi-core scalability, advanced security, compression, high-availability, ACID properties, consistent reads, developer agility with JSON, and much more.

Oracle Globally Distributed Database is built on the Oracle Database Global Data Services feature, so to plan your topology you must understand the Global Data Services architecture and management.

Oracle Globally Distributed Database enables you to deploy a global database, where a single logical database could be distributed over multiple geographies. This makes it possible to satisfy data privacy regulatory requirements (Data Sovereignty) as well as allows to store particular data close to its consumers (Data Proximity).

Data sovereignty generally refers to how data is governed by regulations specific to the region in which it originated. These types of regulations can specify where data is stored, how it is accessed, how it is processed, and the life-cycle of the data. With the exponential growth of data crossing borders and public cloud regions, more than 100 countries now have passed regulations concerning where data is stored and how it is transferred. Personally identifiable information (PII) in particular increasingly is subject to the laws and governance structures of the nation in which it is collected. Data transfers to other countries often are restricted or allowed based on whether that country offers similar levels of data protection, and whether that nation collaborates in forensic investigations.

5.2 Using True Cache with Global Data Services

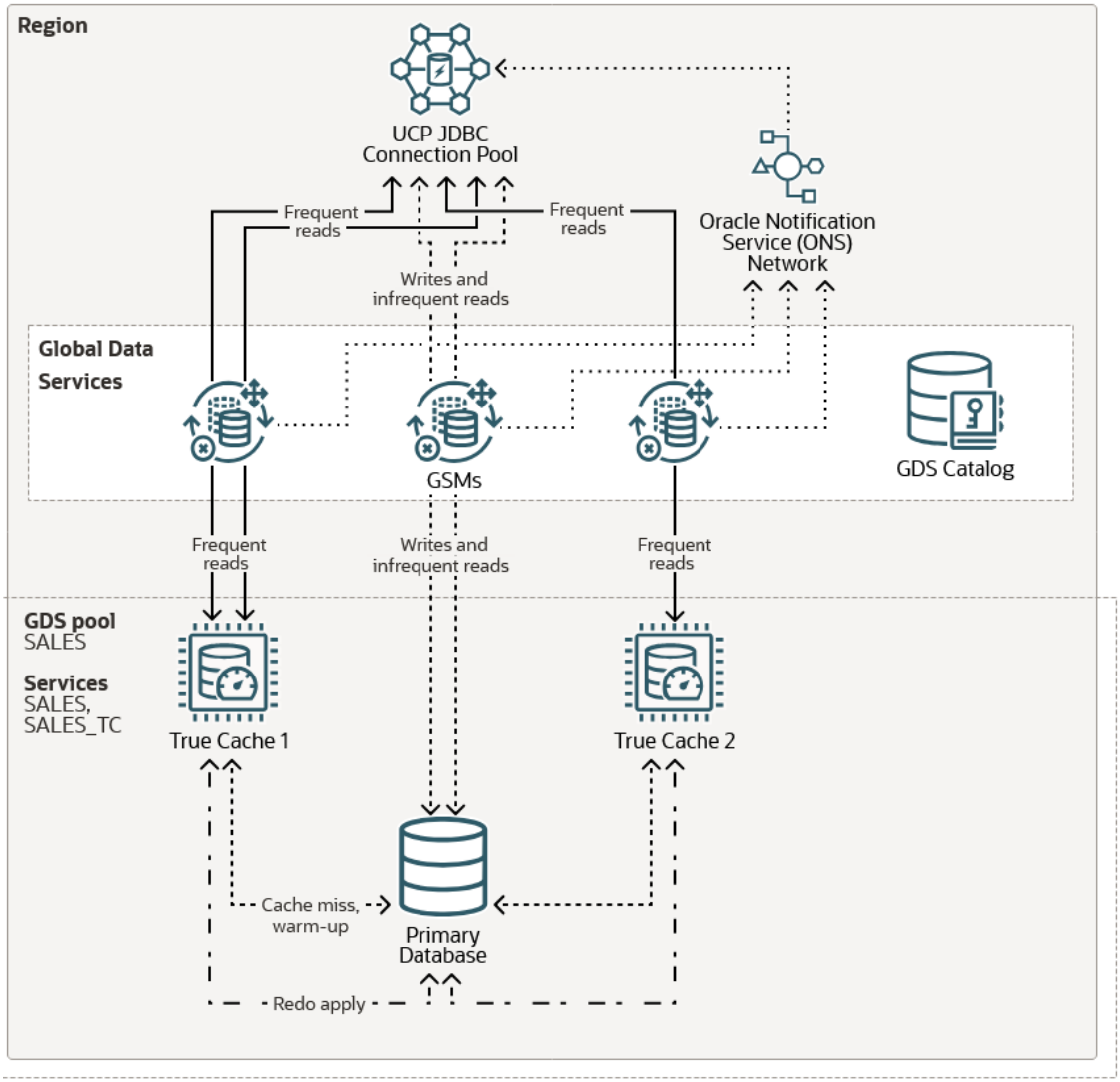
Oracle True Cache is an in-memory, consistent, and automatically managed SQL cache for Oracle Database. True Cache improves application response time while reducing the load on the database. Automatic cache management and consistency simplify application development, reducing development effort and cost.

You can deploy Oracle True Cache with Oracle Database Global Data Services (GDS) to manage workload routing, dynamic load balancing, and service failover across multiple True Caches and other database replicas.

Global services are functionally similar to the local database application services that are provided by single instance or Oracle Real Application Clusters (Oracle RAC) databases. The main 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.

This diagram illustrates a basic True Cache configuration with GDS in a single region:

Figure 5-1 True Cache Integration with GDS



In this example, there is one region with two True Caches and one primary database. True Cache reads from the primary database to warm up the cache or when there's a cache miss. After a block is cached, it's updated automatically through redo apply from the primary database. Applications are configured so that frequent reads go to the True Caches and writes and infrequent reads go to the primary database.

The GDS configuration includes the following components:

- The GDS catalog stores configuration data for the GDS configuration. In this example, the GDS catalog is hosted outside the primary database.
- Global service managers (GSMs) provide service-level load balancing, failover, and centralized management of services in the GDS configuration. For high availability, the best practice is to include two GSMs in each region.
- The GDS pool provides a common set of global services to the primary database and True Caches. A region can contain multiple GDS pools, and these pools can span multiple regions.In this example, the `SALES` pool provides the `SALES` global service for the primary database and the `SALES_TC` global service for the True Caches.

- An Oracle Notification Service (ONS) server is located with each GSM. All ONS servers in a region are interconnected in an ONS server network. The GSMs create runtime load balancing advisories and publish them to the Oracle universal connection pool (UCP) JDBC connection pool through the ONS server network. Applications request a connection from the pool and the requests are sent to True Caches or the primary database depending on the application configuration and the GDS configuration.

True Cache provides the following integrations with Oracle Global Data Services (GDS).

- GDS provides the `-role TRUE_CACHE` option for global services.
- The True Cache application programming model using `JDBC SetReadOnly()` supports global services.
- GDS provides load balancing and service failover between multiple True Caches.

Deploying True Cache with Oracle GDS has the following restrictions.

- When adding True Cache services in GDSCTL, the `-failover_primary` option requires the patch for bug 36740927.
- If the application uses the JDBC programming model, both the primary database service and True Cache service names must be fully qualified with the domain name (for example, `sales.example.com` and `sales_tc.example.com`). This is because GDS has a default domain name and is different from the database's `domain_name` parameter. This also limits the fully qualified service name to a maximum of 64 characters.

For more detailed instructions regarding the deployment of Oracle True Cache with Oracle GDS, please see Deploying Oracle True Cache with Oracle Global Data Services

5.3 Supported Replication Technologies and Implementation Architectures

In environments where databases are replicated for high availability, disaster recovery, or read scaling, Oracle GDS provides:

- **Global Service Management**: GDS creates a global service abstraction, masking the complexity of the underlying replicated databases from applications.
- **Workload Routing**: GDS intelligently routes client requests to the appropriate database instances based on factors like database role (read-write, read-only), region, replication-lag, and resource capacity.
- **Load-balancing**: GDS dynamically balances workloads taking into account load conditions and network latency.
- **Failover and Switchover**: GDS automates data service failover to standby databases in case of failures, ensuring continuous application availability.
- **Centralized Management**: The GDS catalog provides a central point for managing and monitoring the configuration and health of global services and their associated databases.

Replicated Databases Use Cases

- **Disaster Recovery Systems**: Replicates a primary database to a standby database for failover in case of a primary system failure.
- **Content Management Systems (CMS)**: Read-heavy websites (such as content delivery, ecommerce, airline reservation systems) that replicate data for faster access across multiple regions.

- **Business Intelligence (BI) and Analytics**: Replicated databases handle complex, read-heavy queries on a replica while offloading the primary database.
- **Customer Relationship Management (CRM) Systems**: Replicates customer data for distributed teams, ensuring high availability and quick access to client information.
- **Mobile Applications**: Ensures low-latency read access to globally distributed user bases by replicating databases in multiple regions.

5.3.1 Using Oracle Active Data Guard with Global Data Services

Configure sessions to move in a rolling manner for Oracle Active Data Guard reader farm.

Prerequisites

You must have the following in place for this procedure to work correctly.

- Oracle Active Data Guard configuration using Oracle Database (release 19c or later recommended).
- Global Data Services (GDS) configuration using global service manager (release 19c or later recommended).
- A GDS service has been created to run on all Active Data Guard databases in the configuration.

  For example:

```
GDSCTL> add service -service sales_sb -preferred_all -gdspool sales
 -role physical_standby -notification TRUE
GDSCTL> modify service -gdspool sales -service sales_sb -database mts -
add_instances
 -preferred mts1,mts2
GDSCTL> modify service -gdspool sales -service sales_sb -database stm -
add_instances
 -preferred stm1,stm2
GDSCTL> start service -service sales_sb -gdspool sales
```

1. Check the current status of the services and related instances to ensure that services can be moved successfully.

   Note that the service should only be available on the source standby database at this point.

```
GDSCTL> services
Service "sales_sb.sales.oradbcloud" has 2 instance(s). Affinity: ANYWHERE
   Instance "sales%1", name: "mts1", db: "mts", region: "slc", status:
ready.
   Instance "sales%2", name: "mts2", db: "mts", region: "slc", status:
ready.
```

2. Stop services typically (not using the FORCE option) on the source database where connections are to be removed.
   - This step will quiesce the FAN-aware connection pools using FAN.
   - New connections are directed to other instances offering that service, and idle sessions are disconnected from the pool using the services.

- Existing connections can continue until their work is complete and they are returned to the connection pool.

```
GDSCTL> stop service -service sales_sb -database mts -gdspool sales
```

Allow an agreed upon time for the sessions to disconnect and relocate, then continue with the next steps.

> **Note:**
>
> If you are performing a rolling upgrade of an Active Data Guard reader farm and the services are not running on other Active Data Guard reader nodes, you can complete the service stop on this database before performing the GDSCTL `stop service` described in this step.

3. Disconnect long-running sessions after the current query is completed.

   Preferably, long-running queries have been scheduled to stop or are queued before the window when connections are to be moved. This step handles long-running sessions that are still running and now need to be stopped (killed) abruptly.

4. Log on to the instance that you intend to shut down.
5. Check V$SESSION to see if any sessions are still connected to the service.

```
SQL> SELECT service_name, count(1) FROM v$session
 GROUP BY service_name ORDER BY 2;
```

6. Run the `DBMS_SERVICE.DISCONNECT_SESSION` package for the service you stopped earlier.

   For example:

```
SQL> exec

dbms_service.disconnect_session('oltp_work',DBMS_SERVICE.POST_TRANSACTION);
```

7. Check `V$SESSION` again to ensure that sessions have logged off from the service.

```
SQL> SELECT service_name, count(1) FROM v$session
 GROUP BY service_name ORDER BY 2;
```

8. Start the GDS service on the target database and allow sessions to connect.

```
GDSCTL>start service -service sales_sb -database stm -gdspool sales
```

9. Log on to the target database and check `V$SESSION` to see sessions connected to the service.

```
SQL> SELECT service_name, count(1) FROM v$session
 GROUP BY service_name ORDER BY 2;
```

5.3.2 Using Oracle GoldenGate with Global Data Services

The following Oracle GoldenGate role transition example topology consists of two databases: GG replica1 and GG replica2. Oracle GoldenGate is set up with uni-directional replication, with Extract running initially on GG replica1 and Replicat running initially on GG replica2. The generic steps still apply for bi-directional GoldenGate replicas or downstream mining GoldenGate replicas.

Prerequisites

You must have the following in place for this procedure to work correctly.

- Oracle GoldenGate configuration that uses Oracle Database (19c or higher recommended)
- GoldenGate processes should not connect to the source or target database using the GDS service name, but a dedicated TNS alias. Using the GDS service will cause the database connections to terminate prematurely, causing possible data loss.
- A heartbeat table has been implemented in the GoldenGate source and target databases to track replication latency and ensure the Replicat applied SCN synchronization. The GoldenGate automatic heartbeat table feature should be enabled. Refer to the Oracle GoldenGate Administration Guide for details on the automatic heartbeat table: https://docs.oracle.com/en/middleware/goldengate/core/19.1/gclir/add-heartbeattable.html.
- Global Data Services (GDS) configuration using global service manager (19c or higher recommended)
- GDS service has been created so that it can be run on all databases in the GoldenGate configuration.

  For example:

```
GDSCTL> add service -service sales_sb -preferred_all -gdspool sales
GDSCTL> modify service -gdspool sales -service sales_sb -database mts
 -add_instances -preferred mts1,mts2
GDSCTL> modify service -gdspool sales -service sales_sb -database stm
 -add_instances -preferred stm1,stm2
GDSCTL> start service -service sales_sb -gdspool sales
```

> **Note:**
>
> If you are using the lag tolerance option, specify the lag limit for the global service in seconds. Options for `add service` or `modify service` are `-lag {lag_value | ANY}`.

1. Check the current status of the services and related instances to ensure that they can be moved successfully.

   At this point, the service should only be available on the source database.

```
GDSCTL> services
Service "sales_sb.sales.oradbcloud" has 2 instance(s). Affinity: ANYWHERE
   Instance "sales%1", name: "mts1", db: "mts", region: "slc", status:
ready.
   Instance "sales%2", name: "mts2", db: "mts", region: "slc", status:
ready.
```

2. Stop services (not using the FORCE option) on the source database where connections are to be removed.
   - This step will quiesce the FAN-aware connection pools using FAN.
   - New connections are directed to other instances offering that service, and idle sessions are disconnected from the pool using the services.
   - Existing connections can continue until their work is complete and they are returned to the connection pool.

   ```
   GDSCTL> stop service -service sales_sb -database mts -gdspool sales -force
   ```

   Allow an agreed upon time for the sessions to disconnect and relocate, then continue with the next steps. The time to allow for sessions to drain depends on the workload and user transactions for your application.
3. Disconnect long-running sessions after the current transaction is completed.

   Preferably, long-running batch jobs are scheduled to be stopped or queued before the maintenance window. This step handles long-running sessions that are still running and must be stopped abruptly (e.g., killed). Check with the application developers if these long-running batch jobs are idempotent and recoverable before disconnecting long-running sessions.
4. Log on to the instance that you intend to shut down, and check `V$SESSION` to see if any sessions are still connected to the service.

   ```
   SQL> SELECT service_name, count(1) FROM v$session
    GROUP BY service_name ORDER BY 2;
   ```

5. Run the `DBMS_SERVICE.DISCONNECT_SESSION` package for the service you stopped earlier.

   For example:

   ```
   SQL> exec

   dbms_service.disconnect_session('oltp_work',DBMS_SERVICE.POST_TRANSACTION);
   ```

6. Check `V$SESSION` again to ensure sessions have logged off from the service.

   ```
   SQL> SELECT service_name, count(1) FROM v$session
    GROUP BY service_name ORDER BY 2;
   ```

7. When all sessions associated with the GDS service have been disconnected, verify that all data from the GoldenGate source databases have been replicated to the target database.
   - Record the current database SCN from the GoldenGate SOURCE database.

     ```
     SQL> SELECT current_scn FROM v$database;
     ```

   - On the GoldenGate TARGET database, continue to monitor the Replicat applied SCN using the following query.

     ```
     SQL> SELECT lwm_position FROM v$gg_apply_coordinator;
     ```

   - When the target `LWM_POSITION` SCN is greater than the `CURRENT_SCN` recorded in the first step, it is safe to assume that all transactions have been replicated from the

source to the target database. The users can now be switched over to the GoldenGate target database.

The above steps allow for a graceful switchover. However, if this is a failover event where the source database is unavailable, you can estimate the data loss using the steps below.

1. When using the automatic heartbeat table, use the following query to determine the replication latency.

```
SQL> col Lag(secs) format 999.9
SQL> col "Seconds since heartbeat" format 999.9
SQL> col "GG Path" format a32
SQL> col TARGET format a12
SQL> col SOURCE format a12
SQL> set lines 140
SQL> select remote_database "SOURCE", local_database "TARGET",
incoming_path "GG Path",
 incoming_lag "Lag(secs)", incoming_heartbeat_age "Seconds since
heartbeat" from gg_lag;

SOURCE        TARGET              GG Path
Lag(secs) Seconds since heartbeat
------------  ------------        --------------------------------
--------- ----------------------
         MTS          GDST          EXT_1A ==> DPUMP_1A ==> REP_1A
 7.3                     9.0
```

   The above example shows a possible 7.3 seconds of data loss between the source and target databases.

2. Start the GDS service on the target database and allow sessions to connect.

   Note that if the application workload can accept a certain level of data lag, it is possible to perform this step much earlier than step two listed above.

```
GDSCTL> start service -service sales_sb -database stm -gdspool sales
```

3. Log on to the target database and check `V$SESSION` to see sessions connected to the service.

```
SQL> SELECT service_name, count(1) FROM v$session
 GROUP BY service_name ORDER BY 2;
```

5.3.3 Using RAFT Replication with Global Data Services

Raft replication is a consensus-based replication protocol that facilitates automatic configuration of replication across all shards in a distributed datbase. Raft replication seamlessly integrates with applications, providing transparency in its operation. In case of shard host failures or dynamic changes in the distributed database's composition, Raft replication automatically reconfigures replication settings. The system takes a declarative approach to configure the replication factor, ensuring a specified number of replicas are consistently available.

When Raft replication is enabled, a distributed database contains multiple replication units. A replication unit (RU) is a set of chunks that have the same replication topology. Each RU has multiple replicas placed on different shards. The Raft consensus protocol is used to maintain

consistency between the replicas in case of failures, network partitioning, message loss, or delay.

Swift failover is a key attribute of Raft replication, enabling all nodes to remain active even in the event of a node failure. Notably, this feature incorporates an automatic sub-second failover mechanism, reinforcing both data integrity and operational continuity. Such capabilities make this feature well-suited for organizations seeking a highly available and scalable database system.

Oracle GDS provides GDSCTL commands and options to enable and manage Raft replication in a system-managed sharded database.

Enabling Raft Replication

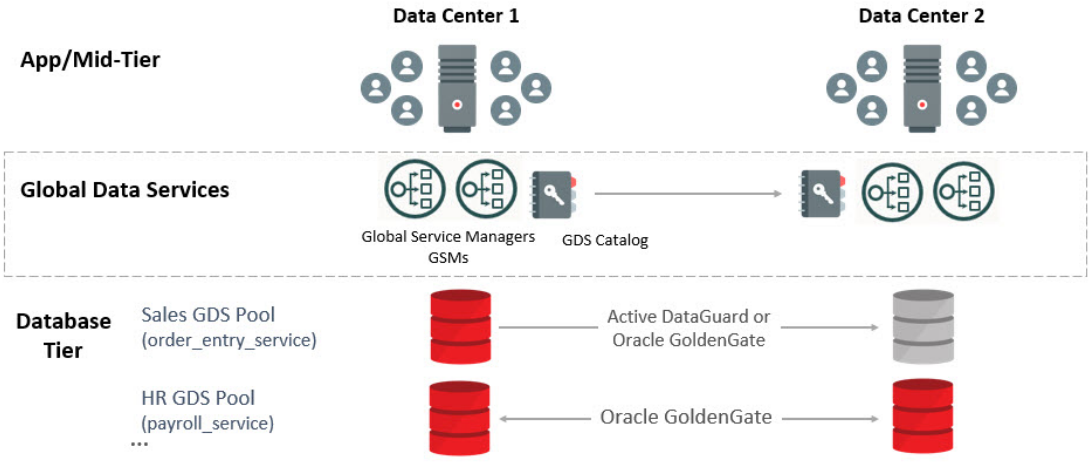
You enable Raft replication when you configure the shard catalog. To enable Raft replication, specify the native replication option in the create shardcatalog command when you create the shard catalog. For example, `gdsctl> create shardcatalog ... -repl native`

> **Note:**
>
> For more information regarding configuring and deploying RAFT Replication see: Raft Replication Configuration and Management

5.3.4 One GDS Infrastructure for Many Replicated Configurations

Architecture example where databases might be replicated for high availability, disaster recovery, or read scaling.

Figure 5-2 GDS Replicated Configurations Architecture Example



5.4 Summary: Replicated Databases vs Distributed Databases

Figure 5-3 Replicated Databases vs Distributed Databases

Feature	Replicated Databases	Distributed Databases
Primary Function	Workload routing, load balancing, service failover	Shard routing, data locality, query optimization
Core Components	Global Service Managers, GDS Catalog	Shard Director, Shard Catalog
Focus	High availability, disaster recovery, read scaling	Global scalability, performance, data distribution
Key Concepts	Database services, replication roles	Data partitioning, parallel query, sharding strategies

5.5 Application Development Considerations

5.5.1 Application Workload Suitability for Global Data Services

Global Data Services (GDS) is best for replication-aware application workloads; it is designed to work in replicated environments. Applications that are suitable for GDS adoption possess any of the following characteristics:

- The application can separate its work into read-only, read-mostly, and read-write services to use Oracle Active Data Guard or Oracle GoldenGate replicas. GDS does not distinguish between read-only, read-write, and read-mostly transactions. The application connectivity has to be updated to separate read-only or read-mostly services from read-write services, and the GDS administrator can configure the global services on appropriate databases. For example, a reporting application can function directly with a read-only service at an Oracle Active Data Guard standby database.
- Administrators should be aware of and avoid or resolve multi-master update conflicts to take advantage of Oracle GoldenGate replicas. For example, an internet directory application with built-in conflict resolution enables the read-write workload to be distributed across multiple databases, each open read-write and synchronized using Oracle GoldenGate multi-master replication.
- Ideally, the application is tolerant of replication lag. For example, a web-based package tracking application that allows customers to track the status of their shipments using a read-only replica, where the replica does not lag the source transactional system by more than 10 seconds.

5.5.2 Using FAN ONS with Global Data Services

Fast Application Notification (FAN) uses the Oracle Notification Service (ONS) for event propagation to all Oracle Database clients, including JDBC, Tuxedo, and listener clients. ONS is installed as part of Oracle Global Data Services, Oracle Grid Infrastructure on a cluster, in an Oracle Data Guard installation, and when Oracle WebLogic is installed. ONS propagates FAN

events to all other ONS daemons it is registered with. No steps are needed to configure or enable FAN on the database server side, with one exception: OCI FAN and ODP FAN require that notification be set to `TRUE` for the service by GDSCTL. With FAN auto-configuration at the client, ONS jar files must be on the `CLASSPATH` or in the `ORACLE_HOME`, depending on your client.

General Best Practices for Configuring FCF Clients

Follow these best practices before progressing to driver-specific instructions.

- Use a dynamic database service. Using FAN requires that the application connects to the database using a dynamic global database service. This is a service created using GDSCTL.
- Do not connect using the database service or PDB service. These services are for administration only and are not supported for FAN. The TNSnames entry or URL must use the service name syntax and follow best practices by specifying a dynamic database service name. Refer to the examples later in this document.
- Use the Oracle Notification Service when you use FAN with JDBC thin, Oracle Database OCI, or ODP.Net clients. FAN is received over ONS. Accordingly, in the Oracle Database, ONS FAN auto-configuration is introduced so that FCF clients can discover the server-side ONS networks and self-configure. FAN is automatically enabled when ONS libraries or jars are present.
- Enabling FAN on most FCF clients is still necessary in the Oracle Database. FAN auto-configuration removes the need to list the global service managers an FCF client needs.
- Listing server hosts is incompatible with location transparency and causes issues with updating clients when the server configuration changes. Clients already use a TNS address string or URL to locate the global service manager listeners.
- FAN auto-configuration uses the TNS addresses to locate the global service manager listeners and then asks each server database for the ONS server-side addresses. When there is more than one global service manager FAN auto-configuration contacts each and obtains an ONS configuration for each one.
- The ONS network is discovered from the URL when using the Oracle Database. An ONS node group is automatically obtained for each address list when `LOAD_BALANCE` is off across the address lists.
- By default, the FCF client maintains three hosts for redundancy in each node group in the ONS configuration.
- Each node group corresponds to each GDS data center. For example, if there is a primary database and several Oracle Data Guard standbys, there are by default three ONS connections maintained at each node group. The node groups are discovered when using FAN auto-configuration.

  With `node_groups` defined by FAN auto-configuration, and `node_groups` (the default), more ONS endpoints are not required. If you want to increase the number of endpoints, you can do this by increasing max connections. This applies to each node group. Increasing to 4 in this example maintains four ONS connections at each node. Increasing this value consumes more sockets.

  ```
  oracle.ons.maxconnections=4 ONS
  ```

- If the client is to connect to multiple clusters and receive FAN events from them, for example in Oracle RAC with a Data Guard event, then multiple ONS node groups are needed. FAN auto-configuration creates these node groups using the URL or TNS name. If automatic configuration of ONS (Auto-ONS) is not used, specify the node groups in the Oracle Grid Infrastructure or oraaccess.xml configuration files.

5.5.3 Client Side Configuration

As a best practice, multiple global service managers should be highly available. Clients should be configured for multiple connection endpoints where these endpoints are global service managers rather than local, remote, or single client access name (SCAN) listeners. For OCI / ODP .Net clients use the following TNS name structure.

```
(DESCRIPTION=(CONNECT_TIMEOUT=90)(RETRY_COUNT=30)(RETRY_DELAY=3)
(TRANSPORT_CONNECT_TIMEOUT=3)
  (ADDRESS_LIST =
   (LOAD_BALANCE=on)
   (ADDRESS=(PROTOCOL=TCP)(HOST=GSM1)(PORT=1522))
   (ADDRESS=(PROTOCOL=TCP)(HOST=GSM2)(PORT=1522))
   (ADDRESS=(PROTOCOL=TCP)(HOST=GSM3)(PORT=1522)))
  (ADDRESS_LIST=
   (LOAD_BALANCE=on)
   (ADDRESS=(PROTOCOL=TCP)(HOST=GSM2)(PORT=1522)))
 (CONNECT_DATA=(SERVICE_NAME=sales)))
```

Always use dynamic global database services created by GDSCTL to connect to the database. Do not use the database service or PDB service, which are for administration only not for application usage and they do not provide FAN and many other features because they are only available at mount. Use the latest client driver aligned with the latest or older RDBMS for JDBC.

Use one `DESCRIPTION` in the TNS names entry or URL Using more causes long delays connecting when `RETRY_COUNT` and `RETRY_DELAY` are used. Set `CONNECT_TIMEOUT=90` or higher to prevent logon storms for OCI and ODP clients.

5.5.4 Configuring FAN for Java Clients Using Universal Connection Pool

The best way to take advantage of FCF with the Oracle Database JDBC thin driver is to use the Universal Connection Pool (UCP) or WebLogic Server Active GridLink.

Setting the pool property `FastConnectionFailoverEnabled` on the Universal Connection Pool enables Fast Connection Failover (FCF). Active GridLink always has FCF enabled by default. Third-party application servers, including IBM WebSphere and Apache Tomcat, support UCP as a connection pool replacement.

For more information about embedding UCP with other web servers, see the following technical briefs.

- Design and deploy WebSphere applications for planned or unplanned database downtimes and runtime load balancing with UCP (https://www.oracle.com/docs/tech/database/planned-unplanned-rlb-ucp-websphere.pdf)
- Design and deploy Tomcat applications for planned or unplanned database downtimes and Runtime Load Balancing with UCP (https://www.oracle.com/docs/tech/database/planned-unplanned-rlb-ucp-tomcat.pdf)

Follow these configuration steps to enable Fast Connection Failover.

1. The connection URL must use the service name syntax and follow best practices by specifying a dynamic database service name and the JDBC URL structure (above and below).
   All other URL formats are not highly available. The URL may use JDBC thin or JDBC OCI.

2. If wallet authentication has not been established, remote ONS configuration is needed. Set the pool property `setONSConfiguration` in a property file as shown in the following example. The property file specified must contain an ons.nodes property and, optionally, properties for `oracle.ons.walletfile` and `oracle.ons.walletpassword`. An example of an `ons.properties` file is shown here.

```
PoolDataSource pds = PoolDataSourceFactory.getPoolDataSource();
pds.setConnectionPoolName("FCFSamplePool");
pds.setFastConnectionFailoverEnabled(true);
 pds.setONSConfiguration("propertiesfile=/usr/ons/ons.properties");
pds.setConnectionFactoryClassName("oracle.jdbc.pool.OracleDataSource");
pds.setURL("jdbc:oracle:thin@((CONNECT_TIMEOUT=4)(RETRY_COUNT=30)
(RETRY_DELAY=3) "+ "
 (ADDRESS_LIST = "+ " (LOAD_BALANCE=on) "+ " ( ADDRESS =
 (PROTOCOL = TCP)(HOST=GSM1)(PORT=1522))) "+ " (ADDRESS_LIST = "+
" (LOAD_BALANCE=on)
 "+ "( ADDRESS = (PROTOCOL = TCP)(HOST=GSM2)(PORT=1522)))"+
 "(CONNECT_DATA=(SERVICE_NAME=service_name)))");
```

3. Ensure the pool property `setFastConnectionFailoverEnabled=true` is set.
4. The `CLASSPATH` must contain `ons.jar`, `ucp.jar`, and the JDBC driver jar file. For example, `ojdbc8.jar`.
5. If you are using JDBC thin with Oracle Database, Application Continuity can be configured to failover the connections after FAN is received.
6. If the database needs different ONS endpoints than those autoconfigured, the ONS endpoints can be enabled.
   In a situation where multiple clusters exist with auto-ons enabled, auto-ons would generate node lists with the following guidelines:

   For EVERY active nodelist `oracle.ons.maxconnections` is set to 3 by default, so there is no need to set this explicitly. This example will result in the ONS client trying to maintain six total connections.

5.5.5 Configuring FAN for OCI Clients

OCI clients embed FAN at the driver level so that all clients can use it regardless of the pooling solution. Ideally, both the server and the client would use release 19c or later.

Configuration for SQL*Plus and PHP

1. Set notification for the service.
2. For PHP clients only, add `oci8.events=On` to `php.ini`.
   Important: If xml is present with `events=-false` or events are not specified, this disables the usage of FAN. To maintain FAN with SQL*Plus and PHP when `oraccess.xml` is in use, set `events=-true`.
3. On the client side, using a client and Oracle Database, enable FAN in xml.

Configuration for OCI Clients

1. Tell OCI where to find ONS Listeners Starting.
   The client installation comes with ONS linked into the client library. Using auto-config, the ONS endpoints are discovered from the TNS address. This automatic method is the recommended approach. Like ODP.Net, manual ONS configuration is also supported using `oraaccess.xml`.

2. Enable FAN high availability events for the OCI connections.
   To enable FAN you edit the OCI file xml to specify the global parameter events. This file is located in `$ORACLE_HOME/network/admin`. See Step 3: Ensure That FAN Is Used for more information.
3. Tell OCI where to find ONS Listeners.
   The client installation comes with ONS linked into the client library. Using auto-config, the ONS endpoints are discovered from the TNS address. This automatic method is the recommended approach. Like ODP.Net, manual ONS configuration is also supported using `oraaccess.xml`.
4. Enable FAN on the server for all OCI clients.
   It is still necessary to enable FAN on the database server for all OCI clients (including SQL*Plus).

5.5.6 Controlling Logon Storms

Small connection pools are strongly recommended, but controlling logon storms can be done with tuning when you have many connections.

Oracle MAA recommends the following tuning on servers that host Global Service Managers.

1. Increase the Listen backlog at the OS level.

   To have the new value take effect without rebooting the server, perform the following as root.

   ```
   echo 8000 > /proc/sys/net/core/somaxconn
   ```

2. To persist the value across reboots, add this setting to /etc/sysctl.conf.

   ```
   net.core.somaxconn=6000
   ```

3. Increase `queuesize` for the global service manager listener.

   Update ora in Oracle home that the listeners are running from to increase the `queuesize` parameter:

   ```
   TCP.QUEUESIZE=6000
   ```

6

Troubleshooting Global Data Services

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

6.1 Obtaining the Running Status of Global Data Services Components

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

```
GDSCTL>status gsm

GDSCTL>status service

GDSCTL>status database
```

6.2 Viewing Static Configuration Information for Global Data Services Components

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

```
GDSCTL>config

GDSCTL>config gsm

GDSCTL>config region

GDSCTL>config gdspool

GDSCTL>config database

GDSCTL>config service

GDSCTL>config invitednode
```

6.3 Networking Issues

This topic includes the following issues:

- `ORA-12514: TNS:listener does not currently know of service requested in connect descriptor`
- `ORA-12516: TNS:listener could not find available handler with matching protocol stack`
- `ORA-12541: TNS:no listener`
- `GSM-40167: VNCR entry "<hostname>" is not resolvable on GSM host`

6.3.1 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.

6.3.2 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.

6.3.3 ORA-12541: TNS:no listener

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

6.3.4 GSM-40167: VNCR entry "<*hostname*>" is not resolvable on GSM host

GSM restart after GSM-upgrade fails with error: `GSM-40167: VNCR entry "<hostname>" is not resolvable on GSM host`

This issue is most likely due to the presence of short hostnames in the VNCR list causing an issue during GSM start in 23ai. If this issue is encountered, remove invitednode of the two short shard hostname entries and then try to start the GSM.

6.4 `add brokerconfig` Related Issues

A common issue that could happen during configuration of GDS environment is absence of database registration on GSM listeners. To detect this customers typically use GDSCTL `status database` command. The output of this command will show:

```
status database
Database: "test" Registered: N State: Ok ONS: Y. Role: N/A Instances: 0
Region: region1
Service: "test_failover" Globally started: Y Started: N
Scan: Y Enabled: Y Preferred: Y
```

The noteworthy part of this output is `Registered` property that shows the value `N`, indicating No registration.

Remember that database registration on a GDS listener requires a region to be assigned to the database. If the `add brokerconfig` command was used to add databases to GDS pool in

presence of two or more regions and the status database command shows No registration, assign the region manually using the `modify database` command:

```
modify database … -region <region_name>
```

If you are adding more than one standby database to the GDS configuration simultaneously, you might receive an `ORA-45539: Database db_name has already been added to another pool`. The solution is to add and sync them one at a time.

6.5 Invalid Objects Related Issues

After installing a Database Release Update (DBRU) or individual patches, some objects in the `GSMADMIN_INTERNAL` schema may become invalid in the database as a whole or in individual containers (PDBs). For example, assume the query below is issued after a DBRU:

```
select owner, object_name, object_type, status from dba_objects
where status='INVALID' and owner='GSMADMIN_INTERNAL' order by owner,
object_name, object_type;
OWNER                 OBJECT_NAME               OBJECT_TYPE      STATUS
CON_ID
-----------------     ----------------------    -------------    -------
------
GSMADMIN_INTERNAL     DBMS_GSM_POOLADMIN        PACKAGE BODY     INVALID    3
GSMADMIN_INTERNAL     CAT_ROLLBACK_TRIGGER      TRIGGER          INVALID    3
GSMADMIN_INTERNAL     REQUEST_DELETE_TRIGGER    TRIGGER          INVALID    3
GSMADMIN_INTERNAL     DONE_TRIGGER              TRIGGER          INVALID    3
...
```

The datapatch script execution failed due to Invalid Objects on GSMADMIN_INTERNAL schema. In order to resolve the invalid object issues, re-running Oracle-provided scripts is necessary by following these steps:

1. Ensure that the GSMADMIN_INTERNAL schema does not contain any user data.

If the Global Data Services (GDS) or Oracle Sharding features are in use, the GSMADMIN_INTERNAL schema may contain important configuration data.

Run the following query as SYS to verify that the schema is empty. For container databases (CDBs), this query MUST be run in CDB$ROOT and in all pluggable databases (PDBs).

```
select count(*) from gsmadmin_internal.cloud;
```

If this query returns 0 (zero), then move on to the next step. If it returns non-zero, contact Oracle Customer Support for further guidance.

2. Connect to the database as SYS using SQL*Plus. If the database is a CDB, connect to CDB$ROOT, not to any individual PDB.

3. Run the following commands:

```
SQL> spool /tmp/invalids.out
SQL> alter session set "_ORACLE_SCRIPT" = true;
SQL> drop user gsmadmin_internal cascade;
SQL> alter session set "_ORACLE_SCRIPT" = false;
```

```
SQL> @?/rdbms/admin/catgwm.sql
SQL> @?/rdbms/admin/dbmsgwm.sql
SQL> @?/rdbms/admin/catgwmcat.sql
SQL> @?/rdbms/admin/prvtgwm.sql
SQL> @?/rdbms/admin/utlrp.sql
SQL> spool off
```

4. Validate that there are no invalid objects owned by GSMADMIN_INTERNAL by running the following query: `select owner, object_name, object_type, status from dba_objects where status='INVALID' and owner='GSMADMIN_INTERNAL' order by owner, object_name, object_type;`

If no rows are returned, then the GSMADMIN_INTERNAL schema is now correct.

If there are still invalid objects, then contact Oracle Customer Support for further guidance and provide the output from the scripts as found in /tmp/invalids.out.

5. Finally, in certain instances, it is possible that some GSMADMIN_INTERNAL objects have been incorrectly created in the SYS schema.

These objects should therefore be dropped since they are duplicates of the correct objects in the GSMADMIN_INTERNAL schema.

To identify objects owned by SYS which should be dropped, run the following query as SYS: `select object_name, object_type from dba_objects where owner = 'SYS' and (object_name, object_type) in (select object_name, object_type from dba_objects where owner = 'GSMADMIN_INTERNAL') order by object_name, object_type;`

This will generate a list of all objects owned by SYS that are also in GSMADMIN_INTERNAL.

Since these objects were created in the wrong schema, they should all be dropped using the appropriate DROP commands from SQL*Plus.

Re-running this query should return 0 (zero) rows when all listed objects have been dropped.

6.6 User and Password Management Issues

This section contains information for troubleshooting user and password issues, such as:

- ORA-01045

6.6.1 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.

6.7 Troubleshooting GDS Issues

This topic includes the following issues:

- `GSM-45034: Connection to GDS catalog is not established`
- Connecting to GDS Configuration Databases When No Global Service Managers Are Running

- Connecting to Catalog Databases When No Global Service Managers Are Running
- Using `SYS_CONTEXT` Parameters in a GDS Environment

6.7.1 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.

6.7.2 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.oradbcloud)
 (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.

6.7.3 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.

6.7.4 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.

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*>

6.8 Troubleshooting GSM Issues

This topic includes the following issues:

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

6.8.1 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.

6.9 GDS Logs and Tracing

This section contains information for locating log files and enabling tracing of GDS components:

- Using Global Service Manager Log and Trace Files
- Enabling and Disabling Tracing on a Global Service Manager

6.9.1 Using Global Data Services Log and Trace Files

If logging and tracing has been enabled for GDS components, you can find the exact location of a given global service manager's log and trace files using the `status gsm` command as shown in the following example.

```
GDSCTL>status gsm

Alias                     MYGSM
Version                   19.1.0.0.1
Start Date                22-OCT-2022 14:05:11
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.

Although not strictly a GDS component, the LISTENER log file can be helpful resolving some issues. 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_gsm1.log`)

If logging and tracing have not been enabled for GDS components, follow the steps below to do so:

1. Enable logging for GDSCTL. On the GSM host, edit the `$ORACLE_HOME/network/admin/gsm.ora` file and add the following: `_GDSCTL=(log=ALL)`

   This will log any commands run using GDSCTL in the following log file on the GSM host. `$ORACLE_HOME/network/admin/GDSCTL.log`

2. Enable tracing for GSM processes.

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

This will generate trace files in `DIAGNOSTICS_DEST` for GSM. For example, `$ORACLE_BASE/diag/gsm/<hostname>/<gsm-name>/trace`

3. Enable tracing on the catalog database.

```
SQL> alter system set events '10798 trace name context forever, level 7';
```

4. Enable tracing on the GDS pool databases.

```
SQL> alter system set events '10798 trace name context forever, level 7';
```

To disable logging and tracing, follow the steps below:

1. Turn off GSM tracing.

```
GDSCTL> set trace_level -gsm <gsm-name> OFF
```

2. Turn off GDSCTL command logging by editing the gsm.ora file and removing the following line:

```
_GDSCTL=(log=ALL)
```

3. Turn off tracing on the catalog database and GDS pool databases.

```
SQL> alter system set events '10798 trace name context forever, level 0';
```

6.9.2 Advanced Global Data Services Troubleshooting

To effectively troubleshoot Global Data Services, it is important to collect adequate data from GDS and related components. Following the steps below will provide a good data pool for troubleshooting issues:

1. Collecting patch information can be useful for code version-related issues:

```
$ cd $ORACLE_HOME/OPatch $ ./opatch lsinventory -detail > lsinventory_info.txt
```

2. Collect all GDS configuration data. Save the following commands in a script and execute it:

```
#!/bin/bash
# Stop script if any command fails
set -e
# Check if gdsctl is available in PATH
if ! command -v gdsctl &> /dev/null
then
    echo "gdsctl could not be found. Please make sure it is installed and available
in PATH."
    exit 1
fi
echo "Starting gdsctl session..."
# Start the gdsctl session
gdsctl << EOF
# List GSMs, GSM status
echo "GDSCTL COMAND: config"
config
echo "GDSCTL COMAND: config gsm -gsm <gsm name>"
#config gsm -gsm <gsm name>
echo "GDSCTL COMAND: gsm status"
gsm status
# List services
echo "GDSCTL COMAND: services"
services
echo "GDSCTL COMAND: config service -service <service name>"
#config service -service <service name>
# List databases
echo "GDSCTL COMAND: databases"
```

```
databases
# Network
echo "GDSCTL COMAND: config vncr"
config vncr
# Validate config
echo "GDSCTL COMAND: validate"
validate
# End of gdsctl commands
EOF
echo "gdsctl session completed successfully."
# Exit the script
exit 0
```

3. Collect GSM listener status.

```
$ lsnrctl status <gsm-name>
```

If it is necessary to contact Oracle Support, the above data will prove useful. In addition, the following files would also be helpful:

From GSM Host: `$ORACLE_HOME/network/admin`

From GSM Host: GSM alert log file

From GSM Host: `GDSCTL.log`

From GSM Host: GSM trace files located in *<DIAGNOSTIC_DEST>*

From GSM Host: `lsinventory`, `lsinventory_info.txt`

From GDS Catalog Host: `lsinventory`, `lsinventory_info.txt`

A

GDSCTL Commands Used For Oracle Globally Distributed Database

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

The following GDSCTL commands are commonly used in an Oracle Globally Distributed Database configuration:

- add cdb
- add credential
- add file
- add gsm
- add invitednode (add invitedsubnet)
- add region
- add service
- add shard
- add shardgroup
- add shardspace
- alter move
- alter move
- alter task
- config
- config backup
- 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 task

- config vncr
- configure
- connect
- copy ru
- create restorepoint
- create shardcatalog
- delete backup
- delete catalog
- deploy
- disable backup
- disable service
- enable backup
- enable service
- list backup
- list restorepoint
- modify catalog
- modify cdb
- modify credential
- modify file
- modify gsm
- modify region
- modify service
- modify shard
- modify shardgroup
- modify shardspace
- move chunk
- move ru
- relocate chunk
- relocate service
- remove cdb
- remove credential
- remove file
- remove gsm
- remove invitednode (remove invitedsubnet)
- remove region
- remove ru
- remove service
- remove shard

- remove shardgroup
- remove shardspace
- restore backup
- services
- set gsm
- set inbound_connect_timeout
- set log_status
- set outbound_connect_timeout
- set trace_level
- split chunk
- sql
- start gsm
- start ru
- start service
- status backup
- status gsm
- status ru
- status service
- stop gsm
- stop ru
- stop service
- switchover ru
- sync ru
- sync schema (synchronize schema)
- validate backup
- validate catalog

> **See Also:**
>
> *Oracle Globally Distributed Database Guide* 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_timeout
- set log_status
- set outbound_connect_timeout
- 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)
- sync schema (synchronize schema)
- 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 Globally Distributed Database configuration.

C.1 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
`-connect` *connect_identifier*	Specify an Oracle Net connect descriptor or net service name that resolves to a connect descriptor for a database in the broker configuration.
`-force`	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.
`-gdspool` *gdspool_name*	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.
`-pwd` *password*	The password for the GSMUSER. If `-pwd` is not specified, then you are prompted for the password.
`-region` *region_name*	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 `-connect` 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.

C.2 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
-connect *connect_identifier*	Specify an Oracle Net connect descriptor or net service name that resolves to a connect descriptor for the database being added as the shard.
-pwd *gsmrootuser_pwd*	Enter the GSMROOTUSER password. If not specified, the user is prompted for the password.
-savename	Store in the shard catalog a net service name specified with the -connect option rather than the connect descriptor mapped to that net service name.
-force	If specified, the existing GDS and sharding configuration on the shard and in the shard catalog with information about this shard will be rewritten.
-cpu_threshold *cpu*	Specify the CPU Utilization percentage threshold.
-disk_threshold *disk*	Specify the average latency in milliseconds of a synchronous single-block read.
-rack *rack_id*	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
```

C.3 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
`-credential credential_name`	Specify the name of the credential to add.
`-osaccount account_name`	Specify the operating system account which will be used for remote jobs.
`-ospassword password`	Specify the corresponding password for the account.
`-windows_domain domain_name`	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
```

C.4 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]
             [-validate_network]
```

Options

Table C-5 GDSCTL add database Options

Option	Description
`-connect connect_identifier`	Specify an Oracle Net connect descriptor or net service name that resolves to a connect descriptor for the database being added.
`-cpu_threshold cpu`	Specifies CPU Utilization percentage threshold.
`-disk_threshold disk`	Specifies the average latency in milliseconds of a synchronous single-block read.
`-gdspool gdspool_name`	The Global Data Services pool to which the database belongs.
`-pwd password`	The password for the GSMUSER. If `-pwd` is not specified, then you are prompted for the password.
`-region region_name`	The Global Data Services region to which the database belongs.
`-savename`	Specify this option to store a net service name specified with the `-connect` option in the Global Data Services catalog, rather than the connect descriptor mapped to that net service name.
`-validate_network`	This flag enables several network validation checks, including checking network connectivity between hosts and checking VNCR entries are valid.

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.

C.5 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 *file_name*	Specify the name of the file object to add.
-source *local_filename*	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
```

C.6 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
`-gdspool gdspool_name_list`	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 `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 (_)).
`-users user_list`	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
```

C.7 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 gmsendpoint]
       [-remote_endpoint remote_endpoint]
       [-trace_level level]
       [-encryption encryption]
       [-validate_network]
```

Options

Table C-9 GDSCTL add gsm Options

Option	Description
`-catalog connect_id`	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 gsmendpoint`	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 gsm_name`	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 listener_port`	Specify the listener port. The default port is 1522.
`-localons ons_port`	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 password`	Specify the password for the `GSMCATUSER`. If you do not specify a password, then you are prompted to enter one.

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

Option	Description
`-region region_name`	Specify the region to which the global service manager belongs. The value for *region_name* 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.
`-remote_endpoint remote_endpoint`	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.
`-remoteons ons_port`	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).
`-trace_level level`	Specify the global service manager trace level (to be used as directed by Oracle Support Services).
`-wpwd password`	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.
`-encryption encryption`	Encryption protocol for Advanced Network Option (ANO) used between GSM, GDSCTL and databases. OFF means that ANO is disabled. (AES256 | AES192 | OFF).
`-validate_network`	This flag enables several network validation checks, including checking network connectivity between hosts and checking VNCR entries are valid.

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
```

C.8 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
            [-validate_network]
```

Options

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

Option	Description
`-catalog catalog_dbname`	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.
`-group group_name`	Specify an alias which defines a group of invited nodes. This alias can be referenced in other commands related to invited nodes.
`-user user_name[/password]`	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.
`vncr_id`	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.
`-validate_network`	This flag enables several network validation checks, including checking network connectivity between hosts and checking VNCR entries are valid.

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.

```
GDSCTL> add invitednode 255.255.255.248
```

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

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

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

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

C.9 add region

Adds a region to a Global Data Services framework or an Oracle Globally Distributed Database configuration.

Syntax

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

Options

Table C-11 GDSCTL add region Options

Option	Description
`-buddy region_name`	Specify the name of the buddy region.
`-region region_list`	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
```

C.10 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 | TRUE_CACHE}]
           [-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]
           [-failover_restore {NONE|LEVEL1|AUTO}]
           [-reset_state {NONE|LEVEL1|LEVEL2|AUTO}]
```

Options

Table C-12 GDSCTL add service Options

Option	Description
`-available` *dbname_list*	Specify a comma-delimited list of available databases on which the service runs if the preferred databases are not available. You *cannot* specify a list of available instances, only databases. You can use the modify service command with the `-server_pool` parameter to specify instance-level preferences. The list of available databases must be mutually exclusive with the list of preferred databases. You *cannot* use this option with the `-preferred_all` option. For True Cache implentations, specify a comma-delimited list of available databases or True Caches on which the service runs if the preferred ones are not available. You can't specify a list of available instances, only databases and True Caches. You can use the `modify service` command with the `-server_pool` parameter to specify instance-level preferences. The list of available databases or True Caches must be mutually exclusive with the list of preferred ones. You cannot use this option with the `-preferred_all` option.
`-clbgoal {SHORT | LONG}`	Connection Load Balancing Goal. Use a value of `SHORT` for this parameter for run-time load balancing, or if using an integrated connection pool. Use a value of `LONG` for this parameter for long running connections, such as batch jobs, that you want balanced by the number of sessions for each node for the service. The default value for this option, if not specified, is `SHORT`. For True Cache services, set the connection load balancing goal to `SHORT` for runtime load balancing.
`-commit_outcome {TRUE | FALSE}`	Enable Transaction Guard; when set to `TRUE`, the commit outcome for a transaction is accessible after the transaction's session fails due to a recoverable outage.
`–drain_timeout`	Set drain time in seconds.
`-dtp {TRUE | FALSE}`	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.

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

Option	Description
`-edition` *`edition_name`*	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 `USE` privilege on the specified edition. If the edition does not exist or if the connect user does not have `USE` privilege on the specified edition, then an error is raised.
`-failover_primary`	If you set the `-role` option to `PHYSICAL_STANDBY`, then you can use this option to enable the service for failover to the primary database. If the `-role` is `TRUE_CACHE`, you can use this option to enable the service to failover to the primary database if no True Caches are available. Also include the primary database in the `-preferred` or `-available list`, or use the `-preferred_all` option.
`-failoverdelay` *`failover_delay`*	For Application Continuity and TAF, this parameter specifies the time delay (in seconds) between reconnect attempts for each incident at failover.
`-failovermethod {NONE | BASIC}`	TAF failover method (for backward compatibility only). If the failover type (`-failovertype`) is set to a value other than `NONE`, then you should choose `BASIC` for this parameter.
`-failoverretry` *`failover_retries`*	For Application Continuity and TAF, this parameter determines the number of attempts to connect after an incident.
`-failovertype {NONE | SESSION | SELECT | TRANSACTION}`	Specify the failover type. To enable Application Continuity for Java, set this parameter to `TRANSACTION`. To enable Transparent Application Failover (TAF) for OCI, set this parameter to `SELECT` or `SESSION`.
`-gdspool` *`gdspool_name`*	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 `gdspool` with access granted to the user, then this the `gdspool` with access granted is used as the default `gdspool`.

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

Option	Description
`-lag {lag_value | ANY}`	Specify the lag for the service in seconds. You can use the keyword `ANY` 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. If a True Cache falls behind the primary database beyond the specified lag time, the service stop forwarding requests to that True Cache until it catches up. The default value for `lag`, if not specified, is `ANY`.
`-locality {ANYWHERE | LOCAL_ONLY}`	Specify the service region locality. If you do not specify this option, then GDSCTL uses the default value of `ANYWHERE` for the service. For True Cache services, set the locality to `LOCAL_ONLY`. This indicates that GDS only routes to True Caches in the same region, regardless of load. Use this option together with -region_failover so that client connections and requests are routed to another region if all True Caches in a region have failed.
`-notification {TRUE | FALSE}`	Enable Fast Application Notification (FAN) for OCI connections.
`-pdbname pdb_name`	Specify the pluggable database name.
`-policy {AUTOMATIC | MANUAL}`	Specify the management policy for the service. If you specify `AUTOMATIC` (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 `MANUAL`, then the service is never automatically restarted upon planned restart of the database. A `MANUAL` setting does not prevent the global service manager from monitoring the service when it is running and restarting it if a failure occurs.
`-preferred dbname_list`	Specify a comma-delimited list of preferred databases or True Caches on which the service runs. You can't specify preferred instances, only databases and True Caches. 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 *cannot* use this option with the `-preferred_all` option.

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

Option	Description
-preferred_all	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 *cannot* use this option with the -preferred and -available options. For True Cache services, indicates that this service will be started on all existing and future True Caches and, if used with - failover_primary, the primary database.
-region_failover	Indicates that the service is enabled for region failover. You can only use this option when you specify LOCAL_ONLY for the -locality option.
-replay_init_time *replay_init_time*	For Application Continuity, this parameter specifies the time (in seconds) after which replay cannot be initiated. The default value is 300 seconds.
-retention *retention_seconds*	If commit_outcome is set to TRUE, then this parameter determines the amount of time (in seconds) that the commit outcome is retained in the database.
-rlbgoal {SERVICE_TIME | THROUGHPUT}	Run-time Load Balancing Goal (for the Load Balancing Advisory). Set this parameter to SERVICE_TIME to balance connections by response time. Set this parameter to THROUGHPUT to balance connections by throughput. For True Cache services, set the runtime load balancing goal to SERVICE_TIME to balance connections by response time instead of by throughput. If you do not use this option, then the value defaults to SERVICE_TIME for the run-time load balancing goal.
-role {[PRIMARY] | [PHYSICAL_STANDBY| TRUE_CACHE] [-failover_primary] | [LOGICAL_STANDBY] |[SNAPSHOT_STANDBY]}	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. For True Cache services, use TRUE_CACHE. This ensure that the service only runs on other True Caches, not on the primary database **See Also:** *Oracle Data Guard Concepts and Administration* for more information about database roles

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

Option	Description
`-service service_name`	Specify the name of the global service. 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. 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, underscore (_), and period (.). 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.
`-session_state {DYNAMIC | STATIC}`	For Application Continuity, this parameter specifies whether the session state that is not transactional is changed by the application. A setting of `DYNAMIC` is recommended for most applications.
`-sql_translation_profile stp_name`	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. This option corresponds to the SQL translation profile parameter in the `DBMS_SERVICE` service attribute. **Notes:** • Before using the SQL translation feature, you must migrate all server-side application objects and data to the Oracle database. • Use the command config service to display the SQL translation profile. **See Also:** *Oracle Database SQL Translation and Migration Guide* for more information about SQL translation
`–stop_option`	Set the default stop option to `NONE`, `IMMEDIATE`, or `TRANSACTIONAL`

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

Option	Description
-table_family*family*	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. 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, '"TESTUSER1.Customers6"'. No quotes are needed if neither name is case sensitive. If this parameter is not specified, but there is currently only one table family, the service created with the add service command is automatically associated with that table family.
-tafpolicy {BASIC | NONE }	TAF policy specification (for administrator-managed databases only).
-failover_restore	Session state restoration for Application Continuity (NONE | LEVEL1 | AUTO)
-reset_state	Reset session state (NONE -default | LEVEL1 | LEVEL2 | AUTO)

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).

Services cannot be added to systems that run Clusterware unless Clusterware is started on the system, even if the target database is not under CRS control. Ensure that Clusterware is started on Clusterware enabled systems before adding a new global service.

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.

> **Note:**
>
> In Database 23ai, the session attribute values FAILOVER_TYPE = TRANSACTION with SESSION_STATE_CONSISTENCY = STATIC are no longer a supported service attribute combination. Instead use one of the following combinations in the service configuration: FAILOVER_TYPE = AUTO with SESSION_STATE_CONSISTENCY = AUTO or FAILOVER_TYPE = TRANSACTION with SESSION_STATE_CONSISTENCY = DYNAMIC These configurations enforce session state tracking in the Oracle database ensuring that session state is preserved at session migration and session failover.

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
  -available 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

C.11 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]
          [-validate_network]
```

Options

Table C-13 GDSCTL add shard Options

Option	Description
`-connect connect_identifier`	Specify an Oracle Net connect descriptor or net service name that resolves to a connect descriptor for the database being added as the shard.
`-pwd password`	Enter the GSMUSER password. If not specified, the user is prompted for the password.
`-savename`	Store in the shard catalog a net service name specified with the `-connect` option rather than the connect descriptor mapped to that net service name.
`-region region_name`	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.
`-force`	If specified, the existing GDS and sharding configuration on the shard and in the shard catalog with information about this shard will be rewritten.
`-cdb cdb_name`	If this parameter is used, the shard must be a PDB and the CDB must already exist in the catalog.
`-cpu_threshold cpu`	Specify the CPU Utilization percentage threshold.
`-disk_threshold disk`	Specify the average latency in milliseconds of a synchronous single-block read.
`{-shardgroup shardgroup_name | -shardspace shardspace_name}`	Specify the name of the shardgroup or shardspace that this shard is being added to. Use `-shardspace` when using this command in a user-defined sharding configuration. Use `-shardgroup` with system-managed and composite sharding configurations.
`-deploy_as {PRIMARY | STANDBY | ACTIVE_STANDBY}`	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: • `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
`-rack rack_id`	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.

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

Option	Description
–replace *old_db_name*	This parameter specifies db_unique_name of the old shard when replacing it. The existing parameters of the ADD SHARD command (such as connect) must refer to attributes for the new (replacement) shard. This parameter is not supported in an Oracle GoldenGate environment.
-gg_service (http|https):*ogg_host:sm_port/deployment*	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: shard1.example.com:9005/shard1.
-validate_network	This flag enables several network validation checks, including checking network connectivity between hosts and checking VNCR entries are valid.

Usage Notes

Before running add shard, you must validate the shard by running the validateShard procedure as described in *Oracle Globally Distributed Database Guide*

- 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 the introduction of Oracle Database 23ai, The default behavior when adding a shard to the topology has changed so that a current version of the schema is captured and applied on the new shard. Previously, all accumulated DDL on the sharded database was applied to the new shard, one by one in order, some of which might be negated by later DDL
- 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. Note that the broker is not configured for a federated database.
- The -DEPLOY_AS option cannot be used in conjunction with -SHARDGROUP for system/composite sharding.

- When using the `-replace` parameter, see *Oracle Globally Distributed Database Guide* for more information about its usage.

Note: Oracle GoldenGate replication support for Oracle Sharding High Availability is deprecated in Oracle Database 21c.

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
```

C.12 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
`-shardgroup shardgroup_name`	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 (_).
`-region region_name`	Specify the name of the region. If not specified, a default region will be used.
`-shardspace shardspace_name`	Specify the name of the shardspace to which to add the shardgroup.

Table C-14 (Cont.) GDSCTL add shardgroup Options

Option	Description
`-deploy_as {PRIMARY | STANDBY | ACTIVE_STANDBY}`	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` • `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
`-repfactor` *number*	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.

Note: Oracle GoldenGate replication support for Oracle Sharding High Availability is deprecated in Oracle Database 21c.

Examples

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

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

C.13 add shardspace

Add a shardspace to the shard catalog.

Syntax

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

Options

Table C-15 GDSCTL add shardspace Options

Option	Description
`-shardspace` *shardspace_name*	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 (_).
`-chunks` *number*	Specify the number of unique chunks in the shardspace. The value of `-chunks` must be greater than 2 times the size of the largest shardgroup in any shardspace. This parameter does not apply to user-defined sharding or a federated database. 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 `DEPLOY` command and is 120 per database of the shardgroup with the biggest number of databases.
`-protectmode` *dg_protection_mode*	Specify the Data Guard protection mode: `MAXPROTECTION`, `MAXAVAILABILITY`,or `MAXPERFORMANCE` (default). This parameter does not apply to Oracle GoldenGate replication.
`-repfactor`	Replication factor (the number of replicas for each piece of data stored in a shardgroup). This parameter can only be used with NATIVE replication and system-managed or composite sharding, and is mandatory in these cases. It doesn't apply to user-defined sharding or a federated database since there are no shardgroups in this case.
`-repunits`	Total number of replication units (SNR only).

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
```

C.14 alter move

Suspend, resume, or cancel scheduled chunk move operations on a sharded database.

Syntax

```
ALTER MOVE {-RESUME|-SUSPEND|-CANCEL}
         [-CHUNK {chunk_id_list}]
         [-SHARD shd_lst]
```

```
[-verbose]
```

Options

Table C-16 GDSCTL ALTER MOVE Options

Option	Description
`-cancel`	Removes chunks from the pending chunk moves list for the specified scope.
`-chunk chunk_id_list`	List comma separated numeric chunk identifiers or use `ALL` for all chunks in the sharded database.
`-resume`	Resumes (reschedules) chunk moves which are in the "move failed" state for the specified scope.
`-shard shd_lst`	A comma separated list of shards.
`-suspend`	Suspends pending chunk moves for the specified scope.
`-verbose`	Enables verbose output mode.

Usage Notes

Suspending chunk moves: Use `ALTER MOVE -SUSPEND` to postpone a pending chunk move.

Resuming chunk moves: Use `ALTER MOVE -RESUME` to place specified chunk moves that are suspended or stalled (in the "move failed" state) into the "scheduled" state, effectively rescheduling them.

Canceling chunk moves: Use `ALTER MOVE -CANCEL` to cancel pending chunk moves. Once a chunk move operation is canceled it cannot be resumed or suspended.

Use the `-chunk` and `-shard` options to filter the scheduled chunk move operations. Use the `-chunk` option to target specific chunk IDs, and use the -shard option to target all chunk moves scheduled to occur to and from the specified database.

You can use the `CONFIG CHUNKS` command to get a list of pending chunk moves.

Chunk moves that are already in process cannot be suspended or canceled. If any chunk in scope for the ALTER MOVE operation is already in any state other than "scheduled" a warning is returned indicating that the move operation for that chunk was not altered.

Examples

Suspend all pending chunk moves:

```
ALTER MOVE -SUSPEND
```

Resume (reschedule) move operations for chunks 3 and 4:

```
ALTER MOVE -RESUME -CHUNK 3,4
```

Cancel all pending chunk moves to and from shard SH_1:

```
ALTER MOVE -CANCEL -SHARD SH_1
```

C.15 alter task

Suspend, resume, or cancel scheduled chunk or replication unit management operations on a sharded database.

Syntax

```
ALTER TASK
         {-RESUME|-SUSPEND|-CANCEL}
         [ -TASK task | [[-CHUNK chunk_id_list] |  [-RU ru_lst] | [-SHARD
shard]]]
         [-verbose]]
```

Options

Table C-17 GDSCTL alter task Options

Option	Description
`-cancel`	Removes chunks for the specified scope from the scheduled list. "`-chunk`" specifies that all listed chunks will be removed, "`-shard`" specifies that all chunk moves to/from this database will be removed. If any chunk in scope is already in a move a warning is returned indicating that the chunk was not removed.
`-chunk`	List of numeric chunk identifiers or `ALL` for all chunks.
`-resume`	Restarts stalled move process, optional parameter "`-shard`" provides a list of databases that will have their "move failed" flags reset before move restarts.
`-ru`	A comma separated list of replication units.
`-shard`	A comma separated list of shards.
`-suspend`	Suspends move for the specified scope. `-chunk` specifies chunks to suspend, "`-shard` specifies that all chunk moves to/from that database will be suspended. If any chunk in scope is already in a move (any state other than "scheduled") a warning is returned indicating that move for that chunk was not suspended.
`-task`	A comma separated list of tasks.
`-verbose`	Enable verbose output mode.

Usage Notes

`RESUME` option is used to restart stalled task. `SUSPEND` is used to postpone task, and `CANCEL` cancels task.

`CHUNK`, `RU` or `SHARD` is used to filter the list of scheduled migration. Use `CONFIG TASK` command to retrieve the list of scheduled tasks.

Examples

Suspend all pending tasks:

```
GDSCTL> alter task -suspend
```

Delete all tasks involving (ru 3 or 4) AND (shards shard1 or shard2):

```
GDSCTL> alter task -cancel -ru 3,4 -shard shard1,shard2
```

Suspend any task involving ru 5 in any way:

```
GDSCTL> alter task -suspend -ru 5
```

Resumes specified tasks:

```
GDSCTL> alter task -resume -task 34,35,36
```

C.16 config

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

Syntax

```
config [-support] [-verbose]
```

Options

Table C-18 GDSCTL config Options

Option	Description
`-support`	If specified, GDSCTL output displays additional information for support purposes.
`-verbose`	Enable verbose mode.

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
```

C.17 config backup

Configure Sharded Database (SDB) Backup

Syntax

```
config backup -rccatalog rc_connstr
              [-target (PRIMARY|STANDBY)]
              [(-destination (ALL|CATALOG|
<shard_list>):<deviceconfig>:<channelconfig>)+]
              [-frequency #level0_days|[#level0_days]:[#level1-days]]
              [-incremental (DIFFERENTIAL|CUMULATIVE)]
              [-retention #recovery_window_days]
              [-cdb sc_cdb_connstr] [-catpwd password]
              [-shard shard_list] [-async]
              [-encryption encryption][-REMOVE]
```

Options

Table C-19 GDSCTL config backup Options

Option	Description
`-rccatalog rc_connstr`	A connection string to the recovery catalog database.
`-target (PRIMARY|STANDBY)`	For shards in Data Guard configurations, database backup can be done at the primary or the standby. The default is the standby. The value for this option is either `PRIMARY` or `STANDBY`.
`-destination {ALL|CATALOG|shard_list:deviceconfig:channelconfig}`	Definition of device types and channels for target databases. It consists of three components: *shard-list*, *deviceconfig* and *channelconfig*. • *shard-list*: specifies a comma separated list of shard identifiers. They can be shard space, shard group or shard names. • *deviceconfig*: configures a device type for the prefixed target databases. It must be a valid device type configuration statement for RMAN. • *channelconfig*: configures a channel for the prefixed target databases. It must be a valid channel configuration statement for RMAN.
`-starttime ALL|CATALOG|shard_list:hh:mm`	Backup start time for individual shards and the sharded database (SDB) catalog database. It must be a local time in a day in the time zone where the target database is located and specified in 24-hour format. The smallest unit for the time is minute. For example, `CATALOG:22:30` specifies that the scheduled backup for the SDB catalog database should be started at 10:30 PM. The default backup start time for a target database is the midnight (00:00). Internally the start time is converted into a UTC time before it's passed to DBMS Scheduler to be set as the job start time

Table C-19 (Cont.) GDSCTL config backup Options

Option	Description
`-frequency` *#level0_days*|*#level0_days*`:`*#level1_days*	Backup repeat intervals for incremental level 0 and level 1 backups in days. The first number is the interval for incremental level 0 backups and the second is that for level 1 backups. If a single number is specified without a following comma, it defines the interval for level 0 backups. If a comma appears in the parameter, then the number to its left is the backup interval for level 0 backups and that to its right is the interval for level 1 backups, but both numbers are optional in this case. The default intervals for level 0 and level 1 backups are respectively 7 and 1 day.
`-incremental DIFFERENTIAL|CUMULATIVE`	Incremental level 1 backup type to either `DIFFERENTIAL` or `CUMULATIVE`. The default is `DIFFERENTIAL`.
`-retention` *#recovery_window_days*	A recovery window for backup files. It must be a positive integer and specified in days. The default is 30 days.
`-cdb` *sc_cdb_connstr*	Required if the SDB catalog database is a PDB. It specifies a connect string for the container database of the SDB catalog database.
`-catpwd` *password*	Password for user GSMCATUSER. Prompted if not specified. This password only needs to be specified once for this command in an entire GDSCTL session.
`-shard` *shard_list*	*shard_list* specifies a comma separated list of shard identifiers. They can be shard space, shard group or shard names. If the same name is used for a shard space, shard group or shard, shard space takes the highest precedence followed by shard group and then shard. There are two special words for this option: `ALL` and `CATALOG`. `ALL` represents the SDB catalog database and all the shards in the SDB while `CATALOG` represents only the SDB catalog database. The default is `all shards`.
`-async`	When specified, all tasks to configure the backup for the shards will run in background. By default, the task will run in foreground. The task for the SDB catalog database will always run in foreground regardless of this flag setting.
`-encryption`	Encryption protocol for Advanced Network Option (ANO) used between GSM, GDSCTL and databases. OFF means that ANO is disabled. (AES256 | AES192 | OFF).
`-remov`	If specified, it removes the backup configuration from the specified shards. Other provided options for the command are ignored.

Examples

The following example configures a backup channel of `DISK` type for the SDB catalog database, two parallel channels of `DISK` type for each of the shards (shard spaces dbs1 and dbs2 are used in the shard list), the backup retention window to 14 days, the level 0 and level 1 incremental backup repeat intervals to 7 and 1 day and the backup start time to 12:00 AM, leaving the incremental backup type to the default `DIFFERENTIAL` and the backup target type to the default `STANDBY`.

```
GDSCTL> config backup -rccatalog rcadmin/rman@inst6 -destination
"CATALOG::configure channel device type disk format '/tmp/rman/backups/
```

```
%d_%U'" -destination "dbs1,dbs2:configure device type disk parallelism
2:configure
channel 1 device type disk format '/tmp/rman/backups/1/%U';configure channel
2
device type disk format '/tmp/rman/backups/2/%U'" -starttime ALL:00:00 -
retention 14 -frequency 7:1 -catpwd gsm -cdb gsm_admin/gsm@instroot1;
```

When `CONFIG BACKUP` is not provided with any parameters, it shows the current backup configuration.

```
GDSCTL> config backup
Recovery catalog database user: rcadmin
Recovery catalog database connect descriptor:
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=den02qxr)(PORT=1521))
(CONNECT_DATA=(SERVICE_NAME=cdb6_pdb1.regress.rdbms.dev.example.com)))
Catalog database root container user: gsm_admin
Catalog database root container connect descriptor:
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=den02qxr)(PORT=1521))
(CONNECT_DATA=(SERVICE_NAME=v1908.regress.rdbms.dev.example.com)))
Backup retention policy in days: 14
Level 0 incremental backup repeat interval in minutes: 10080
Level 1 incremental backup repeat interval in minutes: 1440
Level 1 incremental backup type : DIFFERENTIAL
Backup target type: STANDBY
Backup destinations:
catalog::channel device type disk format '/tmp/rman/backups/%d_%u'
dbs1,dbs2:device type disk parallelism 2:channel 1 device type disk format
'/tmp/rman/backups/1/%u';channel 2 device type disk format '/tmp/rman/
backups/2/%u'
catalog::configure channel device type disk format '/tmp/rman/backups/%d_%u'
dbs1,dbs2:configure device type disk parallelism 2:configure channel 1 device
type disk format '/tmp/rman/backups/1/%u';configure channel 2 device type
disk format '/tmp/rman/backups/2/%u'
Backup start times:
all:00:00
```

C.18 config cdb

Displays properties of a specified CDB.

Syntax

```
config cdb [-cdb cdb_name] [-support][-verbose]
```

Options

Table C-20 GDSCTL config cdb Options

Option	Description
`-cdb cdb_name`	Specify the name of the cdb.
`-support`	If specified, GDSCTL output displays additional information.

Table C-20 (Cont.) GDSCTL config cdb Options

Option	Description
-verbose	Enable verbose mode.

Examples

Display information about CDB called cdb1.

```
GDSCTL> config cdb -cdb cdb1

Name: tstsdbyb
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:
```

C.19 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] )
            [-table_family table family][-verbose]
```

Options

Table C-21 GDSCTL config chunks Options

Option	Description
-chunk *chunk_id*	Specify a numeric chunk ID.
-cross_shard	Show cross-shard placement.
-key *key*	Sharding key
-shard *shd*	The name of the shard.
-shardgroup *sh*	The name of the shardgroup.
-show_reshard	Display information about ongoing chunk management operations.
-superkey *superkey*	Sharding super key. This is only needed for the composite sharding method.

Table C-21 (Cont.) GDSCTL config chunks Options

Option	Description
-support	If specified, GDSCTL output displays additional information.
-table_family	Table family name in the form of <schema name>.<root table name>.
-verbose	Enable verbose mode.

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.

If key or superkey type is date or timestamp then `gdsctl config chunks -key/superkey` should be passed in the format as shown below:

```
"YYYY-MM-DD HH24:MI:SS.FF6"
```

It must be in this order in terms of year/month/date. The example below shows key/super key with DATE type:

```
gdsctl config chunks -key '2000-01-01'
```

For key/super key with timestamp type, pass the value in single quotes:

```
gdsctl config chunks -key 1281 -superkey '"1992-04-09 05:00:00.0"'
```

Examples

The output from `config chunks` is shown below.

```
GDSCTL> config chunks

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

C.20 config credential

Displays remote credentials currently available for shard jobs.

Syntax

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

Options

Table C-22 GDSCTL config credential Options

Option	Description
`-support`	If specified, GDSCTL output displays additional information.
`-verbose`	Enable verbose mode

Usage Notes

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

Examples

Display credentials.

```
GDSCTL> config credential

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

C.21 config database

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

Syntax

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

Options

Table C-23 GDSCTL config database Options

Syntax	Description
`-database` *db_name*	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.
`-verbose`	Enable verbose mode

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
```

C.22 config file

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

Syntax

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

Options

Table C-24 GDSCTL config file Options

Option	Description
`-file file_name`	The name of the file object.
`-support`	If specified, GDSCTL output displays additional information.
`-verbose`	Enable verbose mode

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
```

C.23 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] [-verbose]
```

Options

Table C-25 GDSCTL config gdspool Options

Syntax	Description
`-gdspool gdspool_name`	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.
`-support`	GDSCTL output displays additional support information.
`-verbose`	Enable verbose mode

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
```

C.24 config gsm

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

Syntax

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

Options

Table C-26 GDSCTL config gsm Options

Syntax	Description
`-gsm gsm_name`	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.
`-support`	If specified, GDSCTL output displays additional information.
`-verbose`	Enable verbose mode

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
-----------------------
```

C.25 config region

Displays the static configuration data for the specified region.

Syntax

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

Options

Table C-27 GDSCTL config region Options

Syntax	Description
`-region gsm_name`	Specify the name of a global service manager.
`-support`	If specified, GDSCTL output displays additional information.
`-verbose`	Enable verbose mode

Example

Displays the static configuration data for the specified region.

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

Displays the following output:

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

C.26 config sdb

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

Syntax

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

Options

Table C-28 GDSCTL config sdb Options

Option	Description
-support	If specified, GDSCTL output displays additional information.
-verbose	Enable verbose mode

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
```

C.27 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]
               [-verbose]
```

Options

Table C-29 GDSCTL config service Options

Syntax	Description
`-gdspool` *`gdspool_name`*	Specify the name of the database pool that contains the services. If the name is not specified, and there is only one *`gdspool`* with access granted to the user, it is used as the default *`gdspool`*.
`-service` *`service_name`*	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.
`-support`	If specified, GDSCTL output displays additional information.
`-verbose`	Enable verbose mode

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.sdhdpool.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
```

C.28 config shard

Displays properties of a specified shard.

Syntax

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

Options

Table C-30 GDSCTL config shard Options

Option	Description
`-shard shard_name`	Specify the name of the shard.
`-support`	GDSCTL output displays additional support information.
`-verbose`	Enable verbose mode

Examples

```
GDSCTL> config shard

Name    Shard Group Status State    Region Availability
----    ----------- ------ -----    ------ ------------
den17b dbs1        Ok     Deployed east   ONLINE
den17c dbs2        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.

C.29 config shardgroup

Displays properties of a specified shardgroup.

Syntax

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

Options

Table C-31 GDSCTL config shardgroup Options

Option	Description
`-shardgroup shardgroup_name`	Specify the name of the shardgroup.
`-support`	GDSCTL output displays additional support information.
`-verbose`	Enable verbose mode

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
```

C.30 config shardspace

Displays properties of a specified shardspace.

Syntax

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

Options

Table C-32 GDSCTL config shardspace Options

Option	Description
`-shardspace shardspace_name`	Specify the name of the shardspace. Optional for system-managed sharding.
`-support`	GDSCTL output displays additional support information.
`-verbose`	Enable verbose mode

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
```

C.31 config table family

Displays information about all table families in the sharded database.

Syntax

```
config table family
         [-verbose]
```

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
```

C.32 config task

Display chunk or replication unit management tasks and their statuses.

Syntax

```
CONFIG TASK
         [-oid oid_number]
         [-shard shard_name]
```

Options

Table C-33 GDSCTL config task Options

Option	Description
-oid	Numeric object identifier (shard or replication unit).
-shard	The name of the shard.

Examples

Display all tasks:

```
GDSCTL> CONFIG TASK
         task ID   status            GDS command
         -------   ------            -----------
         43        started           switchover ru -ru 3 -database
cksrd3_ckshard3
```

```
         44        scheduled         switchover ru -ru 5 -database
cksrd2_ckshard2
         45        scheduled         switchover ru -ru 6 -database
cksrd3_ckshard3
```

Display task OID 43:

```
GDSCTL> CONFIG TASK -oid 43
         task ID   status            GDS command
         -------   ------            -----------
         43        started           switchover ru -ru 3 -database
cksrd3_ckshard3
```

Display tasks for shard cksrd2_ckshard2:

```
GDSCTL> CONFIG TASK -shard cksrd2_ckshard2
         task ID   status            GDS command
         -------   ------            -----------
         44        scheduled      switchover ru -ru 5 -database
cksrd2_ckshard2
```

C.33 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]
            [-verbose]
```

Options

Table C-34 GDSCTL config vncr Options

Syntax	Description
`-group group_name`	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.
`-verbose`	Enable verbose mode

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
```

C.34 configure

Sets the GDSCTL parameters.

Syntax

```
configure [-gsmport 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]
          [default_check_time]
          [echo]
          [-encryption encryption]
```

Options

Table C-35 GDSCTL configure Options

Syntax	Description
`-driver THIN | OCI`	Oracle JDBC driver.
`-gsm gsm_name`	Set current global service manager.
`-gsmdebug (1|0)`	Global service manager debug mode.
`-gsmport port`	Default global service manager port.
`-log {ALL | OFF | INFO | FINE | FINER | FINEST | SEVERE | WARNING}`	Set the logging level. The default is `OFF`.
`-log_file log_file`	Set the location of the log file. The default is `$TNS_ADMIN/GDSTL.log`.
`-resolve IP | HOSTNAME| QUAL_HOSTNAME`	Default host resolution for global service manager endpoint.

Table C-35 (Cont.) GDSCTL configure Options

Syntax	Description
`-save_config`	Store configuration changes to GSM.ORA.
`-show`	Show the configuration.
`-showtime ON|OFF`	Print time stamps.
`-spool`	Enable spooling. Warning: prints security-sensitive information to log file.
`-timeout` *seconds*	Global service manager requests timeout in seconds.
`-verbose ON|OFF`	Enable or disable verbose output. The default value is ON.
`-width`	Console width in number of characters (default 80).
`-default_check_time`	Automatic check timeout
`-echo`	Turn echo ON/OFF
`-encryption`	Encryption protocol for Advanced Network Option (ANO) used between GSM, GDSCTL and databases. OFF means that ANO is disabled. (AES256 | AES192 | OFF).

Example

Set the `mygsm` driver to OCI:

```
configure -driver OCI mygsm
```

C.35 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-36 GDSCTL connect Options

Syntax	Description
connect_identifier	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).
password	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.
user_name	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:
```

C.36 copy ru

To instantiate or repair a follower member of a replication unit you can copy another follower in that replication unit from another shard.

Syntax

```
COPY [RU|REPLICATION_UNIT] -RU ru_id
         [-SOURCE source_shard_name] -TARGET target_shard_name
         [-REPLACE repl_shard_name]
         [-timeout timeout]
         [-force]
```

Options

Table C-37 GDSCTL copy ru Options

Option	Description
`-force`	Allow the operation to bypass RAFT replication role checks.
`-replace`	Removes the replication unit follower specified by `-ru` from the database specified in `-replace`.
`-ru`	Replication unit ID

Table C-37 (Cont.) GDSCTL copy ru Options

Option	Description
`-source`	Name of the source shard.
`-target`	Name of the target shard .
`-timeout`	Timeout of connection retention between the time FAN event is sent to clients and chunk going read-only/down (seconds).
`-verbose`	Enable verbose mode.

Usage Notes

The source shard and target shard shouldn't be the replica leader for the given replication unit. If a member of the specified replication unit exists on the target shard, it is replaced by full replica of replication unit from the source shard.

If `-source` is not specified, then an existing follower of the replication unit will be chosen automatically as the source shard.

If the target database doesn't already contain a member of the specified replication unit, then the total number of replicas for the given replication unit should be below replication factor, unless `-replace` is specified.

If `-replace` is specified, the replication unit member is removed from the specified database.

Examples

```
GDSCTL> copy ru -ru 1 -source sh1 -target sh2
```

C.37 create gdscatalog

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

Syntax

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

Options

Table C-38 GDSCTL create gdscatalog 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 confname`	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 db_name`	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 gdspool_name_list`	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.
`-region region_name_list`	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.
`-user user_name[/password]`	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.
`-encryption`	Encryption protocol for Advanced Network Option (ANO) used between GSM, GDSCTL and databases. OFF means that ANO is disabled. (AES256 | AES192 | OFF).
`-validate_network`	This flag enables several network validation checks, including checking network connectivity between hosts and checking VNCR entries are valid.

Usage Notes

The `create gdscatalog` 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.

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

C.38 create restorepoint

Create Global Restore Points.

Syntax

```
create restorepoint [-name restore_point_name]
```

Options

Table C-39 GDSCTL create restorepoint Options

Syntax	Description
`-name` *restore_point_name*	The name of the global restore point to create in the sharded database. If it is not provided, a name is generated for the restore point.

Examples

The example below creates a global restore point in the sharded database with name GRP_MANUAL_1.

```
GGDSCTL> CREATE RESTOREPOINT -NAME GRP_MANUAL_1
```

C.39 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]                                          [-sharding
{system | composite | user}]                    [-chunks number]
                    [-protectmode dg_protection_mode]
                    [-agent_port port]
                    [-for_federated_database] [-encryption encryption]
                    [-repunits repunits]
                    [-encryption encryption]
                    [-validate_network]
```

Options

Table C-40 GDSCTL create shardcatalog options

Command Option	Description
-agent_password *password*	Specify the password to be used for remote scheduler agent registration with the catalog database.
-agent_port *port*	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.
-autovncr {ON|OFF}	This option enables (ON) or disables (OFF) auto VNCR mode. The default value is ON. See the Usage Notes below for important information about this option.

Table C-40 (Cont.) GDSCTL create shardcatalog options

Command Option	Description
`-chunks` *number*	Specify the default number of unique chunks in a shardspace. The value of `-chunks` must be greater than 2 times the size of the largest shardgroup in any shardspace. This parameter does not apply to user-defined sharding or a federated database. It will apply to all shardspaces created for composite sharding if the number of chunks is not specified in the `ADD SHARDSPACE` command. 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 `DEPLOY` command and is 120 per database of the shardgroup with the biggest number of shards
`-configname` *config_name*	Specify the name of the GDS configuration. This name is used as the virtual DB_DOMAIN of the sharded database. 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` *connect_identifier*	Specify the connect identifier for the database in which you want to create the catalog.
`-for_federated_database`	Create a catalog for a federated database.
`-force`	Before creating the new catalog, delete an existing shard or GDS catalog on this database.
`-protectmode` *dg_protection_mode*	Specify the Data Guard protection mode: `MAXPROTECTION`, `MAXAVAILABILITY`, or `MAXPERFORMANCE` (default). For Data Guard replication this parameter applies to any shardspace created without specification of protection mode in the `ADD SHARDSPACE` command.
`-region` *region_name_list*	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.
`-repl DG|NATIVE`	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 `NATIVE` for native RAFT replication. The default value is `DG`. This parameter cannot be modified after the catalog has been created. This parameter cannot be modified after the catalog has been created.

Table C-40 (Cont.) GDSCTL create shardcatalog options

Command Option	Description
`-sdb` *sdb_name*	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, and `COMPOSITE`. This parameter cannot by modified after the catalog has been created. Oracle GoldenGate does not support the user-defined sharding method.
`-repunits` *reunits*	Total number of replication units (this setting only applies to native RAFT replication). By default, Oracle Sharding determines the number of replication units (RUs) in a shardspace and the number of chunks in an RU. Note that in system-managed sharding there is one shardspace named SHARDSPACEORA.
`-shardspace` *shardspace_name_list*	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` *username[/password]*	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.
`-encryption`	Encryption protocol for Advanced Network Option (ANO) used between GSM, GDSCTL and databases. OFF means that ANO is disabled. (AES256 | AES192 | OFF).
`-validate_network`	This flag enables several network validation checks, including checking network connectivity between hosts and checking VNCR entries are valid.

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 `REPFACTOR` 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.
- The `FOR_FEDERATED_DATABASE` option is mutually exclusive with the `SHARDING` option.
- 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.

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

Note: Oracle GoldenGate replication support for Oracle Sharding High Availability is deprecated in Oracle Database 21c.

Examples

The following example creates a shard catalog on the mydb database.

```
GDSCTL> CREATE SHARDCATALOG -DATABASE mydb
```

C.40 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-41 GDSCTL databases Options

Option	Description
`-database db_name`	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.
`-gdspool gdspool_name`	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 *gdspool* with access granted to the user, it is used as the default *gdspool*.
`-gsm gsm_name`	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.

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
```

C.41 delete backup

Deletes sharded database (SDB) backups identified with specific tags from the recovery repository.

Syntax

```
delete backup [-tag tag_list] [-obsolete] [-catpwd password]
              [-shard shard_list] [-async]
```

Options

Table C-42 GDSCTL delete backup Options

Option	Description
`tag` *tag_list*	A comma separated list of tags. The backups identified by these tags will be deleted.
`-obsolete`	If specified, all obsoleted backups are deleted.
`-catpwd` *password*	Password for user GSMCATUSER. Prompted if not specified. It needs to be specified once for the entire GDSCTL session.
`-shard` *shard_list*	*shard_list* specifies a comma separated list of shard identifiers. They can be shard space, shard group or shard names. The default is `no shards`.
`-async`	When specified, all tasks to configure the backup for the shards will run in background. By default, the task will run in foreground. The task for the SDB catalog database will always run in foreground regardless of this flag setting.

Examples

The following example deletes backups with tag `odb_200414205057124_0400` from shard `v1908d_cdb2_pdb1`:

```
GDSCTL> delete backup -shard v1908d_cdb2_pdb1 -tag ODB_200414205057124_0400 -
catpwd gsm

This will delete identified backups, would you like to continue [No]?y

Deleting backups for database "v1908d_cdb2_pdb1" ...

allocated channel: ORA_DISK_1
channel ORA_DISK_1: SID=231 device type=DISK
allocated channel: ORA_DISK_2
channel ORA_DISK_2: SID=467 device type=DISK
deleted backup piece
backup piece handle=/tmp/rman/backups/2/2hutl9e9 RECID=13 STAMP=1037739466
Deleted 1 objects

deleted backup piece
backup piece handle=/tmp/rman/backups/1/2iutl9ed RECID=14 STAMP=1037739469
Deleted 1 objects

Recovery Manager complete.
```

C.42 delete catalog

Deletes the specified catalog.

Syntax

```
delete catalog [-connect [user/[password]@]conn_str]
               [-force]
```

Options

Table C-43 GDSCTL delete catalog Options

Syntax	Description
`-connect`	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.
`-force`	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
```

C.43 deploy

Deploys the sharded database.

Syntax

```
deploy [-no_rebalance] [-validate_network][-timeout timeout]
```

Options

Table C-44 GDSCTL deploy Options

Option	Description
-no_rebalance	Skip automatic chunk migration during incremental deploy.
-timeout	Timeout of connection retention between FAN is sent to clients and chunk going read-only/down (seconds).
-validate_network	This flag enables several network validation checks, including checking network connectivity between hosts and checking VNCR entries are valid.

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. 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
```

C.44 disable backup

Disable Sharded Database (SDB) Backup Jobs.

Syntax

```
disable backup [-catpwd password] [-shard shard_list]
```

Options

Table C-45 GDSCTL disable backup Options

Syntax	Description
`-catpwd password`	Password for user GSMCATUSER. Prompted if not specified. It needs to be specified once for the entire GDSCTL session.
`-shard shard_list`	*shard-list* is a comma separated list of shard identifiers. They can be shard space, shard group or shard names. The default is *all shards*

C.45 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-46 GDSCTL disable service Options

Syntax	Description
`-connect conn_str`	An Oracle Net connect descriptor or net service name that resolves to a connect descriptor for the database (or shard).
`-database db_name`	Specify the name of the database on which to the service is located. If you do not specify this option, then the service is disabled, globally.

Table C-46 (Cont.) GDSCTL disable service Options

Syntax	Description
`-gdspool` *gdspool_name*	Specify the database pool in which the services are located. If not specified and there is only one `gdspool` with access granted to the user, then this one is used as the default `gdspool`.
`-override`	Skip the GDS catalog (used when the GDS catalog is not available).
`-pwd`	The GSMUSER password.
`-service` *service_name_list*	Specify a comma-delimited list of global service names. If you do not use `-service` 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

C.46 enable backup

Enable Sharded Database (SDB) Backup Jobs.

Syntax

```
enable backup [-catpwd password] [-shard shard_list]
```

Options

Table C-47 GDSCTL enable backup Options

Syntax	Description
`-catpwd password`	Password for user GSMCATUSER. Prompted if not specified. It needs to be specified once for the entire GDSCTL session.
`-shard shard_list`	*shard_list* is a comma separated list of shard identifiers. They can be shard space, shard group or shard names. The default is *all shards*

Usage Notes

All backup jobs are initially disabled. They can be enabled by running the `enable database` command. They should be run after SDB backup has been configured with command `CONFIG BACKUP`. An error is reported if this command is run before SDB backup is configured.

C.47 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-48 GDSCTL enable service Options

Syntax	Description
`-connect conn_str`	An Oracle Net connect descriptor or net service name that resolves to a connect descriptor for the database (or shard).
`-database db_name`	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.
`-gdspool gdspool_name`	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.
`-override`	Skip the GDS catalog (used when the GDS catalog is not available).
`-pwd`	The GSMUSER password.
`-service service_name`	Specify a comma-delimited list of global service names. If you do not use `-service` 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

C.48 exit

Quit GDSCTL utility.

Syntax

```
quit | exit
```

C.49 export catalog

Saves the current catalog configuration to a local file.

Syntax

```
export catalog [-force] source
```

Options

Table C-49 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
```

C.50 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-50 GDSCTL help Options

Option	Description
gdsctl_command	Enter any GDSCTL command name to return a help page with syntax, options, usage notes and examples.

C.51 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-51 GDSCTL import catalog Options

Syntax	Description
-catpwd *gsmcatusrpwd*	GSMCATUSER password.
–database *catalog_db_name*	The connect identifier for the database in which to create catalog.

Table C-51 (Cont.) GDSCTL import catalog Options

Syntax	Description
source	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.
-user *gsmadminname[/password]*	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 gdscatalog 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
```

C.52 list backup

List Sharded Database (SDB) Backups

Syntax

```
list backup [-restorepoint restorepoint] [-controlfile]
              [-summary] [-catpwd password] [-shard shard_list]
```

Options

Table C-52 GDSCTL list backup Options

Option	Description
-restorepoint *restorepoint*	An SDB global restore point. If specified, only backups that are usable to restore the specified shards to the specific restore point are listed. Otherwise, all backups for the specified shards are listed.
-controlfile	If specified, only backups usable to restore database control files to a specific restore point are listed.
-summary	If specified, the backups are listed in summary format.

Table C-52 (Cont.) GDSCTL list backup Options

Option	Description
`-catpwd` *password*	Password for user GSMCATUSER. Prompted if not specified. This password only needs to be specified once for this command in an entire GDSCTL session.
`-shard` *shard_list*	*shard_list* specifies a comma separated list of shard identifiers. They can be shard space, shard group or shard names. The default is `no shards`.

Examples

The following example shows how to list the backups of control files from the SDB catalog database in summary.

```
GDSCTL> list backup -shard catalog -controlfile -summary -catpwd gsm

Listing backups for database "v1908" ...

List of Backups
===============
Key     TY LV S Device Type Completion Time #Pieces #Copies Compressed Tag
------- -- -- - ----------- --------------- ------- ------- ---------- ---
1366    B  F  A DISK        13-APR-20       1       1       NO
TAG20200413T234608
1851    B  F  A DISK        14-APR-20       1       1       NO
TAG20200414T000333
1996    B  F  A DISK        14-APR-20       1       1       NO
TAG20200414T001446
2057    B  F  A DISK        14-APR-20       1       1       NO
TAG20200414T001519
2151    B  F  A DISK        14-APR-20       1       1       NO
TAG20200414T012934
3205    B  F  A DISK        14-APR-20       1       1       NO
TAG20200414T202822

Recovery Manager complete.
```

The next example shows the use of the command to list the backups from `shard v1908b_cdb2_pdb1` recoverable to restore point `backup_before_db_maintenance`.

```
GGDSCTL> list backup -shard v1908b_cdb2_pdb1 -restorepoint
BACKUP_BEFORE_DB_MAINTENANCE
"GSMCATUSER" password:

Listing backups for database "v1908b_cdb2_pdb1" ...
List of Backup Sets
===================

BS Key  Type LV Size       Device Type Elapsed Time Completion Time
------- ---- -- ---------- ----------- ------------ ---------------
2998    Incr 0  265.53M    DISK        00:00:06     14-APR-20
        BP Key: 3009   Status: AVAILABLE  Compressed: NO  Tag:
BACKUP_BEFORE_DB_MAINTENANCE
```

```
        Piece Name: /tmp/rman/backups/2/0sutl6oa
  List of Datafiles in backup set 2998
  File LV Type Ckp SCN    Ckp Time  Abs Fuz SCN Sparse Name
  ---- -- ---- ---------- --------- ----------- ------ ----
  11   0  Incr 2678401    14-APR-20              NO    /ade/b/3998875997/
oracle/dbs/cdb2_pdb1_db.f

BS Key  Type LV Size       Device Type Elapsed Time Completion Time
------- ---- -- ---------- ----------- ------------ ---------------
2999    Incr 0  191.61M    DISK        00:00:04     14-APR-20
        BP Key: 3010   Status: AVAILABLE  Compressed: NO  Tag:
BACKUP_BEFORE_DB_MAINTENANCE
        Piece Name: /tmp/rman/backups/1/0tutl6oh
  List of Datafiles in backup set 2999
  File LV Type Ckp SCN    Ckp Time  Abs Fuz SCN Sparse Name
  ---- -- ---- ---------- --------- ----------- ------ ----
  12   0  Incr 2678425    14-APR-20              NO    /ade/b/3998875997/
oracle/dbs/cdb2_pdb1_ax.f
  13   0  Incr 2678425    14-APR-20              NO    /ade/b/3998875997/
oracle/dbs/cdb2_pdb1_xdb.f

Recovery Manager complete.
```

C.53 list restorepoint

List Global Restore Points.

Syntax

```
list restorepoint [-start_time t1] [-end_time t2] [-start_scn_ s1] [-end_scn
s2]
```

Options

Table C-53 GDSCTL list restorepoint Options

Syntax	Description
`-start_time` *time1*	If specified, the command lists restore points that were created at or after this time. It must be specified in the format “YYYY-MM-DD HH:MM:SS[.FFF] where .FFF is a fraction of a second in the precision of milliseconds.
`-end_time` *time2*	If specified, the command lists restore points that were created at or before this time. Refer to the option “start_time” above for its format.
`-start_scn` *scn1*	If specified, the command lists restore points with SCNs equal to or greater than this SCN.
`-end_scn` *scn2*	If specified, the command lists restore points with SCNs equal to or less than this SCN.

Examples

The example below lists the available restore points in the sharded database (SDB) with the SCN between 2600000 and 2700000.

```
GDSCTL> list restorepoint -start_scn 2600000 -end_scn 2700000
Name                            SCN             Create
Time
GRP_200726222838427_0400        2601938         2020-07-26
22:28:39.0
GRP_200726232837677_0400        2613192         2020-07-26
23:28:38.0
GRP_200727002838026_0400        2624200         2020-07-27
00:28:38.0
GRP_200727012838351_0400        2634360         2020-07-27
01:28:38.0
GRP_200727022837961_0400        2645399         2020-07-27
02:28:38.0
GRP_200727032838402_0400        2654898         2020-07-27
03:28:39.0
GRP_200727042837648_0400        2664398         2020-07-27
04:28:38.0
GRP_200727052837932_0400        2673905         2020-07-27
05:28:38.0
GRP_200727062838321_0400        2683840         2020-07-27 06:28:38.0
```

C.54 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-54 GDSCTL modify catalog Options

Syntax	Description
`-agent_password password`	Specify the agent registration password in the catalog for the remote Scheduler agent.
`-agent_port port`	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.
`-autovncr {ON | OFF}`	This option enables (`ON`) or disables (`OFF`) autovncr mode. The default value is `ON`.

Table C-54 (Cont.) GDSCTL modify catalog Options

Syntax	Description
`-newkeys`	Generates a new PKI key pair.
`-newpwd` *`newpassword`*	Used along with `-oldpwd`, sets the `GSMCATUSER` password after changing it on the catalog database.
`-oldpwd` *`oldpassword`*	Used along with `-newpwd`, sets the `GSMCATUSER` password after changing it on the catalog database.
`-pwd` *`password`*	Provides the `GSMCATUSER` password to generate the PKI keys when using `-newkeys`.
`-recover`	Perform catalog recovery to the last consistent state.
`-region` *`region`*	Region that the database, catalog, shard, shardgroup, or global service manager belongs to.

Usage Notes

To use this command, there must be a 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 -autovcnr 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
```

C.55 modify cdb

Modify cdb attributes.

Syntax

```
modify cdb -cdb cdbname_list
            [-connect connect_identifier]
            [-pwd gsmrootuser_pwd]
            [-scan scan_address [-ons port]]
            [-savename]
```

Options

Table C-55 GDSCTL modify cdb Options

Option	Description
-cdb *cdbname_list*	Specify a comma-delimited list of cdb names.
-connect *connect_identifier*	Specify an Oracle Net connect descriptor or a net service name that maps to a connect descriptor for the database that is being modified.
-ons *port*	Specify the ONS port.
-pwd *gsmrootuser_pwd*	Specify the password for GSMROOTUSER.
-savename	Specify this option to store a net service name specified with the -connect option in the Global Data Services catalog, rather than the connect descriptor mapped to that net service name.
-scan *scan_address*	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 -cdb1 cdb1 -pwd new_gsmrootuser_password
```

C.56 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-56 GDSCTL modify credential Options

Option	Description
`-credential credential_name`	Specify the name of the credential to modify.
`-osaccount account_name`	Specify the operating system account which will be used for remote jobs.
`-ospassword password`	Specify the corresponding password for the account.
`-windows_domain domain_name`	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
```

C.57 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-57 GDSCTL modify database Options

Option	Description
`-connect connect_identifier`	Specify an Oracle Net connect descriptor or a net service name that maps to a connect descriptor for the database that is being modified.
`-cpu_threshold cpu`	Specifies CPU Utilization percentage threshold.
`-database dbname_list`	Specify a comma-delimited list of database names.
`-disk_threshold disk`	Specifies the average latency in milliseconds of a synchronous single-block read.
`-gdspool gdspool_name`	Specify the database pool to which the databases belong.
`-ons port`	Specify the ONS port.
`-pwd password`	Specify the password for the GSMUSER.
`-region region_name`	Specify the region to which the databases belong.
`-savename`	Specify this option to store a net service name specified with the `-connect` option in the Global Data Services catalog, rather than the connect descriptor mapped to that net service name.
`-scan scan_address`	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
```

C.58 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-58 GDSCTL modify file Options

Option	Description
`-file file_name`	Specify the name of the file object to update.
`-source local_filename`	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
```

C.59 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-59 GDSCTL modify gdspool Options

Option	Description
`-adduser user_name`	Specify the user to add to the list of GDS pool administrators. This option grants the pool administrator role to the specified user.
`-gdspool database_pool_list`	Specify a comma-delimited list of GDS pool names.
`-removeuser user_name`	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
```

C.60 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]
           [-encryption encryption]
           [-timeout timeout]
```

Options

Table C-60 GDSCTL modify gsm Options

Option	Description
`-catalog` *connect_id*	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.
`-endpoint` *gsmendpoint*	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.
`-gsm` *gsm_name*	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.
`-listener` *listener_port*	Specify the new listener port.
`-localons` *ons_port*	Specify the new local ONS port.
`-pwd` *password*	Specify the password for the `GSMCATUSER` account. If you do not specify a password, then you are prompted to enter one.
`-region` *region_name*	Specify the region to which the global service manager belongs.
`-remote_endpoint` *remote_endpoint*	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.
`-remoteons` *ons_port*	Specify the new remote ONS port.
`-wpwd`	Specify the password for the global service manager wallet.
`-encryption`	Encryption protocol for Advanced Network Option (ANO) used between GSM, GDSCTL and databases. OFF means that ANO is disabled. (AES256 | AES192 | OFF).
`-timeout`	Restart timeout

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
```

C.61 modify region

Modifies the configuration parameters for a region.

Syntax

```
modify region -region region_name_list
              [-buddy region_name]
              [-weights weight]
```

Options

Table C-61 GDSCTL modify region Options

Option	Description
`-buddy region_name`	Specify the name of the buddy region
`-region region_list`	Specify a comma-delimited list of region names
`-weights weight`	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
```

C.62 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-62 GDSCTL modify service Options

Option	Description
`-add_instances [-preferred comma-delimited-list] [-available comma-delimited-list]`	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 –preferred and –available options is optional, but at least one of these must be provided.
`-available db_name_list`	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 `-preferred_all`, then GDSCTL returns an error.
`-cardinality {UNIFORM | SINGLETON}`	Specify the cardinality option for a service running on a policy-managed Oracle RAC database. Services with cardinality set to `UNIFORM` are offered on all database instances. Services with cardinality set to `SINGLETON` are offered on only one database instance.
`-clbgoal {SHORT | LONG}`	For connection load balancing goal: set to `SHORT` if using run-time load balancing, set to `LONG` for long running connections such as batch jobs or older SQL*Forms style. The default value for this option is `SHORT`.
`-commit_outcome {TRUE | FALSE}`	Enable Transaction Guard; when set to `TRUE`, the commit outcome for a transaction is accessible after the transaction's session fails due to a recoverable outage.
`-database db_name`	Specify the name of the database on which you want to modify the service. When `-database` is specified, you must specify exactly one of: • `-server_pool` and/or `-cardinality`. Either is optional, but at least one must appear, both can be used at once. • `-add_instances`, `-modify_instances`, or `-drop_instances`. Exactly one of these three options must be used.
`-dtp {TRUE | FALSE}`	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.

Table C-62 (Cont.) GDSCTL modify service Options

Option	Description
`-drop_instances inst_list`	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.
`-edition edition_name`	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 `USE` privilege on the specified edition. If the edition does not exist or if the connect user does not have `USE` privilege on the specified edition, then an error is raised.
`-failover_primary`	If you set the `-role` option to `PHYSICAL_STANDBY`, then you can use this option to enable the service for failover to the primary database.
`-failoverdelay failover_delay`	For Application Continuity and TAF, the time delay (in seconds) between reconnect attempts for each incident at failover.
`-failovermethod {NONE | BASIC}`	TAF failover method (for backward compatibility only). If `failovertype` is set to either `SESSION` or `SELECT`, then choose `BASIC` for this option.
`-failoverretry failover_retries`	For Application Continuity and TAF, the number of attempts to connect after an incident.
`-failovertype {NONE | SESSION | SELECT | TRANSACTION}`	Specify the failover type. To enable Application Continuity for Java, set this parameter to `TRANSACTION`. To enable Transparent Application Failover (TAF) for OCI, set this parameter to `SELECT` or `SESSION`.
`-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 all new sessions cannot be established to the service.
`-gdspool gdspool_name`	Specify the name of the database pool to which the service belongs. If not specified and there is only one *gdspool* with access granted to user, it is used as the default *gdspool*.
`-lag {lag_value | ANY}`	Specify the lag for the service in seconds. You can use the keyword `ANY` to indicate that there is no upper threshold on the lag time. The default value for the `-lag` option is `ANY`.

Table C-62 (Cont.) GDSCTL modify service Options

Option	Description
`-locality {ANYWHERE | LOCAL_ONLY}`	The service region locality. If you do not use this option, then the default value of `ANYWHERE` is used for the service.
`-modifyconfig`	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.
`-modify_instances [-preferred` *`comma-delimited-list`*`] [-available` *`comma-delimited-list`*`]`	The provided *`comma-delimited-list`* of preferred and available instances is merged with the existing list currently stored in the catalog. If you specify an instance in the *`comma-delimited-list`* that is not already in the stored list, it is added to the stored list in its correct mode (preferred or available.) If you specify in *`comma-delimited-list`* 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. Any instances already in the stored list that are not in the provided list remain unchanged in the stored list. Note that an instance cannot be both preferred and available, it can be in one mode only. `-preferred` and `-available` are optional but at least one list must be provided.
`-new_db` *`database_name`*	Specify the name of the new database on which the service runs. If you attempt to move a service that was configured with `-preferred_all`, then GDSCTL returns an error.
`-notification {TRUE | FALSE}`	Enable Fast Application Notification (FAN) for OCI connections.
`-old_db` *`database_name`*	Specify the name of the old database on which the service runs. If you attempt to move a service that was configured with `-preferred_all`, then GDSCTL returns an error.

Table C-62 (Cont.) GDSCTL modify service Options

Option	Description
`-policy {AUTOMATIC | MANUAL}`	Specify the management policy for the service. If you specify `AUTOMATIC` (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 `MANUAL`, then the service is never automatically restarted upon planned restart of the database. A `MANUAL` setting does not prevent the global service manager from monitoring the service when it is running and restarting it if a failure occurs.
`-pdbname` *`pdb_name`*	Specify the pluggable database name.
`-preferred` *`db_name_list`*	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 `-preferred` 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 `-preferred_all`, then GDSCTL returns an error.
`-preferred_all`	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 `-preferred` and `-available` options.
`-region_failover`	Indicates that the service is enabled for region failover. You can only use this option when you specify `LOCAL_ONLY` for the `-locality` option.
`-replay_init_time` *`replay_init_time`*	For Application Continuity, this parameter specifies the time (in seconds) after which replay is not initiated. Default value is 300 seconds.
`-retention` *`retention_seconds`*	For Transaction Guard (`commit_outcome` set to `TRUE`), this parameter determines the amount of time (in seconds) that the commit outcome is retained in the database.
`-rlbgoal {SERVICE_TIME | THROUGHPUT}`	Run-time Load Balancing Goal. Set this parameter to `SERVICE_TIME` to balance connections by response time. Set this parameter to `THROUGHPUT` to balance connections by throughput. If you do not use this option, then the value defaults to `SERVICE_TIME` for the run-time load balancing goal.
`-role {[PRIMARY] | [PHYSICAL_STANDBY] [-failover_primary] | [LOGICAL_STANDBY] | [SNAPSHOT_STANDBY]}`	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:** *Oracle Data Guard Concepts and Administration* for more information about database roles

Table C-62 (Cont.) GDSCTL modify service Options

Option	Description
`-server_pool` *server_pool_name*	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).
`-service` *service_name*	Specify the name of the global service.
`-session_state {DYNAMIC | STATIC}`	For Application Continuity, this parameter specifies whether the session state that is not transactional is changed by the application. A value of `DYNAMIC` is recommended for most applications.
`-sql_translation_profile` *stp_name*	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. This option corresponds to the SQL translation profile parameter in the `DBMS_SERVICE` service attribute. **Notes:** • Before using the SQL translation feature, you must migrate all server-side application objects and data to the Oracle database. • Use the config service command to display the SQL translation profile. **See Also:** *Oracle Database SQL Translation and Migration Guide* for more information about SQL translation
`-table_family`*family*	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 `CONFIG TABLE FAMILY` output. 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, `'"TESTUSER1.Customers6"'`. No quotes are needed if neither name is case sensitive.
`-tafpolicy {BASIC | NONE }`	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

C.63 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-63 GDSCTL modify shard Options

Option	Description
`-connect connect_identifier`	Specify an Oracle Net connect descriptor or a net service name that maps to a connect descriptor for the database that is being modified.
`-cpu_threshold cpu`	Specifies CPU Utilization percentage threshold.
`-credential credential_name`	Specify the credential to use on the remote machine which specifies the user name and password under which database creation will occur.
`-destination destination_name`	Specify the name of the remote executable agent through which the database will be created.
`-disk_threshold disk`	Specifies the average latency in milliseconds of a synchronous single-block read.
`-ons port`	Specify the ONS port.
`-osaccount account_name`	Specify the operating system account which will be used for remote jobs.

Table C-63 (Cont.) GDSCTL modify shard Options

Option	Description
`-ospassword password`	Specify the corresponding password for the account specified in `-osaccount`.
`-pwd password`	Specify the password for the GSMUSER.
`-region region_name`	Specify the region to which the databases belong.
`-savename`	Specify this option to store a net service name specified with the `-connect` option in the Global Data Services catalog, rather than the connect descriptor mapped to that net service name.
`-scan scan_address`	Specify the SCAN address for a cluster.
`-shard shname_list`	Specify a comma-delimited list of shard names.
`-windows_domain domain_name`	Specify the corresponding domain name if a Windows account has been specified in `-osaccount`.

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
```

C.64 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-64 GDSCTL modify shardgroup Options

Option	Description
`-shardgroup shardgroup_name`	Specify the name of the shardgroup to be modified.
`-region region_name`	Specify the region the shardgroup resides in.
`-shardspace shardspace_name`	Specify the shardspace this shardgroup belongs to.

Table C-64 (Cont.) GDSCTL modify shardgroup Options

Option	Description
`-repfactor number`	Specify the number of replicas for each piece of data stored in this shardgroup.
`-deploy_as {PRIMARY | STANDBY | ACTIVE_STANDBY}`	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 –REPFACTOR 3
```

C.65 modify shardspace

Modify shardspace parameters.

Syntax

```
modify shardspace –shardspace shardspace_name
                  [-chunks number]
                  [-protectmode dg_protection_mode]
                  [-repunits repunits]
                  [-repfactor repfactor]
```

Options

Table C-65 GDSCTL modify shardspace Options

Option	Description
`-shardspace shardspace_name`	Specify the name of the shardspace to be modified.
`-chunks number`	Specify the number of chunks in the shardspace.
`-protectmode dg_protection_mode`	Specify the Data Guard Protection mode: `MAXPROTECTION`, `MAXAVAILABILITY`, or `MAXPERFORMANCE`. This option can only be executed where Data Guard replication technology is used.
`-repfactor`	Replication factor (the number of replicas for each piece of data stored in a shardgroup). This parameter can only be used with NATIVE replication and system-managed or composite sharding, and is mandatory in these cases. It doesn't apply to user-defined sharding or a federated database since there are no shardgroups in this case.
`-repunits`	Total number of replication units (SNR only).

Usage Notes

The number of chunks can only be modified if a shardspace does not contain deployed shards. This command is not applicable to a federated database.

Examples

Modify the GOLD shardspace to have 6000 chunks.

```
GDSCTL> modify shardspace -shardspace gold -chunks 6000
```

C.66 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]
          [-copy]
```

Options

Table C-66 GDSCTL move chunk Options

Option	Description
`-chunk {chunk_id_list | 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.
`-copy`	Copy the chunk instead of moving (OGG only).
`-source shard_name`	Specify the name of the source shard.
`-target shard_name`	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

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
```

C.67 move ru

Move a member replica of a replication unit from one shard to another.

Syntax

```
[ru|replication_unit] -ru ru_id -source source_db
         -target target_db
         [-force]
```

Options

Table C-67 GDSCTL move ru Options

Option	Description
-force	Allow operation to bypass RAFT replication role checks.
-ru	Replication unit ID
-source	Name of the source shard.
-target	Name of the target shard.
-timeout	Timeout of connection retention between FAN is sent to clients and chunk going read-only/down (seconds).

Examples

```
MOVE RU -RU 1 -SOURCE sh1 -TARGET sh2
```

C.68 quit

Quit GDSCTL utility.

Syntax

```
quit | exit
```

C.69 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-68 GDSCTL recover shard Options

Option	Description
`-full`	Full recovery mode.
`-gdspool` *pool*	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` *shard_name*	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
```

C.70 relocate chunk

This command moves a list of chunks in all the replicas of the specified source RU to all the replicas in the target ru.

Syntax

```
relocate chunk -chunk {chunk_id_list | all} -sourceru ru_id
         [-targetru ru_id]
         [-timeout timeout]
```

Options

Table C-69 GDSCTL relocate chunk Options

Option	Description
`-chunk`	List of numeric chunk identifiers or `ALL` for all chunks.

Table C-69 (Cont.) GDSCTL relocate chunk Options

Option	Description
`-sourceru`	Source replication unit ID
`-targetru`	Target replication unit ID
`-timeout`	Timeout of connection retention between FAN is sent to clients and chunk going read-only/down (seconds).
`-verbose`	Enable verbose mode

Usage Notes

The source and target replication unit must be colocated in the same shard. leaders on the same shard and followers on the same shards. if not use `SWITCHOVER` to move the leader and `MOVE RU` to move the followers to colocated shards.

The specified chunks must be in the same source replication unit. If target ru is not specified, an empty target replication unit will be created.

Examples

```
GDSCTL> relocate chunk -chunk 3,4 -sourceru  1 -targetru 2
```

C.71 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-70 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 gdspool_name`	Specify the name of the database pool where the service is located. If not specified and there is only one *gdspool* with access granted to user, it is used as the default *gdspool*.
`-new_db db_name`	Specify the name of the database to which you want to move the service.

Table C-70 (Cont.) GDSCTL relocate service Options

Option	Description
`-newpwd` *newpassword*	Specify the password for the GSMUSER in the database to which the service is being relocated (the target database).
`-old_db` *db_name*	Specify the name of the database where the service is currently located.
`-oldpwd` *oldpassword*	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` *service_name*	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
```

C.72 remove brokerconfig

Removes an Oracle Data Guard broker configuration from a GDS pool.

Syntax

```
remove brokerconfig [-gdspool gdspool_name]
```

Options

Table C-71 GDSCTL remove brokerconfig Options

Syntax	Description
`-gdspool` *gdspool_name*	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.

```
GDSCTL> remove brokerconfig -gdspool myreaderfarm
```

C.73 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-72 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.

Table C-72 (Cont.) GDSCTL remove cdb Options

Option	Description
`-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.

```
GDSCTL> remove cdb -cdb cdb1
```

C.74 remove credential

Removes an existing credential.

Syntax

```
remove credential -credential credential_name
```

Options

Table C-73 GDSCTL remove credential Options

Option	Description
`-credential` *`credential_name`*	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
```

C.75 remove database

Removes databases from a GDS pool.

Syntax

```
remove database [-gdspool gdspool_name]
                {-all | -database db_name_list}
                [-force]
```

Options

Table C-74 GDSCTL remove database Options

Option	Description
`-all`	Removes all databases in the database pool.
`-database db_name_list`	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.
`-gdspool gdspool_name`	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.
`-force`	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 Globally Distributed Database, 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
```

C.76 remove file

Removes an existing file object from the catalog.

Syntax

```
remove file -file file_name
```

Options

Table C-75 GDSCTL remove file Options

Option	Description
`-file file_name`	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`.

```
GDSCTL> remove file -file east_region_db_params
```

C.77 remove gdspool

Removes a GDS pool from the current configuration.

Syntax

```
remove gdspool -gdspool gdspool_name_list
```

Options

Table C-76 GDSCTL remove gdspool Options

Option	Description
`-gdspool gdspool_name_list`	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.

```
GDSCTL> remove gdspool -gdspool tempreaders,myfarm
```

C.78 remove gsm

Removes a global service manager from the configuration.

Syntax

```
remove gsm [-gsm gsm_name]
```

Options

Table C-77 GDSCTL remove gsm Options

Syntax	Description
`-gsm gsm_name`	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
```

C.79 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-78 GDSCTL remove invitednode (remove invitedsubnet) Options

Option	Description
`-group group_name`	Specify an alias which defines a group of VNCRs. This alias can be referenced in other commands related to invited nodes.

Table C-78 (Cont.) GDSCTL remove invitednode (remove invitedsubnet) Options

Option	Description
vncr_id	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.

```
GDSCTL> remove invitednode 198.51.100.22
```

Remove the VNCR alias group EAST_SRV from the catalog.

```
GDSCTL> remove invitednode -group east_srv
```

C.80 remove region

Removes the specified regions from the global service management framework.

Syntax

```
remove region -region region_list
```

Options

Table C-79 GDSCTL remove region Options

Option	Description
-region *region_list*	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.

```
GDSCTL> remove region -region south
```

C.81 remove ru

Remove empty replication unit from a sharded database configuration.

Syntax

```
remove [ru|replication_unit] -ru ru_id
         [-timeout timeout]
```

Options

Table C-80 GDSCTL remove ru Options

Option	Description
`-ru`	Replication unit ID
`-timeout`	GSM (shard director) requests timeout (in seconds).

Usage Notes

Replication unit must be empty prior to its remove. Use `RELOCATE CHUNK` command to move chunks between replication units.

Examples

```
GDSCTL> remove ru -ru 1
```

C.82 remove service

Removes a service from a database pool.

Syntax

```
remove service [-gdspool gdspool_name]
                -service service_name
```

Options

Table C-81 GDSCTL remove service Options

Option	Description
`-gdspool gdspool_name`	Specify the name of the GDS pool from which you want to remove the service. If not specified and there is only one *gdspool* with access granted to user, then it is used as the default *gdspool*.
`-service service_name`	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

C.83 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-82 GDSCTL remove shard Options

Option	Description
`-shard {shard_name_list | ALL}`	Specify a comma-delimited list of shard names to remove, or specify `ALL` to remove all shards from the catalog.
`-shardspace shardspace_list`	Specify a comma-delimited list of names of shardspaces from which to remove all shards.
`-shardgroupshardgroup_list`	Specify a comma-delimited list of names of shardgroups from which to remove all shards.

Table C-82 (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
```

C.84 remove shardgroup

Removes a shardgroup from the shard catalog.

Syntax

```
remove shardgroup -shardgroup shardgroup_name
```

Options

Table C-83 GDSCTL remove shardgroup Options

Option	Description
`-shardgroup`*`shardgroup_name`*	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.

```
GDSCTL> remove shardgroup -shardgroup group1
```

C.85 remove shardspace

Removes a shardspace from the shard catalog.

Syntax

```
remove shardspace -shardspace shardspace_name
```

Options

Table C-84 GDSCTL remove shardspace Options

Option	Description
`-shardspace shardspace_name`	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.

```
GDSCTL> remove shardspace -shardspace gold
```

C.86 restore backup

The restore backup command is used to restore a sharded database to a specific global restore point.

Syntax

```
restore backup [-restorepoint restore_point_name | -scn scn] [-cdb
conn_str]
[-catalog_name pdbname] [-catalog_dbid dbid] [-restore_only | -
recover_only]
[-catpwd password] [-shard shard_list] [-async]
```

Options

Table C-85 GDSCTL restore backup Options

Option	Description
`-restorepoint restore_point_name`	An sharded database global restore point that the specified list of shards will be restored to.
`-scn`	The SCN associated with a global restore point. This option cannot be used with option "-restorepoint". However, to restore the SDB catalog to a specific restore point, the associated SCN must be used. Command "LIST RESTOREPOINT" can be used to list the available global restore points and their associated SCNs.

Table C-85 (Cont.) GDSCTL restore backup Options

Option	Description
-cdb	A connect string to the CDB root of the catalog database. The provided user must have SYSDG privilege in the CDB root and SYSBACKUP privilege for all containers. This option should only be used to restore the SDB catalog.
-catalog_name	The PDB name of the SDB catalog. This option should only be used to restore the SDB catalog.
-catalog_dbid	The DBID of the SDB catalog container database. Both the catalog name and the DBID can be obtained from fixed view v$containers. This option should only be used to restore the SDB catalog.
-catpwd *password*	Password for user GSMCATUSER. Prompted if not specified. This password only needs to be specified once for this command in an entire GDSCTL session.
-recover_only	A flag. If specified, the command only recovers databases. This flag cannot be used with flag "-restore_only".
-restore_only	A flag. If specified, the command restores the databases only without doing database recovery. This flag cannot be used with flag "-recover_only".
-shard *shard_list*	*shard_list* specifies a comma separated list of shard identifiers. They can be shard space, shard group or shard names. The default is `no shards`.
-async	When specified, all tasks to configure the backup for the shards will run in background. By default, the task will run in foreground. The task for the SDB catalog database will always run in foreground regardless of this flag setting.

Examples

The following example restores the control files of shard `v1908c_cdb2_pdb1` to restore point `backup_before_db_maintenance`. The database must be in `NOMOUNT` state. This command alters the database to `MOUNT` state after it has restored the control file..

```
GDSCTL> restore backup -shard v1908c_cdb2_pdb1 -restorepoint
BACKUP_BEFORE_DB_MAINTENANCE -controlfile -catpwd gsm
executing command: SET until clause

Starting restore at 14-APR-20
allocated channel: ORA_DISK_1
channel ORA_DISK_1: SID=441 device type=DISK
allocated channel: ORA_DISK_2
channel ORA_DISK_2: SID=202 device type=DISK

channel ORA_DISK_1: starting datafile backup set restore
channel ORA_DISK_1: restoring control file
channel ORA_DISK_1: reading from backup piece /ade/b/3998875997/oracle/dbs3/
V1908C/autobackup/2020_04_14/o1_mf_s_1037736781_h9dndfrd_.bkp
channel ORA_DISK_1: piece handle=/ade/b/3998875997/oracle/dbs3/V1908C/
autobackup/2020_04_14/o1_mf_s_1037736781_h9dndfrd_.bkp tag=TAG20200414T201301
channel ORA_DISK_1: restored backup piece 1
channel ORA_DISK_1: restore complete, elapsed time: 00:00:01
```

```
output file name=/ade/b/3998875997/oracle/dbs/ct_cf1.f
Finished restore at 14-APR-20

released channel: ORA_DISK_1
released channel: ORA_DISK_2
```

The next example restores the shard `v1908c_cdb2_pdb1` to a restore point `backup_before_db_maintenance`.

```
GDSCTL> restore backup -shard v1908c_cdb2_pdb1 -restorepoint
BACKUP_BEFORE_DB_MAINTENANCE -catpwd gsm
executing command: SET until clause

Starting restore at 14-APR-20
starting full resync of recovery catalog
full resync complete
allocated channel: ORA_DISK_1
channel ORA_DISK_1: SID=460 device type=DISK

channel ORA_DISK_1: starting datafile backup set restore
channel ORA_DISK_1: specifying datafile(s) to restore from backup set
...
channel ORA_DISK_1: restored backup piece 1
channel ORA_DISK_1: restore complete, elapsed time: 00:00:03
Finished restore at 14-APR-20

Starting recover at 14-APR-20
current log archived
using channel ORA_DISK_1

Creating automatic instance, with SID='yhox'
...
executing Memory Script
...

Oracle instance shut down

Removing automatic instance
Automatic instance removed
auxiliary instance file /ade/b/3998875997/oracle/dbs/V1908/datafile/
o1_mf_sysext_h9dx66s0_.dbf deleted
auxiliary instance file /ade/b/3998875997/oracle/dbs/V1908/datafile/
o1_mf_sysaux_h9dx66rp_.dbf deleted
auxiliary instance file /ade/b/3998875997/oracle/dbs/V1908/controlfile/
o1_mf_h9dx5klq_.ctl deleted
Finished recover at 14-APR-20
```

C.87 resume services

Resumes global service activity and traffic routing to the database, previously blocked by `SUSPEND SERVICES` command.

Syntax

```
GDSCTL> RESUME SERVICES -DATABASE target_db
```

Options

Table C-86 GDSCTL resume services Options

Option	Description
target_db	Specify the name of the database

Example

Resumes global service activity and traffic routing to the database:

```
GDSCTL> RESUME SERVICES -DATABASE dba

GDSCTL>
```

C.88 rman

Allow users to submit RMAN commands to a list of shards for execution.

Syntax

```
rman -shard shard_list [-check_syntax] [-from_cdb userid[/password]]
[-catpwd password] [-async] (-cmd_file cmdfile | <quote>rman-stmts<quote>)
```

Options

Table C-87 GDSCTL rman Options

Option	Description
-shard	Specifies a comma-delimited list of shard identifiers. Each shard identifier can be a region, shardspace, shardgroup or shard name. If the same name is used for a region, shardspace, shardgroup or shard, region takes the highest precedence followed by shardspace, shardgroup and then shard. For example, assume a sharded database has a shardspace and a shard both named "foo". When name "foo" is provided in the specified shard list, it is considered a shardspace and expanded to a list of the shards in the shardspace "foo". Two special words can be used for shard list: ALL and CATALOG. "ALL" means the sharded database catalog database and all the shards in the database while "CATALOG" means only the catalog database. For commands where this parameter is optional, if this value is not specified the value defaults to "ALL".
-check_syntax	If specified, it only checks syntax for the input RMAN commands.
-from_cdb	This option provides a CDB common user and a password in the form of "user/password". When this option is used, the provided RMAN commands will be run from the shard root container. The provided user must have SYSBACKUP privilege.
-catpwd	The GSMCATUSER password
-cmd_file	RMAN command file path

Usage Notes

RMAN commands can be supplied either with a command file or directly in the command line. The syntax of the commands is the same as when they are entered at the RMAN prompt.

When the comamnds are supplied in the command line, they must be put in quotation marks. If the commands themselves contain single quotes, then double quotes should be used for the commands.

Some RMAN commands can only be executed from CDB root. If such a command is supplied, option -FROM_CDB must be used. Commands that should be executed from the shard PDBs and those that should be executed from shard CDB root cannot be supplied at the same time.

Some RMAN commands cannot be used inside command files, for example, `HOST`. These commands cannot be used here in the command line or through a command file.

Example

The command allows users to run various RMAN commands against the selected shards. The following example reports the objects needed to be backed up in the shards in shard group DBS1. Assuming that the shard PDB name is PDB1 for all the shards in shard group DBS1:

```
GDSCTL> rman -shard dbs1 -from_cdb 'report need backup of pluggable database pdb1;';
```

Assuming that datafile 1 for shard SHARD1 needs to be backed up. The example below shows how to back up the datafile for shard SHARD1:

```
GDSCTL> rman -shard shard1 -from_cdb 'backup datafile 1;';
```

C.89 run backup

Run Sharded Database (SDB) Backup Jobs.

Syntax

```
run backup [-tag tag] [-catpwd password] [-shard shard_list] [-async]
```

Options

Table C-88 GDSCTL run backup Options

Syntax	Description
`-tag` *tag*	A name for the backup. If not provided, a unique tag is automatically generated for the backup. The size limit for tag is 30.
`-catpwd` *password*	Password for user GSMCATUSER. Prompted if not specified. It needs to be specified once for the entire GDSCTL session.
`-shard` *shard_list*	*shard_list* is a comma separated list of shard identifiers. They can be shard space, shard group or shard names. The default is *no shards*
`-async`	When specified, all tasks to configure the backup for the shards will run in background. By default, the task will run in foreground.

Examples

In the following example, a backup for the shards in shard space `dbs1` is created before a shard maintenance.

```
GDSCTL> run backup -tag backup_before_db_maintenance -shard dbs1 -catpwd gsm

Running on-demand backup for database "v1908b_cdb2_pdb1" ...
executing global script: full_backup_cdb
...
Starting Control File and SPFILE Autobackup at 14-APR-20
piece handle=/ade/b/3998875997/oracle/dbs3/V1908C/autobackup/2020_04_14/
o1_mf_s_1037736781_h9dndfrd_.bkp comment=NONE
Finished Control File and SPFILE Autobackup at 14-APR-20

Recovery Manager complete.
```

C.90 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-89 GDSCTL services Options

Option	Description
`-gsm gsm_name`	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 gsm` command).
`-raw`	If specified, GDSCTL output is presented in a raw, non-parsed format.
`-service service_name`	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 `mthly_report` service:

```
GDSCTL>services -service mthly_report.sales.oradbcloud
```

Returns output similar to the following:

```
Service "mthly_report.sales.oradbcloud" has 1 instance(s). Affinity:
ANYWHERE
    Instance "sales%1", name: "debug", db: "debug", region: "eastcoast",
status: ready.
```

C.91 set dataguard_property

Dynamically updates the value of a specified property of a broker configuration or database.

Syntax

```
set dataguard_property {-shardspace name | -brokerconfig name | -shard name
|
 -shardgroup name} [-reset] [-scope { configuration |
database}]
 'property_name'=property_value
```

Options

Table C-90 GDSCTL set dataguard_property Options

Syntax	Description
`-shardspace` *name*	The name of the shardspace.
`-brokerconfig` *name*	Broker configuration identifier.
`-shard` *name*	The name of the shard.
`-shardgroup` *name*	The name of the shardgroup.
`-reset`	Resets property to default value.

Table C-90 (Cont.) GDSCTL set dataguard_property Options

Syntax	Description
`-scope`	Defines the scope for property: database or broker configuration.

Usage Notes

A database property of a member of a Data Guard Broker configuration can be updated by using the `-SHARD` option. The `-SHARDGROUP` option allows users to update a database property of all databases in a specified shardgroup.

A property of the entire Data Guard Broker configuration can be updated by using the `-BROKERCONFIG` or `-SHARDSPACE` option.

See Oracle Data Guard Broker guide for the complete list of the Broker configuration and database properties.

If none of -shardspace/-shardgroup/-shard/-brokerconfig is specified, then the command will act on all broker configs or databases in the catalog.

If the user specifies `-shardspace`, then the command will only act on the broker configs / shards in that shardspace. Likewise with `-shard` and `-shardgroup`.

Specifying `-scope configuration` will override the default scope of '`database`' for `-shard` and `-shardgroup`. Likewise, specifying `-scope database` will override the default scope of '`configuration`' for `-shardspace` and `-brokerconfig`

Example

```
GDSCTL> set dataguard_property -shard db32 'archivelagtarget'=1200


set dataguard_property -shardgroup west 'archivelagtarget'=1200


set dataguard_property -brokerconfig conf_1 'logxptmode'=async


set dataguard_property -shard us 'logxptmode'=async


set dataguard_property -shardspace silver 'logxptmode'=async
```

C.92 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-91 GDSCTL set gsm Options

Syntax	Description
`-gsm gsm_name`	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 `GSMORA`.

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
```

C.93 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-92 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 gsm_name`	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.
timeout_value	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
```

C.94 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-93 GDSCTL 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 gsm_name`	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 `LOG_STATUS` listener parameter to ON.

```
GDSCLTL> set log_status on -save_config
```

C.95 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-94 GDSCTL set outbound_connect_timeout Options

Option	Description
timeout_value	Specify in seconds the connection timeout value.
-config_only	Update GSM.ORA only, without trying to connect to a running global service manager instance.
-gsm *gsm_name*	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
```

C.96 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-95 GDSCTL set trace_level Options

Option	Description
`-gsm gsm_name`	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 `set gsm` command).
trace_level	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
```

C.97 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-96 GDSCTL set trc_level Options

Option	Description
trace_level	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
-config_only	Update GSM.ORA only, without trying to connect to a running global service manager instance.
-gsm *gsm_name*	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
```

C.98 show ddl

Shows DDL statements execution status.

Syntax

```
show ddl {[-ddl ddl_id] [-count cnt] | [-failed_only]}
         [-support]
         [-verbose]
```

Options

Table C-97 GDSCTL show ddl Options

Option	Description
`-count cnt`	The maximum number of entries to display.
`-ddl ddl_id`	DDL numeric identifier.
`-failed_only`	Use this option to display only errored out statements.
`-support`	If specified, GDSCTL output displays additional support information.
`-verbose`	Enable verbose mode.

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

```
GDSCTL> show ddl -count 20
```

> **Note:**
>
> The `show dll` 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.

C.99 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-98 GDSCTL split chunk Options

Option	Description
`-chunk chunk_id_list`	Specify a comma-separated list of numeric chunk identifiers.

Table C-98 (Cont.) GDSCTL split chunk Options

Option	Description
`-shardspace shard_space_list`	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.

```
GDSCTL> split chunk -chunk 3,4,5
```

C.100 sql

Executes a SQL statement or a PL/SQL stored procedure against a sharded database.

Syntax

```
sql "sql_statement"
```

Options

Table C-99 GDSCTL sql Options

Option	Description
sql_statement	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"
```

C.101 start gsm

Starts a specific global service manager.

Syntax

```
start gsm [-gsm gsm_name] [-validate_network]
```

Options

Table C-100 GDSCTL start gsm Options

Option	Description
`-gsm gsm_name`	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).
`-validate_network`	This flag enables several network validation checks, including checking network connectivity between hosts and checking VNCR entries are valid.

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
```

C.102 start observer

Starts specific services.

Syntax

```
start observer -database db_name
               [-timeout seconds]
```

Options

Table C-101 GDSCTL start observer Options

Option	Description
`-database` *db_name*	The name of the database.
`-timeout` *seconds*	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 *Oracle Globally Distributed Database Guide* 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
```

C.103 start ru

Starts a specified replication unit.

Syntax

```
start ru -ru ru_id [-DATABASE db]
```

Options

Table C-102 GDSCTL start ru Options

Option	Description
`-database`	The name of the database.
`-ru`	Replication unit ID

Usage Notes

If the database is not specified, the `start ru` command runs on all available replicas of the specified replication unit.

Examples

Add the shard to shardgroup GROUP1 of the DB11 database.

```
GDSCTL> start ru -ru 1 -database mydb1
```

C.104 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-103 GDSCTL start service Options

Option	Description
`-database db_name`	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.
`-connect connect_identifier`	Specify an Oracle Net connect descriptor or net service name that resolves to a connect descriptor.
`-gdspool gdspool_name`	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 `gdspool` with access granted to the user, it is used as the default `gdspool`.
`-override`	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.
`-pwd password`	Specify the password of the GSMUSER in the specified database.
`-service service_name`	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

C.105 status

Displays the running status and runtime information for the global service manager.

Syntax

```
status [-gsm gsm_name] [-raw|-verbose|-support]
```

Options

Table C-104 GDSCTL status Options

Option	Description
`-gsm gsm_name`	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
```

C.106 status backup

View the detailed state on the scheduled backup jobs in the specified shards.

Syntax

```
status backup [-start_time t1] [-end_time t2]
         [-catpwd password]
         [-shard shard_list]
         [-READ_LOG rman_log]
         [-OUTPUT_LOG output_log]
         [-DELETE_LOG]
         [-DELETE_LOG_ONLY]
```

Options

Table C-105 GDSCTL status backup Options

Option	Description
-start_time *time1*	The command lists run details of the automated backup jobs that started on or after this time. It must be specified in the format *YYYY-MM-DD HH:MM:SS[.FFF]* where *.FFF* is a fraction of a second in the precision of milliseconds.
-end_time*time2*	If specified, only backups usable to restore database control files to a specific restore point are listed..
-catpwd *password*	Password for user GSMCATUSER. Prompted if not specified. This password only needs to be specified once for this command in an entire GDSCTL session.
-shard *shard_list*	*shard_list* specifies a comma separated list of shard identifiers. They can be shard space, shard group or shard names. The default is `all shards`.
-delete_log	Delete the specified RMAN output file on the shard server.
-delete_log_only	Delete the specified RMAN output file on the shard server without reading it.
-output_log	The specified file must not exist. If the file is not specified with an absolute path, it will be created in the current working directory. The RMAN log will be saved into this file without being displayed in the console.
-read_log	Read the specified RMAN log from the specified shard and display it in the console or save it into the file specified with the option `-OUTPUT_LOG`. The RMAN log name usually comes from the output of this command when it is run without the option `-READ_LOG`.

Examples

The following example shows the job state and all job run details from the sharded database (SDB) catalog and the primary shard `rdbmsb_cdb2_pdb1`.

```
GDSCTL> status backup -catpwd -shard catalog,rdbmsb_cdb2_pdb1;
"GSMCATUSER" password:***

Retrieving scheduler backup job status for database "rdbms" ...
Jobs:
  Incremental Level 0 backup job is enabled
    Job schedule start time: 2020-07-27 00:00:00.000 -0400
Job repeat interval: freq=daily;interval=1
  Incremental Level 1 backup job is enabled
    Job schedule start time: 2020-07-27 00:00:00.000 -0400
    Job repeat interval: freq=minutely;interval=60
  Global restore point create job is enabled
    Job schedule start time: 2020-07-27 23:59:55.960 -0400
    Job repeat interval: freq=hourly

Run Details:
  Incremental Level 1 backup job status: SUCCEEDED
    Job run actual start time: 2020-07-26 14:00:00.177 -0400
    Job run slave process ID: 9023
  Incremental Level 1 backup job status: SUCCEEDED
    Job run actual start time: 2020-07-26 22:00:01.305 -0400
Job run slave process ID: 59526
…
Global restore point create job status: SUCCEEDED
    Job run actual start time: 2020-07-27 15:28:37.603 -0400
    Job run slave process ID: 44227
  …
  Global restore point create job status: SUCCEEDED
    Job run actual start time: 2020-07-27 17:28:38.251 -0400
    Job run slave process ID: 57611

Retrieving scheduler backup job status for database "rdbmsb_cdb2_pdb1" ...
Jobs:
  Incremental Level 0 backup job is enabled
    Job schedule start time: 2020-07-28 00:00:00.000 -0400
    Job repeat interval: freq=daily;interval=1
  Incremental Level 1 backup job is enabled
    Job schedule start time: 2020-07-28 00:00:00.000 -0400
    Job repeat interval: freq=minutely;interval=60

Run Details:
  Incremental Level 1 backup job status: SUCCEEDED
    Job run actual start time: 2020-07-26 14:00:00.485 -0400
    Job run slave process ID: 9056
  …
  Incremental Level 1 backup job status: SUCCEEDED
    Job run actual start time: 2020-07-27 14:33:42.702 -0400
    Job run slave process ID: 9056
  Incremental Level 0 backup job status: SUCCEEDED
```

```
    Job run actual start time: 2020-07-27 00:00:01.469 -0400
    Job run slave process ID: 75176
```

C.107 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-106 GDSCTL status database Options

Option	Description
`-database db_name`	Specify the name of the database on which to check status
`-gdspool gdspool_name`	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 gsm_name`	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 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
```

C.108 status gsm

Displays the status of a specific global service manager.

Syntax

```
status (gsm)? [-gsm gsm_name]
              [-raw | -verbose | -support]
```

Options

Table C-107 GDSCTL status gsm Options

Option	Description
`-gsm gsm_name`	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 support information.
`-verbose`	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 `gdsctl 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
```

C.109 status routing

Displays the runtime routing information status.

Syntax

```
status routing [-by_chunk | -by_instance] [-gsm
gsm_name]
[-raw|-verbose|-support]
```

Options

Table C-108 GDSCTL status routing Options

Option	Description
`-by_chunk`	Group routing table output by chunk.
`-by_instance`	Group routing table information by registered instance (default).
`-gsm`	GSM name
`-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.

Usage Notes

By default the registered chunks are grouped by instance

Example

```
GDSCTL> status routing

GDSCTL>
```

C.110 status ru

Displays runtime information about replication units for Oracle Globally Distributed Database native RAFT replication.

Syntax

```
(STATUS REPLICATION | STATUS RU| RU)
         [-show_offline]
         [-savename]
         [-sort]
         [-gsm gsmname]
         [-catpwd pwd]
         [-wpwd wpwd]
         [-show_chunks]
         [-ru ru]
         [-leaders]
         [-database shard_name]
         [-show_errors [-all]]
         [-show_stats]
```

Options

Table C-109 GDSCTL status ru Options

Option	Description
-all	If not specified, only errors since last recovery are shown.
-catpwd	GSMCATUSER password. Both -CATPWD and -WPWD should be specified if GDSCTL doesn't share a HOME with the GSM (shard director).
-database	Shard name
-gsm	GSM (shard director) name
-leaders	Only leader information is displayed.
-ru	Replication unit ID
-show_chunks	Show chunk distribution across all replication units.
-show_errors	Show replication unit errors.
-show_offline	Show the list of offline (down) shards that have that RU.
-show_stats	Show usage statistics
-sort	Sort the outputby replication unilt ID.

Table C-109 (Cont.) GDSCTL status ru Options

Option	Description
-wpwd	Wallet password

Examples

```
GDSCTL> status ru
Replication units
-------------------------
Database                        RU#  Role      Term  Log Index   Apply SCN
LWM SCN   On-disk SCN
--------                        ---  ----      ----  ---------   ---------
-------   -----------
cdbsh1_sh1                      1    Leader    2        315471
1         0        304531
cdbsh1_sh1                      2    Follower  1        456282      451835
452939        456611
cdbsh1_sh1                      3    Follower  2        262706      260479
261605        262709
cdbsh2_sh2                      2    Leader    1        456282
0         0        446475
cdbsh2_sh2                      1    Follower  2        315471      313342
14393        315479
cdbsh2_sh2                      3    Follower  2        262706      260479
261605        262709
cdbsh3_sh3                      3    Leader    2        262706
1         0        252741
cdbsh3_sh3                      1    Follower  2        315471      314395
314393        315477
cdbsh3_sh3                      2    Follower  1        456282      452941
452939        456583
```

```
GDSCTL> status ru -ru 2
Replication units
-------------------------
Database                        RU#  Role      Term Log Index Apply LogIdx LWM
LogIdx On-disk LogIdx Status
--------                        ---  ----      ---- --------- ------------
---------- -------------- ------
den1b_cdb2_pdb1                 2    Leader    1    1         0
0          2              Ok
den1d_cdb4_pdb1                 2    Follower  1    1         0
0          1              Ok
```

```
den1c_cdb3_pdb1               2    Follower  1    1         0
0          1              Ok


GDSCTL> status ru -ru 2 -show_chunks
Chunks
------------------------
RU#                           From      To
---                           ----      --
2                             8         8

Replication units
------------------------
Database                      RU#  Role      Term Log Index Apply LogIdx LWM
LogIdx On-disk LogIdx Status
--------                      ---  ----      ---- --------- ------------
---------- -------------- ------
den1b_cdb2_pdb1               2    Leader    1    1         0
0          2              Ok
den1d_cdb4_pdb1               2    Follower  1    1         0
0          1              Ok
den1c_cdb3_pdb1               2    Follower  1    1         0
0          1              Ok


GDSCTL> ru -sort -show_chunks
Chunks
------------------------
RU#                           From      To
---                           ----      --
1                             1         7
2                             8         8

Replication units
------------------------
Database                      RU#  Role      Term Log Index Status
--------                      ---  ----      ---- --------- ------
den1b_cdb2_pdb1               1    Leader    1    3658      Ok
den1c_cdb3_pdb1               1    Follower  1    3658      Ok
den1d_cdb4_pdb1               1    Follower  1    3658      Ok
den1b_cdb2_pdb1               2    Leader    1    1         Ok
den1c_cdb3_pdb1               2    Follower  1    1         Ok
den1d_cdb4_pdb1               2    Follower  1    1         Ok


GDSCTL> ru -sort -show_errors
Replication units
------------------------
Database                      RU#  Role      Term Log Index Status
--------                      ---  ----      ---- --------- ------
den1b_cdb2_pdb1               1    Leader    1    3658      Ok
den1c_cdb3_pdb1               1    Follower  1    3658      Ok
den1d_cdb4_pdb1               1    Follower  1    3658      Ok
den1b_cdb2_pdb1               2    Leader    1    1         Ok
den1c_cdb3_pdb1               2    Follower  1    1         Ok
den1d_cdb4_pdb1               2    Follower  1    1         Ok
```

```
ERRORS
------------------------


GDSCTL> ru -sort -database den1b_cdb2_pdb1
Replication units
------------------------
Database                       RU#  Role      Term Log Index Status
--------                       ---  ----      ---- --------- ------
den1b_cdb2_pdb1                1    Leader    1    3658      Ok
den1b_cdb2_pdb1                2    Leader    1    1         Ok
```

C.111 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-110 GDSCTL status service Options

Option	Description
`-gsm gsm_name`	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 gsm` command).
`-raw`	Used by oracle internal components.
`-service service_name`	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 `gdsctl 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.
```

C.112 stop gsm

Stops a specific global service manager.

Syntax

```
stop gsm [-gsm gsm_name]
```

Options

Table C-111 GDSCTL stop gsm Options

Option	Description
`-gsm gsm_name`	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
```

C.113 stop ru

Add a shard to the shard catalog.

Syntax

```
stop ru -ru ru_id [-database db]
```

Options

Table C-112 GDSCTL stop ru Options

Option	Description
`-database`	The name of the database.
`-pwd password`	Replication unit ID

Usage Notes

If the database is not specified, the `start ru` command runs on all available replicas of the specified replication unit.

Examples

```
GDSCTL> stop ru -ru 1 -database mydb1
```

C.114 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-113 GDSCTL stop service Options

Option	Description
`-connect connect_identifier`	Specify an Oracle Net connect descriptor or net service name that resolves to a connect descriptor for the database (or shard).
`-database db_name`	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.
`–drain_timeout`	Set drain time in seconds.

Table C-113 (Cont.) GDSCTL stop service Options

Option	Description
`-force`	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.
`-gdspool` *gdspool_name*	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.
`-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.
`-pwd` *password*	Specify the password of the GSMUSER in the specified database.
`-service` *service_name*	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.
`–stop_option`	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`.

```
GDSCTL> stop service -gdspool readerfarm -service sales_report
```

> **See Also:**
>
> Stopping a Global Service

C.115 suspend services

This command allows users to block database on all GSM listeners.

This command allows users to block database on all GSM listeners. The purpose of this command is to isolate database that runs global services, but can not be accessed by applications.

Syntax

```
GDSCTL> suspend services -database target_db
```

Options

Table C-114 GDSCTL suspend services Options

Option	Description
target_db	Specify the name of the database

Example

In this scenario GSM won't be able to do failover, hence we need to block this database from GSM to initiate failover and guarantee that traffic won't be redirected to this database.

```
GDSCTL> suspend services -database target_db db1

GDSCTL>
```

C.116 switchover ru

Switch leadership for the given replication unit to the specified database.

Syntax

```
switchover [ru|replication_unit] {-ru ru_id -database target_db | -rebalance}
[-timeout time]
```

Options

Table C-115 GDSCTL switchover ru Options

Option	Description
-database	The name of the database.
-rebalance	Perform rebalancing of replication units across shards.

Table C-115 (Cont.) GDSCTL switchover ru Options

Option	Description
`-ru`	Replication unit ID
`-timeout`	Timeout of connection retention between FAN is sent to clients and chunk going read-only/down (seconds).

Usage Notes

If the `REBALANCE` option is specified, an operation of rebalancing replication units and leadership responsibilities is distributed equally across the shards.

Examples

```
GDSCTL> switchover ru -ru 1 -database dba
```

C.117 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-116 GDSCTL sync brokerconfig Options

Option	Description
`-database` *db_name*	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 GDSCTL uses the primary database as the referential database. If a primary database does not exist in the Oracle Data Guard broker configuration, then GDSCTL uses a random database from the pool as the referential database.
`-gdspool` *gdspool_name*	Specify the database pool to which the Oracle Data Guard broker configuration belongs. If not specified and there is only one `gdspool` with access granted to user, that gdspool is used as the default `gdspool`. If the specified database pool does not contain an Oracle Data Guard broker configuration, then GDSCTL 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
```

C.118 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-117 GDSCTL sync database Options

Option	Description
`-database database_name`	Specify the name of a database in the database pool to use as a *referential database*, from which the configuration is queried.
`-gdspool gdspool_name`	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.
- Note that the GDS `sync database` command has the potential to cause services to restart.

Example

Synchronize a database in the default database pool with the database `mydb`.

```
GDSCTL> sync database -database mydb
```

C.119 sync ru

Synchronizes data of the specified replication unit on all shards, erases RAFT logs, and resets log index and term.

Syntax

```
sync[hronize] ru -ru ru_id [-database db]
```

Options

Table C-118 GDSCTL sync ru Options

Option	Description
`-connect connect_identifier`	Specify an Oracle Net connect descriptor or net service name that resolves to a connect descriptor for the database being added as the shard.
`-pwd password`	Enter the GSMUSER password. If not specified, the user is prompted for the password.

Usage Notes

If a database is not specified for the `SYNC RU` command, a replica to synchronize with will be chosen based on the following criteria:

1. Pick the replica that was the the last leader.
2. If not available, pick the replica with greatest apply index.

Examples

```
GDSCTL> sync ru -ru 1 -database mydb1
```

C.120 sync schema (synchronize schema)

Allows common shared schemas across the existing databases to be retrieved. The command compares the schemas on all of the databases and retrieves those that are common.

Syntax

```
sync[hronize] schema [-schema schemalist [-retrieve_only] [-restart [-force]]
| -apply [-skip_first] | -show [[-ddl ddlnum] [-count n] | [-failed_only]]]
```

Options

Table C-119 GDSCTL `sync schema` Options

Option	Description
`-apply`	Specifies that the previously retrieved DDLs should be run in the catalog.

Table C-119 (Cont.) GDSCTL `sync schema` Options

Option	Description
`-count n`	Specifies the maximum number of entries to show.
`-ddl ddlnum`	Specifies the DDL numeric identifier.
`-failed_only`	Shows only errored out statements.
`-force`	Forces sync without user confirmation.
`-restart`	Sync from the beginning, erasing schemas synced earlier.
`-retrieve_only`	Specifies that the DDLs of the common schemas should be retrieved only from the databases and stored in the catalog but not applied.
`-schema schemalist`	Specifies that only the listed schemas will be retrieved. Specify `all` to include all non-Oracle schemas.
`-show`	Shows DDL statements and their execution status.
`-skip_first`	Specifies that the first failed DDL statement is skipped.

Usage Notes

This command is used only when the catalog is created for a federated database, which can be created by using option `-for_federated_database` of the `create shardcatalog` command. This option is mutually exclusive with `-sharding parameter`. The rest of the steps are similar to sharded database environment setup with user-defined sharding [`create shardcatalog`, `add gsm`, `add shardspace`, `add shard`, `deploy`]. After deployment is complete, the `sync schema` command can be run to import specified schemas from shards to the catalog.
The `sync ddl` command combines two operations:
1. Importing and applying schemas on the catalog.
2. Viewing the DDLs generated by combining schemas from shards.
The first operation is the default behavior and it requires a mandatory `-schema` parameter, which is list of schemas to import from shards. Note that `all` can be supplied to the `-schema` parameter to retrieve all non-Oracle schemas common to all shards and which do not exist on the catalog.This operation can be split into two steps using `-retrieve_only` and `-apply` options. The option `-retrieve_only` will retrieve schemas from the shards and generate the required DDL statements to be applied, but it does not execute these statements. To execute them at a later point, the `-apply` option is used. If, for some reason, a DDL execution fails, subsequent statements will not be executed as there could be dependencies on the failed DDL. When `-apply` is run again after fixing the issue, it will start from the first failed DDL statement and continue execution.
The second operation, `-show` is for examining DDL statements and their execution status.

Example

```
GDSCTL> sync schema -schema myschema


GDSCTL> sync schema -schema foo,bar


GDSCTL> sync schema -schema foo,"Bar"


GDSCTL> sync schema -schema all
```

C.121 validate backup

The validate backup command provides sharded database (SDB) backup validation.

Syntax

```
validate backup [-restorepoint restore_point_name] [-controlfile] [-header]
                          [-catpwd password] [-shard shard_list] [-async]
```

Options

Table C-120 GDSCTL validate backup Options

Option	Description
`-restorepoint` *restore_point_name*	A restore point to verify the backups against.
`-controlfile`	If specified, only backups usable to restore database control files to a specific restore point are validated.
`-header`	If specified, it will only validate the backup file headers to determine whether the files on the media correspond to the metadata in the RMAN repository.
`-catpwd` *password*	Password for user GSMCATUSER. Prompted if not specified. This password only needs to be specified once for this command in an entire GDSCTL session.
`-shard` *shard_list*	*shard_list* specifies a comma separated list of shard identifiers. They can be shard space, shard group or shard names. The default is `all shards`.
`-async`	When specified, all tasks to configure the backup for the shards will run in background. By default, the task will run in foreground. The task for the SDB catalog database will always run in foreground regardless of this flag setting.

Examples

The sharded database catalog (SC) database must be open, but the shard databases can be either mounted or open. If the backup validation is for database control files, the shards can be

started nomount.The following example svalidates the backups of the control files from the SDB catalog database recoverable to restore point `backup_before_db_maintenance`.

```
GDSCTL> validate backup -shard catalog -controlfile -restorepoint
BACKUP_BEFORE_DB_MAINTENANCE

Validating backups for database "v1908" ...
executing command: SET until clause

Starting restore at 14-APR-20
allocated channel: ORA_DISK_1
channel ORA_DISK_1: SID=201 device type=DISK
channel ORA_DISK_1: starting validation of datafile backup set
channel ORA_DISK_1: reading from backup piece /ade/b/3998875997/oracle/dbs/
V1908/autobackup/2020_04_14/o1_mf_s_1037669374_h9blkyc8_.bkp
channel ORA_DISK_1: piece handle=/ade/b/3998875997/oracle/dbs/V1908/
autobackup/2020_04_14/o1_mf_s_1037669374_h9blkyc8_.bkp tag=TAG20200414T012934
channel ORA_DISK_1: restored backup piece 1
channel ORA_DISK_1: validation complete, elapsed time: 00:00:01
Finished restore at 14-APR-20
Recovery Manager complete.
```

The next example validates the headers of the backups from shard `v1908b_cdb2_pdb1` recoverable to restore point `backup_before_db_maintenance`.

```
GGDSCTL> validate backup -shard v1908b_cdb2_pdb1 -restorepoint
BACKUP_BEFORE_DB_MAINTENANCE -header

Validating backups for database "v1908b_cdb2_pdb1" ...
executing command: SET until clause

Starting restore at 14-APR-20
allocated channel: ORA_DISK_1
channel ORA_DISK_1: SID=468 device type=DISK
allocated channel: ORA_DISK_2
channel ORA_DISK_2: SID=236 device type=DISK

 List of Backup Sets
===================

BS Key  Type LV Size       Device Type Elapsed Time Completion Time
------- ---- -- ---------- ----------- ------------ ---------------
2998    Incr 0  265.53M    DISK        00:00:06     14-APR-20
        BP Key: 3009   Status: AVAILABLE  Compressed: NO  Tag:
BACKUP_BEFORE_DB_MAINTENANCE
        Piece Name: /tmp/rman/backups/2/0sutl6oa
  List of Datafiles in backup set 2998
  Container ID: 3, PDB Name: CDB2_PDB1
  File LV Type Ckp SCN    Ckp Time  Abs Fuz SCN Sparse Name
  ---- -- ---- ---------- --------- ----------- ------ ----
  11   0  Incr 2678401    14-APR-20              NO    /ade/b/3998875997/
oracle/dbs/cdb2_pdb1_db.f

BS Key  Type LV Size       Device Type Elapsed Time Completion Time
------- ---- -- ---------- ----------- ------------ ---------------
```

```
2999    Incr 0  191.61M    DISK        00:00:04     14-APR-20
        BP Key: 3010   Status: AVAILABLE  Compressed: NO  Tag:
BACKUP_BEFORE_DB_MAINTENANCE
        Piece Name: /tmp/rman/backups/1/0tutl6oh
  List of Datafiles in backup set 2999
  Container ID: 3, PDB Name: CDB2_PDB1
  File LV Type Ckp SCN    Ckp Time  Abs Fuz SCN Sparse Name
---- -- ---- ---------- --------- ----------- ------ ----
  12   0  Incr 2678425    14-APR-20              NO    /ade/b/3998875997/
oracle/dbs/cdb2_pdb1_ax.f
  13   0  Incr 2678425    14-APR-20              NO    /ade/b/3998875997/
oracle/dbs/cdb2_pdb1_xdb.f
validation succeeded for backup piece
Finished restore at 14-APR-20

Recovery Manager complete.
```

C.122 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-121 GDSCTL validate catalog Options

Option	Description
-catpwd *cpwd*	Provides the GSMCATUSER password, otherwise it is read from the local wallet file by default.
-config	Indicates that the validation should be performed on the Global Data Services catalog configuration only.
-database *db_name*	Indicates the name of the database for which the cross-check validation should be performed.
-dbpwd *dpwd*	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.
-gsm *gsm_name*	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
```

C.123 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]
                   [-validate_network]
                   [-show_errors]
                   [-validate_network]
```

Options

Table C-122 GDSCTL validate Options

Option	Description
`-catpwd cpwd`	GSMCATUSER password.
`-config`	If specified, performs validation of GDS catalog configuration only.
`-database db_name`	Performs cross-check validation of the specified database.
`-dbpwd sipwd`	GSMUSER password.
`-gsm gsm`	Global service manager name

Table C-122 (Cont.) GDSCTL validate Options

Option	Description
`-validate_network`	This flag enables several network validation checks, including checking network connectivity between hosts and checking VNCR entries are valid.
`-show_errors`	Show errors only.

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

I

M

R

S

T

V

W