11. Configuring the Database Archiving Mode.

Abstract: Configuring an Oracle database for backup and recovery can be complex. At a minimum, you must understand the archive process, the initialization parameters associated with the archive process, the commands necessary to enable and disable archiving, the commands used to manually archive, and the process of initializing automated archiving. This lesson provides examples of the backup and recovery configuration process. After reading this, you should be comfortable with this process. The archiver and archived logs are the key components you need for the backup and recovery process. Archived logs make it possible for a complete recovery to be conducted. In addition, the archiver or archivers are responsible for creating archived logs so that they are available in the event of a failure. It is important that you understand how to configure the database for archiving and that you need to provide multiple copies of archived logs to reduce the damage caused by failure situations.

Contents

Objectives:

- Describe the differences between Archivelog and Noarchivelog modes.
- Configure a database for Archivelog mode.
- Enable automatic archiving.
- Perform manual archiving of logs.
- Configure multiple archive processes.
- Configure multiple destinations, including remote destinations.

1. Choosing ARCHIVELOG Mode or NOARCHIVELOG Mode

One of the most fundamental backup and recovery decisions that a DBA will make is whether to operate the database in ARCHIVELOG mode or NOARCHIVELOG mode. As you learned earlier, the redo logs record all the transactions that have occurred in a database, and the archived logs are copies of these redo logs. So, the archived logs contain the historical changes, or transactions, that occur in the database. Operating in *ARCHIVELOG mode* means that the database will generate archived logs; operating in *NOARCHIVELOG mode* means that the database will not generate archived logs. This section discusses the differences between ARCHIVELOG and NOARCHIVELOG mode.

1.1. ARCHIVELOG Mode

In ARCHIVELOG mode, the database generates archived log files from the redo logs. This means that the database makes copies of all the historical transactions that have occurred in the database. Here are other characteristics of operating in ARCHIVELOG mode:

- Performing *online (hot) backups* is possible. This type of backup is done when the database is up and running. Therefore, a service outage is not necessary to perform a backup. The ALTER TABLESPACE <TABLESPACE_NAME> BEGIN BACKUP command is issued to perform hot backups. After this command is issued, an OS copy can take place on each tablespace's associated data files. When the OS copy is complete, an ALTER TABLESPACE <TABLESPACE_NAME> END BACKUP command must be issued. These commands must be executed for every tablespace in the database.
- A complete recovery can be performed. This is possible because the archived logs contain all the changes up to the point of failure. All logs can be applied to a backup copy of the database (hot or cold backup). This would apply to all the transactions up to the time of failure. Thus, there would be no data loss or missing transactions.
- Tablespaces can be taken offline immediately.
- Increased disk space is required to store archived logs, and increased maintenance is associated with maintaining this disk space.

1.2. NOARCHIVELOG Mode

In NOARCHIVELOG mode, the database does not generate archived log files from the redo logs. This means that the database is not storing any historical transactions from such logs. Instead, the redo logs are written over each other as needed by Oracle. As a result, the only transactions that can be used in the event of instance failure are in the current redo logs. Operating in NOARCHIVELOG mode has the following characteristics:

- In most cases, a complete restore cannot be performed. This means that a loss of data will occur. The last cold backup will need to be used for recovery.
- The database must be shut down completely for a backup, which means the database will be unavailable to the users of the database during that time. This means that only a cold backup can be performed.
- Tablespaces cannot be taken offline immediately.
- Additional disk space and maintenance is not needed to store archived logs.

2. Understanding Recovery Implications of NOARCHIVELOG

The recovery implications associated with operating a database in NOARCHIVELOG mode are important. A loss of data usually occurs when the last consistent full backup is used for a recovery. Therefore, to reduce the amount of data lost in the event of a failure, frequent cold backups need to be performed. This means that the database could be unavailable to users on a regular basis.

Now that you are familiar with the problems associated with this mode, let's look at examples of when it would not make sense to use NOARCHIVELOG mode and when it would.

Imagine that Manufacturing Company A's database must be available for 24 hours a day to support three shifts of work. This work consists of entering orders, bills of lading, shipping instructions, and inventory adjustments. The shifts are as follows: day shift, 9 A.M. to 5 P.M.; swing shift, 5 P.M. to 1 A.M.; and night shift, 1 A.M. to 9 A.M. If this database is shut down for a cold backup from midnight to 2 A.M., then the night shift and swing shift would be unable to use it during that period. As a result, a NOARCHIVELOG backup strategy would not be workable for Manufacturing Company A.

On the other hand, if Manufacturing Company B's database must be available only during the day shift, from 9 A.M. to 5 P.M., then backups could be performed after 5 P.M. and before 9 A.M. without affecting users. Thus, the DBA could schedule the database to shut down at midnight and perform the backup for two hours. The database would be restarted before 9 A.M., and there would be no interference with the users' work. In the event of a failure, there would be a backup from each evening, and only a maximum of one day's worth of data would be lost. If one day's worth of data loss were acceptable, this would be a workable backup and recovery strategy for Manufacturing Company B.

These examples show that in some situations, operating in NOARCHIVELOG mode makes sense. But there are recovery implications that stem from this choice. One implication is that a loss of data will occur in the event of a failure. Also, there are limited choices on how to recover. The choice is usually to restore the whole database from the last consistent whole backup while the database was shut down (cold backup).

3. Configuring a Database for ARCHIVELOG Mode and Automatic Archiving

Once the determination has been made to run the database in ARCHIVELOG mode, the database will need to be configured properly. You can do this to a new database during database creation or to an existing database via Oracle commands. After the database is in ARCHIVELOG mode, you will most likely configure automatic archiving. *Automatic archiving* frees the DBA up from the manual task of archiving logs with commands before the online redo logs perform a complete cycle.

3.1. Setting ARCHIVELOG Mode

ARCHIVELOG mode can be set during the database creation or by using the ALTER DATABASE ARCHIVELOG command. The database must be mounted, but not open, in order to execute this command. This command stays in force until it is turned off by using the ALTER DATABASE NOARCHIVELOG command. The database must be mounted, but not open, in order to execute this command as well.

The redo log files will be archived to the location specified by the LOG_ARCHIVE_DEST parameter in the init.ora file. By default, the database is in manual archiving mode. This means that as the redo logs become full, the database will hang until the DBA issues the ARCHIVE LOG ALL command, which archives all the online redo log files not yet archived. Figure 1 shows a database configured for ARCHIVELOG mode.

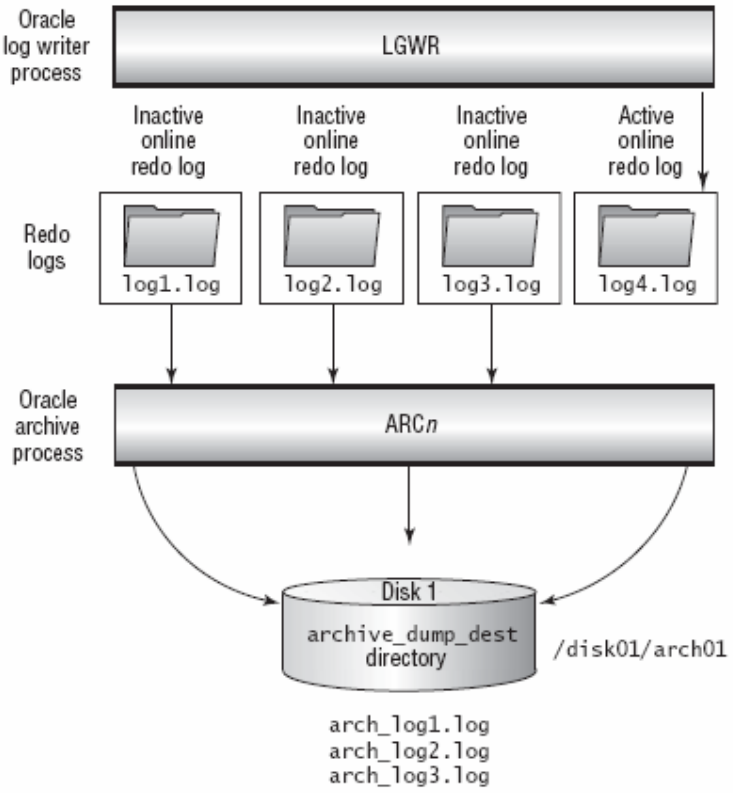

FIGURE 1. A database configured for ARCHIVELOG mode.

Let's look at an example of how to tell whether the database is in ARCHIVELOG or NOARCHIVELOG mode. You will need to run SQL*Plus and execute the following SQL statement that queries from one of the V$ views.

Alternatively, you can perform OS commands, such as ps –ef |grep arch in Unix, that check the process list to see whether the ARCn process is running. This process does the work of copying the archived logs from the redo logs.

This example shows ARCHIVELOG mode using the V$ views:

```
oracle@octilli:~ > sqlplus /nolog
SQL*Plus: Release 9.0.1.0.0 - Production on Tue Sep 25
19:08:25 2001
(c) Copyright 2001 Oracle Corporation. All rights reserved.
SQL> connect /as sysdba
Connected.
SQL> select name,log_mode from v$database;
NAME      LOG_MODE
---------  ------------
ORC9      ARCHIVELOG
SQL>
```

This example shows NOARCHIVELOG mode using the V$ views:

```
oracle@octilli:~ > sqlplus /nolog
SQL*Plus: Release 9.0.1.0.0 - Production on Tue Sep 25
19:08:25 2001
(c) Copyright 2001 Oracle Corporation. All rights reserved.
SQL> connect /as sysdba
Connected.
SQL> select name,log_mode from v$database;
NAME      LOG_MODE
---------  ------------
ORC9      NOARCHIVELOG
SQL>
```

The following is an example of using the Unix OS command ps –ef |grep arch to see whether the archiver process is running. This is more indirect than the V$ table output, but if the archiver process is running, then the database would have to be in ARCHIVELOG mode.

```
oracle@octilli:~ > ps -ef|grep arc
oracle 2097 1 0 00:26 ? 00:00:00 ora_arc0_orc9
oracle 4468 4327 0 19:34 pts/3 00:00:00 grep arc
oracle@octilli:~ >
```

A couple of methods exist for determining the location of the archived logs. The first is to execute the SHOW PARAMETER command and the second is to view the init.ora file. An

example of using the SHOW PARAMETER command to display the value of LOG_ARCHIVE_DEST is as follows:

```
SQL> show parameters log_archive_dest
NAME                                 TYPE        VALUE
------------------------------------ ----------- --------------------
log_archive_dest                     string      oracle/admin/orc9/arch
log_archive_dest_1                   string
log_archive_dest_10                  string
log_archive_dest_2                   string
log_archive_dest_3                   string
log_archive_dest_4                   string
log_archive_dest_5                   string
log_archive_dest_6                   string
log_archive_dest_7                   string
log_archive_dest_8                   string
log_archive_dest_9                   string
log_archive_dest_state_1             string      enable
log_archive_dest_state_10            string      enable
log_archive_dest_state_2             string      enable
log_archive_dest_state_3             string      enable
log_archive_dest_state_4             string      enable
log_archive_dest_state_5             string      enable
log_archive_dest_state_6             string      enable
log_archive_dest_state_7             string      enable
log_archive_dest_state_8             string      enable
log_archive_dest_state_9             string      enable
SQL>
```

An example of viewing a partial init.ora file to display the LOG_ARCHIVE_DEST is listed here:

```
##################################################
# Partial Sample of Init.ora File
```

```
##################################################
##################################################
# Archive Logging
##################################################
log_archive_start=true
log_archive_dest=/oracle/admin/orc9/arch
log_archive_format=archorc9_%s.log
```

3.2. Setting Automatic Archiving

To configure a database for automatic archiving, you must perform a series of steps:

1. Edit the init.ora file and set the LOG_ARCHIVE_START parameter to TRUE. This will automate the archiving of redo logs as they become full.
2. Shut down the database and restart the database by using the command STARTUP MOUNT.
3. Use the ALTER DATABASE ARCHIVELOG command to set ARCHIVELOG mode.
4. Open the database with the ALTER DATABASE OPEN command.

To verify that the database is actually archiving, you should execute the ALTER SYSTEM SWITCH LOGFILE command. After you execute this command, check the OS directory specified by the parameter LOG_ARCHIVE_DEST to validate that archived log files are present. You can also execute the ARCHIVE LOG LIST command to display information that confirms the database is in ARCHIVELOG mode and automatic archival is enabled.

Now let's walk through this process.

1. First, edit the init.ora file and change the parameter LOG_ARCHIVE_START to TRUE. As a result, the database will be in automatic ARCHIVELOG mode. See the example init.ora file below.

```
##################################################
## Partial Sample of Init.ora File
##################################################
##################################################
# Archive Logging
##################################################
log_archive_start=true
log_archive_dest=/oracle/admin/orc9/arch
```

```
log_archive_format=archorc9_%s.log
```

2. Next, run the following commands in SQL:

```
SQL> shutdown
SQL> startup mount
SQL> alter database archivelog;
SQL> alter database open;
```

The automatic archival feature has now been enabled for this database. To verify that the database has been configured correctly, you can perform the following checks.

1. First, perform an ALTER SYSTEM SWITCH LOGFILE. This will need to be done n + 1 times, where n is the number of redo logs in your database.

```
SQL> alter system switch logfile;
```

2. Next, perform a directory listing of the archive destination in LOG_ARCHIVE_DEST. The Unix command pwd displays the current working directory, and ls shows the contents of the directory.

```
oracle@octilli:/oracle/admin/orc9/arch > ls -ltr
total 276
-rw-r----- 1 oracle dba 133632 Sep 25 00:27 archorc9_3.log
-rw-r----- 1 oracle dba 125952 Sep 25 00:28 archorc9_4.log
-rw-r----- 1 oracle dba 1024 Sep 25 00:28 archorc9_5.log
-rw-r----- 1 oracle dba 1536 Sep 25 00:28 archorc9_6.log
-rw-r----- 1 oracle dba 1024 Sep 25 00:28 archorc9_7.log
oracle@octilli:/oracle/admin/orc9/arch >
```

The other way to verify that the automatic archival feature has been enabled is to execute the ARCHIVE LOG LIST command, which displays the status of these settings (as is shown here).

```
SQL> archive log list;
Database log mode Archive Mode
Automatic archival Enabled
Archive destination /oracle/admin/orc9/arch
Oldest online log sequence 6
Next log sequence to archive 8
Current log sequence 8
SQL>
```

WARNING: If you enable ARCHIVELOG mode but forget to enable the automatic archival by not editing the init.ora and changing the LOG_ARCHIVE_START to TRUE, the database will hang when it gets to the last available redo log. You will need to perform manual archiving as a temporary fix or to shut down and start up the database after changing the LOG_ARCHIVE_START parameter.

REAL WORLD SCENARIO

Providing Adequate Space for Archive Logging

The additional disk space necessary for ARCHIVELOG mode is often overlooked or underestimated. For average databases, you should have enough space to keep at least one or two days of archived logs online. In order to estimate the correct size of such files, you will need to estimate the volume during peak transaction volume periods. Peak times are typically when the heaviest transactional activity or batch activity occurs.

This estimated size is multiplied by how many log_archive_dest_n locations are set up. If you perform nightly hot backups on your system, one or two days of online archived logs should meet most of your recovery requirements from the archived log perspective.

But if the archive process doesn't have enough disk space to write archived logs, the database will hang or stop all activity. This hung state will remain until you make more space available by moving older archived logs off your system, by compressing log files, or by setting up an automated job that will remove logs after they have been written to tape. You must be careful not to misplace or delete archived logs when trying to free up space for the archive process. Remember these archived logs could be required in a recovery situation.

4. Manually Archiving Logs

The *manual archiving* of logs consists of enabling the database for ARCHIVELOG mode and then manually executing the ARCHIVE LOG ALL command from SQL. The init.ora parameter LOG_ARCHIVE_START must be set to FALSE to disable automatic archival. The next step is to put the database in ARCHIVELOG mode by performing the following commands:

```
SQL> shutdown
SQL> startup mount
SQL> alter database archivelog;
SQL> alter database open;
```

Now you are ready to perform manual archiving of redo logs by using

```
SQL> archive log all;
```

or

```
SQL> archive log next;
```

The ARCHIVE LOG ALL command will archive all redo logs available for archiving, and the ARCHIVE LOG NEXT command will archive the next group of redo logs.

5. Using init.ora Parameters for Multiple Archive Processes and Locations

The capability to have more than one archived log destination and multiple archivers was first introduced in Oracle8. In Oracle8i and 9i, more than two destinations can be used, providing even greater archived log redundancy.

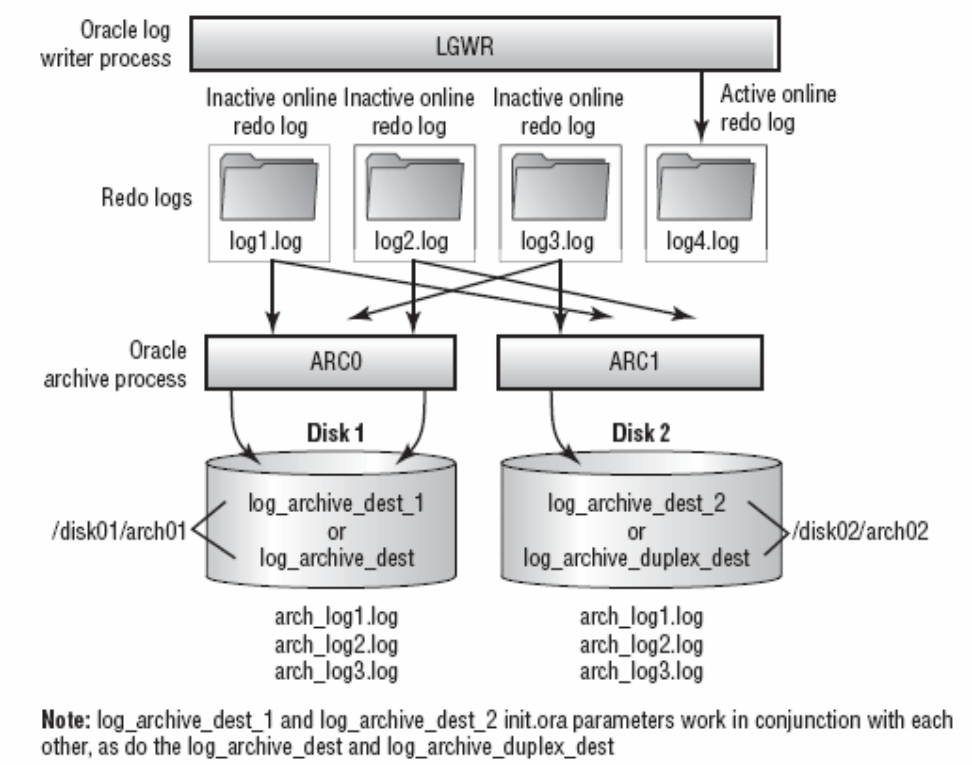

FIGURE 2. ARCHIVELOG mode with multiple destinations.

The main reason for having multiple destinations for archived log files is to eliminate any single point of failure. For example, if you were to lose the disk storing the archived logs before these logs were backed up to tape, the database would be vulnerable to data loss in the event of a failure. If the disk containing the archived logs was lost, then the safest thing to do would be to run a backup. This would ensure that no data would be lost in the event of a database crash from media failure. Having only one archived log location is a single point of failure for the backup process. Hence, Oracle has provided multiple locations, which can be

on different disk drives, so that the likelihood of archived logs being lost is significantly reduced. See Figure 2, which demonstrates ARCHIVELOG mode with multiple destinations.

In addition to reducing the potential archived log loss, one of the multiple locations can be remote. The remote location supports the Oracle standby database, which is an option that can be configured to protect a computing site from a disaster. Let's go over a brief explanation of this option and why it can require remote archived log capabilities.

The Oracle standby database can require archived logs to be moved to a remote server, which is running a copy of the production database. This copy of the production database is in a permanent recovery situation with archived logs from the production database regularly being applied to the standby database.

There are certain initialization parameters that can be used to assure that the archived logs get moved to remote locations. These initialization parameters will be covered in more detail in the upcoming section "Remote Archived Log Locations.".

5.1. Multiple Archive Processes and Locations

Having multiple archive processes can make the archived log creation process faster. If significant volumes of data are going through the redo logs, the archiver can be a point of contention. This means that redo logs could wait or delay database activity while trying to write out an archived log. Furthermore, the archiver has more work to do if the database is writing to multiple destinations. Thus, multiple archive processes can do the extra work to support the additional archive destinations.

To implement these new features, the database must be in ARCHIVELOG mode. To verify that the database is in ARCHIVELOG mode, either run an ARCHIVE LOG LIST command or query the V$DATABASE view.

To configure a database for multiple archive processes and LOG_ARCHIVE_DESTs, you use two sets of init.ora parameters. The first set is based on the destination parameter, which has been slightly changed to LOG_ARCHIVE_DEST_N (where N is a number from 1 to 10). The values for these parameters LOG_ARCHIVE_DEST_1 and LOG_ARCHIVE_DEST_2 are as follows: 'LOCATION = /ORACLE/ADMIN/ORC9/ARCH1', 'LOCATION = /ORACLE/ADMIN/ORC9/ARCH2', and 'LOCATION = /ORACLE/ADMIN/ORC9/REMOTE_ARCH3'. The first set of parameters is listed below in init.ora.

```
##################################################
# Archive Logging
##################################################
log_archive_start=true
#log_archive_dest=/oracle/admin/orc9/arch
log_archive_dest_1='location=/oracle/admin/orc9/arch1'
log_archive_dest_2='location=/oracle/admin/orc9/arch2'
log_archive_dest_3='location=/oracle/admin/orc9/arch_remote3'
```

```
log_archive_max_processes=3
log_archive_format=archorc9_%s.log
```

After these parameters are changed or added, then you will need to start up the database to use the new parameters. These new dump locations will then be in effect.

The second set of parameters are LOG_ARCHIVE_DEST = </SOME/DIRECTORY/LOCATION> and LOG_ARCHIVE_DUPLEX_DEST = </SOME/DIRECTORY/LOCATION>. The following example uses LOG_ARCHIVE_DEST =/ORACLE/ADMIN/ORC9/ARCH1 and LOG_ARCHIVE_DUPLEX_DEST = /ORACLE/ADMIN/ORC9/ARCH2. The main difference in this approach is that you use these parameters if you are going to have only two locations or want to use the same init.ora parameter format supported in 8.0.x. These parameters are mutually exclusive with LOG_ARCHIVE_DEST_N mentioned in the previous example. This second set of parameters can be seen below in the init.ora parameter values.

```
##################################################
# Archive Logging
##################################################
log_archive_start=true
log_archive_dest=/oracle/admin/orc9/arch1
log_archive_duplex_dest=/oracle/admin/orc9/arch2
log_archive_max_processes=2
log_archive_format=archorc9_%s.log
```

This second method of mirroring archived logs is designed to mirror just one copy of the log files, whereas the first method can mirror up to 10 copies, one of which can be a remote database.

5.2. Remote Archived Log Locations

With all remote transactions, there is a greater degree of potential problems. In order to deal with these potential transaction problems, Oracle has developed some specific commands for remote archive destinations. Among these, there are a couple of commands and initialization parameters that determine whether the logs are required to reach the remote location and whether they are required to retry unsuccessful sends.

LOG_ARCHIVE_MIN_SUCCEED_DEST=N is the initialization parameter that determines how many of the 10 maximum archive destinations must successfully receive an archived log before the online redo logs can be written over.

If this is not met, the database will hang. The first parameters specify whether the destinations are mandatory or optional log archive destinations. When you use the MANDATORY parameter, destinations will require log files to arrive at the remote location; if these files don't arrive, the database generating these logs will hang. When you use the OPTIONAL

parameter, the opposite of the MANDATORY parameter, if a log does not make it to remote location, the database will continue processing without hanging.

The REOPEN command causes failed archive destinations to be retried after a certain period of time. Below is an example of this command and the parameters previously mentioned. Note that the REOPEN example uses a service-entered destination for use in standby database mode.

```
LOG_ARCHIVE_DEST_1="LOCATION=/oracle/admin/tst9/arch1 MANDATORY"
LOG_ARCHIVE_DEST_2="LOCATION=/oracle/admin/tst9/arch2 OPTIONAL"
LOG_ARCHIVE_DEST_3="SERVICE=euro_fin_db_V9.0.1 REOPEN=60"
```

TIP: Make sure each LOG_ARCHIVE_DEST_N and LOG_ARCHIVE_DUPLEX_DEST is on a different physical device. The main purpose of these new parameters is to allow a copy of the files to remain intact if a disk were to crash.

6. Summary

The Oracle backup and recovery capability is full featured and robust. It provides many options that support a wide variety of backup and recovery situations. In this lesson, you have seen how to configure an Oracle database for backup and recovery. You have learned the ramifications of operating in ARCHIVELOG as opposed to NOARCHIVELOG mode, and vice versa, and how that choice affects the backup and recovery process. You have seen examples of how the init.ora parameters control the destinations and automation of the archive logging. Finally, you walked through an example that enabled ARCHIVELOG mode and automatic archival of logs in the database.

A solid understanding of the archived log process is fundamental to the backup and recovery process. The decision to enable archive logging has major implications on a DBA's ability to recover the database. In addition, certain recovery options that will be covered in upcoming lessons are dependent on whether or not you have enabled ARCHIVELOG mode. Make sure you understand the reasons for enabling or not enabling the archived log process because this knowledge will benefit you during OCP testing, as well as in real life situations.

References

[1] Oracle9i DBA Fundamentals II.