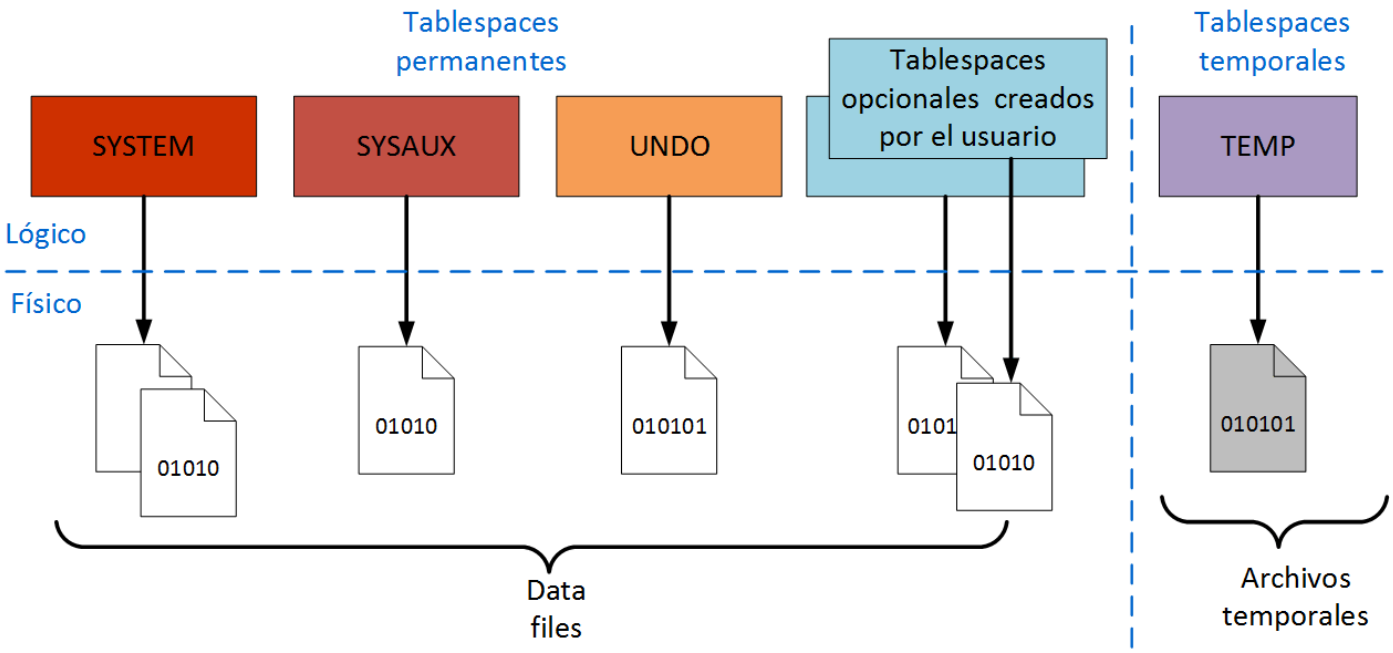
TEMA 6 - Parte 2
Administración de las estructuras lógicas de almacenamiento

6.7. TABLESPACES

- Como se mencionó anteriormente, un tablespace representa una unidad de almacenamiento lógico que agrupa a un conjunto de estructuras lógicas de almacenamiento (tablas, índices, etc.).
- Un tablespace puede ser visualizado como un ***contenedor de segmentos.***
- A nivel físico, los datos del tablespace se almacenan en data files o en archivos temporales.
- Para realizar la creación de un tablespace se emplean las instrucciones
  - `create tablespace`
  - `create temporary tablesapce`
  - `create undo tablespace.`
- Para modificar la configuración de un tablespace se emplean las instrucciones
  - `alter tablespace o`
  - `alter database`
- Para ejecutar estas 2 instrucciones se requieren los privilegios de sistema correspondientes.

- Una base de datos debe contar al menos con 2 tablespaces: `system` y `sysaux`. Sin embargo, se recomienda crear tablespaces adicionales para una mejor operación.



6.8. TIPOS DE TABLESPACES.

En la imagen anterior se observan 2 tipos de tablespaces:

- Tablespaces permanentes
- Tablespaces temporales.

6.8.1. Tablespaces permanentes.

- Encargados de almacenar los datos de los objetos de un esquema (tablas, índices, etc.).
- Los segmentos del tablespace son almacenados en data files de forma física.
- A cada usuario se le asigna un tablespace por default.

6.8.1.1. System tablespace:

- Empleado para almacenar información necesaria para realizar la administración de la base de datos.
- Entre otras cosas, este tablespace incluye:
  - El diccionario de datos.
  - Vistas y tablas que contienen información administrativa de la base de datos.
  - Código compilado (PL/SQL, procedimientos, etc.).

6.8.1.2. Sysaux tablespace

- Representa a un tablespace auxiliar para el tablespace `system`.
- Empleado para almacenar datos de componentes adicionales instalados en la base de datos. De esta forma es posible reducir el número de tablespaces requeridos para la correcta operación de la base de datos.
- Si este tablespace no se encuentra disponible, la BD sigue operando pero con funcionalidad limitada.

6.8.1.3. Undo tablespace

- Empleado para almacenar datos undo.
- La administración de este tipo de tablespaces siempre es automática. La BD administra y determina la mejor configuración que requiere este tipo de datos, en particular para ejecución de consultas que requieren un tiempo mayor de ejecución.
- Pueden existir múltiples tablespaces tipo undo, pero se usa uno solo a la vez.
- En caso de no existir, los datos Undo se almacenan en el tablespace SYSTEM (considerada mala práctica).

6.8.2. Tablespaces temporales

- Contiene datos que persisten únicamente mientras la sesión que los creó existe. Al terminar, los datos se eliminan.
- Los datos de estos tablespaces se guardan en ***archivos temporales.***
- Los datos de un tablespace temporal no generan datos REDO.
- La administración de las extensiones de un temp tablespace siempre es ***locally managed*** y sus extensiones son creadas del mismo tamaño: ***uniform***.
- A nivel general permite la ejecución de múltiples tareas o algoritmos concurrentes que no caben en memoria. Por ejemplo, realizar ordenamientos, etc.
- Entre los datos que se ordenan se encuentran:
  - Resultados intermedios de ordenamiento
  - Datos de tablas temporales e índices temporales
  - LOBs temporales
  - Árboles B-Tree temporales.

- Por default se crea un tablespace temporal llamado `temp` durante la creación de la BD.
- Es posible crear tablespaces temporales adicionales.
- A cada usuario se le asigna un tablespace temporal. Por default todos los usuarios comparten el tablespace `temp`.
- Para determinar los datos del tablespace temporal que se usa en una BD se puede emplear la siguiente instrucción:

```
select property_name, property_value
from database_properties where
property_name='DEFAULT_TEMP_TABLESPACE';
```

PROPERTY_NAME	PROPERTY_VALUE
DEFAULT_TEMP_TABLESPACE	TEMPTS1

- La siguiente instrucción se emplea para verificar el espacio empleado en un tablespace temporal.

```
select * from dba_temp_free_space;
```

	TABLESPACE_NAME	TABLESPACE_SIZE	ALLOCATED_SPACE	FREE_SPACE	SHARED	INST_ID
1	TEMPTS1	20971520	7340032	18874368	SHARED	(null)

- La siguiente instrucción puede emplearse para cambiar el tablespace temporal por default que se emplea en la base de datos por algún otro tablespace temporal creado previamente.

```
alter database default temporary tablespace <tablespace_name>;
```

6.9. MODOS DE UN TABLESPACE

Los modos de un tablespace determinan la forma en la que se acceden

- Read/Write y Read only tablespace.
- Online y Offline tablespaces.

6.9.1. Read/Write y Read Only tablespaces

- Read/Write mode: Los usuarios pueden leer y escribir en el tablespace.
  - Cuando un tablespace se crea, el modo inicial es Read/Write.
  - Los tablespaces `system`, `sysaux` y los temporales siempre deben ser Read/Write.

- Read Only mode: Los usuarios solo pueden leer del tablespace.
  - Esta característica permite que un tablespace pueda ubicarse en medios de solo lectura (DVDs, etc.).
  - Un tablespace Read Only no requiere realizar backups periódicos, tampoco requiere recovery.

6.9.1.1. Transacciones al cambiar a modo Read Only.

- Si existen transacciones activas al intentar cambiar a un tablespace como read only, la instrucción esperará hasta que la transacción concluya.

6.9.2. Online y Offline tablespaces.

- Online: El tablespace está accesible o disponible para los usuarios.
  - Los tablespaces system y los temporales siempre deben estar en modo online.
- Offline: El tablespace no está accesible o disponible para los usuarios, es decir, no se puede realizar operación alguna.
  - Si los usuarios intentan acceder a un tablespace offline, se producirá un error.
  - Transacciones activas con sentencias ejecutadas previamente al momento de pasar un tablespace en modo offline no serán afectadas.
  - Los datos Undo que se pueden generar durante la ejecución de la transacción se guardan en el tablespace `system` y se mueven al tablespace una vez que este vuelve al modo Online.
- Un tablespace puede cambiar de Online a Offline de forma manual o automática. Por ejemplo:
  - Poner un tablespace offline de forma manual para ser respaldado, mantenimiento o para realizar recovery.
  - La BD puede poner un tablespace offline cuando ocurre algún problema. Por ejemplo, el DBWriter intenta escribir en data file reiteradamente sin tener éxito.

6.10. TAMAÑO DE LOS ARCHIVOS DE UN TABLESPACE.

Otra clasificación de los tablespaces es considerando al tipo de archivo que contiene:
- Big file tablespace
- Small file tablespace.

6.10.1. Small file tablespace.

- El tablespace puede contener a un conjunto de tablespaces o archivos temporales.
- Los archivos no pueden ser tan grandes como en el caso de un big file.
- Este tipo de tablespace representa la configuración por default.
- Cada uno de los archivos puede contener aproximadamente hasta 4 millones de bloques (2^{22})

6.10.2. Big file tablespaces.

- Define un solo data file o un solo archivo temporal de tamaño grande (hasta bloques con 4G).

6.10.2.1. Beneficios de un Bigfile tablespace

- Permite incrementar la capacidad de almacenamiento de la base de datos.
- La cantidad total de data files que puede tener un tablespace es limitada. Útil en casos donde se requiera una mayor capacidad de almacenamiento que sobre pasa este límite.
- Las operaciones de mantenimiento se realizan sobre el tablespace en lugar de hacerlo de forma individual a cada archivo.
- Un bigfile tablespace con bloques de datos de 8K puede contener un data file hasta de 32 TB.
- Un bigfile tablespace con bloques de datos de 32K puede contener un data file hasta de 128 TB.
- Puede contener aproximadamente hasta 4 billones de bloques (2^{32})
- Bigfile tablespace permite reducir el tamaño del parámetro `db_files` y como beneficio, se reduce la memoria necesaria en la SGA para almacenar los metadatos requeridos de todos los data files, así como la reducción del tamaño del control file.

- Este tipo de tablespaces permite simplificar su administración. Al ser un solo data file, la sintaxis se simplifica ya que no se requiere especificar instrucciones de administración para cada uno de los data files.

6.11. OTRAS CONFIGURACIONES.

6.11.1. Compresión de datos en un tablespace

- Al momento de crear un tablespace es posible indicar el tipo de compresión de datos que se empleará cuando se almacenen datos en sus tablas e índices.
- El tipo de compresión se puede especificar empleando la cláusula `default`.
- Dichas configuraciones pueden ser sobrescritas al crear tablas o índices que pertenecen al tablespace.

Ejemplos:

```
create tablespace ... default row store compress advanced ... ;

create tablespace ... default index compress advanced high ... ;
```

6.11.2. Cifrado de datos en un tablespace

Permite proteger los datos de un tablespace de accesos no autorizados empleando medios externos a la base de datos, por ejemplo:

  - Acceso directo a los data files desde el sistema operativo.
  - Acceso directo a los backups que se realicen de un tablespace, en especial cuando los respaldos se transportan a otras bases de datos.
- Cuando un tablespace es cifrado, significa que todos sus bloques de datos serán cifrados.
- Todos los tipos de segmentos que pudiera contener el tablespace son cifrados (tablas, índices, lob indexes, lobs, particiones tanto de tablas como de índices).
- Para garantizar el mayor nivel de seguridad, los datos que se escriben en tablespace undo, tablespace temporal, y redo logs también son cifrados.
- El cifrado de tablespaces hace uso de una funcionalidad llamada ***Transparent Data Encryption***.
- ***Transparent Data Encryption*** requiere de la creación de un ***keystore*** para almacenar una ***llave maestra de cifrado.*** Al abrir este keystore, esta estará disponible para todas las sesiones y será cerrada de forma explícita o al detener la base de datos.
- ***Transparent Data Encryption*** soporta los principales algoritmos de cifrado usados en la industria:
  - Advanced Encryption Standard (AES)
  - ARIA
  - GHOST
  - SEED
  - Triple Data Encryption Standard (3DES)

- El algoritmo por default que se emplea es AES con una longitud de llave de cifrado de 128 bits.
- Cifrar datos no genera penalización en cuanto a espacio de almacenamiento, pero si ligera penalización en el uso del procesador.

Ejemplo:

```
alter system set encryption wallet open identified by "wallet_password";

create tablespace encrypt_ts
  datafile '$oracle_home/dbs/encrypt_df.dbf' size 1m
  encryption using 'aes256' encrypt;
```

6.12. ADMINISTRACIÓN DE TABLESPACES.

6.12.1. Uso de múltiples tablespaces.

- Ofrecen una mayor flexibilidad para realizar diversas operaciones en la base de datos.
- Permiten separar los datos de los usuarios finales con los datos del diccionario.
- Permiten asignar los datos de diversas aplicaciones a su propio tablespace, evitando así la dependencia de un mismo tablespace para varias aplicaciones. Si el tablespace compartido falla, fallan todas las aplicaciones.
- Almacenar diferentes data files en diferentes tablespaces y en diferentes discos, mejora las operaciones I/O y reduce contención.
- La administración y configuración se realiza por cada tablespace de forma independiente. Esto permite contar con diversos escenarios de optimización propios de cada conjunto de datos (high update activity , read only activity, temporary storage).
- Posibilidad de realizar backups por tablespace.
- Disponibilidad a nivel de tablespaces. Si un tablespace está en modo offline, los demás no se afectan, continúan en modo online.
- Posibilidad de contar con ***tablespaces transportables***. El tablespace se puede copiar o mover a otra base de datos, inclusive a otro sistema operativo. Esta técnica es mucho más rápida con relación a la copia individual de los objetos que contiene ya que la copia se realiza a nivel físico (copiando sus data files).
- En cuanto seguridad, a cada usuario se le puede asignar una ***cuota*** de almacenamiento para cada tablespace.

Recomendaciones:

- Solo crear los tablespaces que sean necesarios para satisfacer las necesidades de cada base de datos. Esto se decide considerando el diseño y requerimientos de la BD.
- Asignar el mínimo número posible de data files al tablespace.
- Si se requiere agrandar la capacidad de almacenamiento de un tablespace, se tienen las siguientes opciones:
  - Agregar 1 o 2 nuevos data files de tamaño considerable.
  - Crear data files con la propiedad `autoextend` en lugar de crear múltiples data files de menor tamaño.

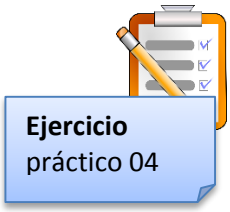
6.12.2. Asignación de cuotas a usuarios

- Para que un usuario pueda crear tablas, clusters, vistas materializadas, índices entre otros, el usuario de be contar con los siguientes privilegios:
  - Otorgarle el privilegio `create` correspondiente al tipo de objeto
  - Otorgarle una cuota de almacenamiento sobre el tablespace donde serán creados los segmentos de dichos objetos.

- La cuota de almacenamiento define el límite de espacio en disco asignado a un usuario. Puede ser ilimitado.
- Los detalles de las cuotas de almacenamiento asignadas a un usuario pueden consultarse en `dba_ts_quotas` o `user_ts_quotas`

Ejemplos:

```
create user myuser identified by mypassword quota unlimited on users;
create user myuser identified by mypassword quota  1GB on users;
create user myuser identified by mypassword quota  150MB on users;
```



6.12.3. Creación de tablesapces.

- Para crear un tablespace se emplean las instrucciones previa asignación del privilegio `create tablespace`

```
create tablespace
create temporary tablespace.
create undo tablespace.
```

A nivel general, para crear un tablespace se deben realizar las siguientes acciones:

- Crear las estructuras de directorios donde se guardarán los archivos que define el tablespace.
- Especificar el tamaño y la ruta completa de los archivos en la cláusula `create tablespace`.
- La instrucción `alter tablespace` se emplea para modificar la configuración del tablespace.

6.12.3.1. Cláusula bigfile / smallfile

- Se emplean para indicar si el tablespace podrá definir a varios data files (`smallfile`) o uno solo (`bigfile`).
- Si se omiten ambas, el default será `smallfile`.

6.12.3.2. Cláusula datafile / tempfile

- Cláusula empleada para especificar los data files que serán asignados a un tablespace.
- Existen múltiples configuraciones de esta cláusula dependiendo las configuraciones de almacenamiento (ASM, Oracle Managed Files). A nivel general las opciones más comunes se muestran a continuación.
  - Ruta y nombre del data file.
  - Cláusula `size`: indica el tamaño del data file.
  - Cláusula `reuse`: Si el archivo ya existe, este se reutiliza y su contenido se sobrescribe (se pierde).
  - Cláusula `autoextend`

6.12.3.3. Cláusula autoextend.

- Esta cláusula se emplea para configurar si un data file o temp file puede crecer de forma automática. Si la cláusula no se especifica, la capacidad de auto extenderse no estará habilitada. Existen las siguientes configuraciones:

- `on`: Habilita la capacidad de auto extensión
- `off`: Deshabilita la capacidad de auto extensión
- `next`: Indica la cantidad en bytes para agrandar o auto extender el data file el cual será aplicado cuando se requiera crear más extensiones. En caso de no especificarse, el archivo se extiende a la longitud de un bloque de datos.
- `maxsize`**:** Indica el tamaño máximo de crecimiento del data file.
- `unlimited`**:** No se limita el crecimiento de data file.

- Esta cláusula no puede emplearse en `create control file, alter database create datafile`.

6.12.3.1. Cláusula blocksize

- Se emplea para sobrescribir el tamaño del bloque de datos que se usará para almacenar los datos de los objetos del tablespace.

6.12.3.2. Cláusula online/offiline

- Establece si el tablespace estará online u offline justo después de ser creado.
- En caso de no especificarse, la opción por default es `online`.

6.12.3.3. Cláusula nologging / logging

- Indica si las operaciones DML que se aplican sobre los objetos del tablespace generarán datos redo
- Estas opciones no aplican para tablespaces undo y temporales.
- La opción por default es `logging`.
- Recordando, los datos REDO son vitales para realizar la recuperación de una base de datos, realizar operaciones `rollback`.
- Los datos undo se almacenan en un tablespace especial para este propósito (undo tablespace).
- En algunos casos es conveniente deshabilitar esta característica para mejorar desempeño.
- Un ejemplo de ello es la instrucción `create table .. as select` la cual no requiere almacenar datos REDO ya que se trata de hacer una copia de datos.
- Para deshabilitar la creación de datos REDO, se especifica la cláusula `nologging` en la instrucción `create tablespace`.
- Lo anterior implica que *ningún* objeto que pertenezca a este tablespace generará datos REDO.
- Es posible sobrescribir este parámetro a nivel de cada objeto, por ejemplo, en tabas. Para ello se especifica `nologging` en la instrucción `create table`.

Ejemplo:

```
create tablespace tbs_01
  datafile 'tbs_f2.dbf' size 40m
  online;
```

- Crea un TS con uno solo data file de 40m sin opción de auto extensión

Ejemplo:

```
create tablespace tbs_03
  datafile 'tbs_f03.dbf' size 20m
  logging;
```

- En este ejemplo el TS también estará online, y generará datos redo (default).

Ejemplo:

```
create tablespace tbs_02
  datafile 'tbs_f5.dbf' size 500k
    reuse
    autoextend on next 500k maxsize 100m
  logging
  online;
```

- En este ejemplo el tablespace reutilizará el archivo en caso de existir. Cuando se requiera más espacio, el data file crecerá 500K y podrá crecer hasta 100 MB.
- Notar que las cláusulas `size`, `reuse`, `autoextend` y su configuración se especifican justo después de la cláusula `datafile`.

Ejemplo:

```
create bigfile tablespace myappts4
  datafile '/u01/app/oracle/oradata/JRCBD2/myappts4.dbf' size 50g;
```

- Creación de un bigfile tablespace.

Ejemplo:

```
create temporary tablespace lmtemp
  tempfile '/u02/oracle/data/lmtemp01.dbf'
  size 20m reuse;
```

- Creación de un tablespace temporal

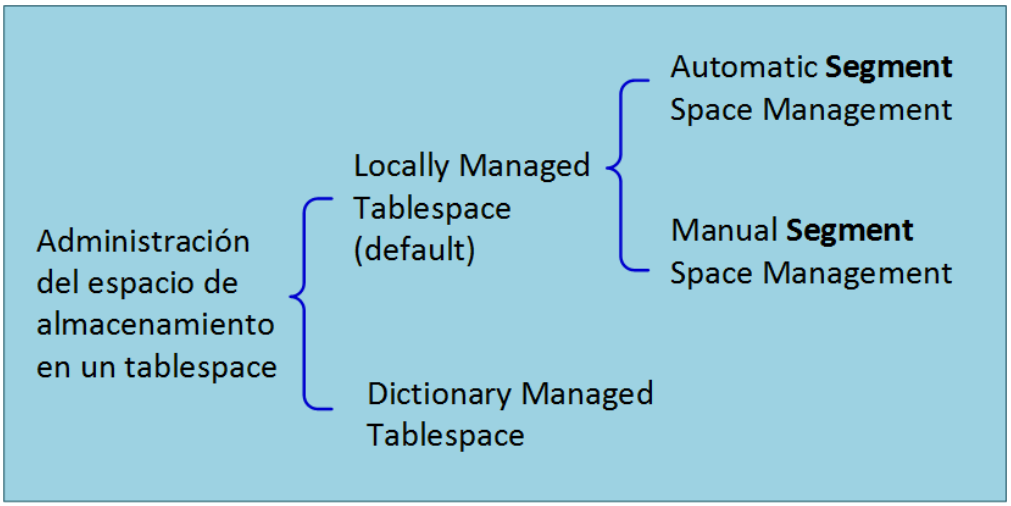
Ejemplo:

```
create undo tablespace undots1
  datafile 'undotbs_1a.dbf'
    size 10m
    reuse
    autoextend on
  retention guarantee;
```

- En el ejemplo anterior se realiza la creación de un tablespace undo.

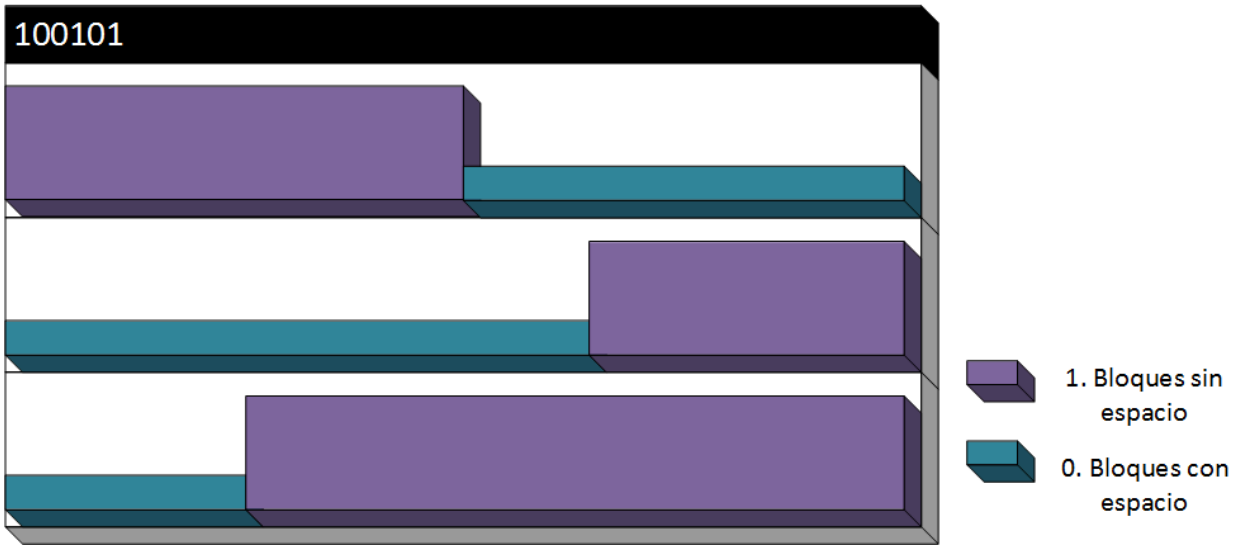
6.12.4. Administración del espacio de almacenamiento en un tablespace

- Cuando un objeto de la base de datos requiere de una nueva extensión, la base de datos aplica un algoritmo para encontrar y asignar extensiones.
- De forma similar, cuando un objeto ya no requiere una extensión, la base de datos emplea un algoritmo para hacer que dicha extensión esté nuevamente disponible para ser empleada.
- Para realizar la administración de las extensiones Existen las siguientes técnicas:
  - Locally managed tablespaces: Hace uso de bitmaps, representa la opción por default y recomendada. Dentro de este tipo de tablespaces, existen a su vez 2 técnicas para administrar los segmentos (revisadas en el tema anterior):
    - Automatic Segment Space Management (**ASSM**)
    - Manual Segment Space Management (**MSSM**).
  - Dictionary Managed tablespaces: Hace uso del diccionario de datos.



6.12.4.1. Locally Managed tablespaces.

- En este tipo de tablespace, la administración de la información de las extensiones empleadas en cada tablespace se realiza a través del uso de un ***mapa de bits*** almacenado en el header del data file. Cada bit representa a un conjunto de bloques:



Ventajas:

- Evita y reduce el uso del diccionario de datos para realizar la administración de las extensiones.
- Permite realizar operaciones de reservación de extensiones de forma concurrente, el *bitmap* puede ser actualizado de forma concurrente.
- Mejora en desempeño. La detección de espacio libre a través de una cadena de bits se simplifica.

Tip: No confundir este mapa de bits con el mapa de bits empleado en ASSM visto en el tema anterior

Ejemplo:

```
create tablespace myappts1
datafile '/u01/app/oracle/oradata/JRCBD2/myappts1.dbf' size 50m
extent management local autoallocate;
```

- En el ejemplo anterior se crea el tablespace llamado `myappts1` con un data file de 50MB.

- Se especifica `extent management` para indicar que la técnica de administración del espacio será del tipo *locally managed*. Se debe especificar esta opción a pesar de ser la opción por default.
- Si los objetos del tablespace son de tamaño variable con extensiones de tamaños variables, se recomienda especificar la cláusula `autoallocate` para realizar la administración de *extensiones* de forma automática (default) en lugar de la opción `uniform`.

Ejemplo:

```
create tablespace myappts2
datafile '/u01/app/oracle/oradata/JRCBD2/myappts2.dbf' size 50m
extent management local uniform size 128k;
```

- En el ejemplo anterior se crea un table space *locally managed* con administración uniforme de extensiones.
- Esto significa que el tamaño de cada extensión es uniforme (del mismo tamaño) y en este caso de 128k. Por default el tamaño es de 1 MB.
- `uniform` puede emplearse cuando se sabe con certeza el tamaño de los objetos que se van a guardar así como el tamaño requerido para cada extensión. La ventaja de esta opción radica en que el espacio se va a utilizar de forma óptima sin dejar huecos de espacio como es el caso de `autoallocate`. El problema es que no siempre se conoce, o no siempre es posible contar con objetos de tamaño uniforme. Por esta razón generalmente `autoallocate` es la mejor opción.

6.12.4.2. Tipos de administración de segmentos para un tablespace locally managed.

- En el tema anterior se revisaron 2 técnicas con las que se realiza la administración del espacio disponible en un segmento (revisar el tema anterior para mayores detalles).
  - Automatic Segment Space Management (**ASSM**)
  - Manual segment space management (**MSSM**).

- Para hacer uso de la administración automática, se debe emplear `segment space management auto`.

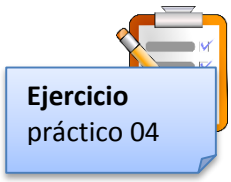
Ejemplo:

```
create tablespace myappts3
datafile '/u01/app/oracle/oradata/JRCBD2/myappts3.dbf' size 50m
extent management local autoallocate
segment space management auto;
```

Algunas notas interesantes:

- La estrategia para administrar la creación de segmentos se especifica al momento de crear el tablespace y este se aplicará a todos los segmentos subsecuentemente creados.
- Esta opción no puede modificarse una vez que el tablespace ha sido creado.
- Si se especifica `local uniform` para administrar la creación de extensiones, cada extensión debe tener al menos 5 bloques de datos.
- Si se especifica la opción `local autoallocate` y si el tamaño del bloque es de 16k o mayor, se crearán extensiones con un tamaño mínimo de 5 bloques redondeados a 64Kb.

- En general se recomienda emplear las opciones `extent management local` y `segment space management autoallocate`



6.12.5. Pasar un tablespace a modo offline.

Existen 3 parámetros que controlan la forma en la que se pueden establecer un tablespace como *offline:*

Parámetro	Descripción
`normal`	• Representa la opción por default. • Empleado cuando no existen errores en los data files del tablespace. • Al no existir errores, el tablespace puede moverse al modo offline sin la necesidad de realizar una posible recuperación de datos. Se crea un checkpoint para garantizar consistencia.
`temporary`	• Existen errores en los data files. • Cuando un data file cae en una condición de error, de forma automática el data file se marca como offline. • En este modo, los data files restantes que aún están en modo online se pasan a modo offline previa ejecución de una operación de checkpoint. • Para regresar el tablespace en modo online, será necesario realizar una operación de recovery sobre los data files con errores.
`immediate`	• Todos los data files del tablespace son puestos en modo offline sin realizar checkpoint. • Lo anterior implica una operación de recovery sobre todos los data files. • `immediate` no puede usarse si la BD se está ejecutando en modo `noarchive` (se revisará más adelante).

Ejemplo:

```
alter tablespace myappts1 offline normal;
```

6.12.6. Pasar un tablespace a modo Online.

- Para activar un tablespace la BD debe estar iniciado en modo open.
- Si le tablespace fue puesto offline en modo `temporary` o `immediate`, se debe realizar recuperación de la BD antes de ser puesto `online`.

Ejemplo:

```
alter tablespace myappts1 online;
```

6.12.1. Pasar un tablespaces a modo read only.

- Para que un tablespace pueda marcarse como read only, se deben cumplir los siguientes requisitos:
  - El tablespace debe estar en modo online
  - No se puede marcar como read only al tablespace `system` o al tablespace `undo` que se esté empleando actualmente.

- El tablespace no debe estar involucrado en un backup online debido a que esta operación modifica el header del data file.
- El tablespace no puede ser temporal.

Ejemplo:

```
alter tablespace myappts1 read only;
```

6.12.2. Pasar un tablespace a modo read/write.

- El usuario debe contar con el privilegio alter tablespace o manage tablespace.
- Para que un tablespaces pueda ser puesto en modo read/write todos los data files y el propio tablespace deben estar en modo ***online***.
- La vista `v$datafile` puede ser consultada para mostrar el status de cada data file.

Ejemplo:

```
alter tablespace myappts1 read write;
```

6.13. MODIFICACIÓN Y MANTENIMIENTO DE TABLESPACES

6.13.1. Incrementar la capacidad de un tablespace.

Para incrementar la capacidad de un tablespace se pueden realizar varias acciones:

- Incrementar el tamaño del data file.
- Agregar un nuevo data file.
- Habilitar el crecimiento automático del data file a través del uso de `autoextend` tanto para small como para big files.

La sintaxis para estas acciones se revisará en el siguiente tema en el que se realiza la administración de los data files. Por ejemplo, para agregar un nuevo data file:

6.13.2. Acciones de mantenimiento para tablespaces locally managed.

- Agregar data files
- Ponerlo offline / online
- Read Only / Read write
- Renombrarlo
- Habilitar autoextend.

Cosas que no se pueden realizar:

- Cambiar la forma de administración de sus segmentos
- Cambiarlo a un tablespace temporal.

Ejemplo:

```
alter tablespace lmtbsb
  add datafile '/u02/oracle/data/lmtbsb02.dbf' size 1m;
```

6.13.3. Acciones de mantenimiento para un bigfile tablespace

- Modificar su tamaño:

```
alter tablespace bigtbs resize 80g;
```

- Uso de autoextend

```
alter tablespace bigtbs autoextend on next 20g;
```

6.13.4. Acciones de mantenimiento para un temporary tablespace

- Agregar un nuevo archivo temporal
- Cambiar su tamaño
- Poner alguno de sus archivos temporales offline / online. Ojo: el tablespace no puede ser puesto offline, pero si alguno de sus data files.
- Eliminar alguno de sus archivos temporales:

```
alter database tempfile '/u02/oracle/data/lmtemp02.dbf' drop
  including datafiles
```

- Liberar espacio en los archivos temporales del tablespace. El espacio que se utiliza para realizar operaciones en el tablespace, por ejemplo, ordenamientos, no se libera de forma inmediata. Solo se marca como espacio disponible. Para liberar este espacio de forma manual se emplea la cláusula `shrink`. La cláusula keep indica el tamaño mínimo que debe conservar el archivo temporal:

```
alter tablespace lmtemp1 shrink space keep 20m;
alter tablespace lmtemp2 shrink tempfile '/u02/oracle/data/lmtemp02.dbf';
```

6.13.5. Renombrar tablespaces.

```
alter tablespace users rename to usersts;
```

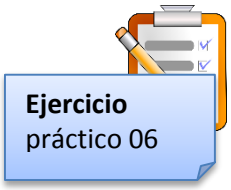
- Al renombrar un TS, la BD actualiza todas las referencias necesarias: control file, diccionario de datos, headers de los data files. Su Id no varía.
- Los tablespaces `system` y `sysaux` no se pueden renombrar.
- Si alguno de los data files o el propio tablespace están marcados como offline estos no se pueden renombrar.
- Para el caso de un tablespace tipo undo, el nombre se debe actualizar en el spfile (parámetro `undo_tablespace`). En casso de no usar un pfile, se genera error.

6.13.6. Eliminar tablespaces

- Importante: Posterior a la eliminación del tablespace, sus datos son ***irrecuperables***. Los segmentos que contiene el tablespace son eliminados.
- De aquí que sea necesario realizar un backup completo de la base de datos.
- Los data files del tablespace pueden ser eliminados en la misma instrucción, o pueden ser eliminados posteriormente de forma manual.
- Si los segmentos del tablespace están siendo empleados, el tablespace no se puede eliminar. Se recomienda primero ponerlo offline y posteriormente eliminarlo.

```
drop tablespace users including contents;
drop tablespace users including contents cascade constraints ;
drop tablespace users including contents and datafiles;
```

- `Cascade constraints` se emplea cuando existen FKs en tablas que no pertenecen al tablespace.



6.14. MOSTRANDO INFORMACIÓN DE LOS TABLESPACES.

La siguiente lista de vistas y vistas dinámicas contienen información acerca de los tablespaces:

```
v$tablespace
v$encrypted_tablespaces
dba_tablespaces, user_tablespaces
dba_tablespace_groups
dba_segments, user_segments
dba_extents, user_extents
dba_free_space, user_free_space
dba_temp_free_space
v$datafile
v$tempfile
dba_data_files
dba_temp_files
v$temp_extent_map
v$temp_extent_pool
v$temp_space_header
dba_users
dba_ts_quotas
v$sort_segment
v$tempseg_usage
```

Ejemplo:

```
select tablespace_name "tablespace",
  initial_extent "initial_ext",
  next_extent "next_ext",
  min_extents "min_ext",
  max_extents "max_ext",
  pct_increase
from dba_tablespaces;
```

TABLESPACE	INITIAL_EXT	NEXT_EXT	MIN_EXT	MAX_EXT	PCT_INCREASE
SYSTEM	65536	(null)	1	2147483645	(null)
SYSAUX	65536	(null)	1	2147483645	(null)
UNDOTBS1	65536	(null)	1	2147483645	(null)
TEMPTS1	1048576	1048576	1	(null)	0
USERS	65536	(null)	1	2147483645	(null)
USERTBS	65536	(null)	1	2147483645	(null)
APPS_TBS	65536	(null)	1	2147483645	(null)
INDX_TBS	65536	(null)	1	2147483645	(null)

- `Initial_extent`: se refiere al tamaño por default que se le asigna a la extensión inicial (en bytes).
- `Next_extent`: se refiere al tamaño por default que se le asigna a las extensiones incrementales.
- `Max_extent`: se refiere al número por default máximo de extensiones que pueden existir en el tablespace.

Ejemplo:

Mostrar los data files y sus correspondientes tablespaces:

```
select  file_name, blocks, tablespace_name
from dba_data_files;
```

FILE_NAME	BLOCKS	TABLESPACE_NAME
/u01/app/oracle/oradata/JRCBD2/system01.dbf	89600	SYSTEM
/u01/app/oracle/oradata/JRCBD2/sysaux01.dbf	70400	SYSAUX
/u01/app/oracle/oradata/JRCBD2/undotbs01.dbf	25600	UNDOTBS1
/u01/app/oracle/oradata/JRCBD2/users01.dbf	64000	USERS
/u01/app/oracle/oradata/JRCBD2/usertbs01.dbf	25600	USERTBS
/u01/app/oracle/oradata/JRCBD2/apps01.dbf	64000	APPS_TBS
/u01/app/oracle/oradata/JRCBD2/indx01.dbf	12800	INDX_TBS