Module 7 – Control File Maintenance

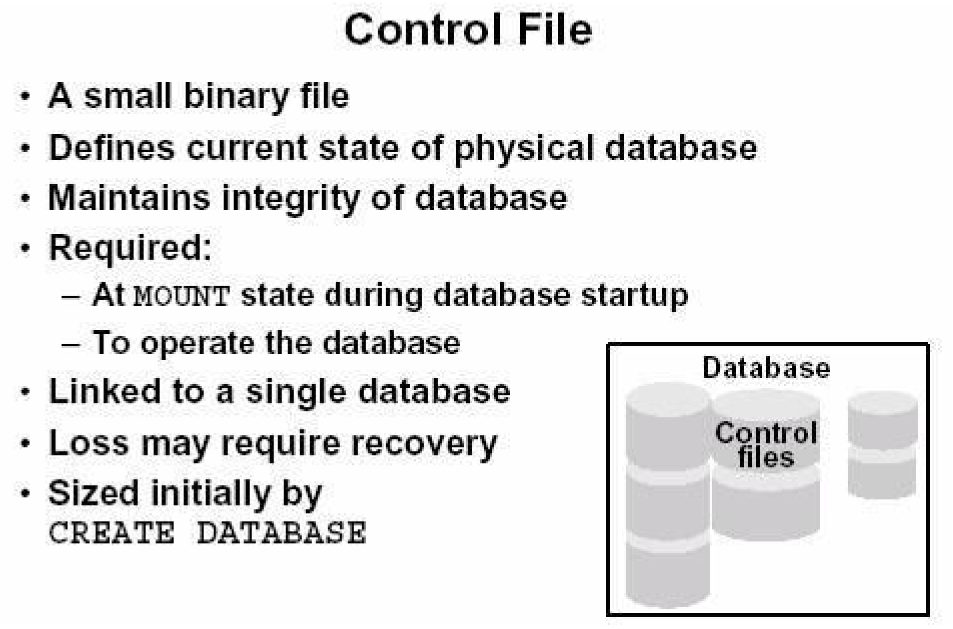
Objectives

These notes explain how:

- a control file is used.
- to examine control files contents.
- to multiplex control files.
- to manage control files with an Oracle Managed Files (OMF) approach.

Introduction

As you've learned from thus far in the course, a **Control File** is a small binary file that stores information needed to startup an Oracle database and to operate the database.



Control File

- A small binary file
- Defines current state of physical database
- Maintains integrity of database
- Required:
  - At `MOUNT` state during database startup
  - To operate the database
- Linked to a single database
- Loss may require recovery
- Sized initially by `CREATE DATABASE`

Database
Control files

- A control file belongs to only one database.

- A control file(s) is created at the same time the database is created based on the **CONTROL_FILES** parameter in the **PFILE**.

- If all copies of the control files for a database are lost/destroyed, then database recovery must be accomplished before the database can be opened.

- An Oracle database reads only the first control file listed in the PFILE; however, it writes continuously to all of the control files (where more than one exists).

- You must never attempt to modify a control file as only the Oracle Server should modify this file.

- While control files are small, the **size** of the file can be significantly influenced by the following **CREATE DATABASE** or **CREATE CONTROLFILE** command parameters if they have large values.
    - MAXLOGFILES
    - MAXLOGMEMBERS
    - MAXLOGHISTORY
    - MAXDATAFILES
    - MAXINSTANCES

Contents of a Control File

Control files record the following information:

- Database name – recorded as specified by the initialization parameter DB_NAME or the name used in the CREATE DATABASE statement.
- Database identifier – recorded when the database is created.
- Time stamp of database creation.
- Names and locations of datafiles and online redo log files. This information is updated if a datafile or redo log is added to, renamed in, or dropped from the database.

- Tablespace information. This information is updated as tablespaces are added or dropped.

- Redo log history – recorded during log switches.

- Location and status of archived logs – recorded when archiving occurs.

- Location and status of backups – recorded by the Recovery Manager utility.

- Current log sequence number –recorded when log switches occur.

- Checkpoint information – recorded as checkpoints are made.

Multiplexing Control Files

Control files should be multiplexed – this means that more than one identical copy is kept and **each copy** is stored to a **separate, physical disk drive** – of course your Server must have multiple disk drives in order to do this. Even if only one disk drive is available, you should still multiplex the control files.

- This eliminates the need to use database recovery if a copy of a control file is destroyed in a disk crash or through accidental deletion.

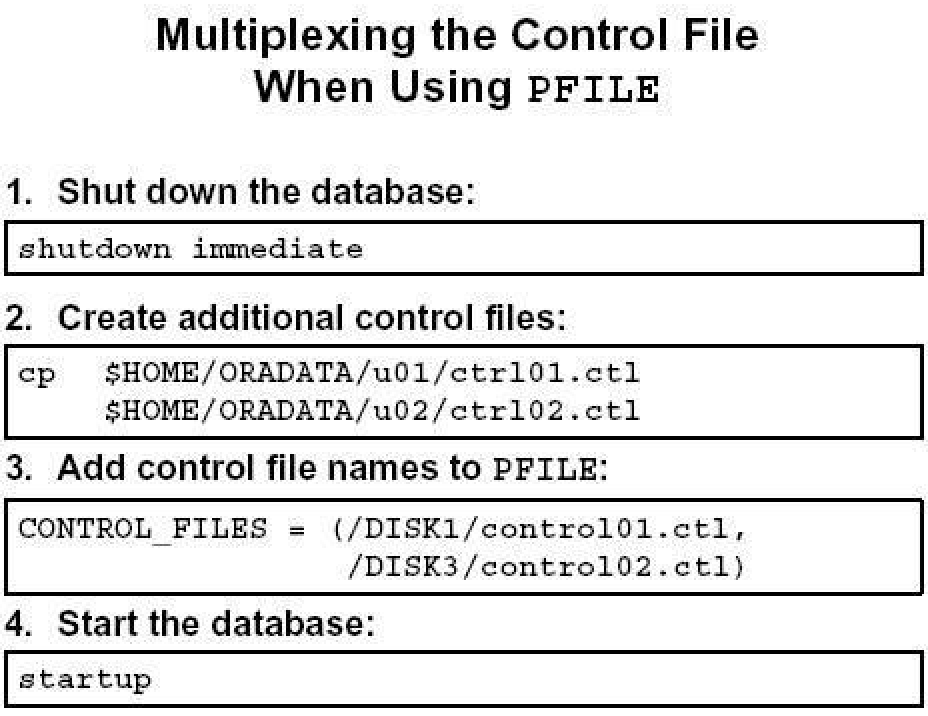
- You can keep up to eight copies of control files – the Oracle Server will automatically update all control files specified in the initialization parameter file to a limit of eight.

- More than one copy of a control file can be created by specifying the location and file name in the **CONTROL_FILES** parameter of the **PFILE** when the database is created.

- During database operation, only the first control file listed in the **CONTROL_FILES** parameter is read, but all control files listed are written to in order to maintain consistency.

- One approach to multiplexing control files is to store a copy to every disk drive used to multiplex redo log members of redo log groups.

You can also add additional control files. When using a **PFILE**, this is accomplished by shutting down the database, copying an existing control file to a new file on a new disk drive, editing the **CONTROL_FILES** parameter of the **PFILE**, then restarting the database.



Multiplexing the Control File When Using PFILE

1. Shut down the database:

```
shutdown immediate
```

2. Create additional control files:

```
cp   $HOME/ORADATA/u01/ctrl01.ctl
     $HOME/ORADATA/u02/ctrl02.ctl
```

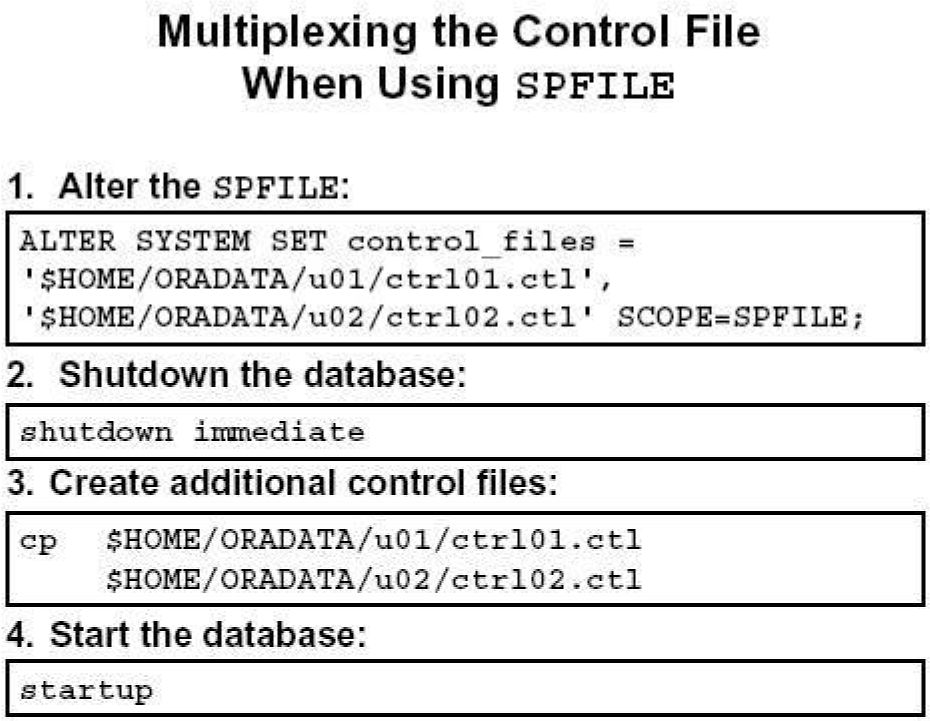
3. Add control file names to PFILE:

```
CONTROL_FILES = (/DISK1/control01.ctl,
                 /DISK3/control02.ctl)
```

4. Start the database:

```
startup
```

If you are using an **SPFILE**, you can use the steps specified in the figure shown here. The difference is you name the control file in the first step and create the copy in step 3.



Multiplexing the Control File When Using SPFILE

1. Alter the SPFILE:

```
ALTER SYSTEM SET control_files =
'$HOME/ORADATA/u01/ctrl01.ctl',
'$HOME/ORADATA/u02/ctrl02.ctl' SCOPE=SPFILE;
```

2. Shutdown the database:

```
shutdown immediate
```

3. Create additional control files:

```
cp   $HOME/ORADATA/u01/ctrl01.ctl
     $HOME/ORADATA/u02/ctrl02.ctl
```

4. Start the database:

```
startup
```

What if a Disk Drive Fails? Recovering a Control File

Use the following steps to recover from a disk drive failure that has one of the database's control files located on the drive.

- Shut down the instance.

- Replace the failed drive.

- Copy a control file from one of the other disk drives to the new disk drive – here we assume that **u02** is the new disk drive and **control02.ctl** is the damaged file.

```
$ cp /u01/oracle/oradata/control01.ctl
/u02/oracle/oradata/control02.ctl
```

- Restart the instance. If the new media (disk drive) does not have the same disk drive name as the damaged disk drive or if you are creating a new copy while awaiting a replacement disk drive, then alter the **CONTROL_FILES** parameter in the **PFILE** prior to restarting the database.

- No media recovery is required.

- If you are awaiting a new disk drive, you can alter the **CONTROL_FILES** parameter to remove the name of the control file on the damaged disk drive – this enables you to restart the database.

Backup Control Files and Create Additional Control Files

Oracle recommends backup of control files every time the physical database structure changes including:

- Adding, dropping, or renaming datafiles.

- Adding or dropping a tablespace, or altering the read/write state of a tablespace.

- Adding or dropping redo log files or groups.

Use the **ALTER DATABASE BACKUP CONTROLFILE** statement to backup control files.

Example:

```
ALTER DATABASE BACKUP CONTROLFILE TO
'/u02/oradata/backup/control.bkp';
```

```
ALTER DATABASE BACKUP CONTROLFILE TO

'C:/oraclexe/oradata/XE/control1.bkp';
```

Now use an SQL statement to produce a trace file (write a SQL script to the trace file) that can be edited and used to reproduce the control file.

```
ALTER DATABASE BACKUP CONTROLFILE TO TRACE;
```

To create additional control files:

1. Shut down the database.

2. Copy an existing control file to a new location, using operating system commands.

3. Edit the **CONTROL_FILES** parameter in the database initialization parameter file to add the new control file name, or to change the existing control filename.

4. Restart the database.

Oracle Managed Files Approach

Control files are automatically created with the Oracle Managed Files (OMF) approach during database creation even if you do not specify file locations/names with the **CONTROL_FILES** parameter—it is preferable to specify file locations/names.

With OMF, if you wish to use the **init.ora** file to manage control files, you must use the filenames generated by OMF.

- The locations are specified by the **DB_CREATE_ONLINE_LOG_DEST_n** parameter.
- If the above parameter is not specified, then their location is defined by the **DB_CREATE_FILE_DEST** parameter.

Control file names generated with OMF can be found within the **alertSID.log** that is automatically generated by the **CREATE DATABASE** command and maintained by the Oracle Server.

Control File Information

Several dynamic performance views and SQL*Plus commands can be used to obtain information about control files.

- **V$CONTROLFILE** – gives the names and status of control files for an Oracle Instance.
- **V$DATABASE** – displays database information from a control file.
- **V$PARAMETER** – lists the status and location of all parameters.
- **V$CONTROLFILE_RECORD_SECTION** – lists information about the control file record sections.
- **SHOW PARAMETER CONTROL_FILES** command – lists the name, status, and location of control files.

The queries shown here were executed against the **DBORCL** database used for general instruction in our department.

```
CONNECT / AS SYSDBA
```

```
SELECT name
FROM v$controlfile;
```

```
NAME
```

```
---------------------------------------
/u01/oradata/DBORCL/DBORCLcontrol01.ctl
/u02/oradata/DBORCL/DBORCLcontrol02.ctl
/u03/oradata/DBORCL/DBORCLcontrol03.ctl


SELECT name, value FROM v$parameter
WHERE name='control_files';


NAME
-------------------------------------------------------
-------------------------
VALUE
-------------------------------------------------------
-------------------------
control_files
/u01/oradata/DBORCL/DBORCLcontrol01.ctl,
/u02/oradata/DBORCL/DBORCLcontrol02.ctl
, /u03/oradata/DBORCL/DBORCLcontrol03.ctl


DESC v$controlfile_record_section;


Name                       Null?    Type
```

```
 --------------------- --------
----------------------------

 TYPE                           VARCHAR2(28)

 RECORD_SIZE                    NUMBER

 RECORDS_TOTAL                  NUMBER

 RECORDS_USED                   NUMBER

 FIRST_INDEX                    NUMBER

 LAST_INDEX                     NUMBER

 LAST_RECID                     NUMBER
```

```
SELECT type, record_size, records_total, records_used

FROM v$controlfile_record_section

WHERE type='DATAFILE';
```

```
TYPE                         RECORD_SIZE RECORDS_TOTAL
RECORDS_USED

---------------------------- ----------- -------------
------------

DATAFILE                             428           100
4
```

The **RECORDS_TOTAL** shows the number of records allocated for the section that stores information on data files.

Several dynamic performance views display information from control files including: V$BACKUP, V$DATAFILE, V$TEMPFILE, V$TABLESPACE, V$ARCHIVE, V$LOG, V$LOGFILE, and others.

END OF NOTES