ORACLE

Oracle Zero Downtime Migration – Logical Online Migration to ExaDB-D on Oracle Database@Azure

Technical Brief

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.
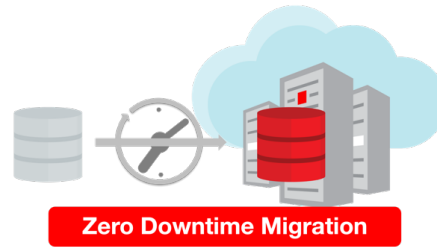
ORACLE

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 migration of workloads into the Oracle Cloud, Engineered Systems, and Oracle Database@ Azure. 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 and any Oracle-owned infrastructure, including Exadata Database Machine On-Premises, Exadata Cloud at Customer (ExaDB-C@C), 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 Exadata Database Service on Dedicated Infrastructure (ExaDB-D) on Oracle Database@Azure, 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@Azure product website.[2]

---

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

ORACLE

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 in Oracle Exadata Database Service on Dedicated Infrastructure (ExaDB-D) on Oracle Database@Azure, 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 Direct Data Transfer 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 and 21c. 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.

Architecture

An architectural overview of the ZDM server, the source database on-premises, the target database on Oracle Exadata Database Service on Dedicated Infrastructure (ExaDB-D) on Oracle Database@Azure, and all networks and components required are described in the diagram below:
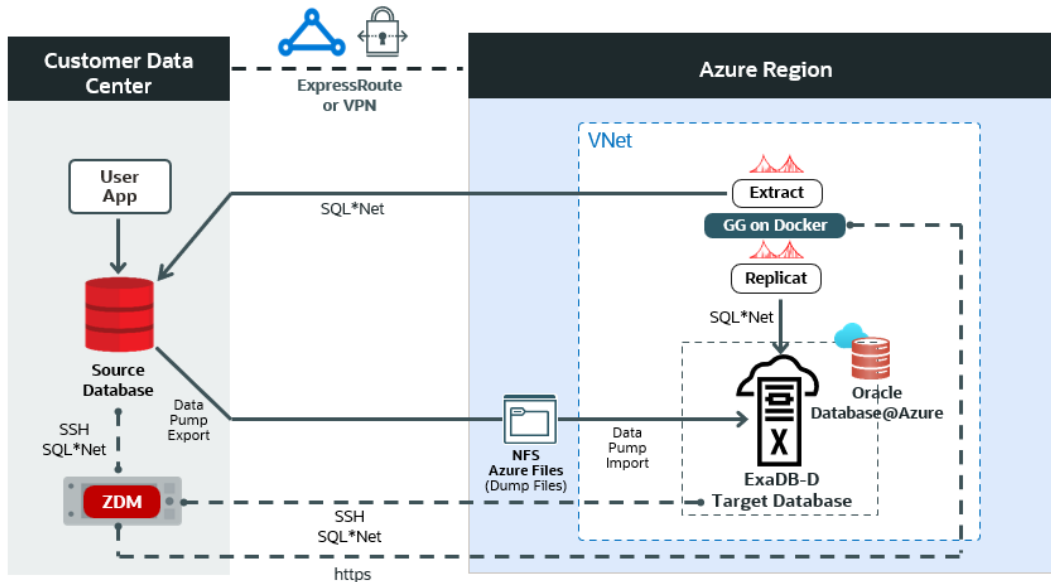

Figure 2. This is a High-Level Architectural overview showcasing the customer data center where the source database and ZDM's server reside. It also shows all connectivity to the target Oracle Exadata Database Service on Dedicated Infrastructure (ExaDB-D) on Oracle Database@Azure.

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***.

---

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

ORACLE

Network and Connectivity

Region
An Oracle Cloud Infrastructure region is a localized geographic area that contains one or more data centers, called availability domains. Regions are independent of other regions, and vast distances can separate them (across countries or continents).

Virtual Cloud Network (VCN) and subnet
A VCN is a customizable, software-defined network that you set up in an Oracle Cloud Infrastructure region. Like traditional data center networks, VCNs give you complete control over your network environment. A VCN can have multiple non-overlapping CIDR blocks that you can change after you create the VCN. You can segment a VCN into subnets, which can be scoped to a region or an availability domain. Each subnet consists of a contiguous range of addresses that don't overlap with the other subnets in the VCN. You can change the size of a subnet after creation. A subnet can be public or private.

OCI Network Security Group (NSG)
A network security group (NSG) provides a virtual firewall for cloud resources with the same security posture. For example, a group of compute instances performs the same tasks and thus needs to use the same set of ports.

Azure VNet
Azure Virtual Network (VNet) is the fundamental building block for your private network in Azure. VNet enables many Azure resources, such as Azure virtual machines (VM), to securely communicate with each other, the internet, and on-premises networks.

Azure Delegated Subnet
Subnet delegation is Microsoft's ability to inject a managed service, specifically a platform-as-a-service service, directly into your virtual network. This means you can designate or delegate a subnet to be a home for an externally managed service inside your virtual network. In other words, that external service will act as a virtual network resource, even though technically it is an external platform-as-a-service service.

Virtual network interface card (VNIC)
The services in Azure data centers have physical network interface cards (NICs). Virtual machine instances communicate using virtual NICs (VNICs) associated with the physical NICs. Each instance has a primary VNIC that's automatically created and attached during launch and is available during the instance's lifetime.

Azure Route table (User Defined Route – UDR)
Virtual route tables contain rules to route traffic from subnets to destinations outside a VNet, typically through gateways. Route tables are associated with subnets in a VNet.

Local Network Virtual Appliance (NVA)
For routing purposes, deploy a Network Virtual Appliance (NVA) within the Oracle Database@Azure VNet following the Microsoft documentation[4][5].

---

[4] https://techcommunity.microsoft.com/t5/fasttrack-for-azure/creating-a-local-network-virtual-appliance-in-azure-for-oracle/ba-p/4218101
[5] https://learn.microsoft.com/en-us/azure/oracle/oracle-db/oracle-database-network-plan

Source Database

For this step-by-step guide, the source database runs on-premises on an Oracle Linux Server 7.7. 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 19.21 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@Azure offers the following products:

- **Oracle Exadata Database Service on Dedicated Infrastructure**
  - 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**
  - Autonomous Database provides an easy-to-use, fully autonomous database that scales elastically, delivers fast query performance, and requires no database administration.

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

Oracle Database@Azure service offers the same low latency as other Azure-native services and meets mission-critical workloads and cloud-native development needs. Users manage the service on the Azure console and with Azure automation tools. The service is deployed in Azure Virtual Network (VNet) and integrated with the Azure identity and access management system. The OCI and Oracle Database metrics and audit logs are natively available in Azure. The service requires that users have an Azure tenancy and an OCI tenancy.
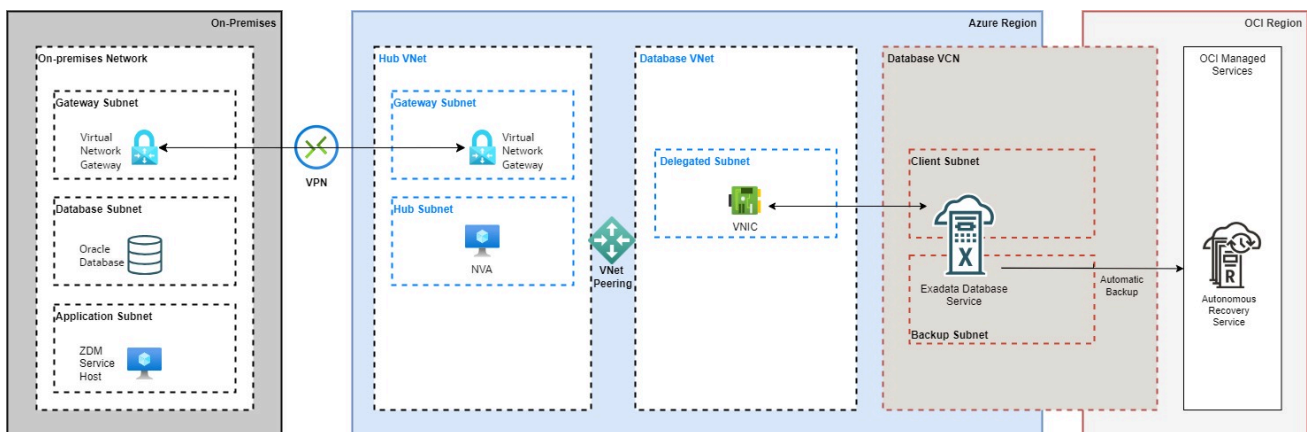

Figure 3 Network and Architectural Overview

ORACLE

For this step-by-step guide, the target platform is Oracle Exadata Database Service on Dedicated Infrastructure (ExaDB-D) on Oracle Database@Azure. The infrastructure contains a 2-node VM cluster. The VM cluster host private IPs are masked for this guide, but as an example, we will use the fictional ta.db.oa.1 and ta.db.oa.2, and the host names are *exadbazure1* and *exadbazure2*.

ZDM requires configuring a placeholder database target environment before beginning the migration process.

The target Oracle database is a 2-node Oracle RAC version 19.22 with multitenant architecture created using Oracle Cloud Console. The database name is *oradb* (the same as the source database), and the database's unique name is *oradb_exa* (different from the source database's unique name).

The HR schema is to be migrated to the target PDB **pdbtgt**.

Source and Target 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.
- 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)[6]
- If the source database is Oracle Database 12.1.0.2 or a later release, apply mandatory RDBMS patches to it.
- 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.
- The DATAPUMP_IMP_FULL_DATABASE role is required for the import operation at the specified target database for the specified target database user.
- Disable auto-purge jobs on the target database immediately after instantiation (Data Pump Import). Otherwise, the purge jobs will cause data inconsistency conditions on the target database, causing GoldenGate Replicat to fail.

Additional Configuration

SSH Key

Check the key format:

```
[zdmuser@zdmhost ~]$ head -n1 id_rsa
```

Create an SSH Key in RSA format (if not already created):

```
[zdmuser@zdmhost ~]$ ssh-keygen -m PEM -t rsa
```

Change an existing SSH key into RSA format (if already created and need to reformat):

```
[zdmuser@zdmhost ~]$ ssh-keygen -p -m PEM -f id_rsa
```

ORACLE

NFS File Share via Azure Files

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 Azure Files service to store the Data Pump dump files. For this step-by-step guide, the file share path is **/azurefilesnfs/dumpfiles/**. The NFS share must be mounted on both the source and target database hosts.

The NFS-mounted path is only readable for the target database user if the Unique Identifiers (UIDs) for the source database user match with the target database user (see Oracle Zero Downtime Migration Product Documentation).

Example of mounting the NFS share:

```
sudo mount -t nfs odaamigration.file.core.windows.net:/odaamigration/testmigration
/azurefilesnfs -o vers=4,minorversion=1,sec=sys
```

ORACLE

Oracle GoldenGate on Docker

For ZDM Logical Online migrations to Oracle Database@Azure, you will run Oracle GoldenGate on a Docker VM on-premises or on an Azure 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. For this step-by-step guide, the docker container runs on Azure IaaS VM on Oracle Linux 7.9. 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[7] 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
total 0
lrwxrwxrwx. 1 opc opc 37 Jul 3 14:41 ogg-credentials.json -> /u02/deployments/ogg-
credentials.json
lrwxrwxrwx. 1 opc opc 36 Jul 3 14:41 ora21c-2113000.tar -> /opt/dockerimages/ora21c-
2113000.tar
drwxr-x---. 3 opc opc 18 Jul 3 15:55 oradiag_opc
```

The file /opt/dockerimages/ora21c-2113000.tar is the GoldenGate Docker image. Copy it to your Azure IaaS VM. Once done, the VM in OCI can be terminated. Its only purpose was to download the ora21c-2113000.tar Docker image file.

Step 2: Set Up the Docker Engine

Set up the Docker engine on the Azure IaaS VM to host the GoldenGate Docker image. In this case, following the Install Docker engine on Oracle Linux 7[8]:

```
# install docker engine
[azureuser@ggdockervm ~]$ sudo yum install docker-engine docker-cli

# start docker service and configure it to start at boot time:
[azureuser@ggdockervm ~]$ sudo systemctl enable --now docker

# ensure docker service is running:
[azureuser@ggdockervm ~]$ systemctl status docker

# display configuration and version of docker engine
[azureuser@ggdockervm ~]$ sudo docker info
```

---

[7] https://cloudmarketplace.oracle.com/marketplace/en_US/listing/96175416
[8] 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-2113000.tar file is the one you copied to this VM in Step 0:

```
[azureuser@ggdockervm ~]$ sudo docker load < ./ora21c-2113000.tar
67d008ba80bc: Loading layer
[==================================================>]  253.7MB/253.7MB
69e9cf1483ea: Loading layer
[==================================================>]  6.144kB/6.144kB
b8c7ceef0daa: Loading layer
[==================================================>]  372.8MB/372.8MB
5e5793be3604: Loading layer
[==================================================>]  26.62kB/26.62kB
377155359816: Loading layer
[==================================================>]  1.739GB/1.739GB
4411981a6090: Loading layer
[==================================================>]  18.43kB/18.43kB
Loaded image: oracle/goldengate:21.13.0.0.0
```

List the images:

```
[azureuser@ggdockervm ~]$ sudo docker image list
REPOSITORY          TAG               IMAGE
ID          CREATED            SIZE
oracle/goldengate  21.13.0.0.0       f9d2aea5f6a7       4 months
ago       2.34GB
```

Step 4: Run the Docker Image

Run the Oracle GoldenGate Docker image as a container:

```
[azureuser@ggdockervm ~]$ sudo docker run --name ogg2113 -p 443:443
docker.io/oracle/goldengate:21.13.0.0.0
```

For more information about the run parameters, visit Running Oracle GoldenGate in a Container[9].

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:

```
[azureuser@ggdockervm ~]$ sudo docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
d9156f6223c3 oracle/goldengate:21.13.0.0.0 "/usr/local/bin/depl…" 2 hours ago Up 2
hours (healthy) 80/tcp, 0.0.0.0:443->443/tcp  ogg2113
```

To start and stop the Docker container:

```
[azureuser@ggdockervm ~]$ sudo docker stop d9156f6223c3
[azureuser@ggdockervm ~]$ sudo docker start d9156f6223c3
```

---

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

Step 5: Ensure Connectivity to Source and Target Databases

Ensure that the source and target database hostnames are resolvable and that you can connect to the source and target databases from within the Docker container. Connect to the docker container:

```
[onpuser@ggdockervm ~]$ sudo docker exec -it ogg2113 /bin/bash
[root@d9156f6223c3 /]#
```

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

```
[root@d9156f6223c3 ]# sqlplus system@onphost:1521/oradb_onp
[root@d9156f6223c3 ]# sqlplus system@exadbazure1:1521/oradb_exa
```

Database Migration Step by Step with ZDM

Step 1: Prepare the Source Database Host On-Premises

Copy the SSH public key of the zdmuser from the ZDM host to the .ssh/authorized_keys file on the source database host for the user you want to use for login, in this case, *onpuser*:

```
#on ZDM host as zdmuser
[zdmuser@zdmhost ~]$ cat .ssh/id_rsa.pub
#on the source database host as user onpuser
[onpuser@onphost ~]$ vi .ssh/authorized_keys
#insert the public key and save the changes
```

Add the target database hostname, IP address, and SCAN name to the /etc/hosts file. As root user:

```
[root@onphost ~]# vi /etc/hosts
#add the following entries
ta.db.oa.1 oradb_exa_sample.oravcn.sample.com target
ta.db.oa.1 demo-scan-sample.oravcn.sample.com target-scan
```

Step 2: Prepare the Source Database On-Premises

Prepare the source database. 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$database;
FORCE_LOGGING
---------------------------------------
NO
SQL> alter database force logging;
SQL> select force_logging from v$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');
```

Step 3: Prepare the target database host on ExaDB-D on Oracle Database@Azure

Copy the SSH public key of the zdmuser from the ZDM host to the .ssh/authorized_keys file on the target database host for the user you want to use for login; in this case, OPC:

```
#on ZDM host as zdmuser
[zdmuser@zdmhost ~]$ cat .ssh/id_rsa.pub

#on the target database hosts as user opc (on all VMs of the VM cluster)
[opc@exadbazure1 ~]$ vi .ssh/authorized_keys
#insert the public key and save the changes
[opc@exadbazure2 ~]$ vi .ssh/authorized_keys
#insert the public key and save the changes

#Add the source database hostname and IP information into the /etc/hosts file. As
root user (on all VMs of the VM cluster)
[root@exadbazure1 ~]# vi /etc/hosts
#add the following entries
aa.bb.sr.db onphost

[root@exadbazure2 ~]# vi /etc/hosts
#add the following entries
aa.bb.sr.db onphost
```

Step 4: Prepare the target database on ExaDB-D on Oracle Database@Azure

Prepare the Target Database, as SYS user:

```
-- on target CDB:
alter system set ENABLE_GOLDENGATE_REPLICATION=TRUE scope=both;

-- on target PDB: create ggadmin on target PDB and grant privileges
alter session set container=pdbtgt;
create user ggadmin identified by <password> default tablespace users temporary
tablespace temp;
alter user ggadmin quota 100M ON USERS;
grant connect, resource to ggadmin;
grant unlimited tablespace to ggadmin;
grant select any dictionary to ggadmin;
grant create view to ggadmin;
grant select any table to ggadmin;
grant insert any table to ggadmin;
grant update any table to ggadmin;
grant delete any table to ggadmin;
grant comment any table to ggadmin;
grant execute on dbms_lock to ggadmin;
exec dbms_goldengate_auth.GRANT_ADMIN_PRIVILEGE('ggadmin');
```

Step 5: Prepare the ZDM Service Host On-Premises

Add the source and target hostnames and IP addresses into the /etc/hosts file. As root user:

```
[root@zdmhost ~]# vi /etc/hosts
#add the following entries
ta.db.oa.1 exadbazure1
ta.db.oa.2 exadbazure2
aa.bb.sr.db onphost
```

Test the SSH connectivity to the source and target database hosts:

```
[zdmuser@zdmhost ~]$ ssh -i /home/zdmuser/.ssh/id_rsa onpuser@onphost
[zdmuser@zdmhost ~]$ ssh -i /home/zdmuser/.ssh/id_rsa opc@exadbazure1
[zdmuser@zdmhost ~]$ ssh -i /home/zdmuser/.ssh/id_rsa opc@exadbazure2
```

Verify that TTY is disabled for the SSH-privileged user. If TTY is disabled, the following command returns the date from the remote host without any errors:

```
[zdmuser@zdmhost ~]$ ssh -oStrictHostKeyChecking=no -i /home/zdmuser/.ssh/id_rsa
onpuser@onphost "/usr/bin/sudo /bin/sh -c date"
[zdmuser@zdmhost ~]$ ssh -oStrictHostKeyChecking=no -i /home/zdmuser/.ssh/id_rsa
opc@exadbazure1 "/usr/bin/sudo /bin/sh -c date"
[zdmuser@zdmhost ~]$ ssh -oStrictHostKeyChecking=no -i /home/zdmuser/.ssh/id_rsa
opc@exadbazure2 "/usr/bin/sudo /bin/sh -c date"
```

These commands should execute without any prompting and return the date from the remote host.

Step 6: Create the Logical Online Migration Response File on the ZDM host

You'll find a template on the ZDM host at $ZDMHOME/rhp/zdm/template/zdm_template.rsp, briefly describing the parameters and their possible values. Here, we will create a new response file with the minimal parameters required. As zdmuser:

```
[zdmuser@zdmhost ~]$ vi /home/zdmuser/logical_online/logical_online.rsp
#add the following parameters and save the changes
# migration method
MIGRATION_METHOD=ONLINE_LOGICAL
DATA_TRANSFER_MEDIUM=NFS

# data pump
DATAPUMPSETTINGS_JOBMODE=SCHEMA
INCLUDEOBJECTS-1=owner:HR
DATAPUMPSETTINGS_METADATAREMAPS-
1=type:REMAP_TABLESPACE,oldValue:USERS,newValue:DATA
DATAPUMPSETTINGS_DATAPUMPPARAMETERS_EXPORTPARALLELISMDEGREE=2
DATAPUMPSETTINGS_DATAPUMPPARAMETERS_IMPORTPARALLELISMDEGREE=2
DATAPUMPSETTINGS_EXPORTDIRECTORYOBJECT_NAME=DUMP_DIR
DATAPUMPSETTINGS_IMPORTDIRECTORYOBJECT_NAME=DUMP_DIR

# on source and target db: select directory_name, directory_path from
dba_directories;
DATAPUMPSETTINGS_EXPORTDIRECTORYOBJECT_PATH=/azurefilesnfs/dumpfiles
DATAPUMPSETTINGS_IMPORTDIRECTORYOBJECT_PATH=/azurefilesnfs/dumpfiles
```

```
# source db (pdb)
SOURCEDATABASE_CONNECTIONDETAILS_HOST=onphost
SOURCEDATABASE_CONNECTIONDETAILS_PORT=1521
SOURCEDATABASE_CONNECTIONDETAILS_SERVICENAME=pdbsrc
SOURCEDATABASE_ADMINUSERNAME=SYSTEM
SOURCEDATABASE_GGADMINUSERNAME=ggadmin

# source cdb
SOURCECONTAINERDATABASE_CONNECTIONDETAILS_HOST=onphost
SOURCECONTAINERDATABASE_CONNECTIONDETAILS_PORT=1521
SOURCECONTAINERDATABASE_CONNECTIONDETAILS_SERVICENAME=oradb
SOURCECONTAINERDATABASE_ADMINUSERNAME=SYSTEM
SOURCECONTAINERDATABASE_GGADMINUSERNAME=c##ggadmin

# target db (pdb)
TARGETDATABASE_CONNECTIONDETAILS_HOST=exadbazure1
TARGETDATABASE_CONNECTIONDETAILS_PORT=1521
TARGETDATABASE_CONNECTIONDETAILS_SERVICENAME=test.ocitestvm.ocitestvnt.test.com
TARGETDATABASE_ADMINUSERNAME=SYSTEM
TARGETDATABASE_GGADMINUSERNAME=ggadmin

# oci cli
OCIAUTHENTICATIONDETAILS_USERPRINCIPAL_USERID=ocid1.user.oc1..aaaaaaaa
OCIAUTHENTICATIONDETAILS_USERPRINCIPAL_TENANTID=ocid1.tenancy.oc1..aaaaaaaa
OCIAUTHENTICATIONDETAILS_USERPRINCIPAL_FINGERPRINT=<fingerpring>
OCIAUTHENTICATIONDETAILS_USERPRINCIPAL_PRIVATEKEYFILE=/home/zdmuser/.oci/oci_api_ke
y.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 7: 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 /home/zdmuser/logical_online/logical_online.rsp \
-sourcenode onphost \
-sourcesid oradb \
-srcauth zdmauth \
-srcarg1 user:azureuser \
-srcarg2 identity_file:/home/zdmuser/.ssh/id_rsa \
-srcarg3 sudo_location:/usr/bin/sudo \
-targetnode exadbazure1 \
-tgtauth zdmauth \
-tgtarg1 user:opc \
-tgtarg2 identity_file:/home/zdmuser/.ssh/id_rsa \
-tgtarg3 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 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
...
Job ID: 1
User: zdmuser
Client: zdmhost
Job Type: "EVAL"
...
Current status: SUCCEEDED
Result file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-1-2024-06-10-
14:59:57.log"
Metrics file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-1-2024-06-10-
14:59:57.json"
...
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-
06-10-14:59:57.log
```

In case troubleshooting is required, please check the ZDM server log on the ZDM Service Host under the following location:

```
$ZDM_BASE/crsdata/<zdm_service_host>/rhp/zdmserver.log.0
```

Step 8: 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 is keeping 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, which in this case is ZDM_MONITOR_GG_LAG.

On the ZDM host as zdmuser:

```
[zdmuser@zdmhost ~]$ $ZDMHOME/bin/zdmcli migrate database \
-rsp /home/zdmuser/logical_online/logical_online.rsp \
-sourcenode onphost \
-sourcesid oradb \
-srcauth zdmauth \
-srcarg1 user:azureuser \
-srcarg2 identity_file:/home/zdmuser/.ssh/id_rsa \
-srcarg3 sudo_location:/usr/bin/sudo \
-targetnode exadbazure1 \
-tgtauth zdmauth \
-tgtarg1 user:opc \
-tgtarg2 identity_file:/home/zdmuser/.ssh/id_rsa \
-tgtarg3 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 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
...
Job ID: 1
User: zdmuser
Client: zdmhost
Job Type: "MIGRATE"
...
Current status: PAUSED
Current Phase: "ZDM_MONITOR_GG_LAG"
Result file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-2-2024-06-10-
15:03:57.log"
Metrics file path: "/home/zdmuser/zdm/zdmbase/chkbase/scheduled/job-2-2024-06-10-
15:03:57.json"
...

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
```

ORACLE

```
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 9: Complete the Migration

Resume the job from the previous step. On the ZDM host as zdmuser, resume the job and also 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 REPLICATs deployments and log files:

```
# connect to the docker container and switch to ogg user
[azureuser@ggdockervm ~]$ sudo docker exec -it ogg2113 /bin/bash
[root@d9156f6223c3 /]# su - ogg

# check the deployment files
[ogg@d9156f6223c3 ~]$ 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@d9156f6223c3 ~]$ view /u02/Deployment/var/log/extract.log
[ogg@d9156f6223c3 ~]$ view /u02/Deployment/var/log/replicat.log
```

Check the Deployment Status:

```
[ogg@d9156f6223c3 ~]$ /u01/ogg/bin/adminclient
Oracle GoldenGate Administration Client for Oracle
Version 21.13.0.0.0 OGGCORE_21.13.0.0.0OGGRU_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