TR-4987: Simplified, Automated Oracle Deployment on Azure NetApp Files with NFS

NetApp database solutions

NetApp
July 31, 2025

Table of Contents

TR-4987: Simplified, Automated Oracle Deployment on Azure NetApp Files with NFS . . . 1
- Purpose . . . 1
- Audience . . . 1
- Solution test and validation environment . . . 1
  - Architecture . . . 1
  - Hardware and software components . . . 2
  - Oracle database configuration in the lab environment . . . 3
  - Key factors for deployment consideration . . . 3
- Solution deployment . . . 3
  - Prerequisites for deployment . . . 3
  - Automation parameter files . . . 5
  - Parameter files configuration . . . 5
  - Playbook execution . . . 8
  - Post execution validation . . . 10
  - Migrate Oracle database to Azure . . . 17
  - Oracle backup, restore, and clone with SnapCenter . . . 17
- Where to find additional information . . . 18

TR-4987: Simplified, Automated Oracle Deployment on Azure NetApp Files with NFS

Allen Cao, Niyaz Mohamed, NetApp

This solution provides overview and details for automated Oracle deployment in Microsoft Azure NetApp Files as primary database storage with NFS protocol and Oracle database is deployed as container database with dNFS enabled.

Purpose

Running performance-intensive and latency-sensitive Oracle workloads in the cloud can be challenging. Azure NetApp Files (ANF) makes it easy for enterprise line-of-business (LOB) and storage professionals to migrate and run demanding Oracle workloads without code change. Azure NetApp Files is widely used as the underlying shared file-storage service in various scenarios, such as new deployment or migration (lift and shift) of Oracle databases from on-premises to Azure.

This documentation demonstrates the simplified deployment of Oracle databases in Azure NetApp files via NFS mounts using Ansible automation. The Oracle database deploys in a container database (CDB) and pluggable databases (PDB) configuration with Oracle dNFS protocol enabled to boost performance. Furthermore, the on-premises Oracle single instance database or PDB can be migrated into a newly deployed container database in Azure using automated PDB relocation methodology with minimal service interruption. It also provides information on fast Oracle database backup, restore, and clone with NetApp SnapCenter UI tool in Azure Cloud.

This solution addresses the following use cases:

- Automated Oracle container database deployment on Azure NetApp files
- Automated Oracle database migration between on-premisses and Azure cloud

Audience

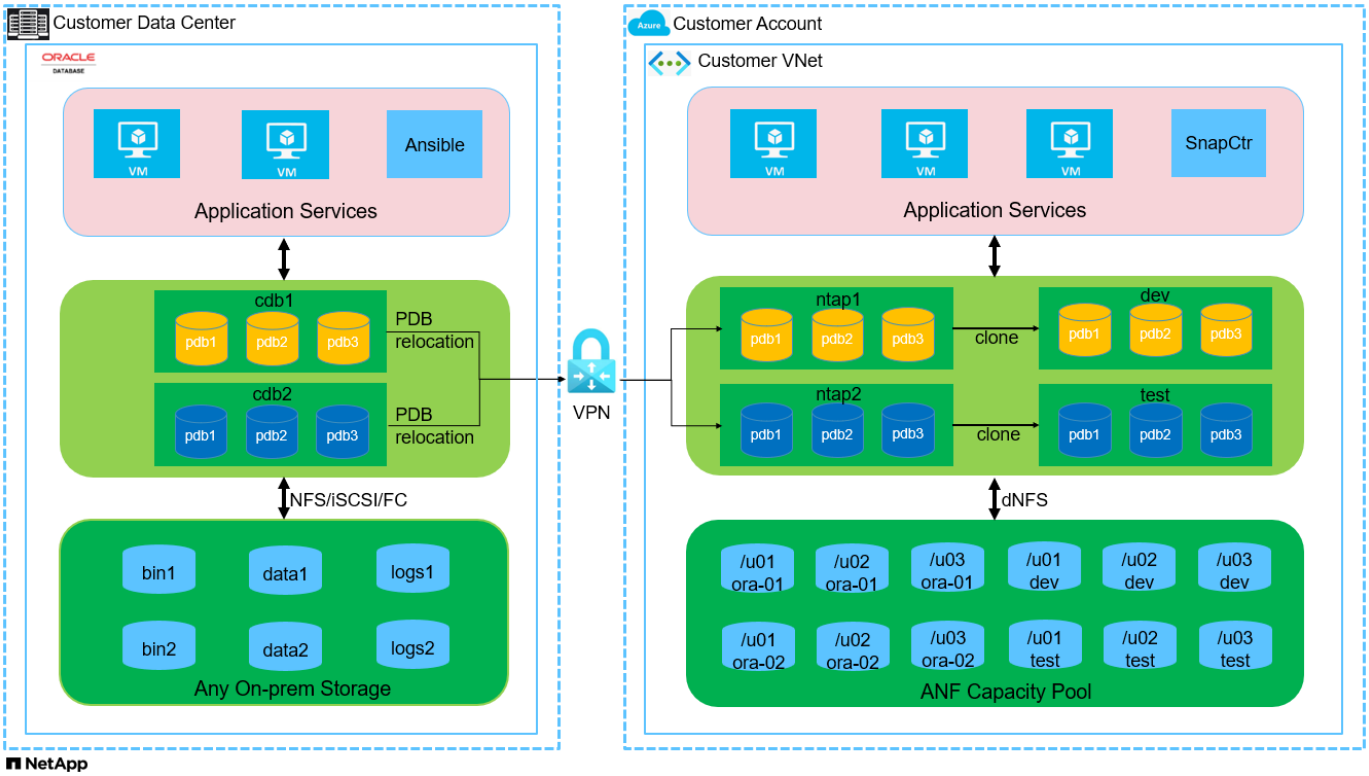
This solution is intended for the following people:

- A DBA who would like to deploy Oracle on Azure NetApp Files.
- A database solution architect who would like to test Oracle workloads on Azure NetApp Files.
- A storage administrator who would like to deploy and manage an Oracle database on Azure NetApp Files.
- An application owner who would like to stand up an Oracle database on Azure NetApp Files.

Solution test and validation environment

The testing and validation of this solution were performed in a lab setting that might not match the final deployment environment. See the section Key factors for deployment consideration for more information.

Architecture



Hardware and software components

Hardware		
Azure NetApp Files	Current offering in Azure by Microsoft	One capacity pool with Premium service level
Azure VM for DB server	Standard_B4ms - 4 vCPUs, 16GiB	Two Linux virtual machine instances for concurrent deployment
Azure VM for SnapCenter	Standard_B4ms - 4 vCPUs, 16GiB	One Windows virtual machine instance
Software		
RedHat Linux	RHEL Linux 8.6 (LVM) - x64 Gen2	Deployed RedHat subscription for testing
Windows Server	2022 DataCenter; Azure Edition Hotpatch - x64 Gen2	Hosting SnapCenter server
Oracle Database	Version 19.18	Applied RU patch p34765931_190000_Linux-x86-64.zip
Oracle OPatch	Version 12.2.0.1.36	Latest patch p6880880_190000_Linux-x86-64.zip
SnapCenter Server	Version 5.0	Workgroup deployment
Open JDK	Version java-11-openjdk	SnapCenter plugin requirement on DB VMs

NFS	Version 3.0	Oracle dNFS enabled
Ansible	core 2.16.2	Python 3.6.8

Oracle database configuration in the lab environment

Server	Database	DB Storage
ora-01	NTAP1(NTAP1_PDB1,NTAP1_PDB2,NTAP1_PDB3)	/u01, /u02, /u03 NFS mounts on ANF capacity pool
ora-02	NTAP2(NTAP2_PDB1,NTAP2_PDB2,NTAP2_PDB3)	/u01, /u02, /u03 NFS mounts on ANF capacity pool

Key factors for deployment consideration

- **Oracle database storage layout.** In this automated Oracle deployment, we provision three database volumes for each database to host Oracle binary, data, and logs by default. The volumes are mounted on Oracle DB server as /u01 - binary, /u02 - data, /u03 - logs via NFS. Dual control files are configured on /u02 and /u03 mount points for redundancy.
- **Multiple DB servers deployment.** The automation solution can deploy an Oracle container database to multiple DB servers in a single Ansible playbook run. Regardless of the number of DB servers, the playbook execution remains the same. You can deploy multiple container databases to a single VM instance by repeating the deployment with different database instance IDs (Oracle SID). But ensure there is sufficient memory on the host to support deployed databases.
- **dNFS configuration.** By using dNFS (available since Oracle 11g), an Oracle database running on an Azure Virtual Machine can drive significantly more I/O than the native NFS client. Automated Oracle deployment configures dNFS on NFSv3 by default.
- **Allocate large size volume to speed up deployment.** ANF file system IO throughput is regulated based on the size of volume. For initial deployment, allocate large size volumes can speed up the deployment. The volumes subsequently can be downsized dynamically without application impact.
- **Database backup.** NetApp provides a SnapCenter software suite for database backup, restore, and cloning with a user-friendly UI interface. NetApp recommends implementing such a management tool to achieve fast (under a minute) snapshot backup, quick (minutes) database restore, and database clone.

Solution deployment

The following sections provide step-by-step procedures for automated Oracle 19c deployment and database migration on Azure NetApp Files with directly mounted database volumes via NFS to Azure VMs.

Prerequisites for deployment

Deployment requires the following prerequisites.

1. An Azure account has been set up, and the necessary VNet and network segments have been created within your Azure account.
2. From the Azure cloud portal, deploy Azure Linux VMs as Oracle DB servers. Create an Azure NetApp Files capacity pool and database volumes for Oracle database. Enable VM SSH private/public key authentication for azureuser to DB servers. See the architecture diagram in the previous section for details about the environment setup. Also referred to Step-by-Step Oracle deployment procedures on Azure VM and Azure NetApp Files for detailed information.

   > For Azure VMs deployed with local disk redundancy, ensure that you have allocated at least 128G in the VM root disk to have sufficient space to stage Oracle installation files and add OS swap file. Expand /tmplv and /rootlv OS partition accordingly. Add 1G free space to rootvg-homelv if it is less than 1G. Ensure the database volume naming follows the VMname-u01, VMname-u02, and VMname-u03 convention.

   ```
   sudo lvresize -r -L +20G /dev/mapper/rootvg-rootlv
   ```

   ```
   sudo lvresize -r -L +10G /dev/mapper/rootvg-tmplv
   ```

   ```
   sudo lvresize -r -L +1G /dev/mapper/rootvg-homelv
   ```

3. From the Azure cloud portal, provision a Windows server to run the NetApp SnapCenter UI tool with the latest version. Refer to the following link for details: Install the SnapCenter Server
4. Provision a Linux VM as the Ansible controller node with the latest version of Ansible and Git installed. Refer to the following link for details: Getting Started with NetApp solution automation^ in section -
   `Setup the Ansible Control Node for CLI deployments on RHEL / CentOS` or
   `Setup the Ansible Control Node for CLI deployments on Ubuntu / Debian`.

   > The Ansible controller node can locate either on-premisses or in Azure cloud as far as it can reach Azure DB VMs via ssh port.

5. Clone a copy of the NetApp Oracle deployment automation toolkit for NFS.

   ```
   git clone https://bitbucket.ngage.netapp.com/scm/ns-
   bb/na_oracle_deploy_nfs.git
   ```

6. Stage following Oracle 19c installation files on Azure DB VM /tmp/archive directory with 777 permission.

```
installer_archives:
  - "LINUX.X64_193000_db_home.zip"
  - "p34765931_190000_Linux-x86-64.zip"
  - "p6880880_190000_Linux-x86-64.zip"
```

7. Watch the following video:

   Simplified and automated Oracle deployment on Azure NetApp Files with NFS

Automation parameter files

Ansible playbook executes database installation and configuration tasks with predefined parameters. For this Oracle automation solution, there are three user-defined parameter files that need user input before playbook execution.

- hosts - define targets that the automation playbook is running against.
- vars/vars.yml - the global variable file that defines variables that apply to all targets.
- host_vars/host_name.yml - the local variable file that defines variables that apply only to a named target. In our use case, these are the Oracle DB servers.

In addition to these user-defined variable files, there are several default variable files that contain default parameters that do not require change unless necessary. The following sections show how to configure the user-defined variable files.

Parameter files configuration

1. Ansible target `hosts` file configuration:

```
# Enter Oracle servers names to be deployed one by one, follow by
each Oracle server public IP address, and ssh private key of admin
user for the server.
[oracle]
ora_01 ansible_host=10.61.180.21 ansible_ssh_private_key_file
=ora_01.pem
ora_02 ansible_host=10.61.180.23 ansible_ssh_private_key_file
=ora_02.pem
```

1. Global `vars/vars.yml` file configuration

```
#################################################################################
##
###### Oracle 19c deployment user configuration variables
######
###### Consolidate all variables from ONTAP, linux and oracle
######
#################################################################################
##

############################################
### ONTAP env specific config variables ###
############################################

# Prerequisite to create three volumes in NetApp ONTAP storage from
System Manager or cloud dashboard with following naming convention:
# db_hostname_u01 - Oracle binary
# db_hostname_u02 - Oracle data
# db_hostname_u03 - Oracle redo
# It is important to strictly follow the name convention or the
automation will fail.


############################################
### Linux env specific config variables ###
############################################

redhat_sub_username: XXXXXXXX
redhat_sub_password: XXXXXXXX


#####################################################
### DB env specific install and config variables ###
#####################################################

# Database domain name
db_domain: solutions.netapp.com

# Set initial password for all required Oracle passwords. Change
them after installation.
initial_pwd_all: XXXXXXXX
```

1. Local DB server `host_vars/host_name.yml` configuration such as ora_01.yml, ora_02.yml …

```
# User configurable Oracle host specific parameters

# Enter container database SID. By default, a container DB is
created with 3 PDBs within the CDB
oracle_sid: NTAP1

# Enter database shared memory size or SGA. CDB is created with SGA
at 75% of memory_limit, MB. The grand total of SGA should not exceed
75% available RAM on node.
memory_limit: 8192

# Local NFS lif ip address to access database volumes
nfs_lif: 172.30.136.68
```

Playbook execution

There are a total of five playbooks in the automation toolkit. Each performs different task blocks and serves different purposes.

```
0-all_playbook.yml - execute playbooks from 1-4 in one playbook run.
1-ansible_requirements.yml - set up Ansible controller with required
libs and collections.
2-linux_config.yml - execute Linux kernel configuration on Oracle DB
servers.
4-oracle_config.yml - install and configure Oracle on DB servers and
create a container database.
5-destroy.yml - optional to undo the environment to dismantle all.
```

There are three options to run the playbooks with the following commands.

1. Execute all deployment playbooks in one combined run.

```
ansible-playbook -i hosts 0-all_playbook.yml -u azureuser -e
@vars/vars.yml
```

2. Execute playbooks one at a time with the number sequence from 1-4.

```
ansible-playbook -i hosts 1-ansible_requirements.yml -u azureuser -e
@vars/vars.yml
```

```
ansible-playbook -i hosts 2-linux_config.yml -u azureuser -e
@vars/vars.yml
```

```
ansible-playbook -i hosts 4-oracle_config.yml -u azureuser -e
@vars/vars.yml
```

3. Execute 0-all_playbook.yml with a tag.

```
ansible-playbook -i hosts 0-all_playbook.yml -u azureuser -e
@vars/vars.yml -t ansible_requirements
```

```
ansible-playbook -i hosts 0-all_playbook.yml -u azureuser -e
@vars/vars.yml -t linux_config
```

```
ansible-playbook -i hosts 0-all_playbook.yml -u azureuser -e
@vars/vars.yml -t oracle_config
```

4. Undo the environment

```
ansible-playbook -i hosts 5-destroy.yml -u azureuser -e
@vars/vars.yml
```

Post execution validation

After the playbook run, login to the Oracle DB server VM to validate that Oracle is installed and configured and a container database is created successfully. Following is an example of Oracle database validation on host ora-01.

1. Validate NFS mounts

```
[azureuser@ora-01 ~]$ cat /etc/fstab

#
# /etc/fstab
# Created by anaconda on Thu Sep 14 11:04:01 2023
#
# Accessible filesystems, by reference, are maintained under
'/dev/disk/'.
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for
more info.
#
# After editing this file, run 'systemctl daemon-reload' to update
systemd
# units generated from this file.
#
/dev/mapper/rootvg-rootlv /                       xfs     defaults
0 0
UUID=268633bd-f9bb-446d-9a1d-8fca4609a1e1 /boot
xfs     defaults        0 0
UUID=89D8-B037          /boot/efi               vfat
defaults,uid=0,gid=0,umask=077,shortname=winnt 0 2
/dev/mapper/rootvg-homelv /home                   xfs     defaults
0 0
/dev/mapper/rootvg-tmplv /tmp                     xfs     defaults
0 0
/dev/mapper/rootvg-usrlv /usr                     xfs     defaults
0 0
/dev/mapper/rootvg-varlv /var                     xfs     defaults
0 0
/mnt/swapfile swap swap defaults 0 0
172.30.136.68:/ora-01-u01 /u01 nfs
rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536 0 0
172.30.136.68:/ora-01-u02 /u02 nfs
rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536 0 0
172.30.136.68:/ora-01-u03 /u03 nfs
rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536 0 0

[azureuser@ora-01 ~]$ df -h
Filesystem                Size  Used Avail Use% Mounted on
devtmpfs                  7.7G     0  7.7G   0% /dev
```

```
tmpfs                       7.8G     0  7.8G   0% /dev/shm
tmpfs                       7.8G  8.6M  7.7G   1% /run
tmpfs                       7.8G     0  7.8G   0% /sys/fs/cgroup
/dev/mapper/rootvg-rootlv    22G   17G  5.8G  74% /
/dev/mapper/rootvg-usrlv     10G  2.0G  8.1G  20% /usr
/dev/mapper/rootvg-varlv    8.0G  890M  7.2G  11% /var
/dev/sda1                   496M  106M  390M  22% /boot
/dev/mapper/rootvg-homelv  1014M   40M  975M   4% /home
/dev/sda15                  495M  5.9M  489M   2% /boot/efi
/dev/mapper/rootvg-tmplv     12G  8.4G  3.7G  70% /tmp
tmpfs                       1.6G     0  1.6G   0% /run/user/54321
172.30.136.68:/ora-01-u01   500G   11G  490G   3% /u01
172.30.136.68:/ora-01-u03   250G  1.2G  249G   1% /u03
172.30.136.68:/ora-01-u02   250G  7.1G  243G   3% /u02
tmpfs                       1.6G     0  1.6G   0% /run/user/1000
```

2. Validate Oracle listener

```
[azureuser@ora-01 ~]$ sudo su
[root@ora-01 azureuser]# su - oracle
Last login: Thu Feb  1 16:13:44 UTC 2024
[oracle@ora-01 ~]$ lsnrctl status listener.ntap1

LSNRCTL for Linux: Version 19.0.0.0.0 - Production on 01-FEB-2024
16:25:37

Copyright (c) 1991, 2022, Oracle.  All rights reserved.

Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=ora-
01.internal.cloudapp.net)(PORT=1521)))
STATUS of the LISTENER
------------------------
Alias                     LISTENER.NTAP1
Version                   TNSLSNR for Linux: Version 19.0.0.0.0 -
Production
Start Date                01-FEB-2024 16:13:49
Uptime                    0 days 0 hr. 11 min. 49 sec
Trace Level               off
Security                  ON: Local OS Authentication
SNMP                      OFF
Listener Parameter File
/u01/app/oracle/product/19.0.0/NTAP1/network/admin/listener.ora
Listener Log File         /u01/app/oracle/diag/tnslsnr/ora-
01/listener.ntap1/alert/log.xml
Listening Endpoints Summary...
```

```
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=ora-
01.hr2z2nbmhnqutdsxgscjtuxizd.jx.internal.cloudapp.net)(PORT=1521)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(KEY=EXTPROC1521)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcps)(HOST=ora-
01.hr2z2nbmhnqutdsxgscjtuxizd.jx.internal.cloudapp.net)(PORT=5500))(
Security=(my_wallet_directory=/u01/app/oracle/product/19.0.0/NTAP1/a
dmin/NTAP1/xdb_wallet))(Presentation=HTTP)(Session=RAW))
Services Summary...
Service "104409ac02da6352e063bb891eacf34a.solutions.netapp.com" has
1 instance(s).
  Instance "NTAP1", status READY, has 1 handler(s) for this
service...
Service "104412c14c2c63cae063bb891eacf64d.solutions.netapp.com" has
1 instance(s).
  Instance "NTAP1", status READY, has 1 handler(s) for this
service...
Service "1044174670ad63ffe063bb891eac6b34.solutions.netapp.com" has
1 instance(s).
  Instance "NTAP1", status READY, has 1 handler(s) for this
service...
Service "NTAP1.solutions.netapp.com" has 1 instance(s).
  Instance "NTAP1", status READY, has 1 handler(s) for this
service...
Service "NTAP1XDB.solutions.netapp.com" has 1 instance(s).
  Instance "NTAP1", status READY, has 1 handler(s) for this
service...
Service "ntap1_pdb1.solutions.netapp.com" has 1 instance(s).
  Instance "NTAP1", status READY, has 1 handler(s) for this
service...
Service "ntap1_pdb2.solutions.netapp.com" has 1 instance(s).
  Instance "NTAP1", status READY, has 1 handler(s) for this
service...
Service "ntap1_pdb3.solutions.netapp.com" has 1 instance(s).
  Instance "NTAP1", status READY, has 1 handler(s) for this
service...
The command completed successfully
```

3. Validate Oracle database and dNFS

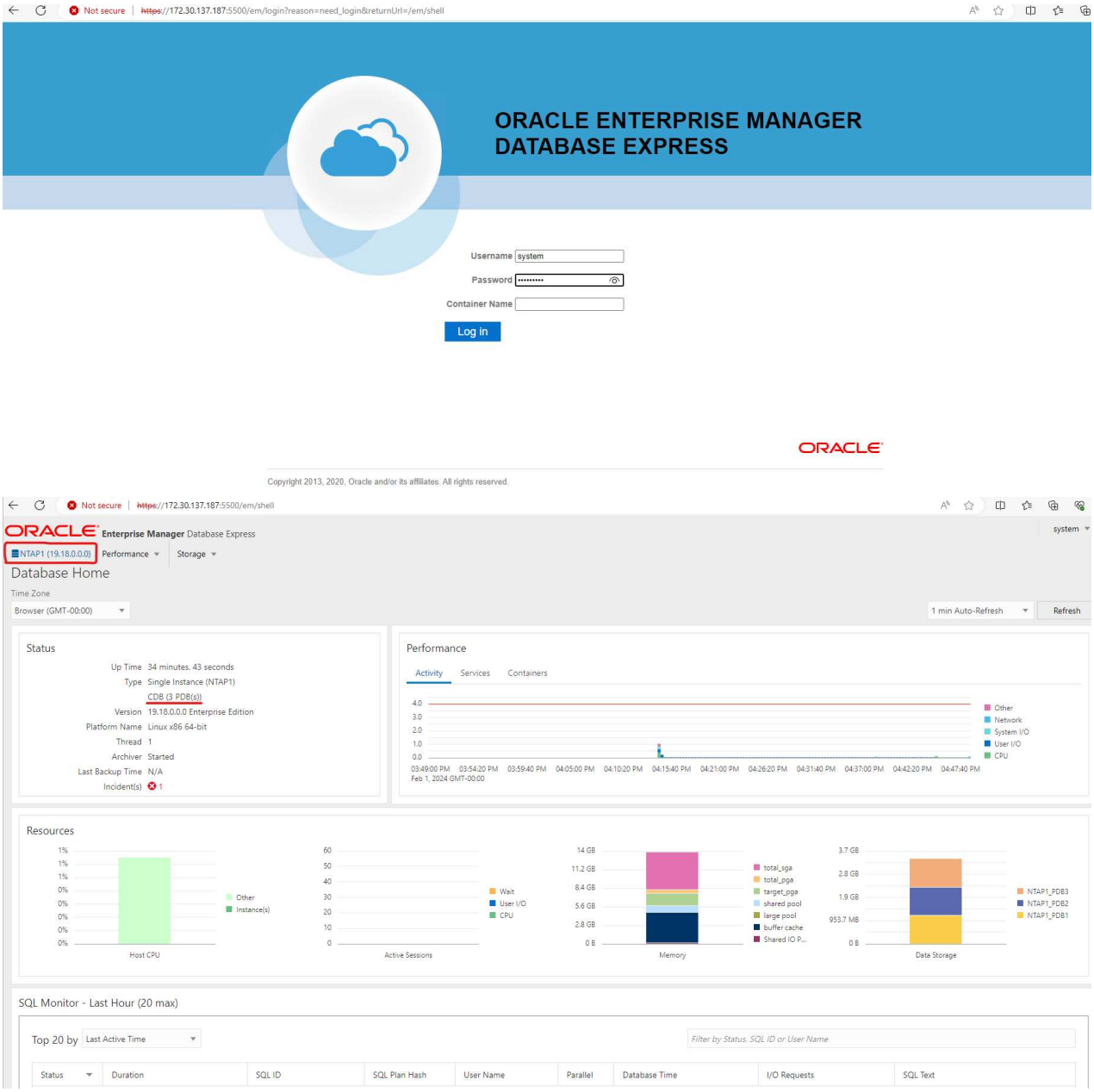
```
[oracle@ora-01 ~]$ cat /etc/oratab
#
# This file is used by ORACLE utilities.  It is created by root.sh
# and updated by either Database Configuration Assistant while
creating
# a database or ASM Configuration Assistant while creating ASM
```

```
instance.

# A colon, ':', is used as the field terminator.  A new line
terminates
# the entry.  Lines beginning with a pound sign, '#', are comments.
#
# Entries are of the form:
#   $ORACLE_SID:$ORACLE_HOME:<N|Y>:
#
# The first and second fields are the system identifier and home
# directory of the database respectively.  The third field indicates
# to the dbstart utility that the database should , "Y", or should
not,
# "N", be brought up at system boot time.
#
# Multiple entries with the same $ORACLE_SID are not allowed.
#
#
NTAP1:/u01/app/oracle/product/19.0.0/NTAP1:Y


[oracle@ora-01 ~]$ sqlplus / as sysdba

SQL*Plus: Release 19.0.0.0.0 - Production on Thu Feb 1 16:37:51 2024
Version 19.18.0.0.0

Copyright (c) 1982, 2022, Oracle.  All rights reserved.


Connected to:
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 -
Production
Version 19.18.0.0.0

SQL> select name, open_mode, log_mode from v$database;

NAME      OPEN_MODE            LOG_MODE
--------- -------------------- ------------
NTAP1     READ WRITE           ARCHIVELOG

SQL> show pdbs

    CON_ID CON_NAME                       OPEN MODE  RESTRICTED
---------- ------------------------------ ---------- ----------
         2 PDB$SEED                       READ ONLY  NO
         3 NTAP1_PDB1                     READ WRITE NO
         4 NTAP1_PDB2                     READ WRITE NO
```

```
         5 NTAP1_PDB3                     READ WRITE NO
SQL> select name from v$datafile;

NAME
------------------------------------------------------------------------
------------
/u02/oradata/NTAP1/system01.dbf
/u02/oradata/NTAP1/sysaux01.dbf
/u02/oradata/NTAP1/undotbs01.dbf
/u02/oradata/NTAP1/pdbseed/system01.dbf
/u02/oradata/NTAP1/pdbseed/sysaux01.dbf
/u02/oradata/NTAP1/users01.dbf
/u02/oradata/NTAP1/pdbseed/undotbs01.dbf
/u02/oradata/NTAP1/NTAP1_pdb1/system01.dbf
/u02/oradata/NTAP1/NTAP1_pdb1/sysaux01.dbf
/u02/oradata/NTAP1/NTAP1_pdb1/undotbs01.dbf
/u02/oradata/NTAP1/NTAP1_pdb1/users01.dbf

NAME
------------------------------------------------------------------------
------------
/u02/oradata/NTAP1/NTAP1_pdb2/system01.dbf
/u02/oradata/NTAP1/NTAP1_pdb2/sysaux01.dbf
/u02/oradata/NTAP1/NTAP1_pdb2/undotbs01.dbf
/u02/oradata/NTAP1/NTAP1_pdb2/users01.dbf
/u02/oradata/NTAP1/NTAP1_pdb3/system01.dbf
/u02/oradata/NTAP1/NTAP1_pdb3/sysaux01.dbf
/u02/oradata/NTAP1/NTAP1_pdb3/undotbs01.dbf
/u02/oradata/NTAP1/NTAP1_pdb3/users01.dbf

19 rows selected.

SQL> select name from v$controlfile;

NAME
------------------------------------------------------------------------
------------
/u02/oradata/NTAP1/control01.ctl
/u03/orareco/NTAP1/control02.ctl

SQL> select member from v$logfile;

MEMBER
------------------------------------------------------------------------
------------
/u03/orareco/NTAP1/onlinelog/redo03.log
```

```
/u03/orareco/NTAP1/onlinelog/redo02.log
/u03/orareco/NTAP1/onlinelog/redo01.log

SQL> select svrname, dirname, nfsversion from v$dnfs_servers;

SVRNAME
----------------------------------------------------------------
------------
DIRNAME
----------------------------------------------------------------
------------
NFSVERSION
----------------
172.30.136.68
/ora-01-u02
NFSv3.0

172.30.136.68
/ora-01-u03
NFSv3.0

SVRNAME
----------------------------------------------------------------
------------
DIRNAME
----------------------------------------------------------------
------------
NFSVERSION
----------------

172.30.136.68
/ora-01-u01
NFSv3.0
```

4. Login to Oracle Enterprise Manager Express to validate database.



Migrate Oracle database to Azure

Oracle database migration from on-premises to the cloud is a heavy-lifting. Using the right strategy and automation can smooth the process and minimize service interruption and downtime. Follow this detailed instruction Database migration from on-premises to Azure cloud to guide your database migration journey.

Oracle backup, restore, and clone with SnapCenter

NetApp recommends SnapCenter UI tool to manage Oracle database deployed in Azure cloud. Please refer to TR-4988: Oracle Database Backup, Recovery, and Clone on ANF with SnapCenter for details.

Where to find additional information

To learn more about the information described in this document, review the following documents and/or websites:

- Oracle Database Backup, Recovery, and Clone on ANF with SnapCenter

  Oracle Database Backup, Recovery, and Clone on ANF with SnapCenter

- Azure NetApp Files

  https://azure.microsoft.com/en-us/products/netapp

- Deploying Oracle Direct NFS

  https://docs.oracle.com/en/database/oracle/oracle-database/19/ladbi/deploying-dnfs.html#GUID-D06079DB-8C71-4F68-A1E3-A75D7D96DCE2

- Installing and Configuring Oracle Database Using Response Files

  https://docs.oracle.com/en/database/oracle/oracle-database/19/ladbi/installing-and-configuring-oracle-database-using-response-files.html#GUID-D53355E9-E901-4224-9A2A-B882070EDDF7

Copyright information

Trademark information