SPK
and Associates

www.spkaa.com
Ph: 888-310-4540

*SPK and Associates*
900 E Hamilton Ave, Ste.100
Campbell, CA 95008

## Oracle Database Cloning Made Easy

As a systems integrator, I spend my time behind a variety of applications every day. Each of these applications stores its data in some type of database. The most common databases I deal with are typically MySQL, Oracle, DB2, Postgres, and MS SQL Server. As such, I'm going to admit that I don't have the mental bandwidth or the willingness to master each and every one. Each database has its own way of doing the same tasks, and learning each nuance is almost impossible unless you're a fulltime DBA.

The most common task a DBA has to deal with is database cloning. In this article, I'm going to discuss the various ways to clone an Oracle database. I chose to focus on Oracle because sometimes it’s not clear what the different methods are, and when one ought to be used over another.

When we refer to the term database cloning, our end goal in most cases is to have the ability to connect to a database that is completely identical to our source database. This means that every object in the source database should be present in the cloned database – tables, indexes, constraints, stored procedures, etc. However in some cases, all we’re interested in is the table data. Another factor to consider is performance. Can your database be cloned quickly enough to support your needs? Lastly, one of the biggest factors that can’t be overlooked is the impact to the users. Can the database be cloned without having to incur any downtime? As we explore the different methods, I’ll outline the different pros and cons of each. Based your particular situation, you’ll now be able to make an informed decision.

## Traditional File Based Cold Clone

The most traditional method of cloning an Oracle database consists of these high level tasks:

- Shut the source database down
- Copy the datafiles to a new location
- Edit the new database creation script
- Startup the new database

Pros:

- Takes every single object from the source database as is: tables, indexes, constraints, stored procedures, packages, etc.
- Allows you to leverage advanced features from your storage device (i.e FlexClones from Netapp) to speed up the copy process
- Provides your application an apples-to-apples environment to be able to replicate issues. For example, what if you were trying to troubleshoot application slowness due to a missing index?
- When leveraging storage device features, this can be one of the fastest methods, particularly with very large databases.

Cons:

- Slightly more difficult to do. Involves both a DBA and storage administrator to complete.
- Inability to exclude certain objects. All or nothing approach.

- Unless you've enabled archive logging, you must take the database down.

### Sample Procedure

- Dump the database creation script from the source database
    - sqlplus>  alter database backup controlfile to trace
    - Locate the trace file that was created.  We will edit this file in order to create our new cloned database.
- Shutdown the source database
    - sqlplus> shutdown immediate;
- Copy over the source data files to a new location
    - cd /u01/oradata/SIDPROD
    - tar cf - * | (cd /u02/oradata/SIDTEST); tar xf –
- Delete the controlfiles from the new cloned database directory
    - rm /u02/oradata/SIDTEST/ctl*
- Edit the trace file (let's call it newdb.sql).
    - Substitute all paths from the old source db to the new db.
    - Substitute the line from:
        - create controlfile reuse database "SIDPROD" noresetlogs
    - Substitute to:
        - create controlfile set database "SIDTEST" resetlogs noarchivelog
- Run the newdb.sql script:
    - export ORACLE_SID=SIDTEST; sqlplus '/as sysdba'
    - @newdb.sql
- Your new cloned db should be up and running.

## Import / Export Method

This method utilizes a pair of tools included with Oracle named imp / exp.  At a high level, these tools connect to a live database and "query" out the data on a table by table basis.

Pros:

- Quick and easy, especially if you only need data from a few tables.
- No downtime needed
- Ability to specify data for only certain users/schemas

Cons:

- For large databases, performance is not great.  Since this connects through Oracle similar to an application, there's overhead involved.
- Since this is a logical dump, you may not be able to troubleshoot certain issues inherent to your source database.

### Sample Procedure

- Login to your source database
- exp sys/manager file=exp.dmp log=exp.log owner=ricky
    - Where file = the target dump file
    - log = log file for any errors encountered
    - owner = name of the schema who's objects you want to export
- Login to your source database
- If you don't have one, you can create one using the DB creation assistant (dbca)
- imp sys/manager file=exp.dmp log=imp.log fromuser=ricky touser=bobby indexes=n rows=y ignore=y grants=n constraints=n statistics=none
    - Where file = the dump file we created earlier
    - log = log file for any import errors we encounter
    - fromuser = the schema that was dumped from the source database
    - touser = the schema that we want to import the objects into
    - indexes = skip the import of indexes?
    - rows = skip the import of the table data?
    - ignore = if any create statements fail, do we want to continue?
    - grants = skip importing any grant statements?
    - constraints = skip importing any constraints?
    - statistics = do we want to import stats from the previous db or recalculate them instead?

## Datapump method

Datapump is another tool included with Oracle as of 10g. It is similar to imp / exp in terms of functionality, however it is a completely redesigned tool optimized to handle bulk data transfer.

Pros:

- Compared to imp/exp, performance is much improved. The tool does not connect to an Oracle database the same way an application does, instead it has the ability to interface directly with the database back end files.
- Those familiar with the imp/exp tools will be able to easily transition to impdp/expdp.
- No database downtime is necessary.

Cons:

- Requires 10g. Not available in 9i.
- Still slower than using the traditional cold clone method

### Sample Procedure

- Login to the source database machine
- Create a database directory object
    - create directory dumpdir as '/disk/backups';
    - grant read, write on directory dumpdir to ricky;

- expdp ricky/password DIRECTORY=dumpdir DUMPFILE=fulldump.dmp
- Login to the destination database machine
- Create a new database using dbca if one does not already exist
- Create the database directory object
    - create directory dumpdir as '/disk/backups';
- impdp system/manager DIRECTORY=dumpdir DUMPFILE=fulldump.dmp

As you can see, there are lots of factors that are involved when trying to decide how to clone an Oracle database. There is a variation of the traditional cold clone method that even allows a dba to take online backups. Perhaps I'll elaborate on how to perform that in a follow up article. But armed with this info, hopefully you'll be able to test each method and determine which works best given your situation.

Stay tuned to our website for more helpful how-tos!

Michael Solinap
Sr. Systems Integrator
SPK & Associates