Cloning database backup

SnapManager Oracle

NetApp
April 15, 2021

Table of Contents

Cloning database backup

If you clone a database, you can perform tasks such as test an upgrade to a database without affecting the database in production, duplicate a master installation to several training systems, or duplicate a master installation as a base installation to other servers, which have similar requirements.

You can perform the following tasks related to cloning:

- Clone a database from an existing backup.
- Clone a database in its current state, which enables you to create the backup and the clone in one procedure.
- Clone a database and use custom plug-in scripts, which run before or after the clone operation.
- Clone a database to the same host on which the database resides.
- Clone a database by using archive log files from the external archive log location.
- Clone a database to an alternate host.
- View a list of clones.
- View detailed clone information.
- Delete clones.

What Cloning is

You can clone a database to create an exact replica of the original database. You can create the clone from a full backup or from the current state of the database.

Some of the advantages of creating a clone by using SnapManager are as follows:

Advantages	Details
Speed	The SnapManager clone operation uses the FlexClone feature available with Data ONTAP. This enables you to quickly clone large data volumes.
Space efficiency	When you create a clone by using SnapManager, space is needed only for the changes between the backup and the clone. A SnapManager clone is a writable Snapshot copy of the original database and can grow as needed. In contrast, a physical clone of the database requires that you have enough space available to copy the entire database.

Virtual copy	You can use the cloned database as if it were the original database. For example, you can use a clone for testing, platform and update checks, multiple simulations against a large data set, and remote office testing and staging. Changes to the clone do not affect the original database. After the database is cloned, the cloned database is fully operational.
Simplicity	You can clone a database to any host by using SnapManager commands.

You must ensure that the following prerequisites are met before a database can be cloned:

- Delete the spfile<SID>.ora file from $ORACLE_HOME\database.
- Delete the init<SID>.ora file from $ORACLE_HOME\database.
- Delete Oracle dump destinations that are specified in the clone specification file.
- Delete the Oracle control files that are specified in the clone specification file.
- Delete the Oracle redo log files that are specified in the clone specification file.

You must give the clone a new system identifier. You cannot simultaneously run two databases with the same system identifier on the same host. You can have a clone on a different host using the same system identifier. You can either give the clone a label or let SnapManager create a label by using the system identifier, date, and time the clone was created.

When you enter a label, you should not include spaces or special characters.

As part of the cloning process, SnapManager creates the necessary Oracle files and parameters for the cloned database. An example of a necessary Oracle file is init<SID>.ora.

When you clone a database, SnapManager creates a new init<SID>.ora file for the database in the $ORACLE_HOME\database directory.

When SnapManager clones the storage for a database, it also creates a new file system mountpoint, but does not change the directory structure under the mountpoint from the SnapManager CLI. However, from the SnapManager GUI, you can change the directory structure and the metadata of the file system.

You can clone a database backup to the host in which the database resides or to an alternate host.

If the database you cloned was using a spfile, SnapManager creates an spfile for the clone. It places this file in the $ORACLE_HOME\database directory and creates the directory structure for the diagnostic files. The file name is spfile <SID>.ora.

Cloning methods

You can clone a database using one of two methods. The method you choose affects the clone create operation.

The following table describes the cloning methods and their effect on the clone create operation and its -reserve option. A LUN can be cloned using either method.

Cloning method
Description
clone create -reserve
LUN cloning
A new clone LUN is created within the same volume.
When -reserve for a LUN is set to yes, space is reserved for the full LUN size within the volume.
Volume cloning
A new FlexClone is created and the clone LUN exists within the new clone volume. Uses FlexClone technology.
When -reserve for a volume is set to yes, space is reserved for the full volume size within the aggregate.

Creating clone specifications

SnapManager for Oracle uses a clone specification XML file, which includes the mappings, options, and parameters for use in the clone operation. SnapManager uses this information to determine where to place the files it clones and how to handle diagnostic information, control files, parameters, and so on.

You can create the clone specification file by using the SnapManager graphical user interface (GUI), command-line interface (CLI), or a text editor.

When you create the clone specification file by using a text editor, you must save it as a .xml file. You can use this XML file for other clone operations.

You can also create a clone specification template and then customize it. You can use the smo clone template command or in the GUI, use the Clone wizard.

SnapManager for Oracle adds a version string to any clone specification template that it generates. SnapManager for Oracle assumes the latest version for any clone specification file that lacks a version string.

If you want to perform remote cloning, do not change the default locations of the data files, redo log files, and control files in the clone specification file. If you change the default location, SnapManager fails to create the clone or creates the clone on a database that does not support Snapshot capability. Therefore, the automatic creation of profile fails.

Though mount point and ASM disk group information are editable from the GUI, you can only change the file name and not the file locations.

You can execute a task multiple times, either with the same or different parameter and value combinations.

1. Open a text file and enter text as shown in the following example:

```
<clone-specification xmlns="http://www.example.com>
     <storage-specification/>
     <database-specification/>
 </clone-specification>
```

2. In the storage specification component, enter the mount points for the data files.

   The storage specification lists the locations for the new storage created for the clone such as data file mount points and raw devices. These items must be mapped from the source to the destination.

   The following example displays the data file mount point syntax that you use in the clone specification:

```
<mountpoint>
     <source>\mnt\path\source_data file_mountpoint</source>
     <destination>\mnt\path\target_data file_mountpoint</destination>
</mountpoint>
```

3. In the database specification component, identify the control file information as a list of the control files that you want created for the clone.

   The database specification specifies the database options for the clone such as control files, redo logs, archive logs, and Oracle parameters.

   The following example displays the control file syntax that you use in clone specification:

```
<controlfiles>
     <file>\mnt\path\clonename\control\control01.ctl</file>
     <file>\mnt\path\clonename\control\control02.ctl</file>
</controlfiles>
```

4. Specify the redo log structure for the clone.

   The following example displays the redo log directory structure for cloning:

```
<redologs>
     <redogroup>
          <file>\mnt\path\clonename\redo\redo01.log</file>
          <number>1</number>
          <size unit="M">100</size>
     </redogroup>
     <redogroup>
         <file>\mnt\path\clonename\redo\redo02.log</file>
         <number>2</number>
         <size unit="M">100</size>
     </redogroup>
</redologs>
```

5. Specify the Oracle parameters that should be set to different values in the cloned database. If you are using Oracle 10, you must specify the following parameters:
   - Background dump
   - Core dump
   - User dump
   - (Optional) Archive logs

   If the parameter values are not set correctly, the clone operation is stopped and you receive an error message.

   If you do not specify the location where archive logs are stored, SnapManager creates the clone in noarchivelog mode. SnapManager copies this parameter information into the init.ora file of the clone.

   + The following example displays the parameter syntax that you use in clone specification:

   +

```
<parameters>
     <parameter>
          <name>log_archive_dest_1</name>
          <value>LOCATION=\mnt\path\clonename\archive</value>
     </parameter>
</parameters>
```

   + You can use a default value by using a default element within the parameter element. In the following example, the os_authentication_prefix parameter will take the default value because the default element is specified:

   +

```
<parameters>
     <parameter>
          <name>os_authent_prefix</name>
          <default></default>
     </parameter>
</parameters>
```

+ You can specify an empty string as the value for a parameter by using an empty element. In the following example, the os_authentication_prefix will be set to an empty string:

+

```
<parameters>
     <parameter>
          <name>os_authent_prefix</name>
          <value></value>
     </parameter>
</parameters>
```

+ NOTE: You can use the value from the source database's init.ora file for the parameter by not specifying any element.

+ If a parameter has multiple values, then you can provide the parameter values separated by commas. For example, if you want to move the data files from one location to another, then you can use the db_file_name_convert parameter and specify the data file paths separated by commas as seen in the following example:

+

```
<parameters>
     <parameter>
          <name>db_file_name_convert</name>
          <value>>\mnt\path\clonename\data
file1,\mnt\path\clonename\data file2</value>
     </parameter>
</parameters>
```

+ If you want to move the log files from one location to another, then you can use the log_file_name_convert parameter and specify the log file paths separated by commas, as seen in the followowing example:

+

```
<parameters>
     <parameter>
          <name>log_file_name_convert</name>

<value>>\mnt\path\clonename\archivle1,\mnt\path\clonename\archivle2</val
ue>
     </parameter>
</parameters>
```

6. Optional: Specify arbitrary SQL statements to execute against the clone when it is online.

   You can use the SQL statements to perform tasks such as re-creating the temp files in the cloned database.

   > You must ensure that a semicolon is not included at the end of the SQL statement.

   The following is a sample SQL statement that you execute as part of the clone operation:

```
<sql-statements>
   <sql-statement>
     ALTER TABLESPACE TEMP ADD
     TEMPFILE 'E:\path\clonename\temp_user01.dbf'
     SIZE 41943040 REUSE AUTOEXTEND ON NEXT 655360
     MAXSIZE 32767M
   </sql-statement>
</sql-statements>
```

Clone specification example

The following example displays the clone specification structure, including both the storage and database specification components, for a Windows environment:

```
<clone-specification xmlns="http://www.example.com>

<storage-specification>
    <storage-mapping>
        <mountpoint>
            <source>D:\oracle\<SOURCE SID>_sapdata</source>
            <destination>D:\oracle\<TARGET SID>_sapdata</destination>
        </mountpoint>
    </storage-mapping>
</storage-specification>

<database-specification>
```

```
    <controlfiles>
        <file>D:\oracle\<TARGET SID>\origlogA\cntrl\cntrl<TARGET
SID>.dbf</file>
        <file>D:\oracle\<TARGET SID>\origlogB\cntrl\cntrl<TARGET
SID>.dbf</file>
        <file>D:\oracle\<TARGET SID>\sapdata1\cntrl\cntrl<TARGET
SID>.dbf</file>
     </controlfiles>

     <redologs>
        <redogroup>
            <file>D:\oracle\<TARGET SID>\origlogA\log_g11m1.dbf</file>
            <file>D:\oracle\<TARGET SID>\mirrlogA\log_g11m2.dbf</file>
            <number>1</number>
            <size unit="M">100</size>
        </redogroup>
        <redogroup>
            <file>D:\oracle\<TARGET SID>\origlogB\log_g12m1.dbf</file>
            <file>D:\oracle\<TARGET SID>\mirrlogB\log_g12m2.dbf</file>
            <number>2</number>
            <size unit="M">100</size>
        </redogroup>
        <redogroup>
            <file>D:\oracle\<TARGET SID>\origlogA\log_g13m1.dbf</file>
            <file>D:\oracle\<TARGET SID>\mirrlogA\log_g13m2.dbf</file>
            <number>3</number>
            <size unit="M">100</size>
        </redogroup>
        <redogroup>
            <file>D:\oracle\<TARGET SID>\origlogB\log_g14m1.dbf</file>
            <file>D:\oracle\<TARGET SID>\mirrlogB\log_g14m2.dbf</file>
            <number>4</number>
            <size unit="M">100</size>
       </redogroup>
    </redologs>

    <parameters>
        <parameter>
            <name>log_archive_dest</name>
            <value>LOCATION=>D:\oracle\<TARGET SID>\oraarch</value>
        </parameter>
        <parameter>
            <name>background_dump_dest</name>
            <value>D:\oracle\<TARGET SID>\saptrace\background</value>
        </parameter>
        <parameter>
```

```
            <name>core_dump_dest</name>
            <value>D:\oracle\<TARGET SID>\saptrace\background</value>
        </parameter>
        <parameter>
            <name>user_dump_dest</name>
            <value>D:\oracle\<TARGET SID>\saptrace\usertrace</value>
        </parameter>
    </parameters>
   </database-specification>
</clone-specification>
```

Related information

Cloning databases and using custom plug-in scripts

Cloning databases from backups

Cloning databases in the current state

Considerations for cloning a database to an alternate host

Cloning databases and using custom plug-in scripts

SnapManager provides a method for using your custom scripts before and after a clone operation occurs. For example, you might have created a custom script that validates a clone database SID and ensures the SID is allowed by your naming policy. Using the SnapManager clone plug-in, you can include your custom scripts and have them run automatically before or after a SnapManager clone operation.

1. View sample plug-in scripts.
2. Create a script from scratch or modify one of the sample plug-in scripts.

   Create your custom script according to SnapManager plug-in script guidelines.
3. Place your custom script in a specified directory location.
4. Update the clone specification XML file and include information about your custom script that should be used during the cloning process.
5. Using a SnapManager command, verify that the custom scripts are operational.
6. When you initiate the clone operation, include the script name and optional parameters.

Cloning databases from backups

You can clone a database from a backup by using the clone create command.

You must first create a clone specification file for the database. SnapManager creates the clone based on the information in this specification file.

You must give the clone a new Oracle system identifier (SID). You cannot run two databases with the same

SID simultaneously on the same host. You can have a clone on a different host that uses the same SID. To designate a unique name for the clone, use -label. If you do not use this option, SnapManager creates a unique name for the clone that includes the SID, date, and time.

After you clone a database, you might want to update your tnsnames.ora files on your client machines with the new cloned database connection information. The tnsnames.ora files are used to connect to an Oracle instance without having to specify the full database information. SnapManager does not update the tnsnames.ora files.

SnapManager always creates a backup including archive log files, if you are using the profile created with -include-with-online-backups. SnapManager allows you to clone only the full database backups.

SnapManager (3.2 or later) allows you to clone the backups containing the data files and archive log files.

If the archive log is available from an external location, you can specify the external location during cloning for recovering the cloned database to a consistent state. You must ensure that the external location is accessible by Oracle. Cloning of the archive log-only backups is not supported.

Though the archive log backup is created along with the online partial backup, you cannot create a database clone by using this backup.

When you specify the external archive log locations for recovering the cloned database to a consistent state, you must ensure that you include the external location names completely in uppercase. In the file system, the names of all the folders and subfolders must be in uppercase because the Oracle database translates the destination path to uppercase and expects the external destination paths, folder names, and subfolder names to be in uppercase. If you specify the external archive log destination paths in lowercase, database might not be able to identify the specified path, and fails to recover the cloned database.

You can clone the database backup from the external archive log file location only for a stand-alone database.

You can specify the -dump option as an optional parameter to collect the dump files after the successful or failed clone create operation.

Cloning datafile backup without archive log backup

When the data files backup does not include the archive log backup, SnapManager for Oracle clones the database based on the System Change Number (SCN) recorded during the backup. If the cloned database cannot be recovered, the Archived log file for thread <number> and change <SCN> required to complete recovery error message is displayed, even though SnapManager for Oracle continues to clone the database, and finally succeeds in creating the clone.

When cloning using the data files backup without including the archive log backup, SnapManager recovers the cloned database until the last archive log SCN, which is recorded during the backup.

1. Create a clone specification file.
2. To create a clone, enter the following command: smo clone create -backup-labelbackup_name -newsidnew_sid-labelclone_label-profileprofile_name-clonespecfull_path_to_clonespecfile [-taskspectaskspec] [-recover-from-location] path1 [,<path2>…]][-dump]

Related information

Cloning databases in the current state

Considerations for cloning a database to an alternate host

Creating clone specifications

The smo clone create command

Creating pretask, post-task, and policy scripts

Variables available in the task scripts for clone operation

Creating task scripts

Storing the task scripts

Cloning databases in the current state

You can create a backup and a clone of the database from the current state of the database by using a single command.

When you specify the profile with the -current option, SnapManager first creates a backup and then a clone from the current state of the database.

In the profile setting, if you have enabled the backup of data files and archive logs together for cloning, whenever you back up the online data files, the archive logs are also backed up. While cloning the database, SnapManager creates the data files backup along with the archive log backup and creates the database clone. If the archive log backup is not included, SnapManager does not create the archive log backup and therefore cannot create the clone of the database.

1. To clone the database in its current state, enter the following command: smo clone create -profileprofile_name-current -labelclone_name-clonespecclonespec.xml

   This command takes a full automatic backup (generating the backup label) and immediately makes a clone from that backup, using an existing clone specification that you want to use.

You can specify the -dump option as an optional parameter to collect the dump files after the successful or failed operations. The dump is collected for both the backup and clone operations.

Cloning database backups without resetlogs

SnapManager enables you to perform flexible cloning so that you can recover the cloned database manually to a desired point in time without opening the database by using resetlogs. You can also manually configure the cloned database as a Data Guard Standby database.

When you can select the -no-resetlogs option while creating the clone, SnapManager performs the following activities to create the cloned database:

1. Executes the preprocessing task activity, if specified, before starting the clone operation
2. Creates the cloned database with the user-specified SID
3. Executes the SQL statements issued against the cloned database.

   Only the SQL statements that can be executed in mount state are successfully executed.

4. Executes the post-processing task activity, if specified.

What tasks you need to do to recover the cloned database manually

1. Mount the archive log backups and recover the cloned database manually by using the archive log files from the mounted path.
2. After performing manual recovery, open the recovered cloned database with -resetlogs option.
3. Create temporary tablespaces, if required.
4. Run the DBNEWID utility.
5. Grant sysdba privilege to the credentials of the cloned database.

While cloning the database backups using the -no-resetlogs option, SnapManager leaves the cloned database in the mounted state for manual recovery.

The cloned database created with the -no-resetlogs option is not a complete database. Therefore you must not perform any SnapManager operations on this database, though SnapManager does not restrict you from performing any operations.

If you do not specify the -no-resetlogs option, SnapManager applies the archive log files, and opens the database with resetlogs.

1. Enter the following command: smo clone create -profileprofile_name [-backup-labelbackup_name | -backup -idbackup_id | current] -newsidnew_sid-clonespecfull_path_to_clonespecfile-no-resetlogs

   If you try to specify both -no-resetlogs and recover-from-location options, SnapManager does not allow you to specify both these options together, and displays the error message: SMO-04084: You must specify either one of the options: -no-resetlogs or -recover-from-location.

Example

```
smo clone create -profile product -backup-label full_offline -newsid
PROD_CLONE -clonespec prod_clonespec.xml -label prod_clone-reserve -no
-reset-logs
```

Considerations for cloning a database to an alternate host

Before you can clone to a host other than the one on which the database resides, there are some requirements that must be met.

The following table shows the source and target host setup requirements:

Prerequisite set up	Requirement
Architecture	Must be the same on both the source and target hosts
Operating system and version	Must be the same on both the source and target hosts

SnapManager for Oracle	Must be installed and running on both the source and target hosts
Credentials	Must be set for the user to access the target host
Oracle	The same software version must be installed on both the source and target hosts. The Oracle Listener must be running on the target host.
Compatible storage stack	Must be the same on both the source and target hosts
Protocol used to access data files	Must be the same on both the source and target hosts
Domain	Both remote host and the host on which the database resides must be in the domain and not in the workgroup

Cloning a database to an alternate host

You can use the clone create command to clone a database backup on an alternate host.

- Create a profile or have an existing profile.
- Create a full backup or have an existing database backup.
- Create a clone specification or have an existing clone specification.

1. To clone a database to an alternate host, enter the following command: smo clone create -backup-label backup_label_name-newsid new_sid-host target_host-label clone_label-commentcomment_text-profileprofile_name-clonespec full_path_to_clonespecfile

   Oracle does not let you run two databases with the same SID simultaneously on the same host. Because of this, you must supply a new SID for each clone. However, you can have a database on another host with the same SID.

Related information

Creating profiles

Cloning databases from backups

Creating clone specifications

The smo clone create command

Viewing a list of clones

You can view a list of clones associated with a particular profile.

The list includes the following information about the clones in a profile:

- The ID for the clone
- Status of the clone operation
- Oracle SID for the clone
- Host on which the clone resides
- Label for the clone

If you specify the -verbose option, the output also shows the comments entered for the clone.

1. To display a list of all clones for a profile, enter the following command smo clone list -profile profile_name [-quiet | -verbose]

Related information

The smo clone list command

Viewing detailed clone information

You can view detailed information about a specific clone by using the clone show command.

The clone show command displays the following information:

- Clone system identifier and clone ID
- Clone operation status
- Clone create start and end date or time
- Clone label
- Clone comment
- Backup label and ID
- Source database
- Backup start and end time
- Database name, tablespaces, and data files
- Host name and file systems containing data files
- Storage system volumes and Snapshot copies backing the clone

1. Enter the following command: smo clone show -profile profile_name [-label label | -id guid]

Related information

The smo clone show command

Deleting clones

You can delete the clones when the size of the Snapshot copy reaches between 10% and 20% of the backup. This also guarantees that the clone has the most current data.

The label is the unique identifier for each clone in a profile. You can use the clone label or ID, but not the system identifier (SID) to delete the clone.

The clone SID and the clone label are not the same.

When you are deleting a clone, the database must be running. Otherwise, many files and directories for the existing clone will not be deleted, resulting in more work before another clone can be created.

The directories specified for certain Oracle parameters in the clone are destroyed when the clone is deleted, and should only contain data for the cloned database: Archive Log Destinations, Background, Core, and User Dump Destinations. The audit files are not deleted.

You cannot delete a clone when the clone is used in other operations.

You can optionally collect the dump files after a successful or failed clone delete operation.

1. Enter the following command: smo clone delete -profile profile_name [-label label | -id guid] [-force][-dump][-quiet]|[-verbose]

Example

```
smo clone delete -profile targetdb1_prof1 -label sales0908_clone1
```

Related information

The smo clone delete command