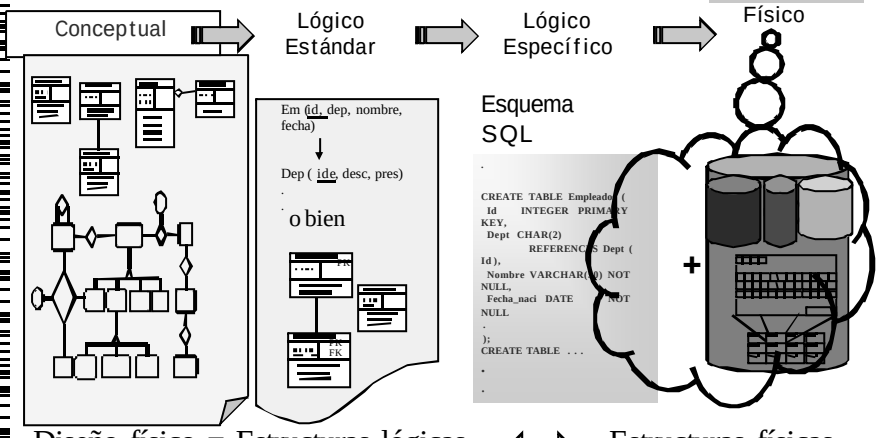
Tema IV: Administración de Bases de Datos

4.1- Diseño Físico

- 4.1.1- Introducción
- 4.1.2- Almacenamiento
- 4.1.3- Tablespaces
- 4.1.4- Segmentos
- 4.1.5- Extensiones
- 4.1.6- Bloques
- 4.1.7- Almacenamiento de tablas en Oracle
- 4.1.8- Índices
- 4.1.9- Clusters
- 4.1.10- Particiones
- 4.1.11- Ajuste de rendimiento

4.1.1- Introducción

Proceso de Diseño de Bases de Datos



Diseño físico = Estructuras lógicas ⟺ Estructuras físicas

4.1.1- Introducción

Diseño Físico: Motivación

- Buscamos implementación "suficientemente" eficiente, en una plataforma concreta
  - Hardware+SO+SGBD+ (aplicación)
- La implementación implícita (probablemente) es
  - Suficiente: en BD para pruebas funcionales, formación, demos
  - Insuficiente: en BD para pruebas de carga, producción

4.1.1- Introducción

Diseño Físico: Tareas

- Objetivo de esta etapa:
  - producir una descripción de la implementación de la base de datos en memoria secundaria. Esta descripción incluye las **estructuras de almacenamiento** y los **métodos de acceso** que se utilizarán para conseguir un acceso eficiente a los datos.
- Tareas:
  - Traducir el esquema lógico global para el SGBD específico.
  - Diseñar la representación física.
  - Diseñar los mecanismos de seguridad.
  - Pruebas de rendimiento. Monitorizar y afinar el sistema.

4.1.1- Introducción

Diseño Físico: Criterios

1. Mejorar el **rendimiento**
   - Espacio en memoria y en disco
   - Tiempo de procesador
   - Tiempo de disco
   - Contención
   - Coste de los procesos auxiliares
2. **Escalabilidad**
   - Volumen de usuarios y datos

4.1.1- Introducción

Diseño Físico: Criterios

3. **Disponibilidad / Integridad**
4. Facilidad de **administración**
5. **Integridad**

- Pero …
  - Medios limitados
  - Criterios contrapuestos
  - Pérdida de independencia

4.1.1- Introducción

Pero además del diseño físico de la BD, en el rendimiento también influyen …

- **Diseño de los procesos (en C/S)**
  - Separación entre BD y lógica
    - Restricciones separadas, Vistas, Proc. Almacenados, Disparadores
  - Conexiones, interacción y tráfico
- **Programación**
  - Los optimizadores no son perfectos
  - Optimización estática
  - Optimización dinámica. Estadísticas
- **Plataformas, la red**

4.1.1- Introducción

Un **buen diseño físico exige conocer** bien:

- Posibilidades del SGBD
- Posibilidades de los equipos de almacenamiento (Ej.: RAID)
- Interacción entre el SO y
  - SGBD
  - Equipos de almacenamiento
- Y cómo los procesos / usuarios utilizan la BD
  - Perfil de uso

4.1.1- Introducción

Diseño Físico: Prototipos

- Diseño "preventivo": Evaluación previa
  - Volúmenes, frecuencias, caminos, …
- Pruebas y prototipos
  - Esqueletos de los procesos críticos
  - Simulación de datos y usuarios
    - Perfil de carga
  - Herramientas de análisis de la ejecución
    - Planes, trazados y mediciones

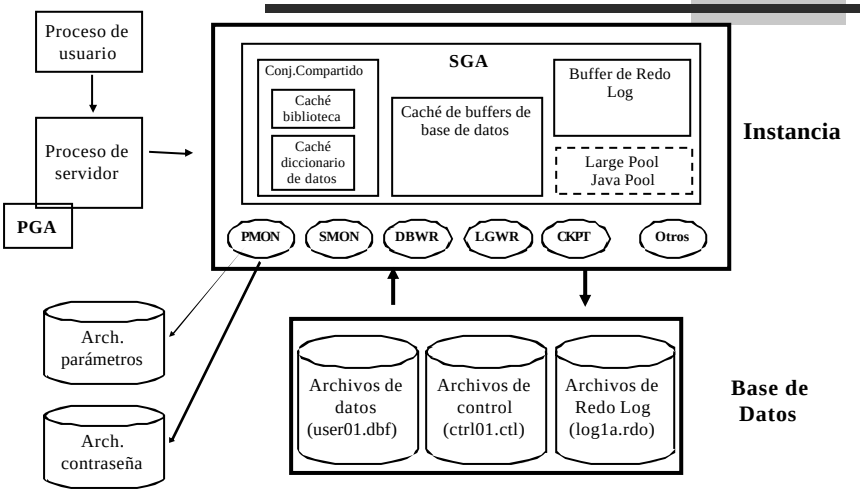
4.1.1- Introducción

Optimización y ajuste según Oracle

- **Objetivos**
  - Código SQL eficiente
  - Reservar recursos apropiados y suficientes (CPU, Memoria, Disco, E/S)
  - Analizar problemas de espera y contención
- **Enfoques**
  - Reactivo: resolver problemas que aparecen en producción
  - Proactivo: diseñar el sistema teniendo en cuenta el rendimiento.

4.1.1- Introducción

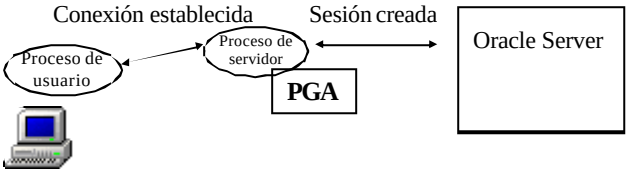
Arquitectura Oracle – Componentes principales



4.1.1- Introducción

Arquitectura Oracle – Componentes principales

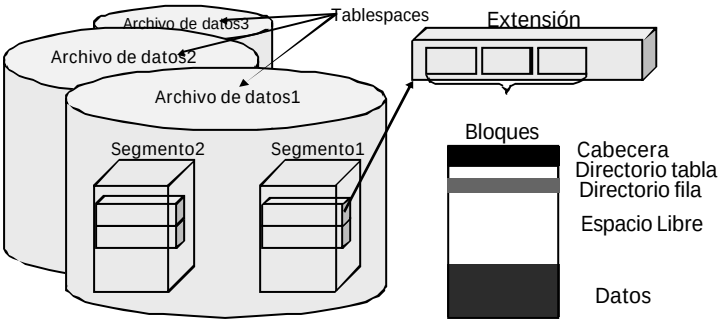
- **Instancia:**
  - **SGA (Área Global del Sistema):** Memoria compartida para almacenar las informaciones de control y los datos de la instancia.
    - Contiene Conjunto compartido, Caché de buffers de Base de Datos, Buffer Redo Log, Large Pool y Java Pool.
  - **PGA (Área Global de Programas o Proceso):** Memoria reservada para cada proceso de usuario que se conecte a una base de datos. Se asigna cuando se crea un proceso y se libera cuando se termina un proceso.



4.1.1- Introducción

Arquitectura Oracle – Componentes principales

- **Base de datos:**
  - Estructuras **lógicas**: jerarquía formada por tablespaces, segmentos, extensiones y bloques
  - Estructuras **físicas**: archivos de datos que forman los tablespace.



4.1.1- Introducción

Arquitectura Oracle – Componentes principales

- Instancias
  - Conjunto de procesos y estructuras en memoria (SGA)
  - Proporciona mecanismos de acceso y control de la BD
  - Los procesos (y parte del almacenamiento principal) son compartidos por todos los usuarios.
  - Sus parámetros están en el fichero init.ora, que se lee al arrancarla.

- Bases de datos
  - Conjunto de datos almacenado y accesible según una estructura lógica de tablas
  - Se divide en Tablespaces (uno o más)
  - Un tablespace consta de ficheros (uno o más)
  - Un fichero sólo pertenece a un tablespace, y puede contener varios objetos

4.1.2- Almacenamiento

El espacio: Nivel lógico y nivel físico

- El espacio lógico
  - visión desde los programas usuarios
  - atributos, filas, **tablas**, …

- El espacio físico
  - visión desde el SO
  - **ficheros** del SO y extensiones

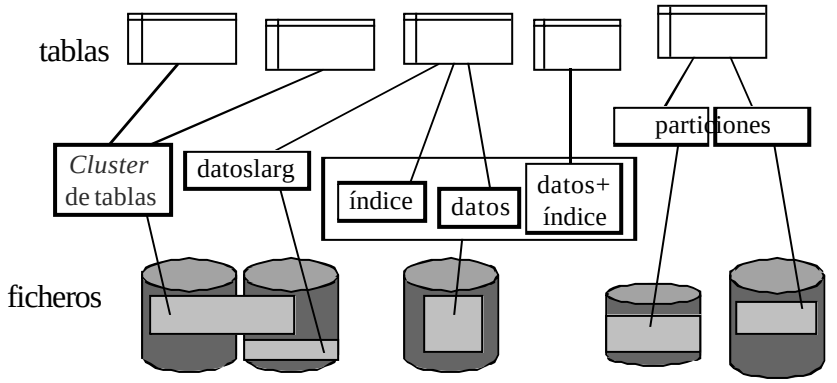
4.1.2- Almacenamiento

Un tercer nivel: El espacio virtual de almacenamiento

- Visión desde el SGBD
  - visión simple del espacio físico, para facilitar el direccionamiento, la independencia, ..
  - espacios (tablespaces,..)
  - campos, registros, páginas
  - particiones
  - clusters
  - índices

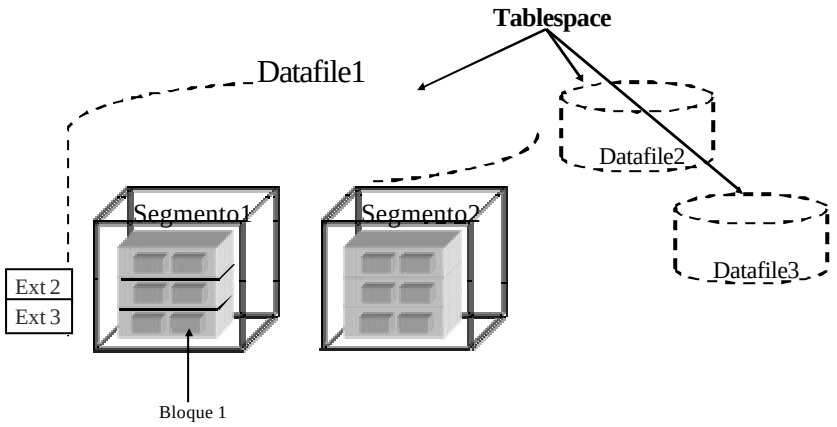
4.1.2- Almacenamiento

Correspondencia entre los tres niveles



4.1.2- Almacenamiento

Jerarquía de almacenamiento en Oracle: Estructuras virtuales

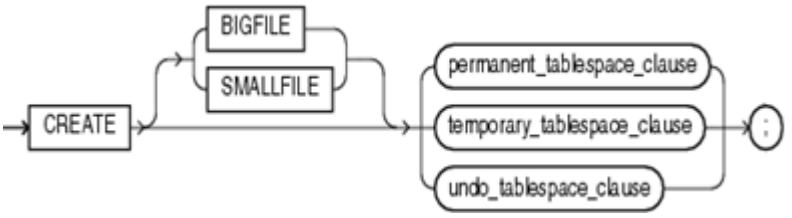


4.1.3- Tablespaces

- Es el espacio lógico de almacenamiento de datos (físicamente en *data files*)
- **Tipos** de tablespaces:
  - SYSTEM: encargado de almacenar
    - el diccionario de datos (tablas con información sobre la propia BD)
    - códigos PL/SQL fuentes y compilados, etc.
  - TEMP: almacena datos temporales
  - ROLLBACK: almacena información transaccional
  - DATA: almacena datos de la aplicación
- Posibles **estados** de un tablespace:
  - En línea (on line): a disposición de las aplicaciones y BDs
  - Desconectado (offline): datos no están disponibles, aunque lo esté la BD

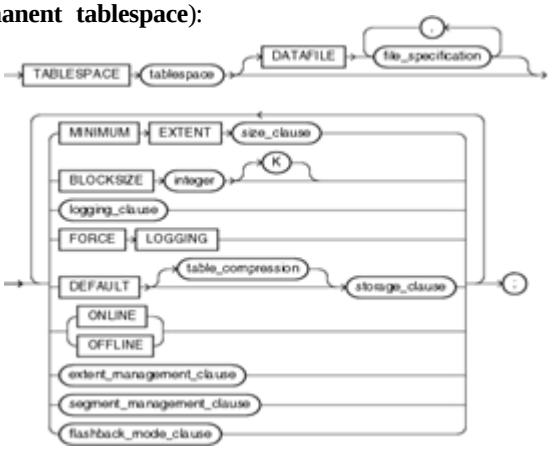
4.1.3- Tablespaces

Sintaxis:



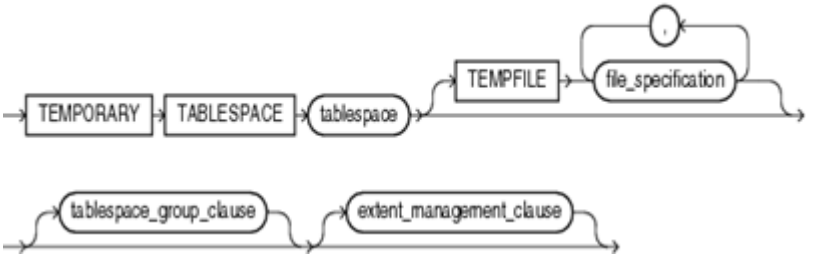
4.1.3- Tablespaces

Sintaxis (**permanent tablespace**):



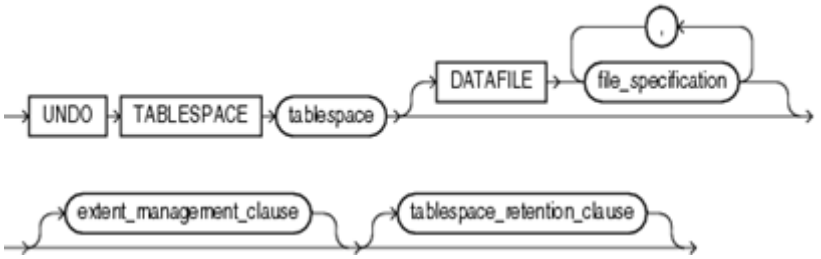
4.1.3- Tablespaces

Sintaxis (**temporal tablespace**):



4.1.3- Tablespaces

Sintaxis (**undo tablespace**):



4.1.3- Tablespaces

Sintaxis:

- **Bigfile** o **smallfile** para grandes o pequeños tablespaces del orden de teras, el defecto es smallfile y es el que se utilizaba hasta ahora (como máximo pude tener 32 Gb)
- **Datafile** para datos de la aplicación.
- **Tempfile** para datos temporales.
  - En general se debe de tener al menos un tablespace por esquema y un temporal general, salvo que la aplicación haga una utilización masiva del temporal, entonces se crea uno para el esquema oportuno

4.1.3- Tablespaces

Ejemplo:

Creamos el tablespace llamado prueba en la que todos sus objetos cuando son creados se guardan en redolog. Se guarda en donde se indica el datafile, con un tamaño de 5 Mb, creciendo de mega en mega hasta 10 Mb como máximo y lo gestiona Oracle automáticamente

```
CREATE SMALLFILE
    TABLESPACE “PRUEBA"
    LOGGING
    DATAFILE 'C:\Path\PRUEBA.ora' SIZE 5M AUTOEXTEND
    ON NEXT 1M MAXSIZE 10M EXTENT MANAGEMENT LOCAL SEGMENT
    SPACE MANAGEMENT AUTO
```


4.1.3- Tablespaces

Ejemplo:

Borra el tablespace, si no tiene objetos no hace falta “including contents” y borra el fichero del sistema operativo (“and datafiles”). Además borra todas las restricciones de clave ajena

```
SQL> DROP TABLESPACE prueba
      INCLUDING CONTENTS and datafiles
      CASCADE CONSTRAINTS
```

4.1.3- Tablespaces

- La forma de gestionar las extensiones de los tablespaces pude ser con la técnica del diccionario, única forma conocida antes de Oracle 8i.
  - Con el manejo del diccionario Oracle modifica una serie de tablas en el diccionario de datos en cuanto una extensión es asignada o liberada. Pero también almacena la información en los segmentos de rollback, actualmente en el "undo". Como las tablas del diccionario y los segmentos de rollback son parte de la instancia de base de datos, el espacio que ellos ocupan está sujeto a las mismas operaciones que otros esquemas de otros tablespaces
  - El manejo local utiliza bitmaps en el propio tablespace. Es el recomendado por Oracle
  - Un manejo ineficiente de las extensiones hace que se produzcan demoras en la creación y asignación de las extensiones

4.1.3- Tablespaces

Ejemplo:

Creamos el tablespace temporal llamado TEMPPRUEBA. Se guarda en donde se indica el tempfile, con un tamaño de 2 Mb, creciendo en 640 Kb. el fichero hasta 32 Mb como máximo. Oracle gestiona automáticamente las extensiones de 1024 en 1024 Kb.

```
CREATE SMALLFILE
    TEMPORARY TABLESPACE "TEMPPRUEBA" TEMPFILE
    'C:\path\TEMPPRUEBA1.ora' SIZE 2M REUSE
    AUTOEXTEND
    ON NEXT  640K MAXSIZE  32767M EXTENT MANAGEMENT
LOCAL UNIFORM
    SIZE 1024K
```

4.1.3- Tablespaces

Ejemplo:
Borramos el tablespace

```
SQL> DROP TABLESPACE TEMPPRUEBA INCLUDING CONTENTS
  and datafiles CASCADE CONSTRAINTS
```

- Los espacios temporales son usados para segmentos temporales, los cuales son creados, manejados y borrados por la Base de Datos.
- Estos segmentos temporales son comúnmente generados por instrucciones como *order by*, *group by* o *create index*

4.1.3- Tablespaces

- El tablespace **undo** se usa para los segmentos de undo, son parecidos a los segmentos de "rollback", recomendado por Oracle su no utilización.
- Aunque puede haber mas de un tablespace de undo por instancia, sólo uno está activo. Los segmentos de undo crecen o disminuyen de acuerdo a las necesidades de las transacciones.

- Se utiliza para:
  - Rollback de las transacciones.
  - Lectura consistente.
  - Operaciones de recuperación de la base de datos.
  - Funcionalidad de Flashback.

4.1.3- Tablespaces

- **Parámetros de almacenamiento**
  - DBA_TABLESPACES
  - USER_TABLESPACES: (visualizar con *select * from user_tablespaces;*)
- Métodos de determinar el **espacio libre y utilizado:** Gestionando las extensiones mediante:
  - El diccionario de datos: actualizar (update) la entrada correspondiente en el diccionario de datos cada vez que se asigna o libera una extensión a un TABLESPACE
  - El propio TABLESPACE (localmente) mantiene un mapa de bits en cada archivo de datos de los bloques o conjuntos de bloques liberados o usados del archivo de datos
  - Oracle actualiza automáticamente (update) el mapa de bits

4.1.4- Segmentos

- Conjunto de extensiones (conjunto de bloques de datos) dedicadas a un objeto de la BD, y almacenado en un fichero
- La cantidad de espacio que utiliza está determinada por sus parámetros de almacenamiento:
  - al crear el objeto (tabla, índice, cluster, Undo segment)
  - utiliza los parámetros de almacenamiento predeterminados del TABLESPACE en el que se almacena (se permite posterior modificación)
- Parámetros de almacenamiento del segmento:
  - INITIAL: tamaño inicial de la extensión
  - NEXT: tamaño de la siguiente extensión
  - PCTINCREASE: factor de incremento geométrico para sucesivas extensiones
  - MAXEXTENTS: número máximo de extensiones
  - MINEXTENTS: número mínimo de extensiones

4.1.4- Segmentos

- Oracle asigna a cada tabla una o más extensiones para formar el segmento de datos de una tabla

- **Tipos de segmentos**:
  - Segmentos de datos: conjunto de extensiones asignadas a una tabla.
    - No puede haber extensiones de otras tablas en un segmento dedicado a una tabla.
  - Segmentos de índices
  - Segmentos de undo: son uno o varios segmentos con información para deshacer transacciones
  - Segmentos temporales

4.1.4- Segmentos

- Cuando las extensiones de un segmento ya no pueden contener más datos el **segmento se amplía con nuevas extensiones**. Así sucesivamente hasta que:
  - no haya más espacio disponible en los datafiles de las tablespaces (no ampliables automáticamente)
  - hasta MAXEXTENTS por segmento (si definido).
  - no haya más espacio (cuota) para ese usuario en el TABLESPACE

- El **tamaño de la siguiente extensión Oracle** lo calcula con la siguiente fórmula:
  - NEXT*(1+(PCTINCREASE/100))
  - Ejemplo (next:4Mb y pctincrease:50%):
    - 1ª extensión: = INITIAL
    - 2ª extensión: = NEXT= 4Mb
    - 3ª extensión: = 4MB*(1+50/100) = 4Mb + 2Mb = 6Mb
    - 4ª extensión: = 9Mb
  - *NOTA: Cuidado con el tamaño del bloque*

4.1.4- Segmentos

- Tipos de segmentos: TABLE, INDEX, ROLLBACK, TEMPORARY, PARTITION, CLUSTER

- Al crear un segmento TABLE:
  - Al menos una extensión
  - Su espacio no se libera hasta que se elimina
  - Se puede utilizar "alter table" para modificar el *storage*. Y el comando "move" para cambiarlo de *tablespace*.
  - Cláusula **PCTFREE** para reservar, en el interior de cada bloque de cada extensión, un porcentaje de espacio para actualizaciones datos
    - Nulos (con valor NULL)
    - Otros valores que impliquen crecimiento del registro
    - *NOTA: Específico de cada aplicación*

4.1.4- Segmentos

Ejemplo:

```
CREATE TABLE ejemplo (
    cod NUMBER(2),
    nombre VARCHAR2(14) )
    STORAGE (INITIAL 100K NEXT 50K MINEXTENTS 1 MAXEXTENTS 50
             PCTINCREASE 5);

SELECT * FROM USER_SEGMENTS;
```

- **COMENTARIOS**
  - A partir de la 3ª extensión se incrementa un 5% el espacio de la extensión anterior:
    - 52,5 K → 52 si el tamaño de bloque fuera 2K

4.1.4- Segmentos

- Al crear un segmento INDEX:
  - Al menos una extensión
  - Su espacio no se libera hasta que se elimina. Pero estos se pueden eliminar automáticamente al eliminar las TABLAS o los CLUSTERS a los que indexan.
  - Se aconseja guardar los índices en diferente TABLESPACE que las tablas, para eliminar contiendas.
  - Se puede utilizar la opción **"REBUILD"** del comando "alter index" para modificar la configuración del *storage* y *tablespace* de un índice.

4.1.4- Segmentos

- Al crear un segmento ROLLBACK:
  - Sus extensiones mantendrán copias temporales de bloques de datos que cambiaron durante alguna transacción.
  - Al menos DOS extensiones.
  - Todas las extensiones del mismo tamaño.
  - Se podrá reducir el tamaño del segmento dinámica o manualmente hasta un tamaño específico (cláusula **optimal**)

Ejemplo:

```
CREATE ROLLBACK segment R1
TABLESPACE user40
STORAGE (INITIAL 2M NEXT 2M MINEXTENTS 2 MAXEXTENTS 249
  OPTIMAL 20M)
```

4.1.4- Segmentos

- Al crear un segmento TEMPORARY:
  - Almacenan datos temporales durante las operaciones (p.e.: consultas de gran tamaño, creación de índices, operaciones de **unión**, etc.)
  - Cada usuario tiene su tablespace temporal (*create user*). Habitualmente se crea un tablespace temporal para cada BD y todos sus usuarios.

```
Select PROPERTY_VALUE from DATABASE_PROPERTIES where
  PROPERTY_NAME = 'DEFAULT_TEMP_TABLESPACE'
```

  - Se pueden crear nuevos tablespaces temporales:

```
Create temporary tablespace

Alter tablespace AAAA temporary; alter tablespace AAAA
  permanent;
```

  - Ver el estado del tablespace

```
Select CONTENT from DBA_TABLESPACES where
  TABLESPACE_NAME="AAAA"
```


4.1.4- Segmentos

- Otras operaciones:
  - Desasignar espacio de los segmentos
    - Reducción tamaño de datafiles
    - Reducción tamaño tablas, clusters, índices (segmentos, etc.)
    - Reconstruir índices.

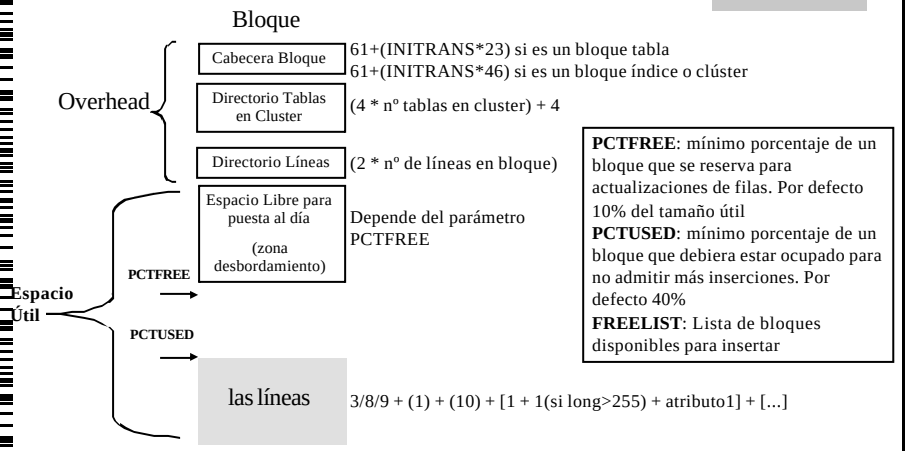
4.1.5- Extensiones

- Conjunto de bloques de datos contiguos de un TABLESPACE
- Cuando se elimina un segmento, sus extensiones se liberan
- Agrupación extensiones libres contiguas cuando *pctincrease*? 0
  - SMON (periódicamente o cuando lo necesita)
  - **`alter tablespace name coalesce;`**
- Asignación de extensiones libres:
  - La más adecuada (suficiente tamaño) cercana a los datos

4.1.6- Bloques

- Unidad de acceso a disco para una BD Oracle (unidad mínima de transferencia de información)
- Su tamaño se define al crear la BD.
- Debe ser múltiplo del tamaño de bloque del S.O. del servidor (entre 2Kb y 32 Kb).
- DB_BLOCK_SIZE: parámetro determina tamaño del bloque
- Parámetros de configuración *(create table / create index ...)*
  - [{PCTFREE *integer*
    | PCTUSED *integer*
    | INITRANS *integer*
    | MAXTRANS *integer*
    | cláusula_almacemanamiento
    }...]

4.1.6- Bloques



4.1.6- Bloques

Ejemplo: Dimensionamiento de la TABLA MUSEO

CAMPO	TIPO	LONG. MAX	TAMAÑO EFECTIVO	PROB. NO NULO
Codigo	NUMBER	4	4	100
Nombre	VARCHAR2	10	10	90
Direccion	VARCHAR2	60	30	10

Nº estimado TUPLAS: 3000
INITRANS = 1
MAXTRANS = 20
Tamaño Bloque = 4 Kb
PCTFREE=10
PCTUSED=85

Tamaño Cabecera: 61+23*1 = 84 bytes

Tamaño Útil = Tamaño bloque – Tamaño Cabecera = 4096-84 = 4014

EspacioPCTFREE = T.Útil*PCTFREE = 4014*10%=401 bytes

EspacioPCTUSED = T.Útil*PCTUSED = 4014*85%= 3412 bytes

EspacioFilaTabla = 3 (Cabecera) + 1(atributo) + 1(atributo largo) + **LogMediaFila =** 3+3+0+18 = 24

LongMediaFila = SUMA(longMediaColumn) = 2 * 1(DirectorioLíneas para 1 línea) + probNoNulo(Atrib1)*TamañoEfectivo(Atrib1) + ... = 2 + 100%*4 + 90%*10 + 10%30 = 2 + 4 + 9 + 3 = 18

Nº Filas por Bloque = TRUNC(EspacioPCTUSED/EspacioFilaTabla) = TRUNC(3412/24) = 142

Nº Bloques = [Nº Estimado Tuplas / Nº Filas por Bloque] = 3000/142 = 22 bloques

TamañoTabla = Nº Bloques*T.ÚtilBloque = 22*4014 = 88308 bytes = 86,24 Kb

4.1.6- Bloques

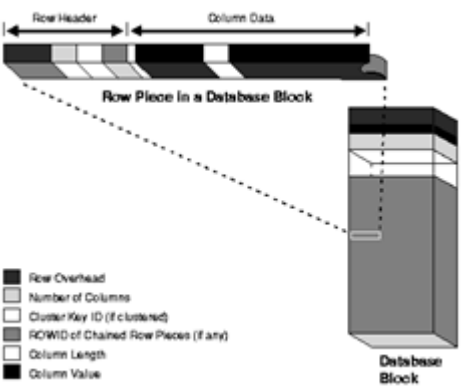
Ejemplo: Dimensionamiento de la TABLA MUSEO

- Determinación del nº de bloques necesarios utilizados para todas las filas de una tabla.
  - Consultado la columna ROWID
    - Selecciona el número de fila y número de bloque utilizado para esa fila
    - Los últimos tres caracteres de la cabecera del bloque indica el número de fila dentro del bloque (16-18)

```
SELECT COUNT(DISTINCT SUBSTR(ROWID,1,16))
FROM museo;
```


4.1.7- Almacenamiento de tablas en Oracle

- Cada fila se almacena en un bloque si tiene espacio suficiente y menos de 256 columnas
- De otra forma la información de la fila se encadena a través de varios bloques.
- Cada fila se compone:
  - Cabecera de fila >= 3 bytes
  - Columna de datos
    - Longitud de los datos
    - Datos
- Cada fila se identifica con ROWID



4.1.7- Almacenamiento de tablas en Oracle

Encadenamiento y reorganización

- Los datos de una fila de una tabla pueden ser demasiado grandes para almacenarlos en un único bloque de datos vacío.

- **Encadenamiento:**
  - Oracle almacena los datos de la fila en una cadena de uno o más bloques de datos (inserción ó modificación)

4.1.7- Almacenamiento de tablas en Oracle

Encadenamiento y reorganización

- **Reorganización**:
  - Si una sentencia UPDATE incrementa la cantidad de datos en una fila, de modo que la fila no se puede dejar en el bloque de datos entonces debe reorganizar:
    - Oracle intenta encontrar otro bloque con espacio libre suficiente para mantener la fila entera
    - Si el bloque está disponible, Oracle mueve la fila entera al nuevo bloque
    - Oracle guarda la parte de la fila original de la fila migrada apuntando al nuevo bloque que contiene la fila actual, el ROWID de la fila migrada no cambia. Los índices no son modificados, así apuntan a la localización original de la fila

4.1.7- Almacenamiento de tablas en Oracle

Almacenamiento básico

- Para aumentar el al rendimiento en el acceso a los datos se pueden utilizar los siguientes métodos:
  - Índices
  - Tablas organizadas por índices
  - Clusters (Agrupamientos) y Hash Clusters (Agrupamiento mediante dispersión)

4.1.8- Índices

Índices: Motivación

- Estructuras auxiliares para mejorar el tiempo de búsqueda
  - WHERE condición
    - Igualdad (una o varias filas)
    - Intervalo
    - Prefijo
- Reducir el tiempo de las operaciones de combinación de tablas
- Mejorar las consultas que requieren agrupación u ordenación
  - ORDER BY, GROUP BY, DISTINCT, ..
- Facilitar la implementación de restricciones
  - Integridad referencial
  - Unicidad

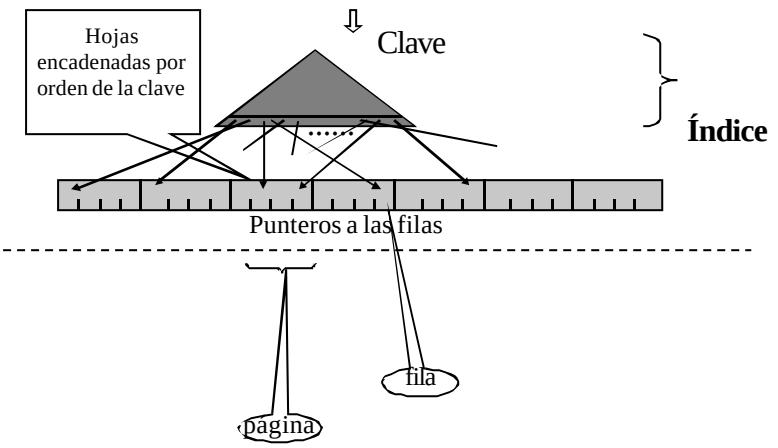
4.1.8- Índices

Índices: Tipos

- Árbol-B+
  - Es el más habitual
- Mapa de bits
- *Hashing*
- Índices de palabras (full-text indexing)
  - Ficheros invertidos
- Etc.

4.1.8- Índices

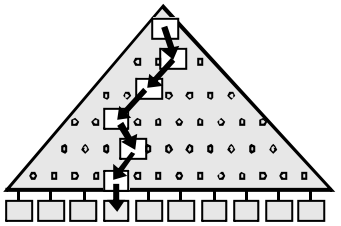
Índices multinivel: Árboles B +



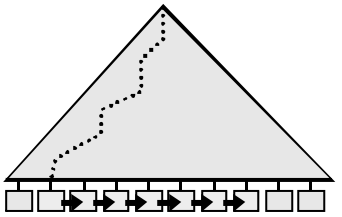
4.1.8- Índices

Índices en Árbol-B +

Búsqueda de un valor



Búsqueda de un rango



4.1.8- Índices

Índices en Árbol-B +

- Son flexibles
- Degeneración limitada
- Tiempo de acceso razonable
- No útiles para pocos valores

4.1.8- Índices

Índices en Árbol-B +

- Los SGBD no suelen hacer fusión
  - para ahorrarse trabajo... posiblemente inútil
  - pero liberan las páginas vacías
- Compresión de índices
  - Nudos intermedios: prefijo discriminador
  - Diferencial / Jerárquica
  - Coste procesador
  - Contención
- Filas no ordenadas físicamente
  - Puede ser más rápido un barrido que usar el índice
  - No confundir estructura con forma de acceso

4.1.8- Índices

Índices multidimensionales: Mapas de Bits

- AND, OR, contadores
- Para pocos valores (región, año, ..) y pocas actualizaciones
- DW/OLAP

iar	***ipe***	***ipr***	ventas	
B6	3	43	375	*1*
B6	3	21	100	*2*
B6	1	21	20	*3*
B7	1	21	100	*4*
.	.	.	.	.
.	.	.	.	*n*

		1234……n
ipr	21	0111…..
	43	1000…..
	:	:
iar	B6	1110….
	B7	0001….
	:	:
ipe	1	1100….
	3	0011….
	:	:

4.1.8- Índices

Índices Hashing

- No util para Orden, Comparaciones > <
- Pero útil para Unicidad. Existencia (IR), Join
- Algoritmo: Usuario, Interno
  - Declaración espacio asignado (degeneración)
  - No posible si el num.filas es imprevisible
- Factor de carga pequeño
- Uno o dos accesos menos que con árbol B+

4.1.8- Índices

Índices en Oracle

- CREATE [{ UNIQUE / BITMAP }] INDEX índice
- ON { tabla (columnasOexpresión) / CLUSTER nomcluster }
- TABLESPACE tablespace detalle almacenamiento
- COMPRESS
- NOSORT
- REVERSED

- Los NULL sólo se indexan si es un índice mapa de bit

4.1.8- Índices

Utilización de índices

- UNIQUE y no UNIQUE
- NULLs
- Coste en espacio
- Coste en tiempo de actualización
- No definir índice si:
  - El SGBD no lo va a utilizar
  - La clave es muy volátil o muy larga

4.1.8- Índices

Tablas organizadas como índices

- Contienen datos que son recuperados más rapidamente que si hubiesen sido almacenados en tablas normales.
- Es una tabla normal con un índice en una o más de sus columnas.
- Índice y tabla se almacenan juntos:
  - Tablas que se consultan por la clave primaria con pocas columnas

4.1.8- Índices

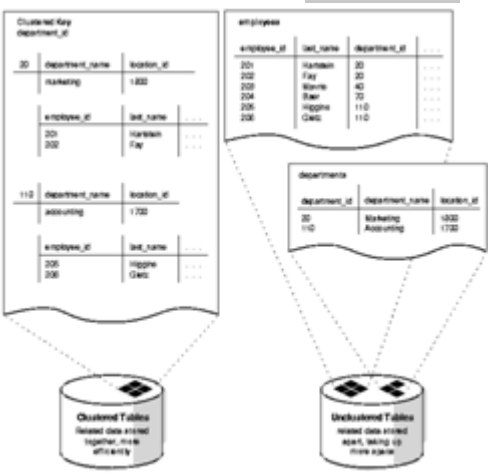
Tablas organizadas como índices

- Ventajas ☺
  - Acceso más rápido a las filas de las tablas ya que están en el mismo bloque
  - Acceso secuencial y por rangos por la clave primaria o un sufijo
  - Ahorro en espacio: la clave se guarda una sola vez
- Desventajas ☹
  - No usar con filas grandes

4.1.9- Clusters

Clusters: Motivación

- Grupos de una o más tablas
- Las filas se guardan físicamente juntas porque se usan generalmente juntas
- Comparten una o más columnas (clave del cluster)



4.1.9- Clusters

Clusters: Ventajas ☺

- En tablas con joins frecuentes o con relación maestro-detalle
  - Disminuye la E/S de disco
  - Mejora el tiempo de acceso
- La clave sólo se almacena una vez, ahorro de espacio

4.1.9- Clusters

Clusters: Problemas ☹

- No usar cuando
  - Valor clave se modifica a menudo
  - En tablas que se recorren completas a menudo
  - Si las filas que se agrupan juntas
    - Varian en número
    - Superan el tamaño de uno o dos bloques

4.1.9- Clusters

Hash Clusters en Oracle

- Similar a los clusters
- A la clave de cada cluster se le aplica una función de dispersión
- ☺ Bueno para consultas con condiciones de igualdad:
  - Un solo acceso
- ☹ Desventaja: Ocupa más espacio

4.1.10- Particiones

- El particionamiento permite descomponer tablas e índices en trozos más pequeños y manejables llamados particiones que son almacenados en segmentos separados
- Partición horizontal física
  - Repartir una tabla en espacios/ficheros/discos, …
  - Paralelismo, bloqueos (locks), copias y recuperación, …
- Partición por: rango, expresión, circular, al azar
- Particiones de índices
- Collocation / Alineación
- Inserciones/supresiones masivas
  - Rápido
  - Índices particionados

4.1.11- Ajuste de rendimiento

Tuning: Motivación

- Ajuste de la Base de Datos para un rendimiento óptimo
- Requiere una perspectiva global del Sistema de Información:
  - Procesos de la empresa
  - Software: SO y SGBD
    - Uso de memoria y uso del almacenamiento
    - Transacciones y procedimientos de recuperación
    - Arquitectura de comunicaciones
  - Hardware
- Cuando hay problemas o se prevén, o periódicamente ....
  - Mediciones/estadísticas
  - Elegir puntos a mejorar
  - Aplicar mejoras una a una e ir comprobando su efecto

4.1.11- Ajuste de rendimiento

Proceso

- **Problema**: el usuario considera que el tiempo de respuesta del sistema es alto
- **Proceso iterativo**
  1. Monitorización del sistema
  2. Diagnóstico
  3. Búsqueda de la solución
     - (Re)diseño apropiado de la aplicación
     - Optimización del código SQL
     - Ajuste de la memoria
     - Ajuste de la E/S
     - Ajuste de los problemas de contención u otros problemas

4.1.11- Ajuste de rendimiento
Principios de Shasha

1.- Estudio global, acción local

a) Identificación del problema
- Panorama global

b) Intervención minimalista

2.- Fragmentar para resolver atascos

a) Detectar el cuello de botella

b) Romperlo
- Mejora del componente causante
  - ej.: índices
- Repartir la carga
  - entre componentes
  - en el tiempo

4.1.11- Ajuste de rendimiento
Principios de Shasha

3.- Lo que cuesta es arrancar
- Minimizar los "arranques"
  - Tiempo de disco: 512b = ½ 64Kb
  - Tiempo de mensaje: 1b ~ 1Kb
  - Conexiones con SGBD: pocas y reaprovecharlas
  - ...

4.- Dar al servidor lo que es suyo
- ¿Dónde poner cada tarea?

5.- Llegar a compromisos entre costes
- accesos a disco vs. procesador
- espacio vs. tiempo
- OLTP vs. OLAP
- ...

4.1.11- Ajuste de rendimiento

Objetivo

- Afinación de los **procesos usuarios**: Uso "eficiente" de las órdenes SQL
  - Optimizador
  - Pruebas con "explicadores" (EXPLAIN) de plan
  - Retoques
  - etc.

  - Sólo los relevantes (periódicamente) y los problemáticos
  - Conexiones y tráfico
  - Restricciones declaradas
  - Disparadores y procedimientos

4.1.11- Ajuste de rendimiento

Objetivo

  - Regeneración
    - Espacio
    - Contigüidad
    - Estadísticas ANALYZE, RUN STATISTICS,...
  - Contención (esperas) / Bloqueos

- Afinación de la **BD**
  - Básico: los índices
  - Distribución entre discos y sistemas

4.1.11- Ajuste de rendimiento

Objetivo

- Afinación de la **plataforma**
  - Hardware
    - Memoria
    - Discos: Número de ejes, Caché en controlador, Niveles de RAID
    - SSD
    - CPU
  - SO, SGBD (parámetros)
- Tendencia a la automatización
  - Wizards/Advisors que aconsejan:
    - índices, particiones, vistas materializadas
  - Investigación en auto-tuning y auto-administración BD

4.1.11- Ajuste de rendimiento

Herramientas

- Para la prevención, identificación y resolución de problemas de rendimiento
- Tienden a la automatización de la ABD
- Herramientas que explotan...
  - Información sobre las operaciones y tráfico entre
    - SGBD, SO , Red
  - Estadísticas mantenidas por el SGBD, sobre..
    - estado de la BD (datos y objetos físicos)
      - num.filas, espacio, histogramas de valores de las columnas
      - grado de degeneración (datos e índice)
    - uso de la BD y los buffers, bloqueos, índices, ..

4.1.11- Ajuste de rendimiento

Herramientas

- Análisis de:

Contención (esperas), bloqueos, colas ..	Alertas (scripts para avisar de "anomalías")
Fragmentación y espacio perdido	Planes de ejecución SQL: ▪ "explicadores" y trazadores ▪ detalle de los pasos de las operaciones y sus tiempos
Encadenamientos y desplazamientos	Comunicación C/S
Uso del buffer-pool /caché	Consumo de recursos por las transacciones
Uso del LOG	Instantáneas y su comparación
Uso de I/O (discos..), RAM, CPU	Tendencias
Uso de los índices	

4.1.11- Ajuste de rendimiento

Herramientas de monitorización

1. Diccionario de Datos (DBA_)
2. Vistas dinámicas de Rendimiento (V$_)
3. Herramientas de monitorización del sistema:
   - Windows: Task Manager, etc...
   - Unix/Linux: top,df...
4. Oracle Enterprise Manager (OEM)
5. StatsPack (AWR/ASH en 10g), Index Tuning Wizard, Quest, Embarcadero...

4.1.11- Ajuste de rendimiento

Diccionario de datos

- Almacena metadatos acerca de
  - Usuarios
  - Objetos: Tablas, índices, paquetes, etc...
  - Roles y Privilegios
  - Almacenamiento (ficheros y tablespaces)
  - Auditoria e información operacional (backup, logs...)
- Información estática sobre la BD
- Sólo lectura (excepto SYS)
- Almacenado en System Tablespace

4.1.11- Ajuste de rendimiento

Vistas del diccionario de datos

- El acceso al diccionario de datos se hace a través de vistas que se identifican con los prefijos:
  - USER: objetos que posee el usuario
  - ALL: objetos sobre los que se tiene permisos
  - DBA: todos los objetos de la base de datos
- Diferentes tipos de vistas:
  - Vistas generales (DBMS y DB)
  - Vistas para monitorización de usuarios (DBA_USERS…)
  - Vistas de auditoría (DBA_AUDIT_OBJECT)
  - Vistas para monitorizar transacciones
  - Vistas para almacenamiento
  - Vistas sobre tipos de objetos: tablas, índices

4.1.11- Ajuste de rendimiento

Vistas generales

- **Dict** (dictionary): vista con los nombres de todas las tablas del diccionario de datos y sus descripciones.
- **GLOBAL_NAME** : nombre global de la BD
- **PRODUCT_COMPONENT_VERSION**: versión de los componentes principales
- **DBA_CATALOG**: tablas, vistas, sinónimos y secuencias en la BD
- **DBA_REGISTRY**: componentes instalados de Oracle
- **NLS_DATABASE_PARAMETERS, NLS_INSTANCE_PARAMETERS, NLS_SESSION_PARAMETERS**: parámetros de localización.
- **DBA_SOURCE**: código fuente de los objetos
- **DBA_JOBS**: monitoriza la definición de procesos en background (jobs)
- **DBA_OBJECT**: objetos de la BD como tablas, índices, paquetes, procedimientos, funciones, dimensiones, vistas materializadas, planes de recursos, tipos, secuencias, sinónimos, disparadores, vistas y particiones

4.1.11- Ajuste de rendimiento

Vistas relacionadas con el almacenamiento

- DBA_EXTENTS: información sobre las extensiones que utilizan los objetos
- DBA_FREE_SPACE: espacio libre en la BD
- DBA_SEGMENTS: información detallada sobre los segmentos: tipo, n bloques...
- DBA_DATA_FILES/DBA_TEMP_FILES
- DBA_TABLESPACES

4.1.11- Ajuste de rendimiento

Vistas relacionadas con los índices y restricciones

- DBA_CONSTRAINTS: restricciones de integridad y tablas sobre la que se aplican
- DBA_CONS_COLUMNS: columnas afectadas por las restricciones
- DBA_INDEXES: índices y sus características
- DBA_IND_COLUMNS: columnas que se han indizado
- INDEX_STATS: resultados del comando analyze, analiza el rendimiento y características de los índices.

4.1.11- Ajuste de rendimiento

Vistas dinámicas de rendimiento

- Se actualizan con la información que la instancia genera durante su ejecución.
- Su información no está disponible si la instancia no está levantada.
- Sólo tenemos acceso para lectura.
- Comienzan con el prefijo V$.

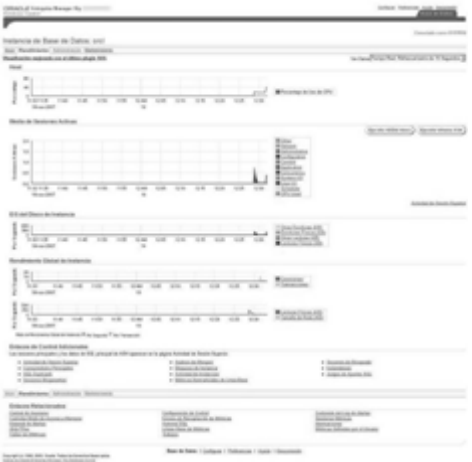
4.1.11- Ajuste de rendimiento

Vistas dinámicas

- Vistas actuales
  - Información actual sobre el estado del sistema
  - Ej: V$SESSION: sesiones abiertas en el sistema
- Vistas acumuladas
  - Cuenta el número de veces que un suceso ha ocurrido desde el inicio de la sesión
  - Se debe hacer la diferencia entre el inicio y el fin
  - Ej: V$FILESTAT: Operaciones de I/O
- Vistas de información
  - Información actual pero no tan dinámica: Estadísticas
  - V$SQL_PLAN: Plan de ejecución para cursores ejecutados recientemente

4.1.11- Ajuste de rendimiento

Oracle Enterprise Manager

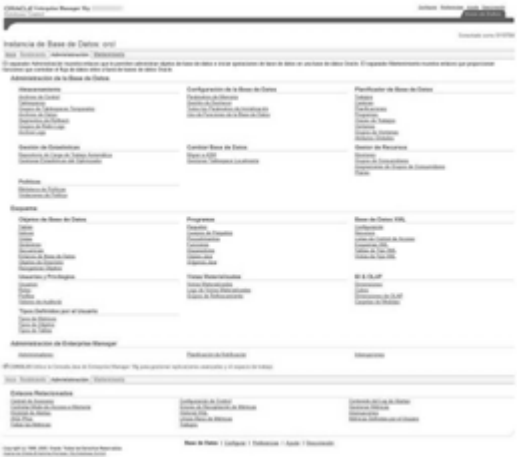


- Estadísticas de Rendimiento

4.1.11- Ajuste de rendimiento

Oracle Enterprise Manager

- Opciones de Administración



4.1.11- Ajuste de rendimiento

Oracle Enterprise Manager

- Opciones de Mantenimiento

4.1.11- Ajuste de rendimiento

Ajuste de la memoria de la instancia

- Acceder a los datos en memoria principal es mucho más rápido que acceder a los datos en disco
- Oracle almacena datos en memoria principal: SGA y PGA
  - Se puede consultar el tamaño de la SGA en V$SGA
- El tamaño correcto de la memoria depende :
  - Tipo de aplicación
  - Número de usuarios
  - Tamaño de las transacciones

4.1.11- Ajuste de rendimiento

SGA: Ajuste de Conjunto Compartido (Shared Pool)

- Afecta directamente al rendimiento de la BD:
  - Caché de biblioteca: código PL/SQL y sentencias SQL recientes
  - Caché de diccionario: Diccionario de datos
- Se modifica conjuntamente el tamaño del conjunto:
  - `alter system`
- Los efectos de un mal dimensionamiento son:
  - Fragmentación
  - Mayor consumo de CPU
  - Mayor intercambio entre memoria y disco

4.1.11- Ajuste de rendimiento

SGA: Caché de Biblioteca

- Contiene versiones analizadas y ejecutables del código SQL y PL/SQL
- Interpretación de SQL
  - Análisis (Parsing)
  - Optimización
  - Ejecución
  - Recuperación
- Análisis y optimización son operaciones costosas:
  - Si la sentencia esta en caché solo es necesario un análisis superficial.

4.1.11- Ajuste de rendimiento

Eficiencia de la Caché de Biblioteca

- Vista dinámicas
  - V$LIBRARYCACHE: estadísticas de acierto
  - V$LIBRARY_CACHE_MEMORY: espacio libre y objetos en caché
  - V$SHARED_POOL_ADVICE: informacón para el ajuste
- Algunos consejos para mejorar la eficiencia
  - Uso de variables
  - Compartir cursores: Cursor_sharing=similar
  - Código compartido

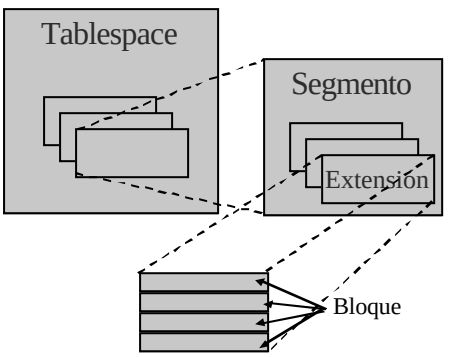
4.1.11- Ajuste de rendimiento

Diseño físico y Gestión del espacio

- Gestión eficiente del espacio en disco afecta al rendimiento de las aplicaciones
  - Asignación dinámica de espacio
  - Número de accesos a discos
  - Fragmentación
  - Encadenamiento de registros

4.1.11- Ajuste de rendimiento

Jerarquía de almacenamiento



- Los objetos de la BD pueden crecer y necesitar más espacio físico
- Cada estructura se guarda en un Tablespace, en su propio segmento
- Cuando crece se asignan nuevas extensiones (cto. de bloques)

4.1.11- Ajuste de rendimiento

Gestión del espacio

- En Oracle10g hay **dos alternativas**:
  - Tablespaces gestionados por el diccionario (DMT)
    - Para la asignación de espacio se consulta el diccionario
    - Los cambios se deben reflejar en los segmentos de rollback, etc
  - Tablespaces gestionados localmente (LMT)
    - La información de control se guarda en cabeceras en los ficheros de datos.
    - Mejora la eficiencia sobre todo en tablespaces locales
    - Permiten la gestión automática
    - Opción por defecto
  - Para convertir de DMT a LMT está el paquete DBMS_SPACE_ADMIN

4.1.11- Ajuste de rendimiento

Uso de Tablespaces

- Es conveniente reservar **diferentes tablespaces** para objetos que tienen características de almacenamiento y uso diferentes:
  - Objetos del diccionario de datos (SYSTEM)
  - Segmentos de rollback, actualmente undo
  - Segmentos temporales
  - Tablas
  - Índices
  - Objetos grandes
- Si es posible deberían estar en **diferentes discos** para permitir el uso en paralelo

4.1.11- Ajuste de rendimiento

Tamaño de las extensiones

- Es necesario **determinar el tamaño de un objeto**
  - Datos iniciales tengan espacio en el segmento
  - Espacio disponible para el crecimiento de los objetos
- Para cada objeto podemos determinar:
  - Tamaño inicial de la extensión
  - Tamaño de la siguiente extensión
    - Uniforme
    - Variable
    - Automático
  - Determinar la forma de manejo de la extensión
    - Manual/Automático

4.1.11- Ajuste de rendimiento

Extensiones grandes: pros y contras

- Pros ☺
  - Evitan las extensiones dinámicas
  - Pequeñas ventajas de rendimiento: búsqueda de tabla completa
  - Lecturas simples contra el mapa de extensiones
- Contras ☹
  - Encontrar espacio libre contiguo
  - Desperdicio de espacio inicialmente

4.1.11- Ajuste de rendimiento

Tamaño del bloque: pequeño

- Generalmente para BDs transaccionales
  - Ventajas: ☺
    - Acceso aleatorio eficiente
    - Bueno para filas pequeñas
    - Reducen la contención de bloques
  - Desventajas ☹
    - Excesiva sobrecarga de información de control
    - Pocas filas por bloque

4.1.11- Ajuste de rendimiento

Tamaño del bloque: grande

- Generalmente para Almacenes de Datos
  - Ventajas ☺
    - Menor sobrecarga
    - Acceso secuencial
    - Filas grandes
    - Lectura secuencial de índices
  - Desventajas ☹
    - Incrementan la contención en los bloques hoja de los índices
    - Desaprovecha espacio en buffer caché

4.1.11- Ajuste de rendimiento

Gestión automática del espacio

- Al crear objetos de la BD en tablespaces gestionados localmente con su propio segmento:
  - La opción AUTOALLOCATE especifica que las nuevas extensiones se crean dinámicamente. La BD decide el tamaño de la extensión
  - La opción AUTO gestiona el espacio libre dentro de los bloques

4.1.11- Ajuste de rendimiento

Estimación del espacio en bloques

- Espacio libre distribuido:
  - PCTFREE: mínimo porcentaje de un bloque de datos que se reserva para actualizaciones de filas. Por defecto 10
  - PCTUSED: mínimo porcentaje de un bloque que debiera estar ocupado para no admitir más inserciones. Por defecto 40
  - FREELIST: Lista de bloques disponibles para insertar

4.1.11- Ajuste de rendimiento

Problemas en bloques

- Encadenamiento: Filas que no caben en un único bloque
- Migración: Tras una operación de actualización la fila no tiene espacio en el bloque
  - La fila se mueve a otro bloque
  - La parte de la fila original se mantiene y apunta a la nueva localización para evitar la modificación de los índices

4.1.11- Ajuste de rendimiento

Problemas en bloques

- Borrados frecuentes provocan tablas ”con huecos”
  - En las operaciones de lectura se leen todos los bloques hasta una marca máxima
- Índices sobre tablas volátiles
  - Los bloques vacíos van a FREELIST
  - Es necesario mantener los bloques aunque sólo tengan una entrada

4.1.11- Ajuste de rendimiento

Estadísticas de almacenamiento

- Es posible obtener datos sobre el estado de las tablas e índices para monitorizar y solucionar los problemas de almacenamiento
  - Comando ANALYZE
  - Paquete DBMS_SPACE

Bibliografía

- Database systems : a practical approach to design, implementation and management. Thomas M. Connolly. Addison-Wesley. 2005.
- Sistemas de bases de datos : un enfoque práctico para diseño, implementación y gestión. Thomas M. Connolly. Addison-Wesley. 2005.
- Desarrollo de Bases de Datos: casos prácticos desde el análisis a la implementación. Dolores Cuadra, Elena Castro, Ana M. Iglesias, Paloma Martínez, Fco. Javier Calle, César de Pablo, Harith Al-Jumaily y Lourdes Moreno. Editorial Ra-Ma. 2007.
- Database System Implementation. Héctor García-Molina, jeffrey D. Ullman and Jeniffer Widom. Prentice Hall. 2000.
- Physical database design : the database professional's guide to exploiting indexes, views, storage, and more. Sam Lightstone. Morgan Kaufmann/Elsevier. 2007.
- Database administration : the complete guide to practices and procedures. Craig Mullins. Addison-Wesley. 2002.
- Oracle 10g : administración y análisis de bases de datos. César Pérez. Editorial Ra-Ma. 2005.
- Database management systems. Raghu Ramakrishnan. McGraw Hill. 2003.
- Fundamentos de Diseño de Bases de Datos. Abrahan Silberschatz and Henry Korth and S. Sudarshan. McGraw Hill. 2007.