15

Using Oracle Managed Files

Oracle Database can manage the files that comprise the database.

- About Oracle Managed Files
  Oracle Managed Files eases database administration, reduces errors, and reduces wasted disk space.
- Enabling the Creation and Use of Oracle Managed Files
  You set certain initialization parameters to enable and use Oracle Managed Files.
- Creating Oracle Managed Files
  You can use Oracle Managed Files to create data files, temp files, control files, redo log files, and archived log.
- Operation of Oracle Managed Files
  The file names of Oracle Managed Files are accepted in SQL statements wherever a file name is used to identify an existing file.
- Scenarios for Using Oracle Managed Files
  Scenarios illustrate how to use Oracle Managed Files.

15.1 About Oracle Managed Files

Oracle Managed Files eases database administration, reduces errors, and reduces wasted disk space.

- What Is Oracle Managed Files?
  Using Oracle Managed Files simplifies the administration of an Oracle Database. Oracle Managed Files eliminates the need for you, the DBA, to directly manage the operating system files that comprise an Oracle Database.
- Who Can Use Oracle Managed Files?
  Oracle Managed Files is most useful for certain types of databases.
- What Is a Logical Volume Manager?
  A logical volume manager (LVM) is a software package available with most operating systems. Sometimes it is called a logical disk manager (LDM). It allows pieces of multiple physical disks to be combined into a single contiguous address space that appears as one disk to higher layers of software.
- What Is a File System?
  A file system is a data structure built inside a contiguous disk address space. A file manager (FM) is a software package that manipulates file systems, but it is sometimes called the file system.
- Benefits of Using Oracle Managed Files
  Oracle Managed Files provides several benefits.
- Oracle Managed Files and Existing Functionality
  Using Oracle Managed Files does not eliminate any existing functionality.

15.1.1 What Is Oracle Managed Files?

Using Oracle Managed Files simplifies the administration of an Oracle Database. Oracle Managed Files eliminates the need for you, the DBA, to directly manage the operating system files that comprise an Oracle Database.

With Oracle Managed Files, you specify file system directories in which the database automatically creates, names, and manages files at the database object level. For example, you need only specify that you want to create a tablespace; you do not need to specify the name and path of the tablespace's data file with the `DATAFILE` clause. This feature works well with a logical volume manager (LVM).

The database internally uses standard file system interfaces to create and delete files as needed for the following database structures:

- Tablespaces
- Redo log files
- Control files
- Archived logs
- Block change tracking files
- Flashback logs
- RMAN backups

Through initialization parameters, you specify the file system directory to be used for a particular type of file. The database then ensures that a unique file, an Oracle managed file, is created and deleted when no longer needed.

This feature does not affect the creation or naming of administrative files such as trace files, audit files, alert logs, and core files.

> **See Also:**
>
> *Oracle Automatic Storage Management Administrator's Guide* for information about Oracle Automatic Storage Management (Oracle ASM), the Oracle Database integrated file system and volume manager that extends the power of Oracle Managed Files. With Oracle Managed Files, files are created and managed automatically for you, but with Oracle ASM, you get the additional benefits of features such as striping, software mirroring, and dynamic storage configuration, without the need to purchase a third-party logical volume manager.

15.1.2 Who Can Use Oracle Managed Files?

Oracle Managed Files is most useful for certain types of databases.

Oracle Managed Files are most useful for the following types of databases:

- Databases that are supported by the following:
  - A logical volume manager that supports striping/RAID and dynamically extensible logical volumes

  - A file system that provides large, extensible files
- Low end or test databases

Because Oracle Managed Files require that you use the operating system file system, you lose control over how files are laid out on the disks, and thus, you lose some I/O tuning ability.

15.1.3 What Is a Logical Volume Manager?

A logical volume manager (LVM) is a software package available with most operating systems. Sometimes it is called a logical disk manager (LDM). It allows pieces of multiple physical disks to be combined into a single contiguous address space that appears as one disk to higher layers of software.

An LVM can make the logical volume have better capacity, performance, reliability, and availability characteristics than any of the underlying physical disks. It uses techniques such as mirroring, striping, concatenation, and RAID 5 to implement these characteristics.

Some LVMs allow the characteristics of a logical volume to be changed after it is created, even while it is in use. The volume may be resized or mirrored, or it may be relocated to different physical disks.

15.1.4 What Is a File System?

A file system is a data structure built inside a contiguous disk address space. A file manager (FM) is a software package that manipulates file systems, but it is sometimes called the file system.

All operating systems have file managers. The primary task of a file manager is to allocate and deallocate disk space into files within a file system.

A file system allows the disk space to be allocated to a large number of files. Each file is made to appear as a contiguous address space to applications such as Oracle Database. The files may not actually be contiguous within the disk space of the file system. Files can be created, read, written, resized, and deleted. Each file has a name associated with it that is used to refer to the file.

A file system is commonly built on top of a logical volume constructed by an LVM. Thus all the files in a particular file system have the same performance, reliability, and availability characteristics inherited from the underlying logical volume. A file system is a single pool of storage that is shared by all the files in the file system. If a file system is out of space, then none of the files in that file system can grow. Space available in one file system does not affect space in another file system. However some LVM/FM combinations allow space to be added or removed from a file system.

An operating system can support multiple file systems. Multiple file systems are constructed to give different storage characteristics to different files as well as to divide the available disk space into pools that do not affect each other.

15.1.5 Benefits of Using Oracle Managed Files

Oracle Managed Files provides several benefits.

Consider the following benefits of using Oracle Managed Files:

- They make the administration of the database easier.

There is no need to invent file names and define specific storage requirements. A consistent set of rules is used to name all relevant files. The file system defines the characteristics of the storage and the pool where it is allocated.

- They reduce corruption caused by administrators specifying the wrong file.

  Each Oracle managed file and file name is unique. Using the same file in two different databases is a common mistake that can cause very large down times and loss of committed transactions. Using two different names that refer to the same file is another mistake that causes major corruptions.

- They reduce wasted disk space consumed by obsolete files.

  Oracle Database automatically removes old Oracle Managed Files when they are no longer needed. Much disk space is wasted in large systems simply because no one is sure if a particular file is still required. This also simplifies the administrative task of removing files that are no longer required on disk and prevents the mistake of deleting the wrong file.

- They simplify creation of test and development databases.

  You can minimize the time spent making decisions regarding file structure and naming, and you have fewer file management tasks. You can focus better on meeting the actual requirements of your test or development database.

- Oracle Managed Files make development of portable third-party tools easier.

  Oracle Managed Files eliminate the need to put operating system specific file names in SQL scripts.

15.1.6 Oracle Managed Files and Existing Functionality

Using Oracle Managed Files does not eliminate any existing functionality.

Existing databases are able to operate as they always have. New files can be created as managed files while old ones are administered in the old way. Thus, a database can have a mixture of Oracle managed and unmanaged files.

15.2 Enabling the Creation and Use of Oracle Managed Files

You set certain initialization parameters to enable and use Oracle Managed Files.

- Initialization Parameters That Enable Oracle Managed Files
  The following table lists the initialization parameters that enable the use of Oracle Managed Files.
- Setting the DB_CREATE_FILE_DEST Initialization Parameter
  The `DB_CREATE_FILE_DEST` initialization parameter specifies the location of important database files.
- Setting the DB_RECOVERY_FILE_DEST Parameter
  Include the `DB_RECOVERY_FILE_DEST` and `DB_RECOVERY_FILE_DEST_SIZE` parameters in your initialization parameter file to identify the default location for the Fast Recovery Area.
- Setting the DB_CREATE_ONLINE_LOG_DEST_n Initialization Parameters
  The `DB_CREATE_ONLINE_LOG_DEST_`*`n`* initialization parameters specify the locations of the redo log files and the control files.

15.2.1 Initialization Parameters That Enable Oracle Managed Files

The following table lists the initialization parameters that enable the use of Oracle Managed Files.

Initialization Parameter	Description
DB_CREATE_FILE_DEST	Defines the location of the default file system directory or Oracle ASM disk group where the database creates data files or temp files when no file specification is given in the create operation. Also used as the default location for redo log and control files if DB_CREATE_ONLINE_LOG_DEST_*n* are not specified.
DB_CREATE_ONLINE_LOG_DEST_*n*	Defines the location of the default file system directory or Oracle ASM disk group for redo log files and control file creation when no file specification is given in the create operation. By changing *n*, you can use this initialization parameter multiple times, where *n* specifies a multiplexed copy of the redo log or control file. You can specify up to five multiplexed copies.
DB_RECOVERY_FILE_DEST	Defines the location of the Fast Recovery Area, which is the default file system directory or Oracle ASM disk group where the database creates RMAN backups when no format option is used, archived logs when no other local destination is configured, and flashback logs. Also used as the default location for redo log and control files or multiplexed copies of redo log and control files if DB_CREATE_ONLINE_LOG_DEST_*n* are not specified. When this parameter is specified, the DB_RECOVERY_FILE_DEST_SIZE initialization parameter must also be specified.

The file system directories specified by these parameters must already exist; the database does not create them. The directory must also have permissions to allow the database to create the files in it.

The default location is used whenever a location is not explicitly specified for the operation creating the file. The database creates the file name, and a file thus created is an Oracle managed file.

Both of these initialization parameters are dynamic, and can be set using the ALTER SYSTEM or ALTER SESSION statement.

See Also:

- *Oracle Database Reference* for additional information about initialization parameters
- "How Oracle Managed Files Are Named"

15.2.2 Setting the DB_CREATE_FILE_DEST Initialization Parameter

The `DB_CREATE_FILE_DEST` initialization parameter specifies the location of important database files.

Include the `DB_CREATE_FILE_DEST` initialization parameter in your initialization parameter file to identify the default location for the database server to create:

- Data files
- Temp files
- Redo log files
- Control files
- Block change tracking files

You specify the name of a file system directory that becomes the default location for the creation of the operating system files for these entities. The following example sets `/u01/app/oracle/oradata` as the default directory to use when creating Oracle Managed Files:

```
DB_CREATE_FILE_DEST = '/u01/app/oracle/oradata'
```

15.2.3 Setting the DB_RECOVERY_FILE_DEST Parameter

Include the `DB_RECOVERY_FILE_DEST` and `DB_RECOVERY_FILE_DEST_SIZE` parameters in your initialization parameter file to identify the default location for the Fast Recovery Area.

The Fast Recovery Area contains:

- Redo log files or multiplexed copies of redo log files
- Control files or multiplexed copies of control files
- RMAN backups (data file copies, control file copies, backup pieces, control file autobackups)
- Archived logs
- Flashback logs

You specify the name of file system directory that becomes the default location for creation of the operating system files for these entities. For example:

```
DB_RECOVERY_FILE_DEST = '/u01/app/oracle/fast_recovery_area'
DB_RECOVERY_FILE_DEST_SIZE = 20G
```

15.2.4 Setting the DB_CREATE_ONLINE_LOG_DEST_n Initialization Parameters

The `DB_CREATE_ONLINE_LOG_DEST_`*n* initialization parameters specify the locations of the redo log files and the control files.

Include the `DB_CREATE_ONLINE_LOG_DEST_`*n* initialization parameters in your initialization parameter file to identify the default locations for the database server to create:

- Redo log files
- Control files

You specify the name of a file system directory or Oracle ASM disk group that becomes the default location for the creation of the files for these entities. You can specify up to five multiplexed locations.

For the creation of redo log files and control files only, this parameter overrides any default location specified in the `DB_CREATE_FILE_DEST` and `DB_RECOVERY_FILE_DEST` initialization parameters. If you do not specify a `DB_CREATE_FILE_DEST` parameter, but you do specify the `DB_CREATE_ONLINE_LOG_DEST_n` parameter, then only redo log files and control files can be created as Oracle Managed Files.

It is recommended that you specify at least two parameters. For example:

```
DB_CREATE_ONLINE_LOG_DEST_1 = '/u02/oradata'
DB_CREATE_ONLINE_LOG_DEST_2 = '/u03/oradata'
```

This allows multiplexing, which provides greater fault-tolerance for the redo log and control file if one of the destinations fails.

15.3 Creating Oracle Managed Files

You can use Oracle Managed Files to create data files, temp files, control files, redo log files, and archived log.

- When Oracle Database Creates Oracle Managed Files
  Oracle Database creates Oracle Managed Files when certain conditions are met.
- How Oracle Managed Files Are Named
  The file names of Oracle Managed Files comply with the Optimal Flexible Architecture (OFA) standard for file naming.
- Creating Oracle Managed Files at Database Creation
  The `CREATE DATABASE` statement can perform actions related to Oracle Managed Files.
- Creating Data Files for Tablespaces Using Oracle Managed Files
  Oracle Database can create data files for tablespaces using Oracle Managed Files when certain conditions are met.
- Creating Temp Files for Temporary Tablespaces Using Oracle Managed Files
  Oracle Database can create temp files for temporary tablespaces using Oracle Managed Files when certain conditions are met.
- Creating Control Files Using Oracle Managed Files
  Oracle Database can create control files using Oracle Managed Files when certain conditions are met.
- Creating Redo Log Files Using Oracle Managed Files
  Redo log files are created at database creation time. They can also be created when you issue either of the following statements: `ALTER DATABASE ADD LOGFILE` and `ALTER DATABASE OPEN RESETLOGS`.
- Creating Archived Logs Using Oracle Managed Files
  Archived logs are created by a background process or by a SQL statement.

15.3.1 When Oracle Database Creates Oracle Managed Files

Oracle Database creates Oracle Managed Files when certain conditions are met.

If you have met any of the following conditions, then Oracle Database creates Oracle Managed Files for you, as appropriate, when no file specification is given in the create operation:

- You have included any of the `DB_CREATE_FILE_DEST`, `DB_RECOVERY_FILE_DEST`, or `DB_CREATE_ONLINE_LOG_DEST_n` initialization parameters in your initialization parameter file.
- You have issued the `ALTER SYSTEM` statement to dynamically set any of `DB_RECOVERY_FILE_DEST`, `DB_CREATE_FILE_DEST`, or `DB_CREATE_ONLINE_LOG_DEST_n` initialization parameters
- You have issued the `ALTER SESSION` statement to dynamically set any of the `DB_CREATE_FILE_DEST`, `DB_RECOVERY_FILE_DEST`, or `DB_CREATE_ONLINE_LOG_DEST_n` initialization parameters.

If a statement that creates an Oracle managed file finds an error or does not complete due to some failure, then any Oracle Managed Files created by the statement are automatically deleted as part of the recovery of the error or failure. However, because of the large number of potential errors that can occur with file systems and storage subsystems, there can be situations where you must manually remove the files using operating system commands.

15.3.2 How Oracle Managed Files Are Named

The file names of Oracle Managed Files comply with the Optimal Flexible Architecture (OFA) standard for file naming.

> **Note:**
>
> The naming scheme described in this section applies only to files created in operating system file systems. The naming scheme for files created in Oracle Automatic Storage Management (Oracle ASM) disk groups is described in *Oracle Automatic Storage Management Administrator's Guide*.

The assigned names are intended to meet the following requirements:

- Database files are easily distinguishable from all other files.
- Files of one database type are easily distinguishable from other database types.
- Files are clearly associated with important attributes specific to the file type. For example, a data file name may include the tablespace name to allow for easy association of data file to tablespace, or an archived log name may include the thread, sequence, and creation date.

No two Oracle Managed Files are given the same name. The name that is used for creation of an Oracle managed file is constructed from three sources:

- The default creation location
- A file name template that is chosen based on the type of the file. The template also depends on the operating system platform and whether or not Oracle Automatic Storage Management is used.
- A unique string created by Oracle Database or the operating system. This ensures that file creation does not damage an existing file and that the file cannot be mistaken for some other file.

As a specific example, file names for Oracle Managed Files have the following format on a Solaris file system:

```
destination_prefix/o1_mf_%t_%u_.dbf
```

where:

- *destination_prefix* is *destination_location*/*db_unique_name*/*datafile*

  where:

  - *destination_location* is the location specified in DB_CREATE_FILE_DEST
  - *db_unique_name* is the globally unique name (DB_UNIQUE_NAME initialization parameter) of the target database. If there is no DB_UNIQUE_NAME parameter, then the DB_NAME initialization parameter value is used.
- %t is the tablespace name.
- %u is an eight-character string that guarantees uniqueness

For example, assume the following parameter settings:

```
DB_CREATE_FILE_DEST   = /u01/app/oracle/oradata
DB_UNIQUE_NAME = PAYROLL
```

Then an example data file name would be:

```
/u01/app/oracle/oradata/PAYROLL/datafile/o1_mf_tbs1_2ixh90q_.dbf
```

Names for other file types are similar. Names on other platforms are also similar, subject to the constraints of the naming rules of the platform.

The examples on the following pages use Oracle managed file names as they might appear with a Solaris file system as an OMF destination.

> **Note:**
>
> The database identifies an Oracle managed file based on its name. If you rename the file, the database is no longer able to recognize it as an Oracle managed file and will not manage the file accordingly.

15.3.3 Creating Oracle Managed Files at Database Creation

The CREATE DATABASE statement can perform actions related to Oracle Managed Files.

> **Note:**
>
> The rules and defaults in this section also apply to creating a database with Database Configuration Assistant (DBCA). With DBCA, you use a graphical interface to enable Oracle Managed Files and to specify file locations that correspond to the initialization parameters described in this section.

- Specifying Control Files at Database Creation
  At database creation, the control file is created in the files specified by the CONTROL_FILES initialization parameter.

- Specifying Redo Log Files at Database Creation
  The `LOGFILE` clause is not required in the `CREATE DATABASE` statement, and omitting it provides a simple means of creating Oracle managed redo log files.
- Specifying the SYSTEM and SYSAUX Tablespace Data Files at Database Creation
  The `DATAFILE` or `SYSAUX DATAFILE` clause is not required in the `CREATE DATABASE` statement, and omitting it provides a simple means of creating Oracle managed data files for the `SYSTEM` and `SYSAUX` tablespaces.
- Specifying the Undo Tablespace Data File at Database Creation
  The `DATAFILE` subclause of the `UNDO TABLESPACE` clause is optional and a file name is not required in the file specification.
- Specifying the Default Temporary Tablespace Temp File at Database Creation
  The `TEMPFILE` subclause is optional for the `DEFAULT TEMPORARY TABLESPACE` clause and a file name is not required in the file specification.
- CREATE DATABASE Statement Using Oracle Managed Files: Examples
  Examples illustrate creating a database with the `CREATE DATABASE` statement when using the Oracle Managed Files feature.

> **See Also:**
>
> *Oracle Database SQL Language Reference* for a description of the `CREATE DATABASE` statement

15.3.3.1 Specifying Control Files at Database Creation

At database creation, the control file is created in the files specified by the `CONTROL_FILES` initialization parameter.

If the `CONTROL_FILES` parameter is not set and at least one of the initialization parameters required for the creation of Oracle Managed Files is set, then an Oracle managed control file is created in the default control file destinations. In order of precedence, the default destination is defined as follows:

- One or more control files as specified in the `DB_CREATE_ONLINE_LOG_DEST_`*n* initialization parameter. The file in the first directory is the primary control file. When `DB_CREATE_ONLINE_LOG_DEST_`*n* is specified, the database does not create a control file in `DB_CREATE_FILE_DEST` or in `DB_RECOVERY_FILE_DEST` (the Fast Recovery Area).
- If no value is specified for `DB_CREATE_ONLINE_LOG_DEST_`*n*, but values are set for both the `DB_CREATE_FILE_DEST` and `DB_RECOVERY_FILE_DEST`, then the database creates one control file in each location. The location specified in `DB_CREATE_FILE_DEST` is the primary control file.
- If a value is specified only for `DB_CREATE_FILE_DEST`, then the database creates one control file in that location.
- If a value is specified only for `DB_RECOVERY_FILE_DEST`, then the database creates one control file in that location.

If the `CONTROL_FILES` parameter is not set and none of these initialization parameters are set, then the Oracle Database default action is operating system dependent. At least one copy of a control file is created in an operating system dependent default location. Any copies of control files created in this fashion are not Oracle Managed Files, and you must add a `CONTROL_FILES` initialization parameter to any initialization parameter file.

If the database creates an Oracle managed control file, and if there is a server parameter file, then the database creates a CONTROL_FILES initialization parameter entry in the server parameter file. If there is no server parameter file, then you must manually include a CONTROL_FILES initialization parameter entry in the text initialization parameter file.

> **See Also:**
>
> Managing Control Files

15.3.3.2 Specifying Redo Log Files at Database Creation

The LOGFILE clause is not required in the CREATE DATABASE statement, and omitting it provides a simple means of creating Oracle managed redo log files.

If the LOGFILE clause is omitted, then redo log files are created in the default redo log file destinations. In order of precedence, the default destination is defined as follows:

- If either the DB_CREATE_ONLINE_LOG_DEST_*n* is set, then the database creates a log file member in each directory specified, up to the value of the MAXLOGMEMBERS initialization parameter.
- If the DB_CREATE_ONLINE_LOG_DEST_*n* parameter is not set, but both the DB_CREATE_FILE_DEST and DB_RECOVERY_FILE_DEST initialization parameters are set, then the database creates one Oracle managed log file member in each of those locations. The log file in the DB_CREATE_FILE_DEST destination is the first member.
- If only the DB_CREATE_FILE_DEST initialization parameter is specified, then the database creates a log file member in that location.
- If only the DB_RECOVERY_FILE_DEST initialization parameter is specified, then the database creates a log file member in that location.

The default size of an Oracle managed redo log file is 100 MB.

Optionally, you can create Oracle managed redo log files, and override default attributes, by including the LOGFILE clause but omitting a file name. Redo log files are created the same way, except for the following: If no file name is provided in the LOGFILE clause of CREATE DATABASE, and none of the initialization parameters required for creating Oracle Managed Files are provided, then the CREATE DATABASE statement fails.

> **See Also:**
>
> " Managing the Redo Log"

15.3.3.3 Specifying the SYSTEM and SYSAUX Tablespace Data Files at Database Creation

The DATAFILE or SYSAUX DATAFILE clause is not required in the CREATE DATABASE statement, and omitting it provides a simple means of creating Oracle managed data files for the SYSTEM and SYSAUX tablespaces.

If the DATAFILE clause is omitted, then one of the following actions occurs:

- If DB_CREATE_FILE_DEST is set, then one Oracle managed data file for the SYSTEM tablespace and another for the SYSAUX tablespace are created in the DB_CREATE_FILE_DEST directory.
- If DB_CREATE_FILE_DEST is not set, then the database creates one SYSTEM and one SYSAUX tablespace data file whose names and sizes are operating system dependent. Any SYSTEM or SYSAUX tablespace data file created in this manner is not an Oracle managed file.

By default, Oracle managed data files, including those for the SYSTEM and SYSAUX tablespaces, are 100MB and autoextensible. When autoextension is required, the database extends the data file by its existing size or 100 MB, whichever is smaller. You can also explicitly specify the autoextensible unit using the NEXT parameter of the STORAGE clause when you specify the data file (in a CREATE or ALTER TABLESPACE operation).

Optionally, you can create an Oracle managed data file for the SYSTEM or SYSAUX tablespace and override default attributes. This is done by including the DATAFILE clause, omitting a file name, but specifying overriding attributes. When a file name is not supplied and the DB_CREATE_FILE_DEST parameter is set, an Oracle managed data file for the SYSTEM or SYSAUX tablespace is created in the DB_CREATE_FILE_DEST directory with the specified attributes being overridden. However, if a file name is not supplied and the DB_CREATE_FILE_DEST parameter is not set, then the CREATE DATABASE statement fails.

When overriding the default attributes of an Oracle managed file, if a SIZE value is specified but no AUTOEXTEND clause is specified, then the data file is *not* autoextensible.

15.3.3.4 Specifying the Undo Tablespace Data File at Database Creation

The DATAFILE subclause of the UNDO TABLESPACE clause is optional and a file name is not required in the file specification.

If a file name is not supplied and the DB_CREATE_FILE_DEST parameter is set, then an Oracle managed data file is created in the DB_CREATE_FILE_DEST directory. If DB_CREATE_FILE_DEST is not set, then the statement fails with a syntax error.

The UNDO TABLESPACE clause itself is optional in the CREATE DATABASE statement. If it is not supplied, and automatic undo management mode is enabled (the default), then a default undo tablespace named SYS_UNDOTS is created and a 20 MB data file that is autoextensible is allocated as follows:

- If DB_CREATE_FILE_DEST is set, then an Oracle managed data file is created in the indicated directory.
- If DB_CREATE_FILE_DEST is not set, then the data file location is operating system specific.

> **See Also:**
>
> " Managing Undo "

15.3.3.5 Specifying the Default Temporary Tablespace Temp File at Database Creation

The `TEMPFILE` subclause is optional for the `DEFAULT TEMPORARY TABLESPACE` clause and a file name is not required in the file specification.

If a file name is not supplied and the `DB_CREATE_FILE_DEST` parameter set, then an Oracle managed temp file is created in the `DB_CREATE_FILE_DEST` directory. If `DB_CREATE_FILE_DEST` is not set, then the `CREATE DATABASE` statement fails with a syntax error.

The `DEFAULT TEMPORARY TABLESPACE` clause itself is optional. If it is not specified, then no default temporary tablespace is created.

The default size for an Oracle managed temp file is 100 MB and the file is autoextensible with an unlimited maximum size.

15.3.3.6 CREATE DATABASE Statement Using Oracle Managed Files: Examples

Examples illustrate creating a database with the `CREATE DATABASE` statement when using the Oracle Managed Files feature.

CREATE DATABASE: Example 1

This example creates a database with the following Oracle Managed Files:

- A `SYSTEM` tablespace data file in directory `/u01/app/oracle/oradata` that is autoextensible up to an unlimited size.
- A `SYSAUX` tablespace data file in directory `/u01/app/oracle/oradata` that is autoextensible up to an unlimited size. The tablespace is locally managed with automatic segment-space management.
- Two online log groups with two members of 100 MB each, one each in `/u02/oradata` and `/u03/oradata`.
- If automatic undo management mode is enabled (the default), then an undo tablespace data file in directory `/u01/app/oracle/oradata` that is 20 MB and autoextensible up to an unlimited size. An undo tablespace named `SYS_UNDOTS` is created.
- If no `CONTROL_FILES` initialization parameter is specified, then two control files, one each in `/u02/oradata` and `/u03/oradata`. The control file in `/u02/oradata` is the primary control file.

The following parameter settings relating to Oracle Managed Files, are included in the initialization parameter file:

```
DB_CREATE_FILE_DEST = '/u01/app/oracle/oradata'
DB_CREATE_ONLINE_LOG_DEST_1 = '/u02/oradata'
DB_CREATE_ONLINE_LOG_DEST_2 = '/u03/oradata'
```

The following statement is issued at the SQL prompt:

```
CREATE DATABASE sample;
```

To create the database with a locally managed SYSTEM tablespace, add the EXTENT MANAGEMENT LOCAL clause:

```
CREATE DATABASE sample EXTENT MANAGEMENT LOCAL;
```

Without this clause, the SYSTEM tablespace is dictionary managed. Oracle recommends that you create a locally managed SYSTEM tablespace.

CREATE DATABASE: Example 2

This example creates a database with the following Oracle Managed Files:

- A SYSTEM tablespace data file in directory /u01/app/oracle/oradata that is autoextensible up to an unlimited size.
- A SYSAUX tablespace data file in directory /u01/app/oracle/oradata that is autoextensible up to an unlimited size. The tablespace is locally managed with automatic segment-space management.
- Two redo log files of 100 MB each in directory /u01/app/oracle/oradata. They are not multiplexed.
- An undo tablespace data file in directory /u01/app/oracle/oradata that is 20 MB and autoextensible up to an unlimited size. An undo tablespace named SYS_UNDOTS is created.
- A control file in /u01/app/oracle/oradata.

In this example, it is assumed that:

- No DB_CREATE_ONLINE_LOG_DEST_*n* initialization parameters are specified in the initialization parameter file.
- No CONTROL_FILES initialization parameter was specified in the initialization parameter file.
- Automatic undo management mode is enabled.

The following statements are issued at the SQL prompt:

```
ALTER SYSTEM SET DB_CREATE_FILE_DEST = '/u01/app/oracle/oradata';
CREATE DATABASE sample2 EXTENT MANAGEMENT LOCAL;
```

This database configuration is not recommended for a production database. The example illustrates how a very low-end database or simple test database can easily be created. To better protect this database from failures, at least one more control file should be created and the redo log should be multiplexed.

CREATE DATABASE: Example 3

In this example, the file size for the Oracle Managed Files for the default temporary tablespace and undo tablespace are specified. A database with the following Oracle Managed Files is created:

- A 400 MB SYSTEM tablespace data file in directory /u01/app/oracle/oradata. Because SIZE is specified, the file in not autoextensible.
- A 200 MB SYSAUX tablespace data file in directory /u01/app/oracle/oradata. Because SIZE is specified, the file in not autoextensible. The tablespace is locally managed with automatic segment-space management.
- Two redo log groups with two members of 100 MB each, one each in directories /u02/oradata and /u03/oradata.

- For the default temporary tablespace `dflt_ts`, a 10 MB temp file in directory `/u01/app/oracle/oradata`. Because `SIZE` is specified, the file in not autoextensible.
- For the undo tablespace `undo_ts`, a 100 MB data file in directory `/u01/app/oracle/oradata`. Because `SIZE` is specified, the file is not autoextensible.
- If no `CONTROL_FILES` initialization parameter was specified, then two control files, one each in directories `/u02/oradata` and `/u03/oradata`. The control file in `/u02/oradata` is the primary control file.

The following parameter settings are included in the initialization parameter file:

```
DB_CREATE_FILE_DEST = '/u01/app/oracle/oradata'
DB_CREATE_ONLINE_LOG_DEST_1 = '/u02/oradata'
DB_CREATE_ONLINE_LOG_DEST_2 = '/u03/oradata'
```

The following statement is issued at the SQL prompt:

```
CREATE DATABASE sample3
EXTENT MANAGEMENT LOCAL
DATAFILE SIZE 400M
SYSAUX DATAFILE SIZE 200M
DEFAULT TEMPORARY TABLESPACE dflt_ts TEMPFILE SIZE 10M
UNDO TABLESPACE undo_ts DATAFILE SIZE 100M;
```

> **See Also:**
>
> *Oracle Multitenant Administrator's Guide*

15.3.4 Creating Data Files for Tablespaces Using Oracle Managed Files

Oracle Database can create data files for tablespaces using Oracle Managed Files when certain conditions are met.

- About Creating Data Files for Tablespaces Using Oracle Managed Files
  When certain conditions are met, the following SQL statements can create data files for tablespaces using Oracle Managed Files: `CREATE TABLESPACE`, `CREATE UNDO TABLESPACE`, and `ALTER TABLESPACE ... ADD DATAFILE`.
- CREATE TABLESPACE: Examples
  Examples illustrate creating tablespaces with Oracle Managed Files.
- CREATE UNDO TABLESPACE: Example
  An example illustrates creating an undo tablespace.
- ALTER TABLESPACE: Example
  An example illustrates adding an Oracle managed autoextensible data file to a tablespace.

15.3.4.1 About Creating Data Files for Tablespaces Using Oracle Managed Files

When certain conditions are met, the following SQL statements can create data files for tablespaces using Oracle Managed Files: `CREATE TABLESPACE`, `CREATE UNDO TABLESPACE`, and `ALTER TABLESPACE ... ADD DATAFILE`.

The following statements can create data files:

- `CREATE TABLESPACE`

- `CREATE UNDO TABLESPACE`
- `ALTER TABLESPACE ... ADD DATAFILE`

When creating a tablespace, either a permanent tablespace or an undo tablespace, the `DATAFILE` clause is optional. When you include the `DATAFILE` clause, the file name is optional. If the `DATAFILE` clause or file name is not provided, then the following rules apply:

- If the `DB_CREATE_FILE_DEST` initialization parameter is specified, then an Oracle managed data file is created in the location specified by the parameter.
- If the `DB_CREATE_FILE_DEST` initialization parameter is not specified, then the statement creating the data file fails.

When you add a data file to a tablespace with the `ALTER TABLESPACE...ADD DATAFILE` statement the file name is optional. If the file name is not specified, then the same rules apply as discussed in the previous paragraph.

By default, an Oracle managed data file for a permanent tablespace is 100 MB and is autoextensible with an unlimited maximum size. However, if in your `DATAFILE` clause you override these defaults by specifying a `SIZE` value (and no `AUTOEXTEND` clause), then the data file is *not* autoextensible.

> **See Also:**
>
> - "Specifying the SYSTEM and SYSAUX Tablespace Data Files at Database Creation"
> - "Specifying the Undo Tablespace Data File at Database Creation"
> - " Managing Tablespaces"

15.3.4.2 CREATE TABLESPACE: Examples

Examples illustrate creating tablespaces with Oracle Managed Files.

> **See Also:**
>
> *Oracle Database SQL Language Reference* for a description of the `CREATE TABLESPACE` statement

CREATE TABLESPACE: Example 1

The following example sets the default location for data file creations to `/u01/oradata` and then creates a tablespace `tbs_1` with a data file in that location. The data file is 100 MB and is autoextensible with an unlimited maximum size.

```
SQL> ALTER SYSTEM SET DB_CREATE_FILE_DEST = '/u01/oradata';
SQL> CREATE TABLESPACE tbs_1;
```

CREATE TABLESPACE: Example 2

This example creates a tablespace named `tbs_2` with a data file in the directory `/u01/oradata`. The data file initial size is 400 MB, and because the SIZE clause is specified, the data file is not autoextensible.

The following parameter setting is included in the initialization parameter file:

```
DB_CREATE_FILE_DEST = '/u01/oradata'
```

The following statement is issued at the SQL prompt:

```
SQL> CREATE TABLESPACE tbs_2 DATAFILE SIZE 400M;
```

CREATE TABLESPACE: Example 3

This example creates a tablespace named `tbs_3` with an autoextensible data file in the directory `/u01/oradata` with a maximum size of 800 MB and an initial size of 100 MB:

The following parameter setting is included in the initialization parameter file:

```
DB_CREATE_FILE_DEST = '/u01/oradata'
```

The following statement is issued at the SQL prompt:

```
SQL> CREATE TABLESPACE tbs_3 DATAFILE AUTOEXTEND ON MAXSIZE 800M;
```

CREATE TABLESPACE: Example 4

The following example sets the default location for data file creations to `/u01/oradata` and then creates a tablespace named `tbs_4` in that directory with two data files. Both data files have an initial size of 200 MB, and because a `SIZE` value is specified, they are not autoextensible

```
SQL> ALTER SYSTEM SET DB_CREATE_FILE_DEST = '/u01/oradata';
SQL> CREATE TABLESPACE tbs_4 DATAFILE SIZE 200M, SIZE 200M;
```

15.3.4.3 CREATE UNDO TABLESPACE: Example

An example illustrates creating an undo tablespace.

The following example creates an undo tablespace named `undotbs_1` with a data file in the directory `/u01/oradata`. The data file for the undo tablespace is 100 MB and is autoextensible with an unlimited maximum size.

1. Set the following initialization parameter:

   ```
   DB_CREATE_FILE_DEST = '/u01/oradata'
   ```

2. Issue the following SQL statement:

   ```
   SQL> CREATE UNDO TABLESPACE undotbs_1;
   ```

> **See Also:**
>
> *Oracle Database SQL Language Reference* for a description of the `CREATE UNDO TABLESPACE` statement

15.3.4.4 ALTER TABLESPACE: Example

An example illustrates adding an Oracle managed autoextensible data file to a tablespace.

This example adds an Oracle managed autoextensible data file to the `tbs_1` tablespace. The data file has an initial size of 100 MB and a maximum size of 800 MB.

1. Set the following initialization parameter:

   ```
   DB_CREATE_FILE_DEST = '/u01/oradata'
   ```

2. Issue the following SQL statement:

   ```
   SQL> ALTER TABLESPACE tbs_1 ADD DATAFILE AUTOEXTEND ON MAXSIZE 800M;
   ```

> **See Also:**
>
> *Oracle Database SQL Language Reference* for a description of the `ALTER TABLESPACE` statement

15.3.5 Creating Temp Files for Temporary Tablespaces Using Oracle Managed Files

Oracle Database can create temp files for temporary tablespaces using Oracle Managed Files when certain conditions are met.

- About Creating Temp Files for Temporary Tablespaces Using Oracle Managed Files
  When certain conditions are met, the following SQL statements can create temp files for tablespaces using Oracle Managed Files: `CREATE TEMPORARY TABLESPACE` and `ALTER TABLESPACE ... ADD TEMPFILE`.
- CREATE TEMPORARY TABLESPACE: Example
  An example illustrates creating a temporary tablespace.
- ALTER TABLESPACE... ADD TEMPFILE: Example
  An example illustrates adding a temp file to a temporary tablespace.

15.3.5.1 About Creating Temp Files for Temporary Tablespaces Using Oracle Managed Files

When certain conditions are met, the following SQL statements can create temp files for tablespaces using Oracle Managed Files: `CREATE TEMPORARY TABLESPACE` and `ALTER TABLESPACE ... ADD TEMPFILE`.

The following statements that create temp files are relevant to the discussion in this section:

- `CREATE TEMPORARY TABLESPACE`
- `ALTER TABLESPACE ... ADD TEMPFILE`

When creating a temporary tablespace the `TEMPFILE` clause is optional. If you include the `TEMPFILE` clause, then the file name is optional. If the `TEMPFILE` clause or file name is not provided, then the following rules apply:

- If the DB_CREATE_FILE_DEST initialization parameter is specified, then an Oracle managed temp file is created in the location specified by the parameter.
- If the DB_CREATE_FILE_DEST initialization parameter is not specified, then the statement creating the temp file fails.

When you add a temp file to a tablespace with the ALTER TABLESPACE...ADD TEMPFILE statement the file name is optional. If the file name is not specified, then the same rules apply as discussed in the previous paragraph.

When overriding the default attributes of an Oracle managed file, if a SIZE value is specified but no AUTOEXTEND clause is specified, then the data file is *not* autoextensible.

> **See Also:**
>
> "Specifying the Default Temporary Tablespace Temp File at Database Creation"

15.3.5.2 CREATE TEMPORARY TABLESPACE: Example

An example illustrates creating a temporary tablespace.

The following example sets the default location for data file creations to /u01/oradata and then creates a tablespace named temptbs_1 with a temp file in that location. The temp file is 100 MB and is autoextensible with an unlimited maximum size.

```
SQL> ALTER SYSTEM SET DB_CREATE_FILE_DEST = '/u01/oradata';
SQL> CREATE TEMPORARY TABLESPACE temptbs_1;
```

> **See Also:**
>
> *Oracle Database SQL Language Reference* for a description of the CREATE TABLESPACE statement

15.3.5.3 ALTER TABLESPACE... ADD TEMPFILE: Example

An example illustrates adding a temp file to a temporary tablespace.

The following example sets the default location for data file creations to /u03/oradata and then adds a temp file in the default location to a tablespace named temptbs_1. The temp file initial size is 100 MB. It is autoextensible with an unlimited maximum size.

```
SQL> ALTER SYSTEM SET DB_CREATE_FILE_DEST = '/u03/oradata';
SQL> ALTER TABLESPACE TBS_1 ADD TEMPFILE;
```

> **See Also:**
>
> *Oracle Database SQL Language Reference* for a description of the ALTER TABLESPACE statement

15.3.6 Creating Control Files Using Oracle Managed Files

Oracle Database can create control files using Oracle Managed Files when certain conditions are met.

- About Creating Control Files Using Oracle Managed Files
  When certain conditions are met, the `CREATE CONTROLFILE` SQL statements can create control files using Oracle Managed Files.
- CREATE CONTROLFILE Using NORESETLOGS Keyword: Example
  An example illustrates creating a control file using the `CREATE CONTROLFILE` statement with the `NORESETLOGS` keyword.
- CREATE CONTROLFILE Using RESETLOGS Keyword: Example
  An example illustrates creating a control file using the `CREATE CONTROLFILE` statement with the `RESETLOGS` keyword.

15.3.6.1 About Creating Control Files Using Oracle Managed Files

When certain conditions are met, the `CREATE CONTROLFILE` SQL statements can create control files using Oracle Managed Files.

When you issue the `CREATE CONTROLFILE` statement, a control file is created (or reused, if `REUSE` is specified) in the files specified by the `CONTROL_FILES` initialization parameter. If the `CONTROL_FILES` parameter is not set, then the control file is created in the default control file destinations. The default destination is determined according to the precedence documented in "Specifying Control Files at Database Creation".

If Oracle Database creates an Oracle managed control file, and there is a server parameter file, then the database creates a `CONTROL_FILES` initialization parameter for the server parameter file. If there is no server parameter file, then you must create a `CONTROL_FILES` initialization parameter manually and include it in the initialization parameter file.

If the data files in the database are Oracle Managed Files, then the database-generated file names for the files must be supplied in the `DATAFILE` clause of the statement.

If the redo log files are Oracle Managed Files, then the `NORESETLOGS` or `RESETLOGS` keyword determines what can be supplied in the `LOGFILE` clause:

- If the `NORESETLOGS` keyword is used, then the database-generated file names for the Oracle managed redo log files must be supplied in the `LOGFILE` clause.
- If the `RESETLOGS` keyword is used, then the redo log file names can be supplied as with the `CREATE DATABASE` statement. See "Specifying Redo Log Files at Database Creation".

The sections that follow contain examples of using the `CREATE CONTROLFILE` statement with Oracle Managed Files.

See Also:

- *Oracle Database SQL Language Reference* for a description of the `CREATE CONTROLFILE` statement
- "Specifying Control Files at Database Creation"

15.3.6.2 CREATE CONTROLFILE Using NORESETLOGS Keyword: Example

An example illustrates creating a control file using the `CREATE CONTROLFILE` statement with the `NORESETLOGS` keyword.

The following `CREATE CONTROLFILE` statement is generated by an `ALTER DATABASE BACKUP CONTROLFILE TO TRACE` statement for a database with Oracle managed data files and redo log files:

```
CREATE CONTROLFILE
     DATABASE sample
     LOGFILE
       GROUP 1 ('/u01/oradata/SAMPLE/onlinelog/o1_mf_1_o220rtt9_.log',
                '/u02/oradata/SAMPLE/onlinelog/o1_mf_1_v2o0b2i3_.log')
                 SIZE 100M,
       GROUP 2 ('/u01/oradata/SAMPLE/onlinelog/o1_mf_2_p22056iw_.log',
                '/u02/oradata/SAMPLE/onlinelog/o1_mf_2_p02rcyg3_.log')
                 SIZE 100M
     NORESETLOGS
     DATAFILE '/u01/oradata/SAMPLE/datafile/o1_mf_system_xu34ybm2_.dbf'
              SIZE 100M,
              '/u01/oradata/SAMPLE/datafile/o1_mf_sysaux_aawbmz51_.dbf'
              SIZE 100M,
              '/u01/oradata/SAMPLE/datafile/o1_mf_sys_undo_apqbmz51_.dbf'
              SIZE 100M
     MAXLOGFILES 5
     MAXLOGHISTORY 100
     MAXDATAFILES 10
     MAXINSTANCES 2
     ARCHIVELOG;
```

15.3.6.3 CREATE CONTROLFILE Using RESETLOGS Keyword: Example

An example illustrates creating a control file using the `CREATE CONTROLFILE` statement with the `RESETLOGS` keyword.

The following is an example of a `CREATE CONTROLFILE` statement with the `RESETLOGS` option. Some combination of `DB_CREATE_FILE_DEST`, `DB_RECOVERY_FILE_DEST`, and `DB_CREATE_ONLINE_LOG_DEST_n` or must be set.

```
CREATE CONTROLFILE
     DATABASE sample
     RESETLOGS
     DATAFILE '/u01/oradata/SAMPLE/datafile/o1_mf_system_aawbmz51_.dbf',
              '/u01/oradata/SAMPLE/datafile/o1_mf_sysaux_axybmz51_.dbf',
              '/u01/oradata/SAMPLE/datafile/o1_mf_sys_undo_azzbmz51_.dbf'
     SIZE 100M
     MAXLOGFILES 5
     MAXLOGHISTORY 100
     MAXDATAFILES 10
     MAXINSTANCES 2
     ARCHIVELOG;
```

Later, you must issue the `ALTER DATABASE OPEN RESETLOGS` statement to re-create the redo log files. This is discussed in "Using the ALTER DATABASE OPEN RESETLOGS Statement". If the previous log files are Oracle Managed Files, then they are not deleted.

15.3.7 Creating Redo Log Files Using Oracle Managed Files

Redo log files are created at database creation time. They can also be created when you issue either of the following statements: `ALTER DATABASE ADD LOGFILE` and `ALTER DATABASE OPEN RESETLOGS`.

- Using the ALTER DATABASE ADD LOGFILE Statement
  The `ALTER DATABASE ADD LOGFILE` statement lets you later add a new group to your current redo log.
- Using the ALTER DATABASE OPEN RESETLOGS Statement
  If you previously created a control file specifying `RESETLOGS` and either did not specify file names or specified nonexistent file names, then the database creates redo log files for you when you issue the `ALTER DATABASE OPEN RESETLOGS` statement.

> **See Also:**
>
> *Oracle Database SQL Language Reference* for a description of the `ALTER DATABASE` statement

15.3.7.1 Using the ALTER DATABASE ADD LOGFILE Statement

The `ALTER DATABASE ADD LOGFILE` statement lets you later add a new group to your current redo log.

The file name in the `ADD LOGFILE` clause is optional if you are using Oracle Managed Files. If a file name is not provided, then a redo log file is created in the default log file destination. The default destination is determined according to the precedence documented in "Specifying Redo Log Files at Database Creation".

If a file name is not provided and you have not provided one of the initialization parameters required for creating Oracle Managed Files, then the statement returns an error.

The default size for an Oracle managed log file is 100 MB.

You continue to add and drop redo log file members by specifying complete file names.

> **See Also:**
>
> - "Specifying Redo Log Files at Database Creation"
> - "About Creating Control Files Using Oracle Managed Files"

Adding New Redo Log Files: Example

The following example creates a log group with a member in `/u01/oradata` and another member in `/u02/oradata`. The size of each log file is 100 MB.

The following parameter settings are included in the initialization parameter file:

```
DB_CREATE_ONLINE_LOG_DEST_1 = '/u01/oradata'
DB_CREATE_ONLINE_LOG_DEST_2 = '/u02/oradata'
```

The following statement is issued at the SQL prompt:

```
SQL> ALTER DATABASE ADD LOGFILE;
```

15.3.7.2 Using the ALTER DATABASE OPEN RESETLOGS Statement

If you previously created a control file specifying `RESETLOGS` and either did not specify file names or specified nonexistent file names, then the database creates redo log files for you when you issue the `ALTER DATABASE OPEN RESETLOGS` statement.

The rules for determining the directories in which to store redo log files, when none are specified in the control file, are the same as those discussed in "Specifying Redo Log Files at Database Creation".

15.3.8 Creating Archived Logs Using Oracle Managed Files

Archived logs are created by a background process or by a SQL statement.

Archived logs are created in the `DB_RECOVERY_FILE_DEST` location when:

- The `ARC` or `LGWR` background process archives an online redo log or
- An `ALTER SYSTEM ARCHIVE LOG CURRENT` statement is issued.

For example, assume that the following parameter settings are included in the initialization parameter file:

```
DB_RECOVERY_FILE_DEST_SIZE = 20G
DB_RECOVERY_FILE_DEST      = '/u01/oradata'
LOG_ARCHIVE_DEST_1         = 'LOCATION=USE_DB_RECOVERY_FILE_DEST'
```

15.4 Operation of Oracle Managed Files

The file names of Oracle Managed Files are accepted in SQL statements wherever a file name is used to identify an existing file.

These file names, like other file names, are stored in the control file and, if using Recovery Manager (RMAN) for backup and recovery, in the RMAN catalog. They are visible in all of the usual fixed and dynamic performance views that are available for monitoring data files and temp files (for example, `V$DATAFILE` or `DBA_DATA_FILES`).

The following are some examples of statements using database-generated file names:

```
SQL> ALTER DATABASE
  2> RENAME FILE '/u01/oradata/mydb/datafile/o1_mf_tbs01_ziw3bopb_.dbf'
  3> TO '/u01/oradata/mydb/tbs0101.dbf';

SQL> ALTER DATABASE
  2> DROP LOGFILE '/u01/oradata/mydb/onlinelog/o1_mf_1_wo94n2xi_.log';

SQL> ALTER TABLE emp
  2> ALLOCATE EXTENT
  3> (DATAFILE '/u01/oradata/mydb/datafile/o1_mf_tbs1_2ixfh90q_.dbf');
```

You can backup and restore Oracle managed data files, temp files, and control files as you would corresponding non Oracle Managed Files. Using database-generated file names does

not impact the use of logical backup files such as export files. This is particularly important for tablespace point-in-time recovery (TSPITR) and transportable tablespace export files.

There are some cases where Oracle Managed Files behave differently, including operations that drop files or rename file, and operations involving standby databases.

- Dropping Data Files and Temp Files
  Unlike files that are not managed by the database, when an Oracle managed data file or temp file is dropped, the file name is removed from the control file and the file is automatically deleted from the file system.
- Dropping Redo Log Files
  When an Oracle managed redo log file is dropped, its Oracle Managed Files are deleted. You specify the group or members to be dropped.
- Renaming Files
  With Oracle Managed Files, SQL statements that rename files do not actually rename the files on the operating system, but rather, the names in the control file are changed.
- Managing Standby Databases
  The data files, control files, and redo log files in a standby database can be managed by the database. This is independent of whether Oracle Managed Files are used on the primary database.

15.4.1 Dropping Data Files and Temp Files

Unlike files that are not managed by the database, when an Oracle managed data file or temp file is dropped, the file name is removed from the control file and the file is automatically deleted from the file system.

The statements that delete Oracle Managed Files when they are dropped are:

- `DROP TABLESPACE`
- `ALTER DATABASE TEMPFILE ... DROP`

You can also use these statements, which always delete files, Oracle managed or not:

- `ALTER TABLESPACE ... DROP DATAFILE`
- `ALTER TABLESPACE ... DROP TEMPFILE`

15.4.2 Dropping Redo Log Files

When an Oracle managed redo log file is dropped, its Oracle Managed Files are deleted. You specify the group or members to be dropped.

The following statements drop and delete redo log files:

- `ALTER DATABASE DROP LOGFILE`
- `ALTER DATABASE DROP LOGFILE MEMBER`

15.4.3 Renaming Files

With Oracle Managed Files, SQL statements that rename files do not actually rename the files on the operating system, but rather, the names in the control file are changed.

The following statements are used to rename files:

- `ALTER DATABASE RENAME FILE`

- `ALTER TABLESPACE ... RENAME DATAFILE`

You must specify each file name using the conventions for file names on your operating system when you issue this statement.

> **Note:**
>
> If the old file is an Oracle managed file and it exists, then it is deleted.

15.4.4 Managing Standby Databases

The data files, control files, and redo log files in a standby database can be managed by the database. This is independent of whether Oracle Managed Files are used on the primary database.

When recovery of a standby database encounters redo for the creation of a data file, if the data file is an Oracle managed file, then the recovery process creates an empty file in the local default file system location. This allows the redo for the new file to be applied immediately without any human intervention.

When recovery of a standby database encounters redo for the deletion of a tablespace, it deletes any Oracle managed data files in the local file system. Note that this is independent of the `INCLUDING DATAFILES` option issued at the primary database.

15.5 Scenarios for Using Oracle Managed Files

Scenarios illustrate how to use Oracle Managed Files.

- Scenario 1: Create and Manage a Database with Multiplexed Redo Logs
  An example illustrates creating and managing a database with multiplexed redo logs.
- Scenario 2: Create and Manage a Database with Database and Fast Recovery Areas
  An example illustrates creating and managing a database with both database and fast recovery areas.
- Scenario 3: Adding Oracle Managed Files to an Existing Database
  An example illustrates adding Oracle Managed Files to an existing database.

15.5.1 Scenario 1: Create and Manage a Database with Multiplexed Redo Logs

An example illustrates creating and managing a database with multiplexed redo logs.

In this scenario, a DBA creates a database where the data files and redo log files are created in separate directories. The redo log files and control files are multiplexed. The database uses an undo tablespace, and has a default temporary tablespace. The following are tasks involved with creating and maintaining this database.

1. Setting the initialization parameters

   The DBA includes three generic file creation defaults in the initialization parameter file before creating the database. Automatic undo management mode (the default) is also specified.

   ```
   DB_CREATE_FILE_DEST = '/u01/oradata'
   DB_CREATE_ONLINE_LOG_DEST_1 = '/u02/oradata'
   ```

```
DB_CREATE_ONLINE_LOG_DEST_2 = '/u03/oradata'
UNDO_MANAGEMENT = AUTO
```

The `DB_CREATE_FILE_DEST` parameter sets the default file system directory for the data files and temp files.

The `DB_CREATE_ONLINE_LOG_DEST_1` and `DB_CREATE_ONLINE_LOG_DEST_2` parameters set the default file system directories for redo log file and control file creation. Each redo log file and control file is multiplexed across the two directories.

2. Creating a database

Once the initialization parameters are set, the database can be created by using this statement:

```
SQL> CREATE DATABASE sample
2>   DEFAULT TEMPORARY TABLESPACE dflttmp;
```

Because a `DATAFILE` clause is not present and the `DB_CREATE_FILE_DEST` initialization parameter is set, the `SYSTEM` tablespace data file is created in the default file system (`/u01/oradata` in this scenario). The file name is uniquely generated by the database. The file is autoextensible with an initial size of 100 MB and an unlimited maximum size. The file is an Oracle managed file. A similar data file is created for the `SYSAUX` tablespace.

Because a `LOGFILE` clause is not present, two redo log groups are created. Each log group has two members, with one member in the `DB_CREATE_ONLINE_LOG_DEST_1` location and the other member in the `DB_CREATE_ONLINE_LOG_DEST_2` location. The file names are uniquely generated by the database. The log files are created with a size of 100 MB. The log file members are Oracle Managed Files.

Similarly, because the `CONTROL_FILES` initialization parameter is not present, and two `DB_CREATE_ONLINE_LOG_DEST_`*n* initialization parameters are specified, two control files are created. The control file located in the `DB_CREATE_ONLINE_LOG_DEST_1` location is the primary control file; the control file located in the `DB_CREATE_ONLINE_LOG_DEST_2` location is a multiplexed copy. The file names are uniquely generated by the database. They are Oracle Managed Files. Assuming there is a server parameter file, a `CONTROL_FILES` initialization parameter is generated.

Automatic undo management mode is specified, but because an undo tablespace is not specified and the `DB_CREATE_FILE_DEST` initialization parameter is set, a default undo tablespace named `UNDOTBS` is created in the directory specified by `DB_CREATE_FILE_DEST`. The data file is a 20 MB data file that is autoextensible. It is an Oracle managed file.

Lastly, a default temporary tablespace named `dflttmp` is specified. Because `DB_CREATE_FILE_DEST` is included in the parameter file, the temp file for `dflttmp` is created in the directory specified by that parameter. The temp file is 100 MB and is autoextensible with an unlimited maximum size. It is an Oracle managed file.

The resultant file tree, with generated file names, is as follows:

```
/u01
     /oradata
          /SAMPLE
               /datafile
                    /o1_mf_system_cmr7t30p_.dbf
                    /o1_mf_sysaux_cmr7t88p_.dbf
                    /o1_mf_sys_undo_2ixfh90q_.dbf
                    /o1_mf_dflttmp_157se6ff_.tmp
/u02
     /oradata
          /SAMPLE
               /onlinelog
```

```
                    /o1_mf_1_0orrm31z_.log
                    /o1_mf_2_2xyz16am_.log
               /controlfile
                    /o1_mf_cmr7t30p_.ctl
/u03
     /oradata
          /SAMPLE
               /onlinelog
                    /o1_mf_1_ixfvm8w9_.log
                    /o1_mf_2_q89tmp28_.log
               /controlfile
                    /o1_mf_x1sr8t36_.ctl
```

The internally generated file names can be seen when selecting from the usual views. For example:

```
SQL> SELECT NAME FROM V$DATAFILE;

NAME
--------------------------------------------------
/u01/oradata/SAMPLE/datafile/o1_mf_system_cmr7t30p_.dbf
/u01/oradata/SAMPLE/datafile/o1_mf_sysaux_cmr7t88p_.dbf
/u01/oradata/SAMPLE/datafile/o1_mf_sys_undo_2ixfh90q_.dbf

3 rows selected
```

3. Managing control files

   The control file was created when generating the database, and a `CONTROL_FILES` initialization parameter was added to the parameter file. If needed, then the DBA can re-create the control file or build a new one for the database using the `CREATE CONTROLFILE` statement.

   The correct Oracle managed file names must be used in the `DATAFILE` and `LOGFILE` clauses. The `ALTER DATABASE BACKUP CONTROLFILE TO TRACE` statement generates a script with the correct file names. Alternatively, the file names can be found by selecting from the `V$DATAFILE`, `V$TEMPFILE`, and `V$LOGFILE` views. The following example re-creates the control file for the sample database:

```
CREATE CONTROLFILE REUSE
  DATABASE sample
  LOGFILE
    GROUP 1('/u02/oradata/SAMPLE/onlinelog/o1_mf_1_0orrm31z_.log',
            '/u03/oradata/SAMPLE/onlinelog/o1_mf_1_ixfvm8w9_.log'),
    GROUP 2('/u02/oradata/SAMPLE/onlinelog/o1_mf_2_2xyz16am_.log',
            '/u03/oradata/SAMPLE/onlinelog/o1_mf_2_q89tmp28_.log')
  NORESETLOGS
  DATAFILE '/u01/oradata/SAMPLE/datafile/o1_mf_system_cmr7t30p_.dbf',
           '/u01/oradata/SAMPLE/datafile/o1_mf_sysaux_cmr7t88p_.dbf',
           '/u01/oradata/SAMPLE/datafile/o1_mf_sys_undo_2ixfh90q_.dbf',
           '/u01/oradata/SAMPLE/datafile/o1_mf_dflttmp_157se6ff_.tmp'
  MAXLOGFILES 5
  MAXLOGHISTORY 100
  MAXDATAFILES 10
  MAXINSTANCES 2
  ARCHIVELOG;
```

   The control file created by this statement is located as specified by the `CONTROL_FILES` initialization parameter that was generated when the database was created. The `REUSE` clause causes any existing files to be overwritten.

4. Managing the redo log

To create a new group of redo log files, the DBA can use the `ALTER DATABASE ADD LOGFILE` statement. The following statement adds a log file with a member in the `DB_CREATE_ONLINE_LOG_DEST_1` location and a member in the `DB_CREATE_ONLINE_LOG_DEST_2` location. These files are Oracle Managed Files.

```
SQL> ALTER DATABASE ADD LOGFILE;
```

Log file members continue to be added and dropped by specifying complete file names.

The `GROUP` clause can be used to drop a log group. In the following example the operating system file associated with each Oracle managed log file member is automatically deleted.

```
SQL> ALTER DATABASE DROP LOGFILE GROUP 3;
```

5. Managing tablespaces

   The default storage for all data files for future tablespace creations in the `sample` database is the location specified by the `DB_CREATE_FILE_DEST` initialization parameter (`/u01/oradata` in this scenario). Any data files for which no file name is specified, are created in the file system specified by the initialization parameter `DB_CREATE_FILE_DEST`. For example:

   ```
   SQL> CREATE TABLESPACE tbs_1;
   ```

   The preceding statement creates a tablespace whose storage is in `/u01/oradata`. A data file is created with an initial of 100 MB and it is autoextensible with an unlimited maximum size. The data file is an Oracle managed file.

   When the tablespace is dropped, the Oracle Managed Files for the tablespace are automatically removed. The following statement drops the tablespace and all the Oracle Managed Files used for its storage:

   ```
   SQL> DROP TABLESPACE tbs_1;
   ```

   Once the first data file is full, the database does not automatically create a new data file. More space can be added to the tablespace by adding another Oracle managed data file. The following statement adds another data file in the location specified by `DB_CREATE_FILE_DEST`:

   ```
   SQL> ALTER TABLESPACE tbs_1 ADD DATAFILE;
   ```

   The default file system can be changed by changing the initialization parameter. This does not change any existing data files. It only affects future creations. This can be done dynamically using the following statement:

   ```
   SQL> ALTER SYSTEM SET DB_CREATE_FILE_DEST='/u04/oradata';
   ```

6. Archiving redo information

   Archiving of redo log files is no different for Oracle Managed Files, than it is for unmanaged files. A file system location for the archived redo log files can be specified using the `LOG_ARCHIVE_DEST_`*n* initialization parameters. The file names are formed based on the `LOG_ARCHIVE_FORMAT` parameter or its default. The archived logs are not Oracle Managed Files.

7. Backup, restore, and recover

   Since an Oracle managed file is compatible with standard operating system files, you can use operating system utilities to backup or restore Oracle Managed Files. All existing methods for backing up, restoring, and recovering the database work for Oracle Managed Files.

15.5.2 Scenario 2: Create and Manage a Database with Database and Fast Recovery Areas

An example illustrates creating and managing a database with both database and fast recovery areas.

In this scenario, a DBA creates a database where the control files and redo log files are multiplexed. Archived logs and RMAN backups are created in the Fast Recovery Area. The following tasks are involved in creating and maintaining this database:

1. Setting the initialization parameters

   The DBA includes the following generic file creation defaults:

   ```
   DB_CREATE_FILE_DEST = '/u01/oradata'
   DB_RECOVERY_FILE_DEST_SIZE = 10G
   DB_RECOVERY_FILE_DEST = '/u02/oradata'
   LOG_ARCHIVE_DEST_1 = 'LOCATION = USE_DB_RECOVERY_FILE_DEST'
   ```

   The `DB_CREATE_FILE_DEST` parameter sets the default file system directory for data files, temp files, control files, and redo logs.

   The `DB_RECOVERY_FILE_DEST` parameter sets the default file system directory for control files, redo logs, and RMAN backups.

   The `LOG_ARCHIVE_DEST_1` configuration '`LOCATION=USE_DB_RECOVERY_FILE_DEST`' redirects archived logs to the `DB_RECOVERY_FILE_DEST` location.

   The `DB_CREATE_FILE_DEST` and `DB_RECOVERY_FILE_DEST` parameters set the default directory for log file and control file creation. Each redo log and control file is multiplexed across the two directories.

2. Creating a database
3. Managing control files
4. Managing the redo log
5. Managing tablespaces

   Tasks 2, 3, 4, and 5 are the same as in Scenario 1, except that the control files and redo logs are multiplexed across the `DB_CREATE_FILE_DEST` and `DB_RECOVERY_FILE_DEST` locations.

6. Archiving redo log information

   Archiving online logs is no different for Oracle Managed Files than it is for unmanaged files. The archived logs are created in `DB_RECOVERY_FILE_DEST` and are Oracle Managed Files.

7. Backup, restore, and recover

   An Oracle managed file is compatible with standard operating system files, so you can use operating system utilities to backup or restore Oracle Managed Files. All existing methods for backing up, restoring, and recovering the database work for Oracle Managed Files. When no format option is specified, all disk backups by RMAN are created in the `DB_RECOVERY_FILE_DEST` location. The backups are Oracle Managed Files.

15.5.3 Scenario 3: Adding Oracle Managed Files to an Existing Database

An example illustrates adding Oracle Managed Files to an existing database.

Assume in this case that an existing database does not have any Oracle Managed Files, but the DBA would like to create new tablespaces with Oracle Managed Files and locate them in directory `/u03/oradata`.

1. Setting the initialization parameters

   To allow automatic data file creation, set the `DB_CREATE_FILE_DEST` initialization parameter to the file system directory in which to create the data files. This can be done dynamically as follows:

   ```
   SQL> ALTER SYSTEM SET DB_CREATE_FILE_DEST = '/u03/oradata';
   ```

2. Creating tablespaces

   Once `DB_CREATE_FILE_DEST` is set, the `DATAFILE` clause can be omitted from a `CREATE TABLESPACE` statement. The data file is created in the location specified by `DB_CREATE_FILE_DEST` by default. For example:

   ```
   SQL> CREATE TABLESPACE tbs_2;
   ```

   When the `tbs_2` tablespace is dropped, its data files are automatically deleted.