**3****Tablespaces, Datafiles, and Control Files**

This chapter describes tablespaces, the primary logical database structures of any Oracle database, and the physical datafiles that correspond to each tablespace. The chapter includes:

* [Introduction to Tablespaces, Datafiles, and Control Files](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "924)

* [Tablespaces Overview](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "967)

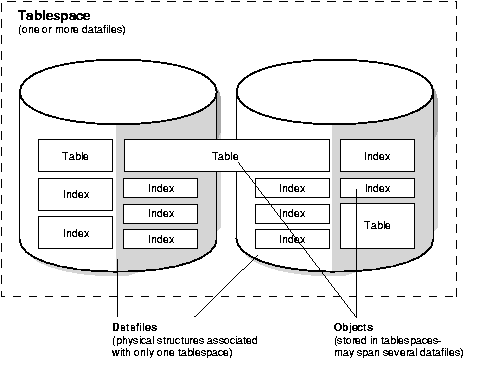
* [Datafiles Overview](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "7388)

* [Control Files Overview](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "10136)

**Introduction to Tablespaces, Datafiles, and Control Files**

Oracle stores data logically in **tablespaces** and physically in **datafiles** associated with the corresponding tablespace. [Figure 3-1](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm#3563) illustrates this relationship.

***Figure 3-1 Datafiles and Tablespaces***

**** [Text description of the illustration cncpt037.gif](https://docs.oracle.com/cd/B10500_01/server.920/a96524/img_text/cncpt037.htm)

Databases, tablespaces, and datafiles are closely related, but they have important differences:

* An Oracle database consists of one or more logical storage units called tablespaces, which collectively store all of the database's data.
* Each tablespace in an Oracle database consists of one or more files called datafiles, which are physical structures that conform to the operating system in which Oracle is running.
* A database's data is collectively stored in the datafiles that constitute each tablespace of the database. For example, the simplest Oracle database would have one tablespace and one datafile. Another database can have three tablespaces, each consisting of two datafiles (for a total of six datafiles).

**Oracle-Managed Files**

Oracle-managed files eliminate the need for you, the DBA, to directly manage the operating system files comprising an Oracle database. You specify operations in terms of database objects rather than filenames. Oracle internally uses standard file

system interfaces to create and delete files as needed for the following database structures:

* Tablespaces
* Online redo log files
* Control files

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

|  |
| --- |
| **See Also:**  [*Oracle9i Database Administrator's Guide*](https://docs.oracle.com/cd/B10500_01/server.920/a96521/toc.htm) |

**Allocate More Space for a Database**

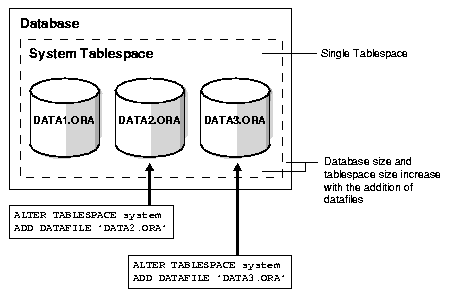
The size of a tablespace is the size of the datafiles that constitute the tablespace. The size of a database is the collective size of the tablespaces that constitute the database.

You can enlarge a database in three ways:

* Add a datafile to a tablespace
* Add a new tablespace
* Increase the size of a datafile

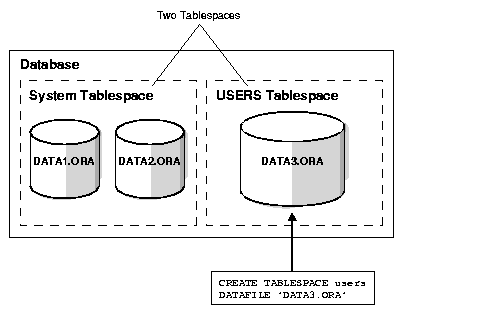
When you add another datafile to an existing tablespace, you increase the amount of disk space allocated for the corresponding tablespace. [Figure 3-2](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm#3682) illustrates this kind of space increase.

***Figure 3-2 Enlarging a Database by Adding a Datafile to a Tablespace***

**** [Text description of the illustration cncpt038.gif](https://docs.oracle.com/cd/B10500_01/server.920/a96524/img_text/cncpt038.htm)

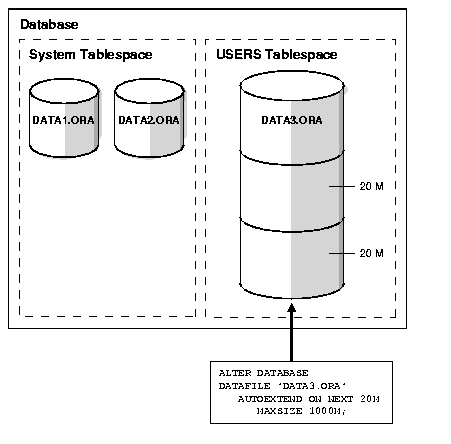
Alternatively, you can create a new tablespace (which contains at least one additional datafile) to increase the size of a database. [Figure 3-3](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm#3692) illustrates this.

***Figure 3-3 Enlarging a Database by Adding a New Tablespace***

**** [Text description of the illustration cncpt039.gif](https://docs.oracle.com/cd/B10500_01/server.920/a96524/img_text/cncpt039.htm)

The third option for enlarging a database is to change a datafile's size or let datafiles in existing tablespaces grow dynamically as more space is needed. You accomplish this by altering existing files or by adding files with dynamic extension properties. [Figure 3-4](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm#3706) illustrates this.

***Figure 3-4 Enlarging a Database by Dynamically Sizing Datafiles***

 [Text description of the illustration cncpt040.gif](https://docs.oracle.com/cd/B10500_01/server.920/a96524/img_text/cncpt040.htm)

|  |
| --- |
| **See Also:**  [*Oracle9i Database Administrator's Guide*](https://docs.oracle.com/cd/B10500_01/server.920/a96521/toc.htm) for more information about increasing the amount of space in your database |

**Tablespaces Overview**

A database is divided into one or more logical storage units called tablespaces. Tablespaces are divided into logical units of storage called **segments**, which are further divided into **extents**. Extents are a collection of contiguous blocks.

This section includes the following topics about tablespaces:

* [The SYSTEM Tablespace](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "1003)

* [Undo Tablespaces](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "10700)

* [Default Temporary Tablespace](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "10742)

* [Using Multiple Tablespaces](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "7963)

* [Managing Space in Tablespaces](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "4346)

* [Multiple Block Sizes](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "11420)

* [Online and Offline Tablespaces](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "3781)

* [Read-Only Tablespaces](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "1116)

* [Temporary Tablespaces for Sort Operations](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "5902)

* [Transport of Tablespaces Between Databases](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "9370)

|  |
| --- |
| **See Also:**  [Chapter 2, "Data Blocks, Extents, and Segments"](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c03block.htm#2528) for more information about segments and extents |

**The SYSTEM Tablespace**

Every Oracle database contains a tablespace named SYSTEM, which Oracle creates automatically when the database is created. The SYSTEM tablespace is always online when the database is open.

To take advantage of the benefits of locally managed tablespaces, you can create a locally managed SYSTEM tablespace, or you can migrate an existing dictionary managed SYSTEM tablespace to a locally managed format.

In a database with a locally managed SYSTEM tablespace, dictionary tablespaces cannot be created. It is possible to plug in a dictionary managed tablespace using the transportable feature, but it cannot be made writable.

|  |
| --- |
| **Note:**  Once a tablespace is locally managed, it cannot be reverted back to being dictionary managed. |

**The Data Dictionary**

The SYSTEM tablespace always contains the data dictionary tables for the entire database. The data dictionary tables are stored in datafile 1.

**PL/SQL Program Units Description**

All data stored on behalf of stored PL/SQL program units (that is, procedures, functions, packages, and triggers) resides in the SYSTEM tablespace. If the database contains many of these program units, then the database administrator must provide the space the units need in the SYSTEM tablespace.

|  |
| --- |
| **See Also:**   * *[Oracle9i Database Administrator's Guide](https://docs.oracle.com/cd/B10500_01/server.920/a96521/toc.htm)* for detailed information about creating or migrating to a locally managed SYSTEM tablespace  * ["Online and Offline Tablespaces"](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "3781) for more information about the permanent online condition of the SYSTEM tablespace  * [Chapter 14, "SQL, PL/SQL, and Java"](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c16sqlpl.htm" \l "15197) and [Chapter 17, "Triggers"](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c18trigs.htm#12312) for more information about the space requirements of PL/SQL program units |

**Undo Tablespaces**

Undo tablespaces are special tablespaces used solely for storing undo information. You cannot create any other segment types (for example, tables or indexes) in undo tablespaces. Each database contains zero or more undo tablespaces. In automatic undo management mode, each Oracle instance is assigned one (and only one) undo tablespace. Undo data is managed within an undo tablespace using undo segments that are automatically created and maintained by Oracle.

When the first DML operation is run within a transaction, the transaction is bound (assigned) to an undo segment (and therefore to a transaction table) in the current undo tablespace. In rare circumstances, if the instance does not have a designated undo tablespace, the transaction binds to the system undo segment.

|  |
| --- |
| **Caution:**  Do not run any user transactions before creating the first undo tablespace and taking it online. |

Each undo tablespace is composed of a set of undo files and is locally managed. Like other types of tablespaces, undo blocks are grouped in extents and the status of each extent is represented in the bitmap. At any point in time, an extent is either allocated to (and used by) a transaction table, or it is free.

**Creation of Undo Tablespaces**

A database administrator creates undo tablespaces individually, using the CREATE UNDO TABLESPACE statement. It can also be created when the database is created, using the CREATE DATABASE statement. A set of files is assigned to each newly created undo tablespace. Like regular tablespaces, attributes of undo tablespaces can be modified with the ALTER TABLESPACE statement and dropped with the DROP TABLESPACE statement.

|  |
| --- |
| **Note:**  An undo tablespace cannot be dropped if it is being used by any instance or contains any undo information needed to recover transactions. |

**Assignment of Undo Tablespaces**

You assign an undo tablespace to an instance in one of two ways:

* At instance startup. You can specify the undo tablespace in the initialization file or let the system choose an available undo tablespace.
* While the instance is running. Use ALTER SYSTEM SET UNDO\_TABLESPACE to replace the active undo tablespace with another undo tablespace. This method is rarely used.

You can add more space to an undo tablespace by adding more data files to the undo tablespace with the ALTER TABLESPACE statement.

You can have more than one undo tablespace and switch between them. Use the Database Resource Manager to establish user quotas for undo tablespaces. You can specify the retention period for undo information.

|  |
| --- |
| **See Also:**  [*Oracle9i Database Administrator's Guide*](https://docs.oracle.com/cd/B10500_01/server.920/a96521/undo.htm#ADMIN013) for detailed information about creating and managing undo tablespaces |

**Default Temporary Tablespace**

When the SYSTEM tablespace is locally managed, you must define a default temporary tablespace when creating a database. A locally managed SYSTEM tablespace cannot be used for default temporary storage.

If SYSTEM is dictionary managed and if you do not define a default temporary tablespace when creating the database, then SYSTEM is still used for default temporary storage. However, you will receive a warning in ALERT.LOG saying that a default temporary tablespace is recommended and will be necessary in future releases.

**How to Specify a Default Temporary Tablespace**

Specify a default temporary tablespace when you create a database, using the DEFAULT TEMPORARY TABLESPACE extension to the CREATE DATABASE statement.

If you drop the default temporary tablespace, then the SYSTEM tablespace is used as the default temporary tablespace.

|  |
| --- |
| **Note:**  You cannot make the default temporary tablespace permanent or take it offline. |

|  |
| --- |
| **See Also:**  [*Oracle9i SQL Reference*](https://docs.oracle.com/cd/B10500_01/server.920/a96540/toc.htm) for more information about defining and altering default temporary tablespaces |

**Using Multiple Tablespaces**

A very small database may need only the SYSTEM tablespace; however, Oracle Corporation recommends that you create at least one additional tablespace to store user data separate from data dictionary information. This gives you more flexibility in various database administration operations and reduces contention among dictionary objects and schema objects for the same datafiles.

You can use multiple tablespaces to perform the following tasks:

* Control disk space allocation for database data
* Assign specific space quotas for database users
* Control availability of data by taking individual tablespaces online or offline
* Perform partial database backup or recovery operations
* Allocate data storage across devices to improve performance

A database administrator can use tablespaces to do the following actions:

* Create new tablespaces
* Add datafiles to tablespaces
* Set and alter default segment storage settings for segments created in a tablespace
* Make a tablespace read-only or read/write
* Make a tablespace temporary or permanent
* Drop tablespaces.

**Managing Space in Tablespaces**

Tablespaces allocate space in extents. Tablespaces can use two different methods to keep track of their free and used space:

* **Locally managed tablespaces**: Extent management by the tablespace
* **Dictionary managed tablespaces:** Extent management by the data dictionary

When you create a tablespace, you choose one of these methods of space management. You cannot alter the method at a later time.

|  |
| --- |
| **Note:**  If you do not specify extent management when you create a tablespace, then the default is locally managed. |

|  |
| --- |
| **See Also:**  ["Extents Overview"](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c03block.htm#17523) |

**Locally Managed Tablespaces**

A tablespace that manages its own extents maintains a bitmap in each datafile to keep track of the free or used status of blocks in that datafile. Each bit in the bitmap corresponds to a block or a group of blocks. When an extent is allocated or freed for reuse, Oracle changes the bitmap values to show the new status of the blocks. These changes do not generate rollback information because they do not update tables in the data dictionary (except for special cases such as tablespace quota information).

Locally managed tablespaces have the following advantages over dictionary managed tablespaces:

* Local management of extents automatically tracks adjacent free space, eliminating the need to coalesce free extents.
* Local management of extents avoids recursive space management operations. Such recursive operations can occur in dictionary managed tablespaces if consuming or releasing space in an extent results in another operation that consumes or releases space in a data dictionary table or rollback segment.

The sizes of extents that are managed locally can be determined automatically by the system. Alternatively, all extents can have the same size in a locally managed tablespace and override object storage options.

The LOCAL clause of the CREATE TABLESPACE or CREATE TEMPORARY TABLESPACE statement is specified to create locally managed permanent or temporary tablespaces, respectively.

**Segment Space Management in Locally Managed Tablespaces**

When you create a locally managed tablespace using the CREATE TABLESPACE statement, the SEGMENT SPACE MANAGEMENT clause lets you specify how free and used space within a segment is to be managed. Your choices are:

* AUTO

This keyword tells Oracle that you want to use bitmaps to manage the free space within segments. A bitmap, in this case, is a map that describes the status of each data block within a segment with respect to the amount of space in the block available for inserting rows. As more or less space becomes available in a data block, its new state is reflected in the bitmap. Bitmaps enable Oracle to manage free space more automatically; thus, this form of space management is called automatic segment-space management.

* MANUAL

This keyword tells Oracle that you want to use free lists for managing free space within segments. Free lists are lists of data blocks that have space available for inserting rows. MANUAL is the default.

|  |
| --- |
| **See Also:**   * + *[Oracle9i SQL Reference](https://docs.oracle.com/cd/B10500_01/server.920/a96540/toc.htm)* for details about SQL statements  * + *[Oracle9i Database Administrator's Guide](https://docs.oracle.com/cd/B10500_01/server.920/a96521/toc.htm)* for more information about managing SQL statements  * + ["Determine the Number and Size of Extents"](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c03block.htm" \l "5476)  * + ["Temporary Tablespaces for Sort Operations"](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "5902) for more information about temporary tablespaces |

**Dictionary Managed Tablespaces**

If you created your database with an earlier version of Oracle, then you could be using dictionary managed tablespaces. For a tablespace that uses the data dictionary to manage its extents, Oracle updates the appropriate tables in the data dictionary whenever an extent is allocated or freed for reuse. Oracle also stores rollback information about each update of the dictionary tables. Because dictionary tables and rollback segments are part of the database, the space that they occupy is subject to the same space management operations as all other data.

**Multiple Block Sizes**

The block size of the SYSTEM tablespace is the **standard block size**. This is set when the database is created and can be any valid size.

You can specify up to four block sizes, in addition to a standard block size. In the initialization file, you can configure subcaches within the buffer cache for each of these block sizes. Subcaches can also be configured while an instance is running. You can create tablespaces having any of these block sizes. The standard block size is used for the system tablespace and most other tablespaces.

|  |
| --- |
| **Note:**  All partitions of a partitioned object must reside in tablespaces of a single block size. |

Multiple block sizes are useful primarily when transporting a tablespace from an OLTP database to an enterprise data warehouse. This facilitates transport between databases of different block sizes.

|  |
| --- |
| **See Also:**   * ["Transport of Tablespaces Between Databases"](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "9370)  * *[Oracle9i Data Warehousing Guide](https://docs.oracle.com/cd/B10500_01/server.920/a96520/toc.htm)* for information about transporting tablespaces in data warehousing environments |

**Online and Offline Tablespaces**

A database administrator can bring any tablespace other than the SYSTEM tablespace **online** (accessible) or **offline** (not accessible) whenever the database is open. The SYSTEM tablespace is always online when the database is open because the data dictionary must always be available to Oracle.

A tablespace is usually online so that the data contained within it is available to database users. However, the database administrator can take a tablespace offline for maintenance or backup and recovery purposes.

**When a Tablespace Goes Offline**

When a tablespace goes offline, Oracle does not permit any subsequent SQL statements to reference objects contained in that tablespace. Active transactions with completed statements that refer to data in that tablespace are not affected at the transaction level. Oracle saves rollback data corresponding to those completed statements in a deferred rollback segment in the SYSTEM tablespace. When the tablespace is brought back online, Oracle applies the rollback data to the tablespace, if needed.

When a tablespace goes offline or comes back online, this is recorded in the data dictionary in the SYSTEM tablespace. If a tablespace is offline when you shut down a database, the tablespace remains offline when the database is subsequently mounted and reopened.

You can bring a tablespace online only in the database in which it was created because the necessary data dictionary information is maintained in the SYSTEM tablespace of that database. An offline tablespace cannot be read or edited by any utility other than Oracle. Thus, offline tablespaces cannot be transposed to other databases.

Oracle automatically switches a tablespace from online to offline when certain errors are encountered. For example, Oracle switches a tablespace from online to offline when the database writer process, DBW*n*, fails in several attempts to write to a datafile of the tablespace. Users trying to access tables in the offline tablespace receive an error. If the problem that causes this disk I/O to fail is media failure, you must recover the tablespace after you correct the problem.

|  |
| --- |
| **See Also:**   * ["Temporary Tablespaces for Sort Operations"](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "5902) for more information about transferring online tablespaces between databases  * *[Oracle9i Database Utilities](https://docs.oracle.com/cd/B10500_01/server.920/a96652/toc.htm)* for more information about tools for data transfer |

**Use of Tablespaces for Special Procedures**

If you create multiple tablespaces to separate different types of data, you take specific tablespaces offline for various procedures. Other tablespaces remain online, and the information in them is still available for use. However, special circumstances can occur when tablespaces are taken offline. For example, if two tablespaces are used to separate table data from index data, the following is true:

* If the tablespace containing the indexes is offline, then queries can still access table data because queries do not require an index to access the table data.
* If the tablespace containing the tables is offline, then the table data in the database is not accessible because the tables are required to access the data.

If Oracle has enough information in the online tablespaces to run a statement, it does so. If it needs data in an offline tablespace, then it causes the statement to fail.

**Read-Only Tablespaces**

The primary purpose of read-only tablespaces is to eliminate the need to perform backup and recovery of large, static portions of a database. Oracle never updates the files of a read-only tablespace, and therefore the files can reside on read-only media such as CD-ROMs or WORM drives.

|  |
| --- |
| **Note:**  Because you can only bring a tablespace online in the database in which it was created, read-only tablespaces are not meant to satisfy archiving or data publishing requirements. |

Read-only tablespaces cannot be modified. To update a read-only tablespace, first make the tablespace read/write. After updating the tablespace, you can then reset it to be read-only.

Because read-only tablespaces cannot be modified, and as long as they have not been made read-write at any point, they do not need repeated backup. Also, if you need to recover your database, you do not need to recover any read-only tablespaces, because they could not have been modified.

|  |
| --- |
| **See Also:**   * *[Oracle9i Database Administrator's Guide](https://docs.oracle.com/cd/B10500_01/server.920/a96521/toc.htm)* for more information about changing a tablespace to read-only or read/write mode  * *[Oracle9i SQL Reference](https://docs.oracle.com/cd/B10500_01/server.920/a96540/toc.htm)* for more information about the ALTER TABLESPACE statement  * *[Oracle9i Backup and Recovery Concepts](https://docs.oracle.com/cd/B10500_01/server.920/a96519/toc.htm)* for more information about recovery |

**Temporary Tablespaces for Sort Operations**

You can manage space for sort operations more efficiently by designating **temporary tablespaces** exclusively for sorts. Doing so effectively eliminates serialization of space management operations involved in the allocation and deallocation of sort space.

All operations that use sorts, including joins, index builds, ordering, computing aggregates (GROUP BY), and collecting optimizer statistics, benefit from temporary tablespaces. The performance gains are significant with Real Application Clusters.

**Sort Segments**

A temporary tablespace can be used only for sort segments. A temporary tablespace is not the same as a tablespace that a user designates for temporary segments, which can be any tablespace available to the user. No permanent schema objects can reside in a temporary tablespace.

Sort segments are used when a segment is shared by multiple sort operations. One sort segment exists for every instance that performs a sort operation in a given tablespace.

Temporary tablespaces provide performance improvements when you have multiple sorts that are too large to fit into memory. The sort segment of a given temporary tablespace is created at the time of the first sort operation. The sort segment expands by allocating extents until the segment size is equal to or greater than the total storage demands of all of the active sorts running on that instance.

|  |
| --- |
| **See Also:**  [Chapter 2, "Data Blocks, Extents, and Segments"](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c03block.htm#2528) for more information about segments |

**Creation of Temporary Tablespaces**

You can create temporary tablespaces by using the CREATE TABLESPACE or CREATE TEMPORARY TABLESPACE statement.

|  |
| --- |
| **See Also:**   * ["Temporary Datafiles"](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "10561) for information about TEMPFILES  * ["Managing Space in Tablespaces"](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "4346) for information about locally managed and dictionary managed tablespaces  * *[Oracle9i SQL Reference](https://docs.oracle.com/cd/B10500_01/server.920/a96540/toc.htm)* for information about the CREATE TABLESPACE, CREATE TEMPORARY TABLESPACE, and ALTER TABLESPACE statements  * *[Oracle9i Database Performance Tuning Guide and Reference](https://docs.oracle.com/cd/B10500_01/server.920/a96533/toc.htm)* for information about setting up temporary tablespaces for sorts and hash joins |

**Transport of Tablespaces Between Databases**

A **transportable tablespace** lets you move a subset of an Oracle database from one Oracle database to another on the same platform. You can clone a tablespace and plug it into another database, copying the tablespace between databases, or you can unplug a tablespace from one Oracle database and plug it into another Oracle database, moving the tablespace between databases on the same platform.

Moving data by transporting tablespaces can be orders of magnitude faster than either export/import or unload/load of the same data, because transporting a tablespace involves only copying datafiles and integrating the tablespace metadata. When you transport tablespaces you can also move index data, so you do not have to rebuild the indexes after importing or loading the table data.

|  |
| --- |
| **Note:**  You can transport tablespaces only between Oracle databases that use the same character set and that run on compatible platforms from the same hardware vendor. |

**How to Move or Copy a Tablespace to Another Database**

To move or copy a set of tablespaces, you must make the tablespaces read-only, copy the datafiles of these tablespaces, and use export/import to move the database information *(***metadata***)* stored in the data dictionary. Both the datafiles and the metadata export file must be copied to the target database. The transport of these files can be done using any facility for copying flat files, such as the operating system copying facility, ftp, or publishing on CDs.

After copying the datafiles and importing the metadata, you can optionally put the tablespaces in read/write mode.

|  |
| --- |
| **Note:**  In a database with a locally managed SYSTEM tablespace, dictionary tablespaces cannot be created. It is possible to plug in a dictionary managed tablespace using the transportable feature, but it cannot be made writable. |

|  |
| --- |
| **See Also:**   * *[Oracle9i Database Administrator's Guide](https://docs.oracle.com/cd/B10500_01/server.920/a96521/toc.htm)* for details about how to move or copy tablespaces to another database  * *[Oracle9i Database Utilities](https://docs.oracle.com/cd/B10500_01/server.920/a96652/toc.htm)* for import/export information  * ["The SYSTEM Tablespace"](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm" \l "1003) for |

**Datafiles Overview**

A tablespace in an Oracle database consists of one or more physical **datafiles**. A datafile can be associated with only one tablespace and only one database.

Oracle creates a datafile for a tablespace by allocating the specified amount of disk space plus the overhead required for the file header. When a datafile is created, the operating system under which Oracle runs is responsible for clearing old information and authorizations from a file before allocating it to Oracle. If the file is large, this process can take a significant amount of time. The first tablespace in any database is always the SYSTEM tablespace, so Oracle automatically allocates the first datafiles of any database for the SYSTEM tablespace during database creation.

|  |
| --- |
| **See Also:**  Your Oracle operating system-specific documentation for information about the amount of space required for the file header of datafiles on your operating system |

**Datafile Contents**

When a datafile is first created, the allocated disk space is formatted but does not contain any user data. However, Oracle reserves the space to hold the data for future segments of the associated tablespace--it is used exclusively by Oracle. As the data grows in a tablespace, Oracle uses the free space in the associated datafiles to allocate extents for the segment.

The data associated with schema objects in a tablespace is physically stored in one or more of the datafiles that constitute the tablespace. Note that a schema object does not correspond to a specific datafile; rather, a datafile is a repository for the data of any schema object within a specific tablespace. Oracle allocates space for the data associated with a schema object in one or more datafiles of a tablespace. Therefore, a schema object can span one or more datafiles. Unless table **striping** is used (where data is spread across more than one disk), the database administrator and end users cannot control which datafile stores a schema object.

|  |
| --- |
| **See Also:**  [Chapter 2, "Data Blocks, Extents, and Segments"](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c03block.htm#2528) for more information about use of space |

**Size of Datafiles**

You can alter the size of a datafile after its creation or you can specify that a datafile should dynamically grow as schema objects in the tablespace grow. This functionality enables you to have fewer datafiles for each tablespace and can simplify administration of datafiles.

|  |
| --- |
| **Note:**  You need sufficient space on the operating system for expansion. |

|  |
| --- |
| **See Also:**  [*Oracle9i Database Administrator's Guide*](https://docs.oracle.com/cd/B10500_01/server.920/a96521/toc.htm) for more information about resizing datafiles |

**Offline Datafiles**

You can take tablespaces offline or bring them online at any time, except for the SYSTEM tablespace. All of the datafiles of a tablespace are taken offline or brought online as a unit when you take the tablespace offline or bring it online, respectively.

You can take individual datafiles offline. However, this is usually done only during some database recovery procedures.

**Temporary Datafiles**

Locally managed temporary tablespaces have temporary datafiles (**tempfiles**), which are similar to ordinary datafiles, with the following exceptions:

* Tempfiles are always set to NOLOGGING mode.
* You cannot make a tempfile read-only.
* You cannot rename a tempfile.
* You cannot create a tempfile with the ALTER DATABASE statement.
* When you create or resize tempfiles, they are not always guaranteed allocation of disk space for the file size specified. On certain file systems (for example, UNIX) disk blocks are allocated not at file creation or resizing, but before the blocks are accessed.

|  |
| --- |
| **Caution:**  This enables fast tempfile creation and resizing; however, the disk could run of space later when the tempfiles are accessed. |

* Tempfile information is shown in the dictionary view DBA\_TEMP\_FILES and the dynamic performance view V$TEMPFILE, but not in DBA\_DATA\_FILES or the V$DATAFILE view.

|  |
| --- |
| **See Also:**  ["Managing Space in Tablespaces"](https://docs.oracle.com/cd/B10500_01/server.920/a96524/c04space.htm#4346) for more information about locally managed tablespaces |

**Control Files Overview**

The database control file is a small binary file necessary for the database to start and operate successfully. A control file is updated continuously by Oracle during database use, so it must be available for writing whenever the database is open. If for some reason the control file is not accessible, then the database cannot function properly.

Each control file is associated with only one Oracle database.

**Control File Contents**

A control file contains information about the associated database that is required for access by an instance, both at startup and during normal operation. Control file information can be modified only by Oracle; no database administrator or user can edit a control file.

Among other things, a control file contains information such as:

* The database name
* The timestamp of database creation
* The names and locations of associated datafiles and online redo log files
* Tablespace information
* Datafile offline ranges
* The log history
* Archived log information
* Backup set and backup piece information
* Backup datafile and redo log information
* Datafile copy information
* The current log sequence number
* Checkpoint information

The database name and timestamp originate at database creation. The database name is taken from either the name specified by the initialization parameter DB\_NAME or the name used in the CREATE DATABASE statement.

Each time that a datafile or an online redo log file is added to, renamed in, or dropped from the database, the control file is updated to reflect this physical structure change. These changes are recorded so that:

* Oracle can identify the datafiles and online redo log files to open during database startup
* Oracle can identify files that are required or available in case database recovery is necessary

Therefore, if you make a change to the physical structure of your database (using ALTER DATABASE statements), then you should immediately make a backup of your control file.

Control files also record information about checkpoints. Every three seconds, the checkpoint process (CKPT) records information in the control file about the checkpoint position in the online redo log. This information is used during database recovery to tell Oracle that all redo entries recorded before this point in the online redo log group are not necessary for database recovery; they were already written to the datafiles.

|  |
| --- |
| **See Also:**   * *[Oracle9i Recovery Manager User's Guide](https://docs.oracle.com/cd/B10500_01/server.920/a96566/toc.htm)*  * *[Oracle9i User-Managed Backup and Recovery Guide](https://docs.oracle.com/cd/B10500_01/server.920/a96572/toc.htm)*   for information about backing up a database's control file |

**Multiplexed Control Files**

As with online redo log files, Oracle enables multiple, identical control files to be open concurrently and written for the same database.

By storing multiple control files for a single database on different disks, you can safeguard against a single point of failure with respect to control files. If a single disk that contained a control file crashes, then the current instance fails when Oracle attempts to access the damaged control file. However, when other copies of the current control file are available on different disks, an instance can be restarted easily without the need for database recovery.

If *all* control files of a database are permanently lost during operation, then the instance is aborted and media recovery is required. Media recovery is not straightforward if an older backup of a control file must be used because a current copy is not available. Therefore, it is strongly recommended that you adhere to the following practices:

* Use multiplexed control files with each database
* Store each copy on a different physical disk
* Use operating system mirroring
* Monitor backups