

SELECT[®] Online

The Journal of the International Oracle Users Group

Select Journal
Home

Regular Columns

Past Issues

General Information

IOUG Home



Select Magazine - January 1998

Volume 5, No. 2

Cloning an Oracle Database — Faster

By Jay Mehta

Learn how cloning a database can be faster and more helpful than traveling the traditional import/export route.

The recent news of cloning a sheep didn't come as a surprise to the Oracle DBA community as the concept of cloning isn't new to any of us. An Oracle DBA routinely performs a procedure to clone databases to support the life cycle of an application. During the application life cycle, it's not uncommon to have one or more databases to support production, acceptance, quality assurance, test and development environments. An Oracle DBA is responsible for building databases for these environments, and most of the times, it's cloning: build production from acceptance database, test from development database, quality assurance from production database, and so on.

Let's take an example of database cloning that is very common: the users of HRPROD database have been experiencing poor performance. HRTEST, a test database being significantly smaller than HRPROD database, it's not possible to duplicate HRPROD's problems using HRTEST database. You need the data from the production environment to resolve these problems, but you don't (and shouldn't) want to use your production database for testing and tuning. You need to clone your production environment. This is just one example where database cloning is required. We will refer to this example throughout this article to illustrate the procedures.

Export/Import Approach

Most DBAs utilize export/import approach to copy a database. Let's assume that you want to copy HRTEST database from HRPROD database, then here is a list of high-level steps you need to follow:

- Export HRPROD database to the dump file in consistent mode
- Create HRTEST database with create database command
- Run catalog.sql, catproc.sql and other scripts to build catalog
- Create additional rollback segments and necessary tablespaces in HRTEST
- Import a dump file extracted from HRPROD into HRTEST
- Make sure that import worked without any problems. If not, fix problems.

On occasion, the export/import approach is the best, or perhaps the only choice. For example, you want to copy a database from the Unix server to the Windows NT server or to a box with different version of operating system that does not offer file level compatibility. Another example is when you want to reorganize a database that is highly fragmented due to update and/or delete activities. During import, Oracle reorganizes data and removes fragmentation. This approach is also useful when you want to resize the target database by pre-creating tablespaces, tables or indexes of appropriate size. But the following are the main disadvantages of export/import approach:

Export Time: The time to export a database is directly proportional to the amount of data stored in the database: the larger the database, the longer it takes to export. On a mid-range Unix box, it's not unusual for an export to take 4-to-6 hours to dump a 10GB database. This article explains an approach that does not require export.

Consistent Export: During database export, especially for a large database with heavy activities, you might

encounter "snapshot too old" error if rollback segments wrap around and overwrite before-image information which export needs to reconstruct a consistent snapshot. To resolve "snapshot too old" error, you need to increase the size and/or number of the rollback segments. If this error reoccurs, then you need to export database while keeping all users out of the database that may not be acceptable for systems that run 24 hours a day, 7 days a week. The approach detailed in this article does not encounter errors like "snapshot too old," and use of backup datafiles eliminates any need for additional downtime.

Import Time: The time to import a database could be even more than the time to export as import, being a write operation, is more resource intensive that may lead to system overloading. Like export operation, the time to import a database is directly proportional to the size of the database, and usually more than the time to export a database. The approach detailed in this article does not require an import operation.

Backup/Restore/Rename/Recover Approach

This approach is based on the concept of backup, restore, rename and recover, and doesn't require export/import. Export/import approach is a high-level approach which involves exporting data from the database and then importing back into the database. That's pretty redundant. It's more like emptying a bucket into an another one, and then filling it back with the same water. Export copies data from the datafiles into the dump file and then import puts the data back into the datafiles from the dump file. Then why not copy the datafiles directly which doesn't require copying data back and forth? And that's what this approach does. This approach copies data at the datafile level. So your target database will inherit all the properties, good and bad, of the source database, but this approach can be significantly faster, especially for a large database.

Under this approach, we restore all the datafiles from the backup, but these files can't be used without renaming the database name in the headers of the datafiles as it's not possible to have two Oracle databases with the same name on the same server. So we rename the database. If necessary, we apply archived redo logs for database point-in-time recovery.

We will use the example described above to illustrate the backup/restore/rename approach. Let's say we want to clone HRPROD(Human Resource Production) database and call it HRTEST(Human Resource Test). HRTEST database will reside on the same server as HRPROD database. Note that you can use this approach even if target database is on the another server. A step-by-step approach for cloning HRTEST database is explained below:

Step 1: Determine Disk Space Requirements for HRTEST

The first step is to determine the amount of free disk space necessary on the server for HRTEST database. Since HRTEST is going to be a clone of HRPROD, free disk space needed to build HRTEST is equal to the disk space used by HRPROD database. The script, fs_usage.sql, can be used to determine the disk space used by HRPROD database, and the output produced by it is shown in Figure 1. As you can see in Figure 1, the space usage report is broken down by the file system or directory or sub-directory, depending upon the configuration of the database and the server, but we will be using the term file systems in the subsequent sections. Note that this query is written specifically for Unix platforms, but can be easily utilized on other platforms as well by replacing "/" character in the query with "]" for Open VMS platforms or "\" for Windows NT platforms.

```
rem *****
rem Name: fs_usage.sql
rem Author: Jay Mehta
rem Description: Disk space usage report
rem Usage Notes: Needs access to v$datafile view
rem *****
set pagesize 60
set heading on
set feedback off
column filesys format A30 heading "File System"
column filesys_size FORMAT 9,999,999,999 Heading "Bytes"
break on report
compute sum of filesys_size on report
spool filesys_usage.lst
select substr(name,1,instr(name,'/',-1,1)-1) filesys,
       sum(bytes) filesys_size
from sys.v$datafile
group by substr(name,1,instr(name,'/',-1,1)-1)
/
spool off
```

Figure 1: HRPROD Database : Space Usage Report

File System	Bytes
/oracle/fs01a/HRPROD	120,586,240
/oracle/fs03a/HRPROD	314,572,800
/oracle/fs04a/HRPROD	220,200,960
sum	655,360,000

Make sure that you have enough free disk space on the server not only to create HRTEST database, but also for the database growth and administrative scripts/logs. Note that this script does not include space utilized by controlfiles and redo logfiles.

Step 2: Choose Physical Database Layout for HRTEST

The next step is to determine the physical database layout for HRTEST database, i.e., determine the placement of datafile, controlfiles and redo logfiles. The number of physical database layouts that are available for HRTEST varies. Typically, it will depend on factors such as the configuration of the server, free disk space available, performance desired, etc. For example, you can choose a physical database layout for optimal performance or efficient disk space usage or a combination. A physical database layout achieves optimal performance by spreading datafiles across multiple disks and controllers. A physical layout achieves efficient disk space usage by storing all the database files on one disk.

To simplify matters, let's choose a physical database layout for HRTEST that is identical to the physical database layout of HRPROD. In other words, HRTEST's file systems map one-to-one HRPROD's file systems, as shown in *Figure 2*. As you can see from *Figure 1*, HRPROD database uses three file systems. Under one-to-one mapping, each file system of HRPROD will be mapped to one and only one file system. As shown in *Figure 2*, /oracle/fs01a/HRPROD will be mapped to /oracle/fs11a/HRTEST, and so on. In other words, HRPROD's datafiles that reside on /oracle/fs01a/HRPROD will be copied or restored to /oracle/fs11a/HRTEST and so on. From *Figure 1* and 2, the free disk space required in /oracle/fs11a/HRTEST file system is 20,971,520 bytes, and so on.

HRPROD's File Systems	HRTEST's File Systems
/oracle/fs01a/HRPROD	/oracle/fs11a/HRTEST
/oracle/fs03a/HRPROD	/oracle/fs13a/HRTEST
/oracle/fs04a/HRPROD	/oracle/fs14a/HRTEST

Figure 2: File Systems Mapping

Step 3: Update init.ora and config.ora

The next step is to configure parameters in initHRTEST.ora and configHRTEST.ora. If the HRTEST database does not exist on the server, then copy HRPROD's configuration files, rename them to initHRTEST.ora and configHRTEST.ora, and make appropriate changes. If HRTEST database already exists, then you have configuration files in place. You need to watch for those parameters, as shown in *Figure 3*, that either include database name in its values or contain physical locations of the file systems. Depending upon requirements, you might want to change memory parameters including sort_area_size, db_block_buffers, and other memory parameters if necessary.

```
* ifile
* db_name
* control_files
* audit_file_dest, core_dump_dest, background_dump_dest,
  log_archive_dest user_dump_dest, core_dump_dest
* mts parameters
```

Figure 3

Step 4: Generate Control File Script

The next step is to generate a script that will create the controlfiles for HRTEST database. Generate a text copy of HRPROD's controlfile using the sqldb or srvrmgr command: ALTER DATABASE BACKUP CONTROLFILE TO TRACE. A script generated by this command can be found in the location specified by background_dump_dest parameter of init.ora, and is shown in *Figure 4*. Rename this file to control.ctl as it's easy to remember.

```
# The following commands will create a new control file and use it
# to open the database.
# No data other than log history will be lost. Additional logs may
# be required for media recovery of offline data files. Use this
# only if the current version of all online logs are available.
STARTUP NOMOUNT
CREATE CONTROLFILE REUSE DATABASE "HRPROD" NORESETLOGS ARCHIVELOG
    MAXLOGFILES 16
    MAXLOGMEMBERS 4
    MAXDATAFILES 1022
    MAXINSTANCES 1
    MAXLOGHISTORY 100
LOGFILE
    GROUP 1 '/oracle/fs01a/HRPROD/log1a.rdo' SIZE 10M,
    GROUP 2 '/oracle/fs01a/HRPROD/log2a.rdo' SIZE 10M,
    GROUP 3 '/oracle/fs01a/HRPROD/log3a.rdo' SIZE 10M
DATAFILE
    '/oracle/fs01a/HRPROD/system01.dbf' SIZE 50M,
    '/oracle/fs01a/HRPROD/tools01.dbf' SIZE 5M,
    '/oracle/fs04a/HRPROD/rbs01.dbf' SIZE 50M,
    '/oracle/fs01a/HRPROD/temp01.dbf' SIZE 60M,
    '/oracle/fs03a/HRPROD/users01.dbf' SIZE 100M,
    '/oracle/fs04a/HRPROD/users_idx01.dbf' SIZE 60M,
    '/oracle/fs03a/HRPROD/eq_data01.dbf' SIZE 200M,
    '/oracle/fs04a/HRPROD/eq_idx01.dbf' SIZE 100M
;
# Recovery is required if any of the datafiles are restored backups,
# or if the last shutdown was not normal or immediate.
RECOVER DATABASE
# All logs need archiving and a log switch is needed.
ALTER SYSTEM ARCHIVE LOG ALL;
# Database can now be opened normally.
ALTER DATABASE OPEN;
```

Figure 4: HRPROD's Control File

Step 5: Edit the Control File Script

Make the following changes to control.ctl:

- Delete everything except the create controlfile statement from the control.ctl file. The control.ctl file should include only create controlfile statement as it is advisable to execute one command at a time, determine its success or failure and then proceed to the next step.
- Remove all occurrences of REUSE in the control.ctl file as REUSE option grants Oracle an authorization to overwrite any existing files of the same size without any prompt or notification at runtime which could be disastrous, particularly when you are cloning a live production database.
- Replace NORESETLOGS with RESETLOGS as RESETLOGS option is required with the use of backup controlfile by Oracle.
- Replace HRPROD with HRTEST in create controlfile statement as HRTEST is the name of the new database. Add SET, as shown in *Figure 4*, just before DATABASE word in the script. This step renames the name of the database to HRTEST.
- Replace the path names of all the datafiles and logfiles with the path names for HRTEST database. Refer to the mapping chart for the replacements. Since we have chosen one-to-

one mapping, global search and replace function of text editor can be used to speed up path name replacement.

- If necessary, change the values of MAXLOGFILES, MAXDATAFILES, and other parameters. You can also take this opportunity to change the size of redo log files or add or remove redo log groups and/or members. You can also choose the desired mode for redo log archiving.
- Check and recheck this trace file to avoid any potential disasters.

```
CREATE CONTROLFILE SET DATABASE "HRTEST" RESETLOGS ARCHIVELOG
    MAXLOGFILES 16
    MAXLOGMEMBERS 4
    MAXDATAFILES 1022
    MAXINSTANCES 1
    MAXLOGHISTORY 100

LOGFILE
GROUP 1 '/oracle/fs53a/HRTEST/log1a.rdo' SIZE 10M,
GROUP 2 '/oracle/fs53a/HRTEST/log2a.rdo' SIZE 10M,
GROUP 3 '/oracle/fs53a/HRTEST/log3a.rdo' SIZE 10M

DATAFILE
'/oracle/fs11a/HRPROD/system01.dbf' SIZE 50M,
'/oracle/fs11a/HRPROD/tools01.dbf' SIZE 5M,
'/oracle/fs14a/HRPROD/rbs01.dbf' SIZE 50M,
'/oracle/fs11a/HRPROD/temp01.dbf' SIZE 60M,
'/oracle/fs13a/HRPROD/users01.dbf' SIZE 100M,
'/oracle/fs14a/HRPROD/users_idx01.dbf' SIZE 60M,
'/oracle/fs13a/HRPROD/eq_data01.dbf' SIZE 200M,
'/oracle/fs14a/HRPROD/eq_idx01.dbf' SIZE 100M
;
```

Figure 5: Updated Control File

Step 6: Restore Datafiles

The purpose of this step is to restore HRPROD's datafiles to the HRTEST's file systems. During this restore, datafiles are not restored to the HRPROD's file systems, but to HRTEST's file systems. It's more like a situation in which you lost HRPROD database completely due to disk failure, and you need to restore datafiles to alternative file systems.

A copy of HRPROD's datafiles can be obtained from either cold backup or hot backup. The backup/recovery strategy employed varies, often greatly, from database-to-database and shop-to-shop. The idea here, however, is to copy all the datafiles produced by the hot or cold backup to the file systems designated in Step 2. Make sure that you do not overwrite any of HRPROD's datafiles during backup restore.

If you are not doing hot or cold backup or if HRPROD can be brought down for the time that is required to copy datafiles, then copy all HRPROD's datafiles to the mapped file systems, as determined in the Step 2 using operating system commands. Make sure that database is down during file copy. Since we have chosen one-to-one file system mapping, multiple files can be copied using wild cards. For example, use "cp /oracle/fs01a/HRPROD/* /oracle/fs11a/HRTEST" on the Unix platforms.

If HRTEST database already exists, then delete all HRTEST's database files, including datafiles, controlfiles and redo logfiles, before copy or restore.

Step 7: Restore Archived Redo Logs

If you want to roll forward HRTEST to a particular time, make sure you have all the necessary archived redo logs from the time backup was taken up to the time you want HRTEST to roll forward. If you are using hot backup files to restore, then you will need archived redo logs. Not only you can restore archived redo logs from tape, if necessary, but you can also apply the last archived redo log generated by Oracle. Note that you cannot carry-over transactions that are in the active redo logfile of HRPROD, but you can perform a log switch in HRPROD to archive the current redo logfile and then apply that archived redo log.

To apply archived redo logs to HRTEST, you need to point log_archive_dest parameter of

HRTEST's init.ora file to the location where you have restored archived redo logs from tapes. Make sure that you have enough free disk space to restore archived redo logs for the duration of roll forward process, and restored archived redo logs can be deleted afterwards.

Step 8: Set-up HRTEST environment

Set Unix environment variables including ORACLE_SID by using oraenv or coraenv. Make sure that ORACLE_SID is set to HRTEST.

Step 9: Create Control File

The next step is to create control files for HRTEST database. As shown in *Figure 6*, start the HRTEST instance with startup nomount command and run control.ctl by @control.ctl. Oracle creates the control files, specified by control_files parameter of initHRTEST.ora and mounts the database. If initHRTEST.ora isn't stored in the default location, then start the instance with pfile option. Note that SQLDBA is started in line mode. For the Oracle 7.2 or later versions, SQLDBA is replaced by server manager.

Step 10: Recover Database, if necessary

If you have copied HRPROD's datafiles while HRPROD being down or restored datafiles from the cold backup, then no recovery action is required. But if you would like to roll forward database, then media recovery should be performed. If you have restored a database from hot backup, then media recovery is required. Perform the media recovery by using recover database using backup controlfile command. Oracle will suggest the name and location of the archived redo log it needs. If archived redo logs aren't in the location suggested by Oracle, then change the location and continue. A log of recovery actions is shown in *Figure 6*. Start HRTEST database using alter database open resetlogs. Oracle responds by the message, "Instance Started", and you have successfully completed cloning.

```
SQLDBA>
SQLDBA> startup nomount ;
ORACLE instance started.
SQLDBA>
SQLDBA> @control.ctl
Statement processed.
SQLDBA>
SQLDBA> alter database open resetlogs ;
ORA-01195: online backup of file 1 needs more recovery to be consistent
ORA-01110: data file 1: '/oracle/fs53a/HRTEST/system01.dbf'
SQLDBA>
SQLDBA> recover database using backup controlfile ;
.....
.....
Applying suggested logfile...
Log applied.
.....
.....
Media recovery canceled.
SQLDBA>
SQLDBA> alter database open resetlogs ;
Statement processed.
SQLDBA> exit
SQL*DBA complete.
```

Figure 6: SQLDBA Session

Step 11: Rename GLOBAL_NAME

The last step in this procedure is to rename the GLOBAL_NAME of the database that can be done by the following statement:

```
ALTER DATABASE RENAME GLOBAL_NAME to HRTEST ;
```

Summary

The approach outlined in this article can be significantly faster than the export/import approach, particularly for a large database. I tested database cloning for a 3GB database using both the approaches, and backup/rename/recover approach was three times faster: three hours compared to nine hours for export/import approach. But a word of caution is warranted here. Backup/Rename/Recover approach requires in-depth knowledge of Oracle database administration, in general, and principles of backup/recovery in particular. Here you are killing two birds with one stone: restoring a backup and renaming the database. And don't forget to backup your target database if it already exists before you perform this procedure. You need to make sure that your backup is valid and if necessary, can be used to recover a database successfully. Happy cloning!

About the Author

Jay Mehta, a certified Oracle DBA, works as a consultant in the metropolitan D.C. area. He is the owner of Oberon System, a consulting firm. He has written articles for Select and Oracle magazines. He can be reached via email at jmehta@compuserve.com or 71034.261@compuserve.com. Your comments are welcome.

[Download Acrobat Reader](#)

Copyright 2003 by the International Oracle Users Group