CHAPTER 3
Planning and Managing Tablespaces
How a DBA configures the layout of the tablespaces in a database directly affects the performance and manageability of the database. In this chapter, we’ll review the different types of tablespaces as well as how temporary tablespace usage can drive the size and number of tablespaces in a database leveraging the temporary tablespace group feature introduced in Oracle 10g.

I’ll also show how Oracle’s Optimal Flexible Architecture (OFA), supported since Oracle 7, helps to standardize the directory structure for both Oracle executables and the database files themselves; Oracle Database 12c further enhances OFA to complement its original role of improving performance to enhancing security and simplifying cloning and upgrade tasks.

A default installation of Oracle provides the DBA with a good starting point, not only creating an OFA-compliant directory structure but also segregating segments into a number of tablespaces based on their function. We’ll review the space requirements for each of these tablespaces and provide some tips on how to fine-tune the characteristics of these tablespaces.

Using Oracle Automatic Storage Management (ASM) as your logical volume manager makes tablespace maintenance easier and more efficient by automatically spreading out the segments within a tablespace across all disks of an ASM disk group. Adding datafiles to a tablespace is almost trivial when using ASM; using bigfile tablespaces means you only have to allocate a single datafile for the tablespace. In both cases, you don’t need to specify, or even need to know, the name of the datafile itself within the ASM directory structure.

In Oracle Database 12c, container databases (CDBs) and pluggable databases (PDBs) in a multitenant database architecture change how some tablespaces are used and managed in a pluggable database. All permanent tablespaces can be associated with one and only one database—either the CDB or one PDB. In contrast, temporary tablespaces or temporary tablespace groups are managed at the CDB level and are used by all PDBs within the CDB. See Chapter 10 for an in-depth discussion of the Oracle Database 12c multitenant architecture.

At the end of the chapter, I’ll provide some guidelines to help you place segments into different tablespaces based on their type, size, and frequency of access, as well as ways to identify hotspots in one or more tablespaces.

Tablespace Architecture

A prerequisite to competently setting up the tablespaces in your database is understanding the different types of tablespaces and how they are used in an Oracle database. In this section, we’ll review the different types of tablespaces and give some examples of how they are managed. In addition, I’ll review the types of tablespaces by category—permanent tablespaces (SYSTEM, SYSAUX, and so on), temporary tablespaces, undo tablespaces, and bigfile tablespaces—and describe their function. Finally, I’ll also discuss Oracle’s Optimal Flexible Architecture (OFA) and how it can ease maintenance tasks.

Tablespace Types

The primary types of tablespaces in an Oracle database are permanent, undo, and temporary. Permanent tablespaces contain segments that persist beyond the duration of a session or a transaction.

Although the undo tablespace may have segments that are retained beyond the end of a session or a transaction, it provides read consistency for SELECT statements that access tables being modified as well as provides undo data for a number of the Oracle Flashback features of the database. Primarily, however, undo segments store the previous values of columns being updated or deleted. This ensures
that if a user’s session fails before the user issues a COMMIT or a ROLLBACK, the UPDATEs, INSERTs, and DELETEs will be removed and will never be accessible by other sessions. Undo segments are never directly accessible by a user session, and undo tablespaces may only have undo segments.

As the name implies, temporary tablespaces contain transient data that exists only for the duration of the session, such as space to complete a sort operation that will not fit in memory.

Bigfile tablespaces can be used for any of these three types of tablespaces, and they simplify tablespace management by moving the maintenance point from the datafile to the tablespace. Bigfile tablespaces consist of one and only one datafile. There are a couple of downsides to bigfile tablespaces, however, and they will be presented later in this chapter.

Permanent
The SYSTEM and SYSAUX tablespaces are two examples of permanent tablespaces. In addition, any segments that need to be retained by a user or an application beyond the boundaries of a session or transaction should be stored in a permanent tablespace.

SYSTEM Tablespace User segments should never reside in the SYSTEM or SYSAUX tablespace, period. If you do not specify a default permanent or temporary tablespace when creating users, the database-level default permanent and temporary tablespaces are used.

If you use the Oracle Universal Installer (OUI) to create a database for you, a separate tablespace other than SYSTEM is created for both permanent and temporary segments. If you create a database manually, be sure to specify both a default permanent tablespace and a default temporary tablespace, as in the sample CREATE DATABASE command that follows.

```sql
CREATE DATABASE rjbdb
  USER SYS IDENTIFIED BY melsm25
  USER SYSTEM IDENTIFIED BY welisa45
  LOGFILE GROUP 1 ('/u02/oracle11g/oradata/rjbdb/redo01.log') SIZE 100M,
    GROUP 2 ('/u04/oracle11g/oradata/rjbdb/redo02.log') SIZE 100M,
    GROUP 3 ('/u06/oracle11g/oradata/rjbdb/redo03.log') SIZE 100M
  MAXLOGFILES 5
  MAXLOGMEMBERS 5
  MAXLOGHISTORY 1
  MAXDATAFILES 100
  MAXINSTANCES 1
  CHARACTER SET US7ASCII
  NATIONAL CHARACTER SET AL16UTF16
  DATAFILE '/u01/oracle11g/oradata/rjbdb/system01.dbf' SIZE 2G REUSE
  EXTENT MANAGEMENT LOCAL
  SYSAUX DATAFILE '/u01/oracle11g/oradata/rjbdb/sysaux01.dbf'
    SIZE 800M REUSE
  DEFAULT TABLESPACE USERS
    DATAFILE '/u03/oracle11g/oradata/rjbdb/users01.dbf'
    SIZE 4G REUSE
  DEFAULT TEMPORARY TABLESPACE TEMPTS1
    TEMPFILE '/u01/oracle11g/oradata/rjbdb/temp01.dbf'
    SIZE 500M REUSE
  UNDO TABLESPACE undotbs
    DATAFILE '/u02/oracle11g/oradata/rjbdb/undotbs01.dbf'
    SIZE 400M REUSE AUTOEXTEND ON MAXSIZE 2G;
```
As of Oracle 10g, the SYSTEM tablespace is locally managed by default; in other words, all space usage is managed by a bitmap segment in the first part of the first datafile for the tablespace. In a database where the SYSTEM tablespace is locally managed, the other tablespaces in the database must also be locally managed or they must be read-only. Using locally managed tablespaces takes some of the contention off the SYSTEM tablespace because space allocation and deallocation operations for a tablespace do not need to use data dictionary tables. More details on locally managed tablespaces can be found in Chapter 6. Other than to support the import of a transportable tablespace that is dictionary managed from a legacy database, there are no advantages to having a dictionary-managed tablespace in your database.

**SYSAUX Tablespace**  
Like the SYSTEM tablespace, the SYSAUX tablespace should not have any user segments. The contents of the SYSAUX tablespace, broken down by application, can be reviewed using Oracle Enterprise Manager Database Express (EM Express) or Cloud Control 12c. You can edit the SYSAUX tablespace in Cloud Control 12c by choosing Administration | Storage | Tablespaces and clicking the SYSAUX link in the tablespace list. Figure 3-1 shows a graphical representation of the space usage within SYSAUX.

If the space usage for a particular application that resides in the SYSAUX tablespace becomes too high or creates an I/O bottleneck through high contention with other applications that use the SYSAUX tablespace, you can move one or more of these applications to a different tablespace. Any SYSAUX occupant listed in Figure 3-1 that has a Change Tablespace link available can be moved by clicking the link and then choosing a destination tablespace in the field shown in Figure 3-2. The XDB objects will be moved to the SYSAUX2 tablespace. An example of moving a SYSAUX occupant to a different tablespace using the command line interface can be found in Chapter 6.

The SYSAUX tablespace can be monitored just like any other tablespace; later in this chapter, I’ll show how EM Cloud Control can help us to identify hotspots in a tablespace.

**Undo**

Multiple undo tablespaces can exist in a database, but only one undo tablespace can be active at any given time for a single database instance. Undo tablespaces are used for rolling back transactions, for providing read consistency for SELECT statements that run concurrently with DML statements on the same table or set of tables, and for supporting a number of Oracle Flashback features, such as Flashback Query.

The undo tablespace needs to be sized correctly to prevent ORA-01555 “Snapshot too old” errors and to provide enough space to support initialization parameters such as UNDO_RETENTION. More information on how to monitor, size, and create undo tablespaces can be found in Chapter 7.

**Temporary**

More than one temporary tablespace can be online and active in the database, but until Oracle 10g, multiple sessions by the same user would use the same temporary tablespace because only one default temporary tablespace could be assigned to a user. To solve this potential performance bottleneck, Oracle supports *temporary tablespace groups*. A temporary tablespace group is a synonym for a list of temporary tablespaces.

A temporary tablespace group must consist of at least one temporary tablespace; it cannot be empty. Once a temporary tablespace group has no members, it no longer exists.

One of the big advantages of using temporary tablespace groups is to provide a single user with multiple sessions with the ability to use a different actual temporary tablespace for each session. In the diagram shown in Figure 3-3, the user OE has two active sessions that need temporary space for performing sort operations.
Instead of a single temporary tablespace being assigned to a user, the temporary tablespace group is assigned; in this example, the temporary tablespace group TEMPGRP has been assigned to OE. However, because there are three actual temporary tablespaces within the TEMPGRP temporary tablespace group, the first OE session may use temporary tablespace TEMP1, and the SELECT statement executed by the second OE session may use the other two temporary tablespaces, TEMP2 and TEMP3, in parallel. Before Oracle 10g, both sessions would use the same temporary tablespace, potentially causing a performance issue.
FIGURE 3-2. Using EM Cloud Control 12c to move a SYSAUX occupant

FIGURE 3-3. Temporary tablespace group TEMPGRP
Creating a temporary tablespace group is very straightforward. After creating the individual tablespaces TEMP1, TEMP2, and TEMP3, we can create a temporary tablespace group named TEMPGRP as follows:

```
SQL> alter tablespace temp1 tablespace group tempgrp;
Tablespace altered.
SQL> alter tablespace temp2 tablespace group tempgrp;
Tablespace altered.
SQL> alter tablespace temp3 tablespace group tempgrp;
Tablespace altered.
```

Changing the database’s default temporary tablespace to TEMPGRP uses the same command as assigning an actual temporary tablespace as the default; temporary tablespace groups are treated logically the same as a temporary tablespace:

```
SQL> alter database default temporary tablespace tempgrp;
Database altered.
```

To drop a tablespace group, we must first drop all its members. Dropping a member of a tablespace group is accomplished by assigning the temporary tablespace to a group with an empty string (in other words, removing the tablespace from the group):

```
SQL> alter tablespace temp3 tablespace group '';
Tablespace altered.
```

As you might expect, assigning a temporary tablespace group to a user is identical to assigning a temporary tablespace to a user; this assignment can happen either when the user is created or at some point in the future. In the following example, the new user JENWEB is assigned the temporary tablespace TEMPGRP:

```
SQL> create user jenweb identified by pi4001
    2     default tablespace users
    3     temporary tablespace tempgrp;
User created.
```

Note that if we did not assign the tablespace during user creation, the user JENWEB would still be assigned TEMPGRP as the temporary tablespace because it is the database default from our previous CREATE DATABASE example.

A couple of changes were made to the data dictionary views in Oracle Database 10g and Oracle Database 11g to support temporary tablespace groups. The data dictionary view DBA_USERS still has the column TEMPORARY_TABLESPACE, as in previous versions of Oracle, but this column may now contain either the name of the temporary tablespace assigned to the user or the name of a temporary tablespace group:

```
SQL> select username, default_tablespace, temporary_tablespace
    2    from dba_users where username = 'JENWEB';

USERNAME     DEFAULT_TABLESPACE TEMPORARY_TABLESPACE
------------- ------------------ ------------------
JENWEB        USERS              TEMPGRP

1 row selected.
```
The new data dictionary view DBA_TABLESPACE_GROUPS shows the members of each temporary tablespace group:

```
SQL> select group_name, tablespace_name from dba_tablespace_groups;

GROUP_NAME                   TABLESPACE_NAME
---------------------------- ----------------------------
TEMPGRP                      TEMP1
TEMPGRP                      TEMP2
TEMPGRP                      TEMP3

3 rows selected.
```

As with most every other feature of Oracle that can be accomplished with the command line, assigning members to temporary tablespace groups or removing members from temporary tablespace groups can be performed using EM Cloud Control 12c. In Figure 3-4, we can add or remove members from a temporary tablespace group.

**Bigfile**

A bigfile tablespace eases database administration because it consists of only one datafile. The single datafile can be up to 128TB (terabytes) in size if the tablespace block size is 32KB; if you use the more common 8KB block size, 32TB is the maximum size of a bigfile tablespace. Many of the commands previously available only for maintaining datafiles can now be used at the tablespace level if the tablespace is a bigfile tablespace. Chapter 6 reviews how BIGFILE tablespaces are created and maintained.

The maintenance convenience of bigfile tablespaces can be offset by some potential disadvantages. Because a bigfile tablespace is a single datafile, a full backup of a single large datafile will take significantly longer than a full backup of several smaller datafiles (with the same total size as the single bigfile tablespace) even when Oracle uses multiple slave processes per datafile. If your bigfile tablespaces are read-only or if only changed blocks are backed up on a regular basis, the backup issue may not be critical in your environment. If you use the SECTION SIZE option in RMAN, available as of Oracle Database 11g, then an entire bigfile tablespace (and therefore the entire datafile) can be backed up in parallel.

**Optimal Flexible Architecture**

Oracle’s Optimal Flexible Architecture (OFA) provides guidelines to ease the maintenance of the Oracle software and database files as well as improve the performance of the database by placing the database files such that I/O bottlenecks are minimized.

Although using OFA is not strictly enforced when you’re installing or maintaining an Oracle environment, using OFA makes it easy for someone to understand how your database is organized on disk, preventing that phone call in the middle of the night during the week you’re on vacation!

OFA is slightly different depending on the type of storage options you use: either an ASM environment or a standard operating system file system that may or may not be using a third-party logical volume manager or RAID-enabled disk subsystem. In either case, the Database Configuration Assistant can create an OFA-compliant datafile directory structure for you.
Chapter 3: Planning and Managing Tablespaces

Non-ASM Environment

In a non-ASM environment on a Unix server, at least three file systems on separate physical devices are required to implement OFA recommendations. Starting at the top, the recommended format for a mount point is `/<string const><numeric key>`, where `<string const>` can be one or several letters and `<numeric key>` is either two or three digits. For example, on one system we may have mount points `/u01`, `/u02`, `/u03`, and `/u04`, with room to expand to an additional 96 mount points without changing the file-naming convention. Figure 3-5 shows a typical Unix file system layout with an OFA-compliant Oracle directory structure.

There are two instances on this server: an ASM instance to manage disk groups and a standard RDBMS instance (dw).
Software Executables  The software executables for each distinct product name reside in the directory `<string const><numeric key><directory type><product owner>`, where `<string const>` and `<numeric key>` are defined previously, `<directory type>` implies the type of files installed in this directory, and `<product owner>` is the name of the user that owns and installs the files in this directory. For example, `/u01/app/oracle` would contain application-related files (executables) installed by the user `oracle` on the server. The directory `/u01/app/apache` would contain the executables for the middleware web server installed from a previous version of Oracle.

As of Oracle 10g, the OFA standard makes it easy for the DBA to install multiple versions of the database and client software within the same high-level directory. The OFA-compliant Oracle home path, corresponding to the environment variable `ORACLE_HOME`, contains a suffix that corresponds to the type and incarnation of the installation. For example, one installation of
Chapter 3: Planning and Managing Tablespaces

Oracle 12c, one installation of Oracle 11g, two different installations of Oracle 10g, and one installation of Oracle9i may reside in the following three directories:

/u01/app/oracle/product/9.2.0.1
/u01/app/oracle/product/10.1.0/db_1
/u01/app/oracle/product/10.1.0/db_2
/u01/app/oracle/product/11.1.0/db_1
/u01/app/oracle/product/12.1.0/dbhome_1

At the same time, the Oracle client executables and configuration may be stored in the same parent directory as the database executables:

/u01/app/oracle/product/12.1.0/client_1

Some installation directories will never have more than one instance for a given product; for example, Oracle Grid Infrastructure (one installation per server) will be installed in the following directory given the previous installations:

/u01/app/oracle/product/12.1.0/grid

Because Grid Infrastructure can be installed only once on a system, it does not have an incrementing numeric suffix.

**Database Files** Any non-ASM Oracle datafiles reside in /<mount point>/oradata/<database name>, where <mount point> is one of the mount points we discussed earlier, and <database name> is the value of the initialization parameter DB_NAME. For example, /u02/oradata/rac0 and /u03/oradata/rac0 would contain the non-ASM control files, redo log files, and datafiles for the instance rac0, whereas /u05/oradata/dev1 would contain the same files for the dev1 instance on the same server. The naming convention for the different file types under the oradata directory are detailed in Table 3-1.

Although Oracle tablespace names can be as long as 30 characters, it is advisable to keep the tablespace names eight characters or less in a Unix environment. Because portable Unix filenames are restricted to 14 characters, and the suffix of an OFA datafile name is <n>.dbf, where n is two digits, a total of six characters are needed for the suffix in the file system. This leaves eight characters for the tablespace name itself.

<table>
<thead>
<tr>
<th>File Type</th>
<th>Filename Format</th>
<th>Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control files</td>
<td>control.ctl</td>
<td>None.</td>
</tr>
<tr>
<td>Redo log files</td>
<td>redo&lt;n&gt;.log</td>
<td>n is a two-digit number.</td>
</tr>
<tr>
<td>Datafiles</td>
<td>&lt;tn&gt;.dbf</td>
<td>t is an Oracle tablespace name, and n is a two-digit number.</td>
</tr>
</tbody>
</table>

**TABLE 3-1.** OFA-Compliant Control File, Redo Log File, and Datafile Naming Conventions
Only control files, redo log files, and datafiles associated with the database `<database name>` should be stored in the directory `/<mount point>/oradata/<database name>`. For the database `ord` managed without ASM, the datafile names are as follows:

```
SQL> select file#, name from v$datafile;
```

<table>
<thead>
<tr>
<th>FILE#</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>/u05/oradata/ord/system01.dbf</td>
</tr>
<tr>
<td>2</td>
<td>/u05/oradata/ord/undotbs01.dbf</td>
</tr>
<tr>
<td>3</td>
<td>/u05/oradata/ord/sysaux01.dbf</td>
</tr>
<tr>
<td>4</td>
<td>/u05/oradata/ord/users01.dbf</td>
</tr>
<tr>
<td>5</td>
<td>/u09/oradata/ord/example01.dbf</td>
</tr>
<tr>
<td>6</td>
<td>/u09/oradata/ord/oe_trans01.dbf</td>
</tr>
<tr>
<td>7</td>
<td>/u05/oradata/ord/users02.dbf</td>
</tr>
<tr>
<td>8</td>
<td>/u06/oradata/ord/logmnr_rep01.dbf</td>
</tr>
<tr>
<td>9</td>
<td>/u09/oradata/ord/big_users.dbf</td>
</tr>
<tr>
<td>10</td>
<td>/u08/oradata/ord/idx01.dbf</td>
</tr>
<tr>
<td>11</td>
<td>/u08/oradata/ord/idx02.dbf</td>
</tr>
<tr>
<td>12</td>
<td>/u08/oradata/ord/idx03.dbf</td>
</tr>
<tr>
<td>13</td>
<td>/u08/oradata/ord/idx04.dbf</td>
</tr>
<tr>
<td>14</td>
<td>/u08/oradata/ord/idx05.dbf</td>
</tr>
<tr>
<td>15</td>
<td>/u08/oradata/ord/idx06.dbf</td>
</tr>
<tr>
<td>16</td>
<td>/u08/oradata/ord/idx07.dbf</td>
</tr>
<tr>
<td>17</td>
<td>/u08/oradata/ord/idx08.dbf</td>
</tr>
</tbody>
</table>

17 rows selected.

Other than file numbers 8 and 9, all the datafiles in the `ord` database are OFA compliant and are spread out over four different mount points. The tablespace name in file number 8 is too long, and file number 9 does not have a numeric two-digit counter to represent new datafiles for the same tablespace.

**ASM Environment**

In an ASM environment, the executables are stored in the directory structure presented previously; however, if you browsed the directory `/u02/oradata` in Figure 3-5, you would see no files. All the control files, redo log files, and datafiles for the instance `dw` are managed by the ASM instance `+ASM` on this server.

The actual datafile names are not needed for most administrative functions because ASM files are all Oracle Managed Files (OMF). This eases the overall administrative effort required for the database. Within the ASM storage structure, an OFA-like syntax is used to subdivide the file types even further:

```
SQL> select file#, name from v$datafile;
```

```
<table>
<thead>
<tr>
<th>FILE#</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+DATA/dw/datafile/system.256.622426913</td>
</tr>
<tr>
<td>2</td>
<td>+DATA/dw/datafile/sysaux.257.622426915</td>
</tr>
<tr>
<td>3</td>
<td>+DATA/dw/datafile/undotbs1.258.622426919</td>
</tr>
</tbody>
</table>
```
Within the disk groups +DATA and +RECOV, we see that each of the database file types, such as datafiles, control files, and online log files, has its own directory. Fully qualified ASM filenames have the format

```
<group>/<dbname>/<file type>/<tag>.<file>.<incarnation>
```

where <group> is the disk group name, <dbname> is the database to which the file belongs, <file type> is the Oracle file type, <tag> is information specific to the file type, and the pair <file>.<incarnation> ensures uniqueness within the disk group.

Automatic Storage Management is covered in Chapter 6.

### Oracle Installation Tablespaces

Table 3-2 lists the tablespaces created with a standard Oracle 12c installation using the Oracle Universal Installer (OUI); the EXAMPLE tablespace is optional; it is installed if you specify that you want the sample schemas created during the installation.
As mentioned previously in this chapter, no user segments should ever be stored in the SYSTEM tablespace. The clause DEFAULT TABLESPACE in the CREATE DATABASE command helps to prevent this occurrence by automatically assigning a permanent tablespace for all users that have not explicitly been assigned a permanent tablespace. An Oracle installation performed using the OUI will automatically assign the USERS tablespace as the default permanent tablespace.

The SYSTEM tablespace will grow more quickly the more you use procedural objects such as functions, procedures, triggers, and so forth, because these objects must reside in the data dictionary. This also applies to abstract datatypes and Oracle’s other object-oriented features.

As with the SYSTEM tablespace, user segments should never be stored in the SYSAUX tablespace. If one particular occupant of the SYSAUX tablespace takes up too much of the available space or significantly affects the performance of other applications that use the SYSAUX tablespace, you should consider moving the occupant to another tablespace.

Instead of one very large temporary tablespace, consider using several smaller temporary tablespaces and creating a temporary tablespace group to hold them. As you found out earlier in this chapter, this can improve the response time for applications that create many sessions with the same username. For Oracle container databases and pluggable databases (in Oracle’s multitenant architecture, new to Oracle Database 12c), the container database owns the temporary tablespace used by all plugged-in databases.

Even though a database may have more than one undo tablespace, only one undo tablespace can be active at any given time for a given instance. If more space is needed for an undo tablespace, and AUTOEXTEND is not enabled, another datafile can be added. One undo tablespace must be available for each node in a Real Application Clusters (RAC) environment because each instance manages its own undo.
 USERS
The USERS tablespace is intended for miscellaneous segments created by each database user, and it’s not appropriate for any production applications. A separate tablespace should be created for each application and segment type; later in this chapter I’ll present some additional criteria you can use to decide when to segregate segments into their own tablespace.

EXAMPLE
In a production environment, the EXAMPLE tablespace should be dropped; it takes up hundreds of megabytes of disk space and has examples of all types of Oracle segments and data structures. A separate database should be created for training purposes with these sample schemas; for an existing training database, the sample schemas can be installed into the tablespace of your choice by using the scripts in $ORACLE_HOME/demo/schema.

Segment Segregation
As a general rule of thumb, you want to divide segments into different tablespaces based on their type, size, and frequency of access. Furthermore, each of these tablespaces would benefit from being on its own disk group or disk device; in practice, however, most shops will not have the luxury of storing each tablespace on its own device. The following list identifies some of the conditions you might use to determine how segments should be segregated among tablespaces. The list is not prioritized because the priority depends on your particular environment. Using ASM eliminates many of the contention issues listed with no additional effort by the DBA. ASM is discussed in detail in Chapter 4. In most of these scenarios the recommendations primarily enhance manageability over performance to enhance availability.

■ Big segments and small segments should be in separate tablespaces, especially for manageability and reclaiming empty space from a large table.
■ Table segments and their corresponding index segments should be in separate tablespaces (if you are not using ASM and each tablespace is stored in its own set of disks).
■ A separate tablespace should be used for each application.
■ Segments with low usage and segments with high usage should be in different tablespaces.
■ Static segments should be separated from high DML segments.
■ Read-only tables should be in their own tablespace.
■ Staging tables for a data warehouse should be in their own tablespace.
■ Tablespaces should be created with the appropriate block size, depending on whether segments are accessed row by row or in full table scans.
■ Tablespaces should be allocated for different types of activity, such as primarily UPDATEs, primarily read-only, or temporary segment usage.
■ Materialized views should be in a separate tablespace from the base table.
■ For partitioned tables and indexes, each partition should be in its own tablespace.

Using EM Cloud Control 12c, you can identify overall contention on any tablespace by identifying hotspots, either at the file level or at the object level. We’ll cover performance tuning, including resolving I/O contention issues, in Chapter 8.
Summary
The basic logical building block of a database is the tablespace. It consists of one or more physical datafiles, only one datafile if you create a bigfile tablespace. Whether you’re creating a permanent, undo, or temporary tablespace you can create those tablespaces as bigfile tablespaces for ease of management.

When you create tablespaces or other objects, you can use Optimal Flexible Architecture (OFA) to automatically create an appropriate OS file name and directory location. This is even more useful in an ASM environment where you only need to specify the disk group name; Oracle puts it in the right directory location automatically and you may never need to know where in the ASM file structure Oracle places the object.

In a default Oracle database installation, Oracle creates five required tablespaces: SYSTEM, SYSAUX, TEMP, UNDOTBS1, and USERS; if you choose to install the sample schemas they will exist in the EXAMPLE tablespace. You will most likely create many more tablespaces in your environment to segregate applications to their own tablespace or to restrict how much disk space a tablespace may use for that application.