ORACLE

Oracle Zero Downtime Migration – Logical Online Migration to ADB-S on Oracle Database@Google Cloud

Technical Brief

November, 2024, Version [1.0]

ORACLE

Purpose statement

This document provides an overview of features and enhancements included in ZDM 21.5. It is intended solely to help you assess the business benefits of upgrading to ZDM 21.5 and planning for the implementation and upgrade of the product features described.

Disclaimer

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

ORACLE

Table of contents

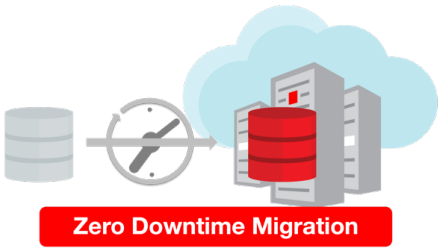


Figure 1. The Oracle Zero Downtime Migration Logo comprises a Database and a Clock with an arrow pointing to a Database deployed in the Cloud.

Purpose

Oracle customers are rapidly increasing their workload migration into the Oracle Cloud, Engineered Systems, and Oracle Database@Google Cloud. However, migrating workloads has been a source of challenges for many years. Migrating database workloads from one system to another or into the Cloud is easier said than done.

Based on years of experience migrating Oracle workloads, Oracle has developed Zero Downtime Migration (ZDM). ZDM is Oracle's premier solution for a simplified and automated migration experience, providing zero to negligible downtime for the production system depending on the migration scenario. ZDM allows you to migrate your on-premises Oracle Databases directly and seamlessly to and between Oracle Database@Azure, Oracle Database@ Google Cloud, Oracle Database@AWS, and any Oracle-owned infrastructure, including Exadata Database Machine On-Premises, Exadata Cloud at Customer, and Oracle Cloud Infrastructure. Oracle ZDM supports a wide range of Oracle Database versions and, as the name implies, ensures minimal to no production database impact during the migration.

ZDM follows Oracle Maximum Availability Architecture (MAA) principles[1] and incorporates products such as GoldenGate and Data Guard to ensure High Availability and an online migration workflow that leverages technologies such as the Recovery Manager, Data Pump, and Database Links.

This technical brief is a step-by-step guide for migrating your on-premises Oracle Databases to Oracle Autonomous Database Serverless (ADB-S)on Oracle Database@Google Cloud, with ZDM's Logical Online workflow.

Oracle ZDM will run on a separate node and connect to Source and Target to perform the migration. This guide will cover all requirements for installing the Oracle ZDM Service Host, the Source Database, the Target Database recipient of the migration process, and the networking used. The migration process will be dissected and done in a step-by-step fashion. This guide will answer the most frequently asked questions regarding the product and the overall migration process.

For more information on Oracle Zero Downtime Migration, please visit ZDM's product website and Oracle Database@Google Cloud product website.[2]

[1] https://oracle.com/goto/maa
[2] https://www.oracle.com/goto/zdm
https://www.oracle.com/cloud/google/oracle-database-at-google-cloud/

Zero Downtime Migration

Oracle Zero Downtime Migration (ZDM) is the Oracle Maximum Availability Architecture (MAA)-recommended solution to migrate Oracle Databases to the Oracle Cloud. ZDM's inherent design keeps in mind the migration process as straightforward as possible to ensure the most negligible impact on production workloads. The Source Database to be migrated can be on-premises, deployed on Oracle Cloud Infrastructure, or a 3rd Party Cloud. The Target Database deployment can be on Oracle Database@Azure, Oracle Database@Google Cloud, Oracle Database@AWS, Database Cloud Service on Oracle Cloud Infrastructure (OCI) Virtual Machine, Exadata Cloud Service, Exadata Cloud at Customer, and Autonomous Database. ZDM automates the entire migration process, reducing the chance of human errors. ZDM leverages Oracle Database-integrated high availability (HA) technologies such as Oracle Data Guard and GoldenGate and follows all MAA best practices that ensure no significant downtime of production environments. Oracle ZDM supports both Physical and Logical Migration workflows. This technical brief covers a step-by-step guide for the Logical Online Migration Workflow.

A standard Logical Online migration with Oracle GoldenGate and Data Pump Export and Import will take the following steps:

1. Download and Configure ZDM.
2. ZDM Starts Database Migration.
3. ZDM Configures an Oracle GoldenGate Extract Microservice.
4. ZDM Starts a Data Pump Export Job.
5. ZDM Starts a Data Pump Import Job.
6. ZDM Configures an Oracle GoldenGate Replicat Microservice.
7. ZDM Monitors Oracle GoldenGate Replication.
8. ZDM Switches Over.
9. ZDM Validates, Cleans Up, and Finalizes.

Supported Configurations

Oracle ZDM supports Oracle Database versions 11.2.0.4, 12.1.0.2, 12.2.0.1, 18c, 19c, 21c, and 23ai. ZDM's physical migration workflow requires the Source and Target Databases to be in the same database release.

Oracle ZDM supports Source Oracle Databases hosted on Linux, Solaris, and AIX operating systems. Oracle ZDM supports single-instance databases, Oracle RAC One Node databases, or Oracle RAC databases as sources. Oracle ZDM supports Oracle Database Enterprise & Standard Edition as Source and Target Databases. ZDM's physical migration workflow supports only Source Databases hosted on Linux platforms.

Architecture

An architectural overview of the ZDM server, the source database on-premises, the target database on Oracle Autonomous Database Serverless (ADB-S) on Oracle Database@Google Cloud, the Oracle GoldenGate on Docker, and all networks and components required are described in the diagram below:

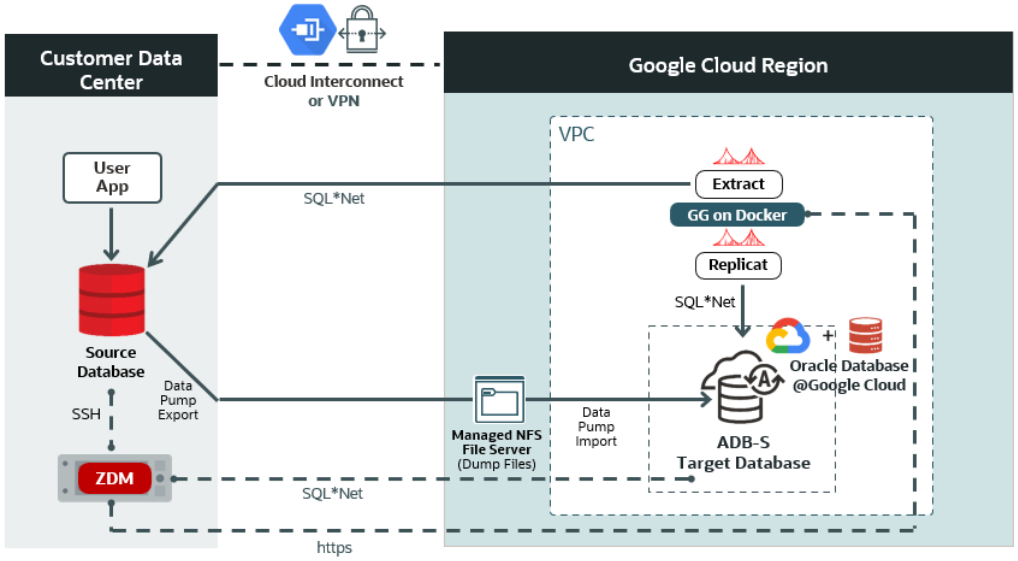


Figure 2. This is a High-Level Architectural overview showcasing the customer data center where the source database and ZDM's server reside. It also shows all connectivity to the target Oracle Autonomous Database Serverless (ADB-S) on Oracle Database@Google Cloud.

Zero Downtime Migration Service Host

Zero Downtime Migration Service Host Requirements

Oracle Zero Downtime Migration installation must take place on a separate host, which must fulfill the following requirements:

- Linux host running on Oracle 7, 8, or RHEL 8 (only these OS platforms/versions are supported).
- 100 GB of free storage space. This space is required for all the logs that ZDM will generate.
- A `zdm` group and a `zdmuser` as part of this group.
- The following packages must be installed:
  - `glibc-devel`
  - `expect`
  - `unzip`
  - `libaio`
  - `oraclelinux-developer-release-el7`
- All hostnames and IP addresses to be used must be present as entries in the `/etc/hosts` file.

For more information on the ZDM Service Host requirements and setting up ZDM on RHEL platforms, please refer to Oracle ZDM's product documentation, specifically "Setting Up Zero Downtime Migration Software" section[3].

For this step-by-step guide, the ZDM Service Host runs on-premises on an Oracle Linux Server 8.9. The host private IP is masked for this guide, but as an example, we will use the fictional ***zz.dd.mm.hh,*** and the hostname is ***zdmhost***.

On the ZDM host, as root, add the ADB Private Endpoint URL, for example, xyz.adb.region-1.oraclecloud.com, to the /etc/hosts file to be resolved to the ADB Private Endpoint IP, for example, ***aa.bb.cc.dd***. *If you do not have access to this information yet, please configure it once it is available* after provisioning the target Autonomous Database.

[3] https://docs.oracle.com/en/database/oracle/zero-downtime-migration/index.html

Network and Connectivity

Google Cloud Region
A Google Cloud region is a geographical area that contains data centers and infrastructure for hosting resources. It is made up of zones that are isolated from each other within the region.

Google Cloud Project
A Google Cloud Project is required to use Google Workspace APIs and build Google Workspace add-ons or apps. A Cloud project forms the basis for creating, enabling, and using all Google Cloud services, including managing APIs, enabling billing, adding and removing collaborators, and managing permissions.

Google Virtual Private Cloud
Google Cloud Virtual Private Cloud (VPC) provides networking functionality to Compute Engine virtual machine (VM) instances, Google Kubernetes Engine (GKE) containers, database services, and serverless workloads. VPC provides global, scalable, and flexible networking for your cloud-based service.

Google Cloud Interconnect
Cloud Interconnect extends your on-premises network to the Google network through a highly available, low-latency connection. You can use Dedicated Interconnect to connect directly to Google or Partner Interconnect to connect to Google through a supported service provider.

Autonomous Database
Oracle Autonomous Database is a fully managed, preconfigured database environments that you can use for transaction processing and data warehousing workloads. You do not need to configure or manage any hardware, or install any software. Oracle Cloud Infrastructure handles creating the database, as well as backing up, patching, upgrading, and tuning the database.

Source Database

The source database runs on-premises on an Oracle Linux Server 7.7 for this step-by-step guide. The host's private IP is masked for this guide, but as an example, we will use the fictional **aa.bb.sr.db** address, and the hostname is **onphost**. The source Oracle database is a single-instance Enterprise Edition database version 19.22 with multitenant architecture. The database name is **oradb**, and its unique name is **oradb_onp**.

The HR schema to be migrated resides in the source PDB **pdbsrc**.

Target Database

Oracle Database@Google Cloud offers the following products:

- **Oracle Exadata Database Service on Dedicated Infrastructure (ExaDB-D)**
  - You can provision flexible Exadata systems that allow you to add database compute servers and storage servers to your system anytime after provisioning.

- **Oracle Autonomous Database Serverless (ADB-S)**
  - Autonomous Database provides an easy-to-use, fully autonomous database that scales elastically, delivers fast query performance, and requires no database administration.

Oracle Database@Google Cloud integrates Oracle Exadata Database Service, Oracle Real Application Clusters (Oracle RAC), and Oracle Data Guard technologies into the Google Cloud platform. The Oracle Database service runs on Oracle Cloud Infrastructure (OCI) and is co-located in Google's data centers. The service offers features and price parity with OCI.

Oracle Database@Google Cloud service offers the same low latency as other Google-native services and meets mission-critical workloads and cloud-native development needs. Users manage the service on the Google Cloud console and with Google Cloud automation tools. The service is deployed in Google Virtual Private Cloud (VPC). The service requires that users have a Google Cloud Project and an OCI tenancy.

For this step-by-step guide, the target platform is Oracle Autonomous Database Serverless (ADB-S) on Oracle Database@Google Cloud. ZDM requires configuring a placeholder database target environment before beginning the migration process.

Enhanced Security for Outbound Connections with Private Endpoints

Setting the ROUTE_OUTBOUND_CONNECTIONS database property to the value PRIVATE_ENDPOINT enforces that all outgoing connections to a target host are subject to and limited by the private endpoint's egress rules.

```
ALTER DATABASE PROPERTY SET ROUTE_OUTBOUND_CONNECTIONS = 'PRIVATE_ENDPOINT';
```

NFS File Share via Google Cloud Managed NFS File Server

ZDM Logical Online migration workflow uses Oracle Data Pump export and import to migrate the data from the source to the target database. An NFS file share is provided through the Google Cloud-managed NFS File Server to store the Data Pump dump files.

The IP address of the NFS server is masked for this guide, but as an example, we will use the fictional aa.an.fs.pe address. The NFS path is **aa.an.fs.pe:/zdm_share**
The NFS share must be mounted on both the source database host and the target Autonomous Database.

To mount the NFS Share on the source database server:

As *root*:

```
mkdir -p /mnt/nfs3zdm
```

```
mount -o rw aa.an.fs.pe:/zdm_share /mnt/nfs3zdm
```

Make sure the *Oracle* user has access to the NFS mount

```
chown oracle:oinstall /mnt/nfs3zdm
```

As *oracle* user:

```
touch /mnt/nfs3zdm/test.txt
```

On the source PDB:

```
SQL> create directory DATA_PUMP_DIR_NFS as '/mnt/nfs3zdm';
```

Pre-Requisites

Source Database Pre-Requisites

- Oracle GoldenGate requires a unique row identifier on the source and target tables to locate the correct target rows for replicated updates and deletes.[4]
- The character set on the source database must be the same as the target database.
- If the source is Oracle Database 11.2, apply mandatory 11.2.0.4 RDBMS patches on the source database. See My Oracle Support note Oracle GoldenGate -- Oracle RDBMS Server Recommended Patches (Doc ID 1557031.1)[5]
- If the source database is Oracle Database 12.1.0.2 or a later release, apply mandatory RDBMS patches.
- If the source is Oracle Database Standard Edition 2, available with Oracle Database 18c or 19c, and lower than DBRU 19.11, apply the RDBMS patch for bug 29374604 -- Integrated Extract not starting against Oracle RDBMS Standard Edition.

Source Database Preparation

For online logical migrations, set STREAMS_POOL_SIZE to at least 2GB. See MOS Note 2078459.1 for the recommendation 1GB STREAMS_POOL_SIZE per integrated extract + additional 25 percent.
As SYS user:

```
-- Set streams_pool_size to 2G
SQL> alter system set streams_pool_size=2G scope=both;

-- Set global_names to false
SQL> alter system set global_names=false;

-- Enable ARCHIVELOG mode:
SQL> select log_mode from v$database;
LOG_MODE
------------
NOARCHIVELOG
SQL> shutdown immediate;
SQL> startup mount
SQL> alter database archivelog;
SQL> alter database open;
SQL> select log_mode from v$database;
LOG_MODE
------------
ARCHIVELOG
-- Enable FORCE LOGGING to ensure that all changes are found in the redo by the
Oracle GoldenGate Extract process:
SQL> select force_logging from v$database;
FORCE_LOGGING
---------------------------------------
NO
SQL> alter database force logging;
```

[4] https://docs.oracle.com/en/middleware/goldengate/core/21.3/gghdb/sql-server-preparing-system-oracle-goldengate.html#GUID-299C1451-50E7-49CC-87C6-BFA995C52249
[5] https://support.oracle.com/rs?type=doc&id=1557031.1

```
SQL> select force_logging from v$database;
FORCE_LOGGING
---------------------------------------
YES

-- Enable database minimal supplemental logging:
SQL> select minimal from dba_supplemental_logging;
MINIMAL
----------
NO
SQL> alter database add supplemental log data;
SQL> select minimal from dba_supplemental_logging;
MINIMAL
----------
YES


-- Enable initialization parameter ENABLE_GOLDENGATE_REPLICATION:
SQL> alter system set ENABLE_GOLDENGATE_REPLICATION=TRUE scope=both;
System altered.

-- In case of Multitenant, create the user c##ggadmin in CDB$ROOT:
SQL> create user c##ggadmin identified by VerySecretPw_22 default tablespace users
temporary tablespace temp;
grant connect, resource to c##ggadmin;
grant unlimited tablespace to c##ggadmin;
alter user c##ggadmin quota 10G on users;
grant select any dictionary to c##ggadmin;
grant create view to c##ggadmin;
grant execute on dbms_lock to c##ggadmin;
grant set container to c##ggadmin container=all;
exec dbms_goldengate_auth.GRANT_ADMIN_PRIVILEGE('c##ggadmin',container=>'all');

-- Create a GoldenGate administration user, ggadmin (in the PDB in case of
Multitenant):
SQL> alter session set container=pdbsrc;
create user ggadmin identified by VerySecretPw_22 default tablespace users temporary
tablespace temp;
grant connect, resource to ggadmin;
grant unlimited tablespace to ggadmin;
alter user ggadmin quota 10G on users;
grant select any dictionary to ggadmin;
grant create view to ggadmin;
grant execute on dbms_lock to ggadmin;
exec dbms_goldengate_auth.GRANT_ADMIN_PRIVILEGE('ggadmin');
```

ZDM Service Host

On the ZDM host, as root, add the Autonomous Database Private Endpoint URL **sample.adb.us-region-1.oraclecloud.com** to the /etc/hosts file to be resolved to the Autonomous Database Private Endpoint IP **aa.dd.bb.ss**.

Access Network File System from Autonomous Database

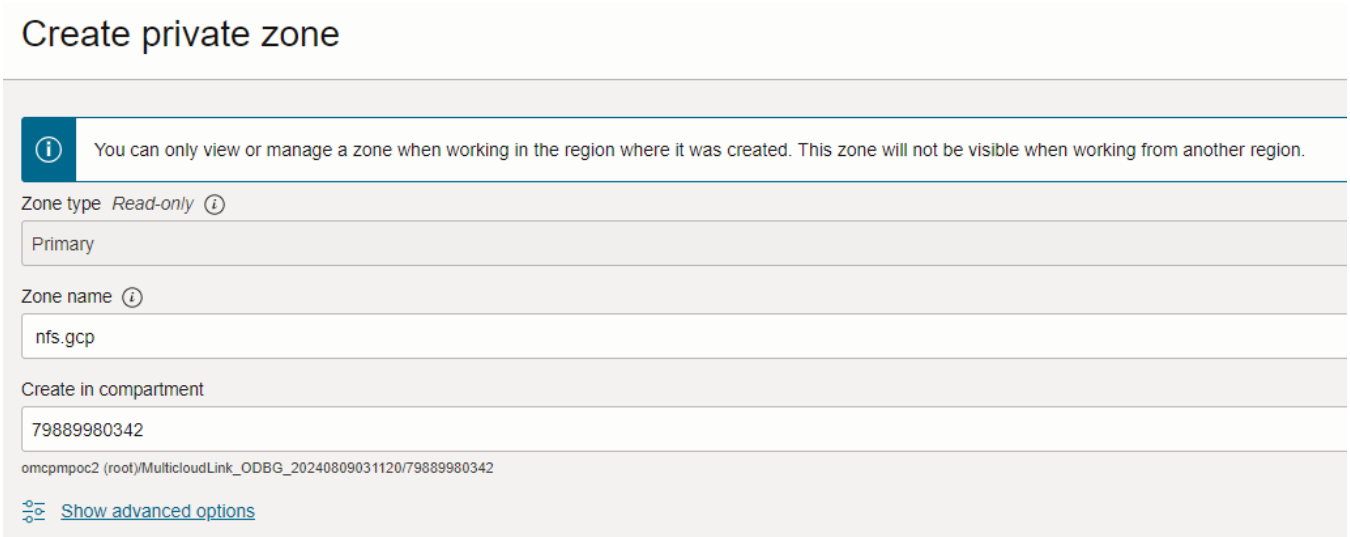
You can attach a Network File System to a directory location in your Autonomous Database[6]. This allows you to load data from Oracle Cloud Infrastructure File Storage in your Virtual Cloud Network (VCN), Google Cloud-managed NFS Server[7], or any other Network File System in on-premises data centers. Depending on the Network File System version you want to access, both **NFSv3** and **NFSv4** are supported.

Step 1: Add NFS Server Name to OCI DNS VCN Resolver

Bear in mind that if the OCI tenancy is a new tenancy created within the Oracle Database@Google Cloud provisioning process, the limits for OCI private DNS and A records might need to be increased. To increase the limits, open a Service Request with Oracle Support. A limit of at least three records is required.

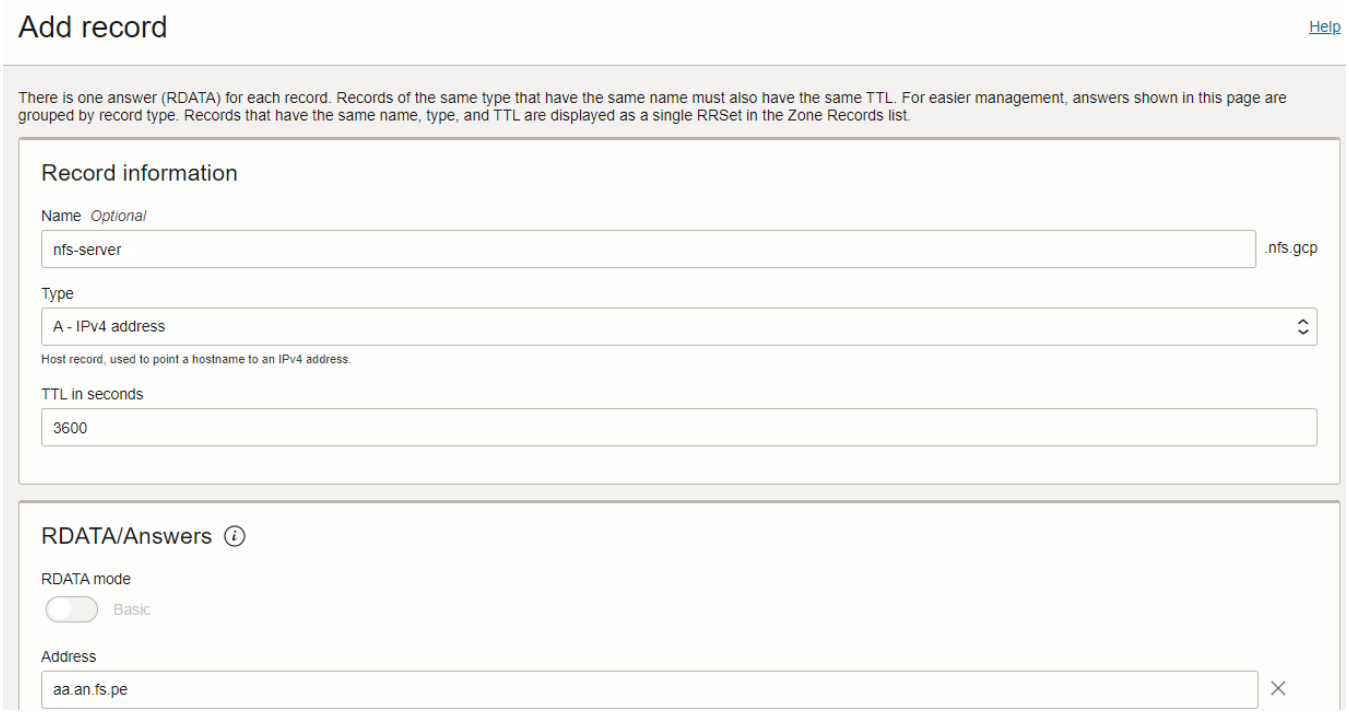
Follow these steps to create an A-record in OCI DNS to resolve the NFS server name:

1. From the Oracle Autonomous Database details page in Google Cloud, click on the MANAGE IN OCI link.
2. From the Oracle Autonomous Database details page in OCI, click on the virtual cloud network in the Network section.
3. On the network details page, click on the DNS Resolver.
4. On the private resolver details page, click on the default private view.
5. Click the create zone button and create a new zone using the name of your choice, e.g., ***nfs.gcp***

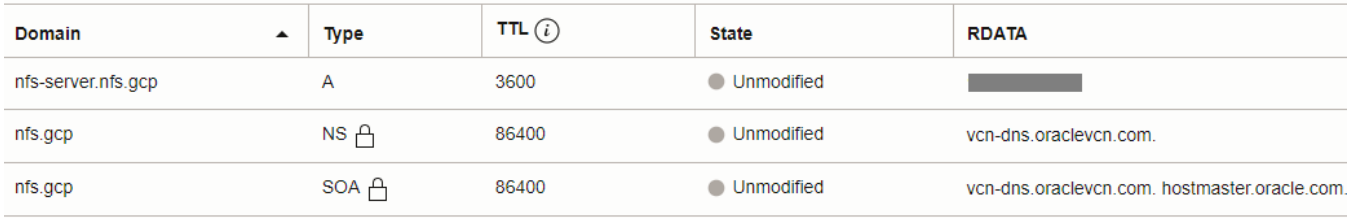


[6] https://docs.oracle.com/en/cloud/paas/autonomous-database/serverless/adbsb/load-oci-file-storage.html#GUID-7C396A7A-D20A-40F7-99D7-50B85B9B18DC
[7] https://cloud.google.com/filestore/docs

6. Click on the newly created zone, manage records, and add a record with the name of your choice, e.g., ***nfs-server***. Replace aa.an.fs.pe with the actual IP address of the Google Cloud-managed NFS server.



7. Publish the changes.



Domain	Type	TTL	State	RDATA
nfs-server.nfs.gcp	A	3600	Unmodified	[redacted]
nfs.gcp	NS	86400	Unmodified	vcn-dns.oraclevcn.com.
nfs.gcp	SOA	86400	Unmodified	vcn-dns.oraclevcn.com. hostmaster.oracle.com.

8. Update the Network Security Group (NSG) in OCI to allow network traffic flow from the VPC where the NFS server resides.

On the target database:

Step 2: Add the NFS Mount FQDN to the Access Control List (ACL)

```
SQL> exec DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE(host => 'nfs-server.nfs.gcp', ace =>
xs$ace_type(privilege_list => xs$name_list('connect', 'resolve'), principal_name =>
'ADMIN', principal_type => xs_acl.ptype_db));

PL/SQL procedure successfully completed.
```

Step 3: Create a Directory on the Autonomous Database

```
SQL> CREATE or replace DIRECTORY FSS_DIR AS 'fss';

Directory created.
```

Step 4: Attach NFS to Autonomous Database

Set the NFS version accordingly in the parameter "**params => JSON_OBJECT('nfs_version' value <value>)**".

```
SQL> BEGIN
DBMS_CLOUD_ADMIN.ATTACH_FILE_SYSTEM(
file_system_name => 'GCPNFS',
file_system_location => 'nfs-server.nfs.gcp:/zdm_share',
directory_name => 'FSS_DIR',
description => 'Attach GCP NFS',
params => JSON_OBJECT('nfs_version' value 3)
);
END;
/

PL/SQL procedure successfully completed.

SQL> SELECT object_name FROM DBMS_CLOUD.LIST_FILES('FSS_DIR');

OBJECT_NAME
--------------------------------------------------------------------------------
test.txt
```

Additional Configuration

SSH Key

ZDM connects via SSH to the Source Database servers; hence, an SSH key pair for the zdmuser is required. As zdmuser, run the following:

```
[zdmuser@zdmhost ~]$ mkdir ~/.ssh
[zdmuser@zdmhost ~]$ chmod 700 ~/.ssh
[zdmuser@zdmhost ~]$ /usr/bin/ssh-keygen -t rsa
Generating public/private rsa key pair.

Enter file in which to save the key (/home/zdmuser/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/zdmuser/.ssh/id_rsa.
Your public key has been saved in /home/zdmuser/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:keyfingerprintsample zdmuser@zdmhost
[zdmuser@zdmhost ~]$ cd ~/.ssh
[zdmuser@zdmhost .ssh]$ cat id_rsa.pub >> authorized_keys
[zdmuser@zdmhost .ssh]$ chmod 600 authorized_keys
```

You can find more information on ZDM Product's documentation section, "Generating a Private SSH Key Without a Passphrase."[8]

[8] https://docs.oracle.com/en/database/oracle/zero-downtime-migration/index.html

Before continuing with the migration environment setup, rename the id_rsa.pub file to <zdm_service_host_name>.ppk

On the ZDM Service Host.

```
[zdmuser@zdmhost .ssh]$ cd /home/zdmuser/.ssh

[zdmuser@zdmhost .ssh]$ mv id_rsa zdm.ppk
```

Authentication Token

The OCI user requires an Authentication Token, which can be created from the user's detail page. Click the "Auth Tokens" option and the "Generate Token" button. ZDM uses the Auth Token during the migration; hence, it is of the utmost importance that it is securely copied and stored.

OCI CLI Command Line Tool

The Oracle Cloud Infrastructure command-line tool (OCI CLI) accesses OCI resources during the migration, among other tasks. To install the OCI CLI on the ZDM Service Host, as the zdmuser, run as follows:

```
[zdmuser@zdmhost ~]$ sudo yum install python36-oci-cli
```

API Signing Public Key and Configuration File

ZDM uses an API Signing Public Key to call REST APIs. First, you need to create the API Keys. Do so by accessing the terminal on the ZDM Service Host, and as the zdmuser, run the following:

```
[zdmuser@zdmhost ~]$ mkdir .oci
[zdmuser@zdmhost ~]$ cd .oci
[zdmuser@zdmhost ~]$ openssl genrsa -out /u01/app/zdmhome/.oci/oci_api_key.pem 2048
[zdmuser@zdmhost ~]$ openssl rsa -pubout -in /u01/app/zdmhome/.oci/oci_api_key.pem -out
/u01/app/zdmhome/.oci/oci_api_key_public.pem
[zdmuser@zdmhost ~]$ cat oci_api_key_public.pem
```

Copy the catted 'oci_api_key_public.pem' file and save it; you will need it in the next step. Include the "Begin Public Key" and "End Public Key" lines during the copy. Go to your Oracle Cloud OCI Dashboard, navigate to the top right, click on your user profile icon, and select the top option representing your user. Select API Keys and Add API Key. Paste the public OCI API key file above and click Add Key.

You will see a configuration file preview. Copy its contents; you will use them to populate your configuration file in the following step.

As the zdmuser in the ZDM Service Host, create a configuration file in the command prompt; you can use vi/vim or any editor you prefer. In the empty file, paste the configuration file contents copied from above. Replace < path to your private key file > # TODO with the line above; once done, save the file and quit the editor:

```
/u01/app/zdmhome/.oci/oci_api_key.pem
```

Oracle GoldenGate on Docker

For ZDM Logical Online migrations to Oracle Database@Google Cloud, you will run Oracle GoldenGate on a Docker VM on-premises or on a Google Cloud Compute VM. Oracle GoldenGate keeps your source and target databases in sync and enables you to achieve zero to negligible downtime for your Oracle database migrations across database versions and platforms.
The docker container runs on Google Cloud Compute VM on RHEL version 8.10 for this step-by-step guide. The host's private IP is masked for this guide, but as an example, we will use the fictional **aa.bb.gg.do** address, and the hostname is **ggdockervm**.

Install Oracle GoldenGate on Docker

Step 1: Download the GoldenGate Docker Image

On Oracle Cloud, create a VM using the Oracle GoldenGate – Database Migrations image[9] from Marketplace. Search for "Goldengate migrations" in the Marketplace and choose the Database Migrations image. Choose the latest "Oracle DB – Microservices Edition – Promotional" version, and launch the stack.
Once the stack is completed, a VM in OCI Compute Instances will be created with the details you provided during the stack launch. Log in to that VM via SSH and issue the shell list command:

```
-bash-4.2$ ls -l
lrwxrwxrwx. 1 opc opc 36 Oct 3 11:21 ora21c-2115001.tar -> /opt/dockerimages/ora21c-
2115001.tar
```

The file /opt/dockerimages/ora21c-2115001.tar is the GoldenGate Docker image. Copy it to your Google Cloud Compute VM. Once done, the VM in OCI can be terminated. Its only purpose was to download the ora21c-2115001.tar Docker image file.

Step 2: Set Up the Docker Engine

Set up the Docker engine on the Google Cloud Compute VM to host the GoldenGate Docker image. In this case, following the Install Docker engine on Oracle RHEL 8[10]:

```
# add docker repository
[gcpuser@ggdockervm ~]$ sudo yum install -y yum-utils
[gcpuser@ggdockervm ~]$ sudo yum-config-manager --add-repo
https://download.docker.com/linux/rhel/docker-ce.repo

# install docker engine
[gcpuser@ggdockervm ~]$ sudo yum install docker-ce docker-ce-cli containerd.io
docker-buildx-plugin docker-compose-plugin -y

# start and enable docker service
[gcpuser@ggdockervm ~]$ sudo systemctl start docker
[gcpuser@ggdockervm ~]$ sudo systemctl enable docker

# verify docker installation
[gcpuser@ggdockervm ~]$ docker –version
[gcpuser@ggdockervm ~]$ docker compose version

# Add the local user to docker group to run the docker commands with sudo
[gcpuser@ggdockervm ~]$ sudo usermod -aG docker <your_user_name>
[gcpuser@ggdockervm ~]$ newgrp docker
[gcpuser@ggdockervm ~]$ id
```

[9] https://cloudmarketplace.oracle.com/marketplace/en_US/listing/96175416
[10] https://blogs.oracle.com/virtualization/post/install-docker-on-oracle-linux-7

Step 3: Load the Docker Image

Load the Docker image to the Docker engine. The ora21c-2115001.tar file is the one you copied to this VM in Step 0:

```
[gcpuser@ggdockervm ~]$ sudo docker load < ./ora21c-2115001.tar
67d008ba80bc: Loading layer [==================================================>]
253.7MB/253.7MB
45904b12df07: Loading layer [==================================================>]
6.144kB/6.144kB
30d4b06f2cea: Loading layer [==================================================>]
376.3MB/376.3MB
d43b06bdfe01: Loading layer [==================================================>]
27.14kB/27.14kB
369165b4aa0e: Loading layer [==================================================>]
1.787GB/1.787GB
82e31cbe7ea8: Loading layer [==================================================>]
18.43kB/18.43kB
Loaded image: oracle/goldengate:21.15.0.0.1
```

List the images:

```
[gcpuser@ggdockervm ~]$ sudo docker image list
REPOSITORY          TAG           IMAGE ID       CREATED         SIZE
oracle/goldengate   21.15.0.0.1   062a307e99c7   2 months ago    2.4GB
```

Step 4: Run the Docker Image

Run the Oracle GoldenGate Docker image as a container:

```
[gcpuser@ggdockervm ~]$ sudo docker run --name ogg2115 -p 443:443
docker.io/oracle/goldengate:21.15.0.0.1
```

For more information about the run parameters, visit Running Oracle GoldenGate in a Container[11].

The run output will display the ggadmin user password. You will need this later when running the ZDM migration command:

```
--------------------------------------------------------------------------------
--  Password for OGG administrative user 'oggadmin' is 'SamplePassword1234*&=+'
--------------------------------------------------------------------------------
```

Check the status of the Docker container:

```
[gcpuser@ggdockervm ~]$ sudo docker ps -a
CONTAINER ID   IMAGE                              COMMAND                  CREATED
STATUS                        PORTS                                          NAMES
135ed264e10c   oracle/goldengate:21.15.0.0.1   "/usr/local/bin/depl…"   About a
minute ago   Up About a minute (healthy)   80/tcp, 0.0.0.0:443->443/tcp, :::443-
>443/tcp   ogg2115
```

To start and stop the Docker container:

```
[gcpuser@ggdockervm ~]$ sudo docker stop 135ed264e10c
[gcpuser@ggdockervm ~]$ sudo docker start 135ed264e10c
```

[11] https://github.com/oracle/docker-images/tree/main/OracleGoldenGate/21c#running-oracle-goldengate-in-a-container

Step 5: Docker, Instant Client & Wallet Further Configuration Steps

Download[12] Oracle Instant Client locally, and proceed to copy it and the ADB-S wallet to the docker container, and check connectivity to source and target databases from the docker container:

```
[gcpuser@ggdockervm ~]$ sudo docker cp instantclient_21_3.zip 135ed264e10c:/
instantclient_21_3.zip
[gcpuser@ggdockervm ~]$ sudo docker cp Wallet_zdmadb.zip 135ed264e10c:/
Wallet_zdmadb.zip
```

Connect to the docker container:

```
[gcpuser@ggdockervm ~]$ sudo docker exec -it ogg2115 /bin/bash
[root@135ed264e10c /]# mv instantclient_21_3.zip /home/ogg/
[root@135ed264e10c /]# mv Wallet_zdmadb.zip /home/ogg/
[root@135ed264e10c /]# chown ogg:ogg /home/ogg/instantclient_21_3.zip
[root@135ed264e10c /]# chown ogg:ogg Wallet_zdmadb.zip
[root@135ed264e10c /]# su - ogg
[ogg@135ed264e10c /]# unzip instantclient_21_3.zip
[ogg@135ed264e10c /]# chmod 744 instantclient_21_3/sqlplus
```

Ensure the wallet containing certificates for TLS authentication is in the correct location in the GoldenGate Hub as per ZDM's documentation[13]:

- For an Autonomous Database, the wallet file should be in the following directory:
  `/u02/deployments/deployment_name/etc/adb`

- For a co-managed database, the wallet file should be in directory:
  `/u02/deployments/deployment_name/etc`

Log in as the Oracle GoldenGate user, create the required wallet directory, copy the wallet and instant client, and unzip them.

```
[ogg@135ed264e10c ~]$ mkdir /u02/Deployment/etc/adb
[ogg@135ed264e10c ~]$ mv Wallet_zdmadb.zip /u02/Deployment/etc/adb
[ogg@135ed264e10c ~]$ cd /u02/Deployment/etc/adb
[ogg@135ed264e10c adb]$ unzip Wallet_zdmadb.zip
```

Open and Edit the sqlnet.ora file:

```
$ vi sqlnet.ora
```

Copy the following line:

```
DIRECTORY="/u02/Deployment/etc/adb"
```

Export the following environment variables:

```
export LD_LIBRARY_PATH=/home/ogg/instantclient_21_3
export TNS_ADMIN=/u02/Deployment/etc/adb
```

[12] https://docs.oracle.com/en/database/oracle/oracle-database/21/lacli/install-instant-client-using-zip.html
[13] https://docs.oracle.com/en/database/oracle/zero-downtime-migration/21.4/zdmug/preparing-logical-database-migration1.html#GUID-3BD5C670-E0E2-4278-B9C1-B62F0F6E2210

Edit the etc hosts file on the docker container:

```
[root@ggdockervm ~]$ vi /etc/hosts
aa.bb.sr.db onphost
aa.dd.bb.ss sample.adb.us-region-1.oraclecloud.com
```

Check database connectivity to the source and target using the hostnames in the connection string:

```
[ogg@135ed264e10c ]# sqlplus system@onphost:1521/oradb_onp
[ogg@135ed264e10c ]# sqlplus system@zdmadb_high
```

Database Migration Step by Step with ZDM

Step 1: Fill the response file

```
vi /home/zdmuser/logical_online_adb_nfs/logical_online_adb_nfs.rsp
```

```
# migration method
MIGRATION_METHOD=ONLINE_LOGICAL
DATA_TRANSFER_MEDIUM=NFS
# data pump
DATAPUMPSETTINGS_JOBMODE=SCHEMA
DATAPUMPSETTINGS_METADATAREMAPS-1=type:REMAP_TABLESPACE,oldValue:USERS,newValue:DATA
INCLUDEOBJECTS-1=owner:HR
DATAPUMPSETTINGS_EXPORTDIRECTORYOBJECT_NAME=DATA_PUMP_DIR_NFS
DATAPUMPSETTINGS_IMPORTDIRECTORYOBJECT_NAME=FSS_DIR
# source db (pdb)
SOURCEDATABASE_CONNECTIONDETAILS_HOST=onphost
SOURCEDATABASE_CONNECTIONDETAILS_PORT=1521
SOURCEDATABASE_CONNECTIONDETAILS_SERVICENAME=pdbsrc
SOURCEDATABASE_ADMINUSERNAME=SYSTEM
SOURCEDATABASE_GGADMINUSERNAME=ggadmin
# source db (cdb)
SOURCECONTAINERDATABASE_CONNECTIONDETAILS_HOST=onhost2
SOURCECONTAINERDATABASE_CONNECTIONDETAILS_PORT=1521
SOURCECONTAINERDATABASE_CONNECTIONDETAILS_SERVICENAME=oradb
SOURCECONTAINERDATABASE_ADMINUSERNAME=SYSTEM
SOURCECONTAINERDATABASE_GGADMINUSERNAME=c##ggadmin
# target ADB (pdb)
TARGETDATABASE_OCID=ocid1.autonomousdatabase.oc1.iad.aaaa.bbb.cccc.ddddd
TARGETDATABASE_ADMINUSERNAME=ADMIN
TARGETDATABASE_GGADMINUSERNAME=ggadmin
# oci cli
OCIAUTHENTICATIONDETAILS_USERPRINCIPAL_USERID=ocid1.user.oc1..aaaa.bbb.ccccc.ddddd
OCIAUTHENTICATIONDETAILS_USERPRINCIPAL_TENANTID=ocid1.tenancy.oc1.aaa.bbbbb
OCIAUTHENTICATIONDETAILS_USERPRINCIPAL_FINGERPRINT=12:ac:34:cc:aa
OCIAUTHENTICATIONDETAILS_USERPRINCIPAL_PRIVATEKEYFILE=/home/zdmuser/.oci/oci_api_key.pem
OCIAUTHENTICATIONDETAILS_REGIONID=us-ashburn-1


# GoldenGate
GOLDENGATEHUB_ADMINUSERNAME=oggadmin
GOLDENGATEHUB_SOURCEDEPLOYMENTNAME=Local
GOLDENGATEHUB_TARGETDEPLOYMENTNAME=Local
# Private IP of the VM where Docker is running
GOLDENGATEHUB_URL=https://aa.bb.gg.do
GOLDENGATEHUB_ALLOWSELFSIGNEDCERTIFICATE=TRUE
```

Step 2: Evaluate the Configuration

Execute the following command on the ZDM host as zdmuser to evaluate the migration. ZDM will check the source and target database configurations. The actual migration will not be started. On the ZDM host as zdmuser:

```
[zdmuser@zdmhost ~]$ $ZDMHOME/bin/zdmcli migrate database \
-rsp /logical_online_adb_nfs/logical_online_adb_nfs.rsp \
-sourcenode onphost \
-sourcesid oradb \
-srcauth zdmauth \
-srcarg1 user:gcpuser \
-srcarg2 identity_file:/home/zdmuser/.ssh/id_rsa \
-srcarg3 sudo_location:/usr/bin/sudo \
-eval

Enter source database administrative user "SYSTEM" password:
Enter source database administrative user "ggadmin" password:
Enter source container database administrative user "SYSTEM" password:
Enter source container database administrative user "c##ggadmin" password:
Enter target database administrative user "ADMIN" password:
Enter target database administrative user "ggadmin" password:
Enter Oracle GoldenGate hub administrative user "oggadmin" password:
Enter Authentication Token for OCI user "ocid1.user.oc1...":
Enter Data Pump encryption password:
Operation "zdmcli migrate database" scheduled with the job ID "1".
```

If the source database uses ASM for storage management, use -sourcedb <db_unique_name> instead of -sourcesid <SID> in the zdmcli command.

Check the job status. On the ZDM host as zdmuser:

```
[zdmuser@zdmhost ~]$ $ZDMHOME/bin/zdmcli query job -jobid 1
zdmhost.zdm: Audit ID: 272
Job ID: 1
User: zdmuser
Client: zdmhost
Job Type: "EVAL"
Scheduled job command: "zdmcli migrate database -rsp
/home/zdmuser/logical_online_adb/logical_online_adb.rsp -sourcenode onphost -
sourcesid oradb -srcauth zdmauth -srcarg1 user:sinan_petrus_toma_oracle_com -
srcarg2 identity_file:/home/zdmuser/.ssh/id_rsa -srcarg3
sudo_location:/usr/bin/sudo -eval"
Scheduled job execution start time: 2024-10-22T12:33:36Z. Equivalent local time:
2024-10-22 12:33:36
Current status: SUCCEEDED
Result file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-1-2024-10-22-
12:33:53.log"
Metrics file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-1-2024-10-22-
12:33:54.json"
Excluded objects file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-1-
filtered-objects-2024-10-22T12:34:17.696.json"
Job execution start time: 2024-10-22 12:33:54
Job execution end time: 2024-10-22 12:36:08
Job execution elapsed time: 2 minutes 14 seconds
ZDM_VALIDATE_TGT ..... COMPLETED
ZDM_VALIDATE_SRC ..... COMPLETED
ZDM_SETUP_SRC ..... COMPLETED
ZDM_PRE_MIGRATION_ADVISOR ..... COMPLETED
```

```
ZDM_VALIDATE_GG_HUB ................... COMPLETED
ZDM_VALIDATE_DATAPUMP_SETTINGS_SRC .... COMPLETED
ZDM_VALIDATE_DATAPUMP_SETTINGS_TGT .... COMPLETED
ZDM_PREPARE_DATAPUMP_SRC .............. COMPLETED
ZDM_DATAPUMP_ESTIMATE_SRC ............. COMPLETED
ZDM_CLEANUP_SRC ....................... COMPLETED
```

Detailed information about the migration process can be found by monitoring the log file:

```
[zdmuser@zdmhost ~]$ tail -f /home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-1-2024-
10-22-12:33:53.log
```

Step 3: Initiate the Migration

To initiate the actual migration, execute the same command for evaluation, but this time without the -eval parameter. Oracle ZDM allows you to pause the migration process at any given phase. For example, the migration process can be Paused after Oracle GoldenGate keeps the target database in sync with the source. Upon executing the zdm migrate database command, the -pauseafter flag must be entered with the desired pausing stage, ZDM_MONITOR_GG_LAG. On the ZDM host as zdmuser:

```
[zdmuser@zdmhost ~]$ $ZDMHOME/bin/zdmcli migrate database \
-rsp /logical_online_adb_nfs/logical_online_adb_nfs.rsp \
-sourcenode onphost \
-sourcesid oradb \
-srcauth zdmauth \
-srcarg1 user:gcpuser \
-srcarg2 identity_file:/home/zdmuser/.ssh/id_rsa \
-srcarg3 sudo_location:/usr/bin/sudo \
-pauseafter ZDM_MONITOR_GG_LAG

Enter source database administrative user "SYSTEM" password:
Enter source database administrative user "ggadmin" password:
Enter source container database administrative user "SYSTEM" password:
Enter source container database administrative user "c##ggadmin" password:
Enter target database administrative user "ADMIN" password:
Enter target database administrative user "ggadmin" password:
Enter Oracle GoldenGate hub administrative user "oggadmin" password:
Enter Authentication Token for OCI user "ocid1.user.oc1...":
Enter Data Pump encryption password:
Operation "zdmcli migrate database" scheduled with the job ID "2".
```

Check the job status. On the ZDM host as zdmuser:

```
[zdmuser@zdmhost ~]$ $ZDMHOME/bin/zdmcli query job -jobid 2
zdmhost.zdm: Audit ID: 323
Job ID: 2
User: zdmuser
Client: zdmhost
Job Type: "MIGRATE"
Scheduled job command: "zdmcli migrate database -rsp
/home/zdmuser/logical_online_adb/logical_online_adb.rsp -sourcenode onphost -
sourcesid oradb -srcauth zdmauth -srcarg1 user:sinan_petrus_toma_oracle_com -
srcarg2 identity_file:/home/zdmuser/.ssh/id_rsa -srcarg3
sudo_location:/usr/bin/sudo -pauseafter ZDM_MONITOR_GG_LAG"
```

```
Scheduled job execution start time: 2024-10-22T12:40:06Z. Equivalent local time:
2024-10-22 12:40:06
Current status: PAUSED
Current Phase: "ZDM_MONITOR_GG_LAG"
Result file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-2-2024-10-22-
12:40:24.log"
Metrics file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-2-2024-10-22-
12:40:24.json"
Excluded objects file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-2-
filtered-objects-2024-10-22T12:40:45.805.json"
Job execution start time: 2024-10-22 12:40:24
Job execution end time: 2024-10-22 12:49:35
Job execution elapsed time: 9 minutes 11 seconds
Oracle GoldenGate replication metrics:
Extract "EXTCWK4D" status: RUNNING
Extract "EXTCWK4D" trail files generated: 1
Replicat "R6EV8" status: RUNNING
Replicat "R6EV8" trail files applied: 1
End-to-end heartbeat lag 1.45 seconds
Replication throughput: 0.0 GBph
ZDM_VALIDATE_TGT ..................... COMPLETED
ZDM_VALIDATE_SRC ..................... COMPLETED
ZDM_SETUP_SRC ........................ COMPLETED
ZDM_PRE_MIGRATION_ADVISOR ............ COMPLETED
ZDM_VALIDATE_GG_HUB .................. COMPLETED
ZDM_VALIDATE_DATAPUMP_SETTINGS_SRC .... COMPLETED
ZDM_VALIDATE_DATAPUMP_SETTINGS_TGT .... COMPLETED
ZDM_PREPARE_DATAPUMP_SRC ............. COMPLETED
ZDM_DATAPUMP_ESTIMATE_SRC ............ COMPLETED
ZDM_PREPARE_GG_HUB ................... COMPLETED
ZDM_ADD_HEARTBEAT_SRC ................ COMPLETED
ZDM_ADD_SCHEMA_TRANDATA_SRC .......... COMPLETED
ZDM_CREATE_GG_EXTRACT_SRC ............ COMPLETED
ZDM_PREPARE_DATAPUMP_TGT ............. COMPLETED
ZDM_DATAPUMP_EXPORT_SRC .............. COMPLETED
ZDM_TRANSFER_DUMPS_SRC ............... COMPLETED
ZDM_DATAPUMP_IMPORT_TGT .............. COMPLETED
ZDM_POST_DATAPUMP_SRC ................ COMPLETED
ZDM_POST_DATAPUMP_TGT ................ COMPLETED
ZDM_ADD_HEARTBEAT_TGT ................ COMPLETED
ZDM_ADD_CHECKPOINT_TGT ............... COMPLETED
ZDM_CREATE_GG_REPLICAT_TGT ........... COMPLETED
ZDM_START_GG_REPLICAT_TGT ............ COMPLETED
ZDM_MONITOR_GG_LAG ................... COMPLETED
ZDM_PREPARE_SWITCHOVER_APP ........... PENDING
ZDM_ADVANCE_SEQUENCES ................ PENDING
ZDM_SWITCHOVER_APP ................... PENDING
ZDM_POST_SWITCHOVER_TGT .............. PENDING
ZDM_RM_GG_EXTRACT_SRC ................ PENDING
ZDM_RM_GG_REPLICAT_TGT ............... PENDING
ZDM_DELETE_SCHEMA_TRANDATA_SRC ....... PENDING
ZDM_RM_HEARTBEAT_SRC ................. PENDING
ZDM_RM_CHECKPOINT_TGT ................ PENDING
ZDM_RM_HEARTBEAT_TGT ................. PENDING
ZDM_CLEAN_GG_HUB ..................... PENDING
ZDM_POST_ACTIONS ..................... PENDING
ZDM_CLEANUP_SRC ...................... PENDING

Pause After Phase: "ZDM_MONITOR_GG_LAG"
```

Pay attention to the current job status. It is in PAUSED status now. Also, the progress stopped after phase ZDM_MONITOR_GG_LAG was COMPLETED. At this stage, every change in the source database is immediately synchronized with the target database. Resume the job when your application is ready for migration.

Step 4: Complete the Migration

Resume the job from the previous step. On the ZDM host as zdmuser, resume the job and query the status until all phases are completed:

```
[zdmuser@zdmhost ~]$ $ZDMHOME/bin/zdmcli resume job -jobid 2
[zdmuser@zdmhost ~]$ $ZDMHOME/bin/zdmcli query job -jobid 2
...
ZDM_VALIDATE_TGT ..................... COMPLETED
ZDM_VALIDATE_SRC ..................... COMPLETED
ZDM_SETUP_SRC ........................ COMPLETED
ZDM_PRE_MIGRATION_ADVISOR ............ COMPLETED
ZDM_VALIDATE_GG_HUB .................. COMPLETED
ZDM_VALIDATE_DATAPUMP_SETTINGS_SRC .... COMPLETED
ZDM_VALIDATE_DATAPUMP_SETTINGS_TGT .... COMPLETED
ZDM_PREPARE_DATAPUMP_SRC ............. COMPLETED
ZDM_DATAPUMP_ESTIMATE_SRC ............ COMPLETED
ZDM_PREPARE_GG_HUB ................... COMPLETED
ZDM_ADD_HEARTBEAT_SRC ................ COMPLETED
ZDM_ADD_SCHEMA_TRANDATA_SRC .......... COMPLETED
ZDM_CREATE_GG_EXTRACT_SRC ............ COMPLETED
ZDM_PREPARE_DATAPUMP_TGT ............. COMPLETED
ZDM_DATAPUMP_EXPORT_SRC .............. COMPLETED
ZDM_TRANSFER_DUMPS_SRC ............... COMPLETED
ZDM_DATAPUMP_IMPORT_TGT .............. COMPLETED
ZDM_POST_DATAPUMP_SRC ................ COMPLETED
ZDM_POST_DATAPUMP_TGT ................ COMPLETED
ZDM_ADD_HEARTBEAT_TGT ................ COMPLETED
ZDM_ADD_CHECKPOINT_TGT ............... COMPLETED
ZDM_CREATE_GG_REPLICAT_TGT ........... COMPLETED
ZDM_START_GG_REPLICAT_TGT ............ COMPLETED
ZDM_MONITOR_GG_LAG ................... COMPLETED
ZDM_PREPARE_SWITCHOVER_APP ........... COMPLETED
ZDM_ADVANCE_SEQUENCES ................ COMPLETED
ZDM_SWITCHOVER_APP ................... COMPLETED
ZDM_POST_SWITCHOVER_TGT .............. COMPLETED
ZDM_RM_GG_EXTRACT_SRC ................ COMPLETED
ZDM_RM_GG_REPLICAT_TGT ............... COMPLETED
ZDM_DELETE_SCHEMA_TRANDATA_SRC ....... COMPLETED
ZDM_RM_HEARTBEAT_SRC ................. COMPLETED
ZDM_RM_CHECKPOINT_TGT ................ COMPLETED
ZDM_RM_HEARTBEAT_TGT ................. COMPLETED
ZDM_CLEAN_GG_HUB ..................... COMPLETED
ZDM_POST_ACTIONS ..................... COMPLETED
ZDM_CLEANUP_SRC ...................... COMPLETED
```

Known Issues

All common issues are documented and updated periodically in Oracle Zero Downtime Migration's documentation, specifically on the product release note, Known Issues section:

https://docs.oracle.com/en/database/oracle/zero-downtime-migration/

Troubleshooting Oracle GoldenGate Replication

During your migration, you can pause the ZDM migration job after the **ZDM_MONITOR_GG_LAG** phase.

At this stage, every transaction on the source database is replicated via GoldenGate to the target database. If this is not the case, log in to the Docker container and check the EXTRACT and REPLICAT deployments and log files:

```
# connect to the docker container and switch to ogg user
[gcpuser@ggdockervm ~]$ sudo docker exec -it ogg2113 /bin/bash
[root@135ed264e10c /]# su - ogg

# check the deployment files
[ogg@135ed264e10c ~]$ ls -l /u02/Deployment/etc/conf/ogg
total 16
-rw-r-----. 1 ogg ogg 257 Jun 10 19:05 EXT4BL3P.prm
-rw-r--r--. 1 ogg ogg  17 Jun 10 13:05 GLOBALS
-rw-r-----. 1 ogg ogg 260 Jun 10 19:34 RN22M.prm
-rw-r-----. 1 ogg ogg 260 Jun 10 19:57 RN22M.prm.backup

# check the deployments log files
[ogg@135ed264e10c ~]$ view /u02/Deployment/var/log/extract.log
[ogg@135ed264e10c ~]$ view /u02/Deployment/var/log/replicat.log
```

Check the Deployment Status:

```
[ogg@135ed264e10c ~]$ /u01/ogg/bin/adminclient
Oracle GoldenGate Administration Client for Oracle
Version 21.15.0.0.1 OGGCORE_21.15.0.0.1OGGRU_PLATFORMS_240108.2205

Copyright (C) 1995, 2024, Oracle and/or its affiliates. All rights reserved.

Oracle Linux 7, x64, 64bit (optimized) on Jan  9 2024 09:04:36
Operating system character set identified as US-ASCII.

OGG (not connected) 1>
OGG (not connected) 1> connect http://127.0.0.1 as oggadmin password /Qlq1UTGMK+N-
EksH
Using default deployment 'Local'

OGG (http://127.0.0.1 Local) 2> info all
Program     Status      Group       Type             Lag at Chkpt  Time Since Chkpt

ADMINSRVR   RUNNING
DISTSRVR    RUNNING
PMSRVR      RUNNING
RECVSRVR    RUNNING
EXTRACT     RUNNING     EXT4BL3P    INTEGRATED       00:00:01      00:00:05
REPLICAT    ABENDED     RN22M       PARALLEL NONINT  00:04:21      00:00:21
```

In this case, the REPLICAT status was ABENDED. The replicat.log indicated insufficient privileges for the ggadmin user on the target database. After fixing the issue, start the deployment:

```
OGG (http://127.0.0.1 Local) 3> start RN22M
2024-06-10T20:35:54Z  INFO    OGG-00975  Replicat group RN22M starting.
2024-06-10T20:35:54Z  INFO    OGG-15445  Replicat group RN22M started.
```

Troubleshooting Oracle ZDM & Other Resources

For Oracle ZDM log review:

- ZDM Server Logs:
  - Check - $ZDM_BASE/crsdata/<zdm_service_node>/rhp/rhpserver.log.0
- Check source node logs
  - - <oracle_base>/zdm/zdm_<src_db_name>_<job_id>/zdm/log
- Check target node logs.
  - - <oracle_base>/zdm/zdm_<tgt_db_name>_<job_id>/zdm/log

For all Oracle Support Service Requests related to Zero Downtime Migration, please be sure to follow the instructions in My Oracle Support Document:

- SRDC – Data Collection for Database Migration Using Zero Downtime Migration (ZDM) (DOC ID 2595205.1)
- https://support.oracle.com/epmos/faces/DocContentDisplay?id=2595205.1

ORACLE

Connect with us

Call +**1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

blogs.oracle.com facebook.com/oracle twitter.com/oracle