Administración de Bases de Datos
Tema 3: Diseño físico y ajuste de BBDD

Dpto. de Ingeniería Informática

Contenidos

- Introducción
- Diseño físico
  - Objetivos del diseño
  - Comparación entre indexación y dispersión
  - Técnicas de diseño
  - Proceso de diseño de BD
  - Optimización de consultas
- Oracle
  - Arquitectura
  - Estructura lógica y física de la BD
  - Creación de objetos
  - Diccionario de Datos
- Conclusiones
- Referencias

Introducción

- Diseño físico: especificación de estructuras de almacenamiento y caminos de accesos específicos para que las aplicaciones que acceden a la BD tengan un buen rendimiento
- Cada SGBD concreto ofrece varias opciones:
  - Diferentes tipos de índices
  - Agrupamientos de registros, de distinto tipo, relacionados en los mismo bloques de disco (*cluster*)
  - Distintos tipos de técnicas de dispersión (*hashing*)
  - Ajuste de los parámetros físicos de almacenamiento (bloque, *buffer*, ...)

Objetivos del diseño físico

- Criterios para elegir opciones de diseño físico:
  - *Tiempo de respuesta*: es el tiempo entre la introducción de la transacción y la respuesta obtenida. Debe minimizarse
    - Para dar mejor respuesta al usuario que la ejecuta … y para dejar recursos al resto de usuarios
  - *Productividad*: nº posible de transacciones por minuto. Debe maximizarse
  - *Aprovechamiento del espacio*: espacio ocupado por los ficheros de la BD y de sus estructuras de acceso. Debe minimizarse el desperdicio
- El diseño físico es dependiente del SGBD concreto

Comparación entre indexación y dispersión

- La elección entre indexación y dispersión (*hashing*) se debe basar en:
  - Coste de la reorganización periódica del índice o de la estructura de dispersión
  - Frecuencia relativa de inserciones y eliminaciones
  - Tiempos de acceso a los datos
  - Tipos de consultas más frecuentes
    - *SELECT a, b, c...*
    - *FROM t*
    - Si es *WHERE a = c;* → dispersión
    - Si es *WHERE a <m AND a >n;* → indexación

Técnicas de diseño

- Existen tres estrategias de los fabricantes para el diseño físico:
  - El SGBD impone una estructura interna, el DBA no decide
    - Aumenta la independencia física/lógica, pero disminuye la eficacia
  - El DBA diseña la estructura, luego es una tarea extra
    - La independencia puede no conseguirse pero puede mejorar la eficiencia.
  - El SGBD proporciona una estructura a partir de los parámetros que le proporcione el DBA y luego se pueden ajustar y optimizar mejorando el rendimiento de la BD
- La última estrategia es la que suelen ofrecer la mayoría de los SGBD

Técnicas de diseño

- Esta última estrategia tiene las siguientes ventajas:
  - La BD puede funcionar de inmediato
  - Se afina la eficiencia modificando los ajustes
  - Se mantienen la independencia físico/lógica
- Las estructuras más convenientes para una de terminada BD están determinada por los tipos de consultas más frecuentes
  - Debe ayudar a la localización de la tupla solicitada y a reducir los accesos a disco
- Existen diversos elementos pero no todos los SGBD disponen de ellos, o bien, no se le da la opción al DBA para que pueda ajustarlos según las necesidades

Técnicas de diseño

- Elementos que debemos tener en cuenta a la hora del diseño:
  - Registros físicos
  - Bloques
  - Punteros
  - Gestión del espacio de disco
  - Gestión de la memoria intermedia
  - Índices
  - Funciones de dispersión
  - Árboles
  - Caché de consultas
  - Agrupamientos de tablas
  - Técnicas de compresión
  - Redundancia de datos

Técnicas de diseño

- Muchos SBGD permiten almacenar el resultado de una consulta concreta en una caché
  - En MySQL se mira si es exactamente la misma consulta: “select nombre” ≠ “SELECT nombre”
  - El resultado está precalculado en memoria
  - ¿Ventajas?
  - ¿Desventajas?

Técnicas de diseño

- Muchos SBGD permiten almacenar el resultado de una consulta concreta en una caché
  - En MySQL se mira si es exactamente la misma consulta: “select nombre” ≠ “SELECT nombre”
  - El resultado está precalculado en memoria
  - Muuuy rápido
  - Un cambio en una tabla implica recalcular todas las consultas cacheadas que las usen
    - No funciona con RANDOM, SYSDATE, …
  - https://dev.mysql.com/doc/refman/5.7/en/query-cache-operation.html
  - http://www.docplanet.org/mysql/mysql-query-cache-in-depth/

Técnicas de diseño

- Agrupamientos de tablas:
  - Se pueden agrupar físicamente dos o más tablas que comparten un grupo de atributos comunes, llamado clave de agrupamiento
  - Aunque se unen físicamente siguen siendo tablas independientes desde el punto de vista lógico
  - El agrupamiento es transparente para el usuario
  - ¿Ventajas?
  - ¿Desventajas?

Técnicas de diseño

- Agrupamientos de tablas:
  - Se pueden agrupar físicamente dos o más tablas que comparten un grupo de atributos comunes, llamado clave de agrupamiento
  - Aunque se unen físicamente siguen siendo tablas independientes desde el punto de vista lógico
  - El agrupamiento es transparente para el usuario
  - Reduce espacio de disco y facilita las consultas que conlleven producto natural entre ellas
  - Enlentece las consultas que se realicen sólo a una de las tablas

Técnicas de diseño

- Técnicas de compresión:
  - Consiste en tener los datos comprimidos
  - Disminuye el espacio de disco (y las transferencias disco-mem), pero aumenta el proceso: requiere descompresión de datos
  - Transparente para usuario
  - Especialmente interesante para discos más pequeños (SSD) o Arduino
    - https://dev.mysql.com/doc/refman/5.7/en/innodb-compression.html

Técnicas de diseño

- Redundancia de datos:
  - Duplicidad de datos para disminuir los tiempos de respuesta. Se consigue desnormalizando
  - Necesita más espacio de disco a cambio de rapidez de acceso a los datos y una actualización más complicada

Técnicas de diseño

- Ejemplo desnormalización
- Sea una tabla

Técnicas de diseño

- Los recursos disponibles han aumentado facilitando la eficiencia de los SGBD. El DBA debe conocer las características de los productos para elegir los más adecuados

Proceso de diseño de una BD

- Queremos diseñar la estructura lógica y física de una o más BDs para atender las necesidades de información de los usuarios mediante un conjunto definido de aplicaciones
- Objetivos del diseño:
  - Satisfacer los requisitos de contenido de información de los usuarios y aplicaciones especificados
  - Proporcionar una estructura de la información natural y fácil de entender (mantenible)
  - Soportar los requisitos de procesamiento y los objetivos de rendimiento:
    - Tiempo de respuesta
    - Tiempo de procesamiento
    - Espacio de almacenamiento

Proceso de diseño de una BD

- Fases principales del diseño de BD:
  - Obtención y análisis de requisitos:
    - Identificación de las áreas de aplicación y grupos de usuarios
    - Estudio y análisis de toda la documentación relativa a las aplicaciones
    - Estudio del entorno de operación y planes de utilización de la información
    - Cuestionario de preguntas a usuarios, entrevistas, etc
  - El diseño conceptual de la BD implica dos fases en paralelo:
    - Diseño del esquema conceptual
    - Diseño de transacciones
      - Transacciones de recuperación
      - Transacciones de actualización
      - Transacciones mixtas

Proceso de diseño de una BD

- Fases principales del diseño de BD:
  - Todo esto se puede complicar con muchos aspectos:
    - Bases de datos ya creadas con las que tenemos que interoperar
      - Pueden estar llenas de datos
      - Y sin normalizar
      - O sin documentar
    - Dependencia de operaciones de mantenimiento que se encuentran en programas externos

Proceso de diseño de una BD

- Elección de un SGBD:
  - Costes (TOC):
    - Adquisición de software
      - Y mantenimiento
    - Adquisición de hardware
      - Y mantenimiento
    - Creación y conversión de la BD anterior
    - Personal que disponemos (o que podemos contratar)
    - Formación necesaria para el personal
    - Otros costes de operación
  - Cambios: de SGF a SGBD
    - Complejidad de los datos
    - Compartición entre aplicaciones
    - Evolución o crecimiento dinámico de los datos
    - Frecuencia de solicitudes de datos para un caso en concreto
    - Volumen de datos y necesidad de control

Proceso de diseño de una BD

- Transformación al modelo de datos (diseño lógico). Puede realizarse en dos etapas:
  - Transformación independiente del sistema
  - Adaptación de los esquemas a un SGBD específico
- Diseño físico de la BD:
  - Tiempo de respuesta
  - Aprovechamiento del espacio
  - Productividad de las transacciones
- Implementación y ajuste del sistema de BD

Pautas para el diseño de BDR

- Factores que influyen en el diseño físico:
  - Análisis de consultas y transacciones de la BD. Para cada consulta debemos especificar:
    - Atributos a los que se aplica los criterios de selección (*)
    - Atributos implicados en productos de reunión y acceso a múltiples tablas (*)
    - Atributos que ofrece la consulta como respuesta
    - Ficheros a los que tendrá acceso la consulta

    Los atributos (*) son candidatos para la definición de estructuras de acceso (índices/hash)

Pautas para el diseño de BDR

- Factores que influyen en el diseño físico:
  - Análisis de consultas y transacciones de la BD: para cada transacción/operación de actualización debemos especificar:
    - Atributos sobre los que se aplica condiciones de selección para operaciones de eliminación o de modificación (*)
    - Atributos cuyos valores pueden alterar las operaciones (+)
    - Ficheros que se acceden
    - Tipo de operación (acceso/actualización) en cada fichero

    Los atributos (*) son candidatos para estructuras de acceso

    Los atributos (+) son candidatos para evitar estructuras de acceso: su modificación requiere la actualización de dichas estructuras

Pautas para el diseño de BDR

- Ejemplo: sea la consulta X
- ¿Qué campos sería adecuados para indexar?
- Si se hacen X consultas al día (distribuidas tal que … ¿merece la pena indexar?

Pautas para el diseño de BDR

- Factores que influyen en el diseño físico:
  - Análisis de la frecuencia esperada de invocación de consultas y transacciones
    - Qué va a pasar muchas/pocas veces
  - Análisis de las restricciones de tiempo sobre las consultas y transacciones
    - Qué urge y qué no. Algo puede usarse poco pero requerir mínimo tiempo de respuesta
  - Análisis de la frecuencia esperada de las operaciones de actualización
    - Número de recálculos de cachés / estructuras de búsqueda
  - Análisis de las restricciones de unicidad sobre los atributos
    - Facilitan estructuras de búsquedas

Pautas para el diseño de BDR

- Decisiones de diseño de una BD
  - Decisiones de diseño sobre índices
    - Cuándo realizar indexación de atributo(s)
    - Qué atributo o atributos indexar
    - Cuándo establecer un índice de agrupación
  - Cuándo utilizar un hash sobre un índice de árbol
  - Desnormalización para acelerar las consultas
    - Implica y establecer protocolos para actualizar la información
    - Puede producir tiempo de no-servicio mientras se hace
    - Y por favor, documentar el protocolo

Ajustes de BDR

- Supervisar y revisar periódicamente el diseño físico
  - Ejecución más rápida de aplicaciones
  - Disminución tiempo de respuesta consultas/transacciones
  - Mejorar rendimiento total de las transacciones
- Información recogida por los SGBD:
  - Tamaño de tablas
    - Y su evolución en el tiempo
    - ¿Hay límites? De dominio, de hardware, de crecimiento, ...
  - Nº de valores distintos en una columna
  - Nº de veces que una determinada consulta o transacción es ejecutada en un intervalo de tiempo
  - Tiempos requeridos para las diferentes fases del procesamiento de transacciones y consultas

Ajustes de BDR

- El SGBD puede recoger también las estadísticas sobre:
  - Almacenamiento
    - Uso de disco, etc
  - Rendimiento de E/S y dispositivos
    - Junto con el SSOO
  - Procesamiento de consultas/transacciones
  - Bloqueos y registros en el diario
  - Índices

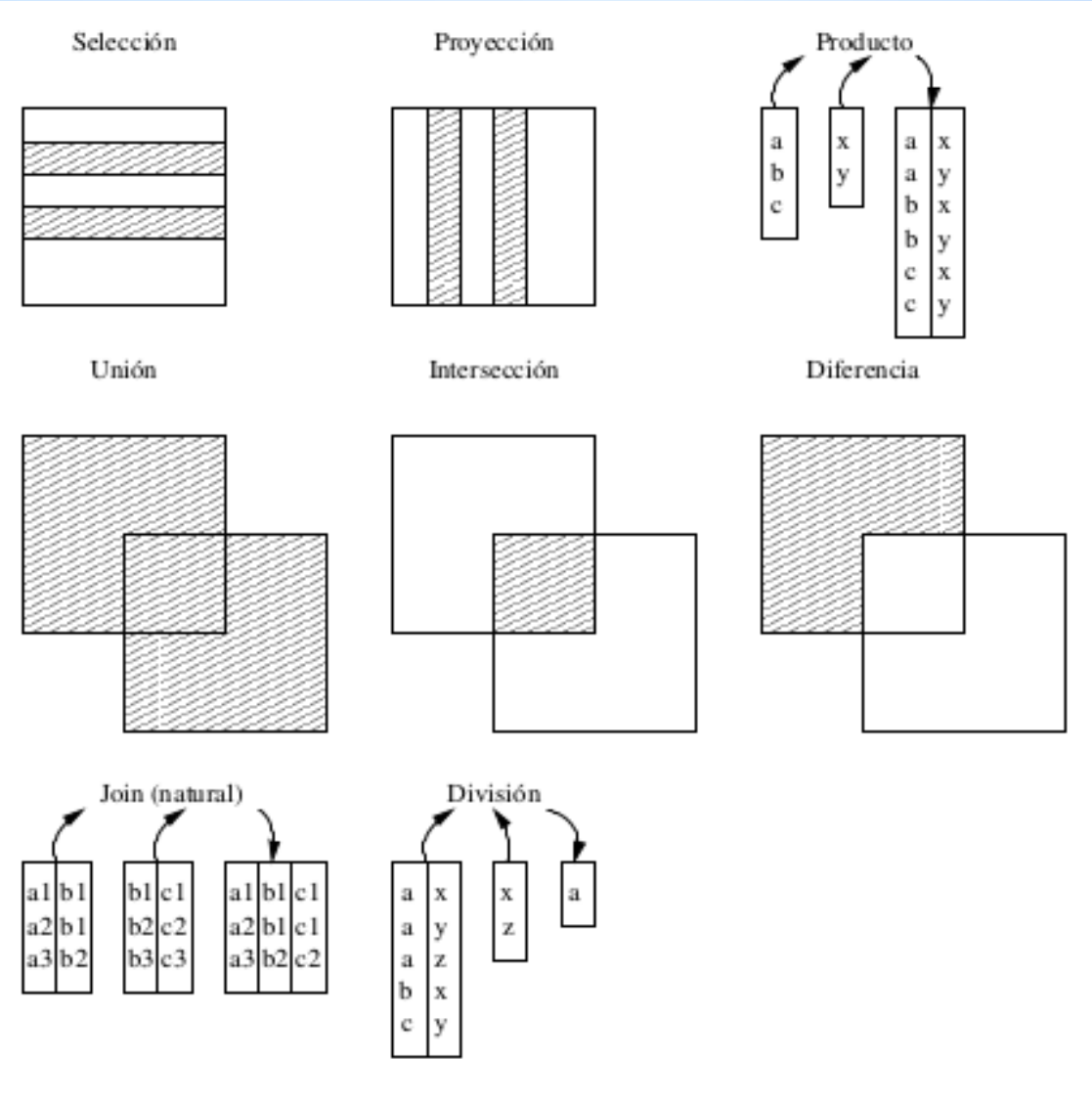
Ajustes de BDR

- Ajuste de índices:
  - Consultas que puedan mejorar su tiempo de ejecución por medio de un índice
  - Eliminar índices que se utilicen poco
    - Se dedica tiempo (y disco) a mantenerlos para poca mejora
  - Índices que sufren muchas actualizaciones por estar definidos sobre atributos que (se detecta que) cambian mucho
- Ajuste del diseño:
  - Desnormalización
  - Control de la redundancia
  - Fragmentación vertical (proyección)
  - Fragmentación horizontal (selección)

Optimización de consultas

- Álgebra relacional: nos permite indicar las operaciones y el orden en que las queremos realizar
  - Deseamos que el sistema invierta los menos recursos posibles
- Un mismo resultado puede obtenerse con varias sentencia → deseamos la más óptima (disminución del tiempo de respuesta)
- Los sistema relacionales tienen un optimizador de consultas que mejora el rendimiento.
- El coste total de una consulta es función de:
  - Tiempo de respuesta
  - Costes de comunicación
  - Acceso a almacenamiento secundario
  - Almacenamiento
  - Hardware de procesamiento (CPU, ...)

Optimización de consultas



Optimización de consultas

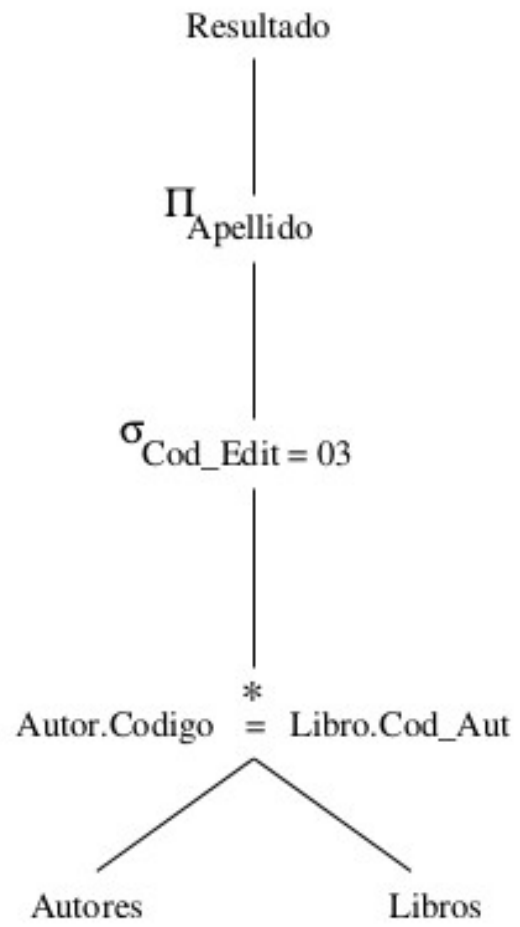
- Criterios prácticos para la formulación de accesos:
  - Realizar las selecciones tan pronto como sea posible
  - Combinar ciertas selecciones con un producto cartesiano anterior, para realizar los productos de unión
  - Combinar secuencias de operadores unarios, como selección y proyección
  - Detectar la posible existencia de subexpresiones comunes en una expresión
    - Si el resultado de una subexpresión no es muy grande, y puede leer de memoria secundaria en mucho menos tiempo de lo que lleva el calcularla, es mejor calcularla previamente y leer el resultado cuando se necesite

Optimización de consultas

- Ejemplo: la Biblioteca de la UCA
  - Deseamos obtener el apellido de los autores que han publicado libros con la editorial de código 3
    - ¿Cómo sería en SQL?
    - ¿Y en álgebra relacional?
  - La consulta se puede hacer de varias formas

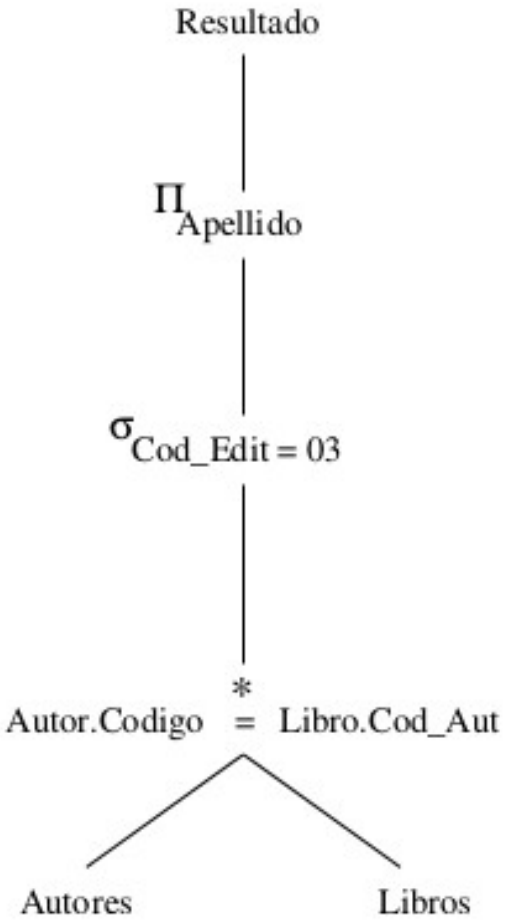
Optimización de consultas

$$\Pi_{Apellido}(\sigma_{Cod\_Edit=03}(Autores \bowtie Libros))$$

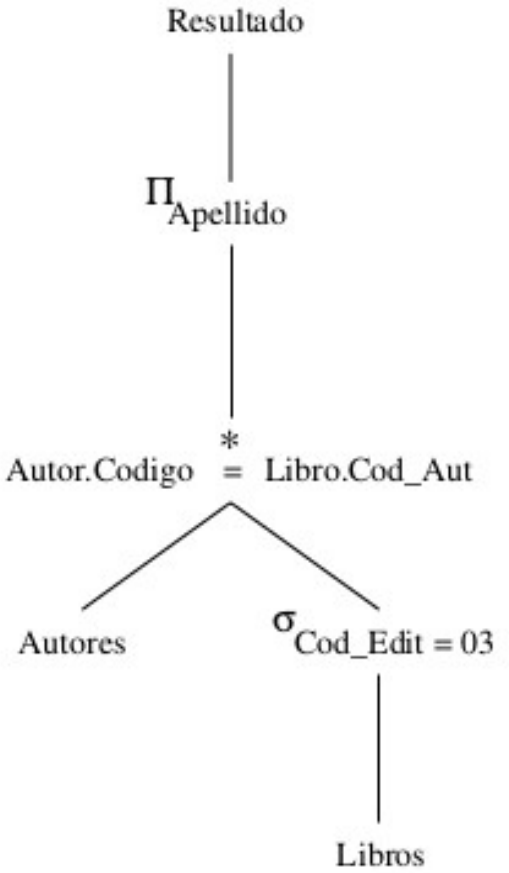


Optimización de consultas

$\Pi_{Apellido}(\sigma_{Cod\_Edit=03}(Autores \bowtie Libros))$

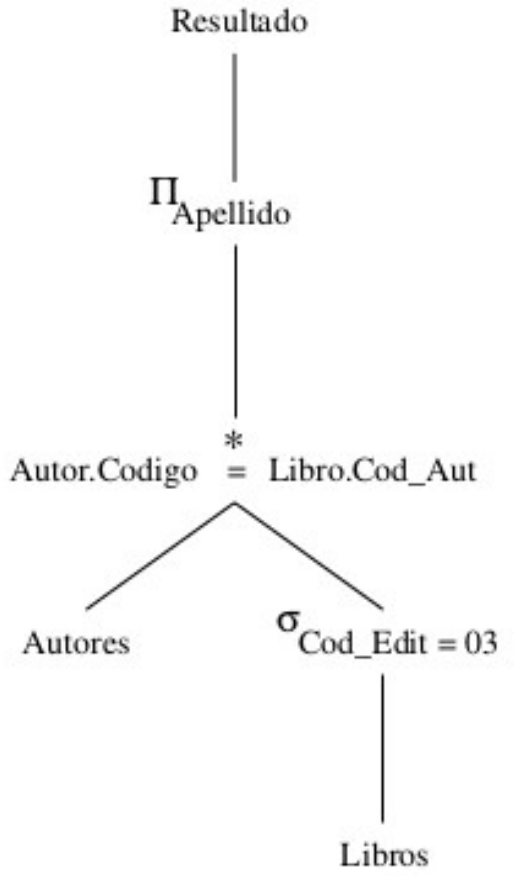


$\Pi_{Apellido}(Autores \bowtie (\sigma_{Cod\_Edit=03}(Libros)))$

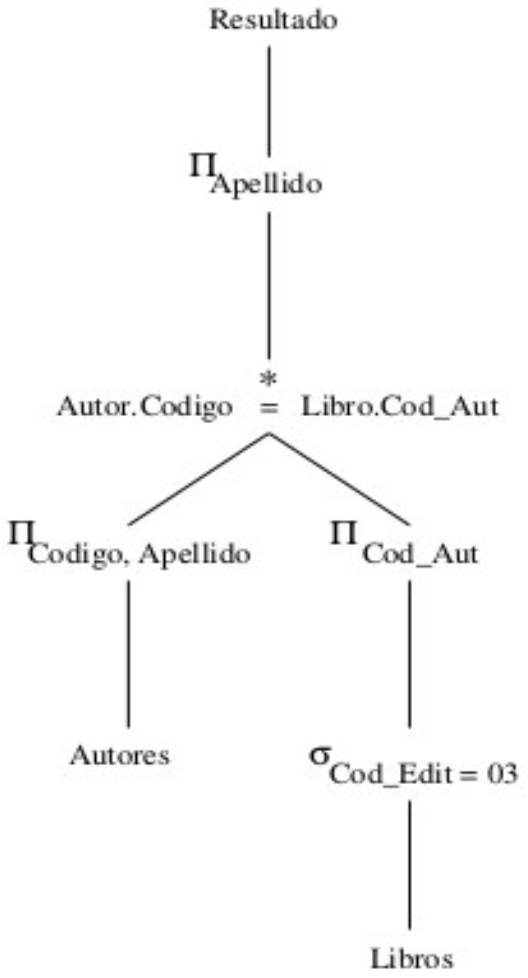


Optimización de consultas

$$\Pi_{Apellido}(Autores \bowtie (\sigma_{Cod\_Edit=03}(Libros)))$$



$$\Pi_{Apellido}(\Pi_{Codigo,Apellido}(Autores) \bowtie (\Pi_{Cod_Aut}(\sigma_{Cod\_Edit=03}(Libros))))$$



Optimización de consultas

- En cada caso de las tres consultas anteriores supongamos que tenemos
  - 30.000 libros, cada uno en un registro de 500KB
  - 40.000 autores, cada uno en registro de 100KB
  - Si cada libro tiene de media 2 autores
  - Si hay 100 libros de la editorial 3
  - ¿Qué necesidades de espacio tendría cada una de las consultas anteriores?
    - selectividad de producto (*join selectivity*): ratio entre la cardinalidad de un producto natural y su máximo posible

Arquitectura

- En Oracle existen dos conceptos básicos para entender su arquitectura:
  - *Base de datos Oracle*: un conjunto de datos, almacenados, físicamente en ficheros y lógicamente en *tablespace*.
  - *Instance*: un conjunto de estructuras de memoria y de procesos en segundo plano que acceden a un conjunto de ficheros. Los parámetros que determinan la composición y tamaño de una instance se almacenan en el fichero init$ORACLE_SID.ora
  - La variable de entorno $ORACLE_SID identifica a la base de datos que queremos acceder

$ORACLE_SID = nombre_id_instancia

db_name = nombre_BD

Arquitectura

- La estructura de una BD Oracle se puede dividir en tres categorías:
  - Estructuras externas a la BD
  - Estructuras internas de áreas de memoria
  - Estructuras internas de la BD

  Vemos cada una por encima

Arquitectura

- Estructuras externas a la BD
  - Ficheros de recuperación o *redo logs*
  - Ficheros de control
- Estructuras internas de áreas de memoria
  - *System Global Area (SGA)* y *Program Global Area (PGA)*
  - *Data Block Buffers*
  - *Dictionary Caché*
  - *Redo Log Buffer*
  - *Shared SQL Pool*
  - *Context Areas*

Arquitectura

- Estructuras internas de la BD:
  - Tablas, columnas, tipos de datos y restricciones
  - Usuarios y esquemas
  - Índices, agrupamientos y agrupamientos hash
  - Vistas
  - Secuencias
  - Procedimientos, funciones, paquetes y disparadores
  - Sinónimos
  - Privilegios y roles
  - Enlaces a DB (Database Links)
  - Segmentos, extensiones y bloques
  - Segmentos de recuperación (Rollback Segments)

Arquitectura

- Utilización de índices en Oracle:
  - Oracle dispone de un módulo de optimización de consultas
  - La presencia o ausencia de índices no requiere cambios en la ejecución de cualquier orden de SQL
  - Los índices son lógica y físicamente independientes de los datos de las tablas asociadas, y requieren espacio de disco
  - Una vez creados, son mantenidos automáticamente por Oracle
  - Se implementan mediante árboles B+
  - No incluye entradas en el índice para tuplas con valores *null* en la columna de indexación

Arquitectura

- El optimizador no usa el índice construido sobre una columna si:
  - La consulta no incluye una cláusula *where*
  - La consulta usa la columna indexada aplicándole una función (substr, ...) u operador (|| , ...)
- El optimizador sí usa el índice cuando:
  - La consulta contiene una cláusula *order by* por la columna indexada
  - La consulta aplica las funciones *max* o *min* sobre una única columna y ésta es la indexada
- Los índices de agrupamiento son densos

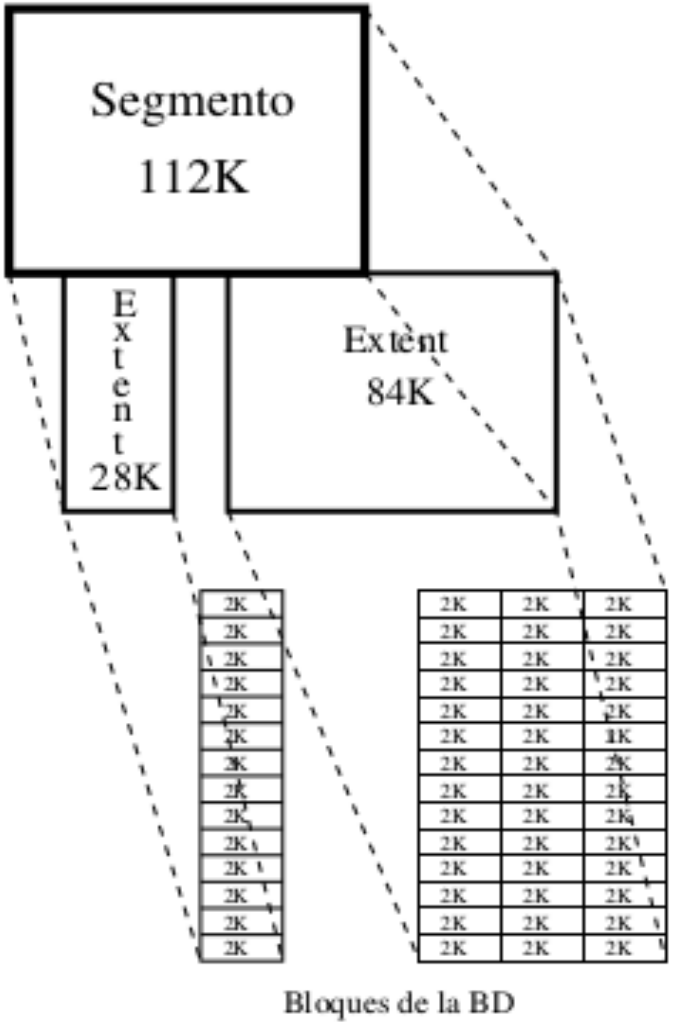
Estructura lógica de la BD

- *Tablespace*: división lógica de una BD
- *Objetos del esquema*: estructura lógica de almacenamiento de datos. Los objetos no se relacionan directamente con un fichero almacenado en disco; se almacenan lógicamente en tablespace de una BD
- *Esquema*: colección de objetos del esquema. Cada usuario tiene asociado un esquema
- Un *tablespace* puede contener objetos de diferentes esquemas, y los objetos de un esquema pueden estar en diferentes table-space
- Los tablespace permiten realizar un diseño lógico óptimo de la BD al separar los objetos según su tipo y actividad

Estructura lógica de la BD

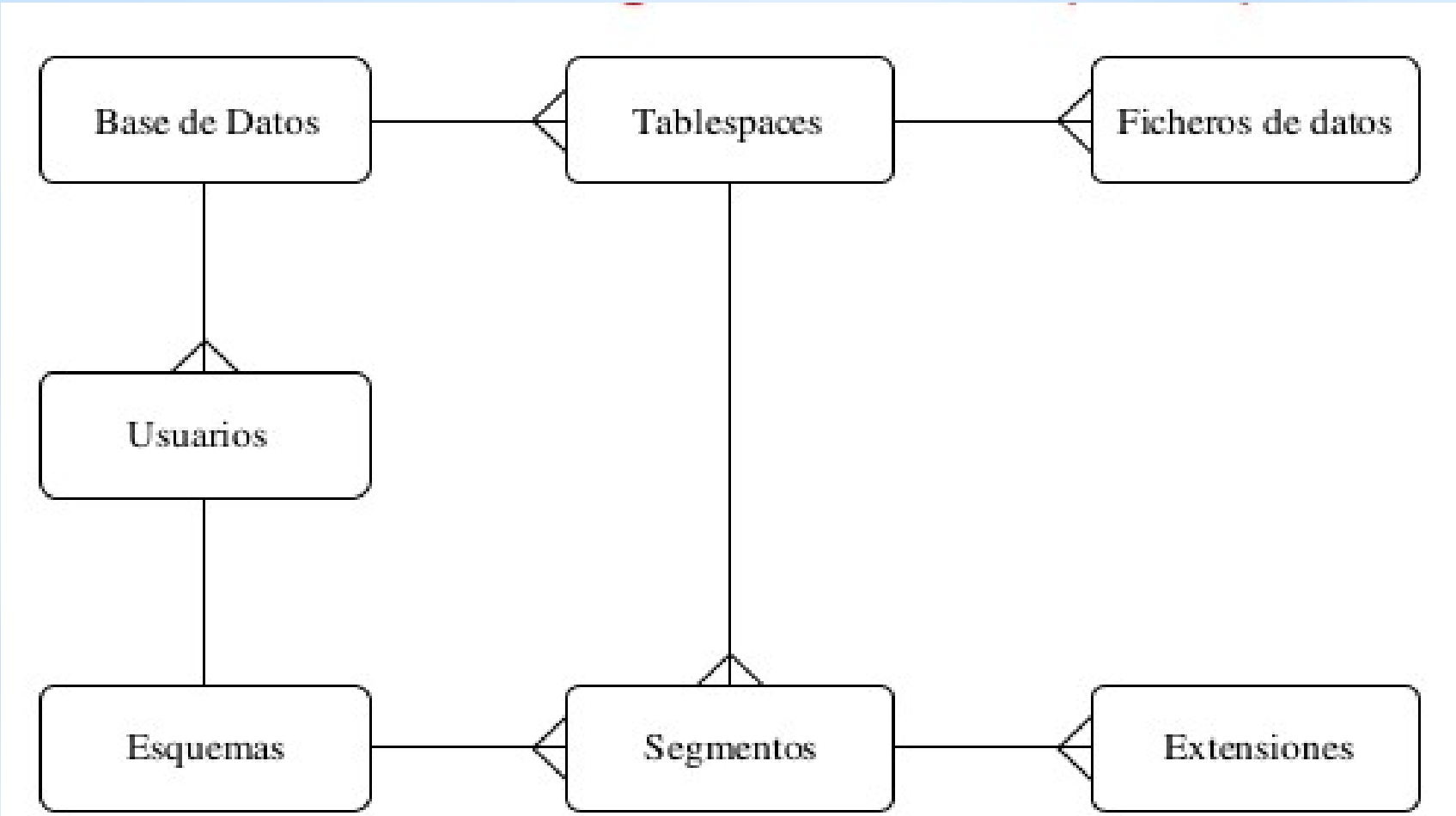
- *Bloques*: unidad más pequeña para el manejo del espacio de disco y de E/S. Se corresponde con un bloque de bytes físicos de disco, o con un múltiplo del bloque del SO
- *Extensiones*: nº de bloques contiguos en disco que almacena el mismo tipo de información
- *Segmentos*: conjunto de extensiones que contiene todos los datos de un tipo específico de estructura lógica de almacenamiento dentro de un *tablespace*
  - Clasificación de segmentos:
    - Datos
    - Índice
    - Recuperación
    - Temporal

Estructura lógica de la BD



Relación entre segmento, extensión y bloque de datos

Estructura lógica de la BD

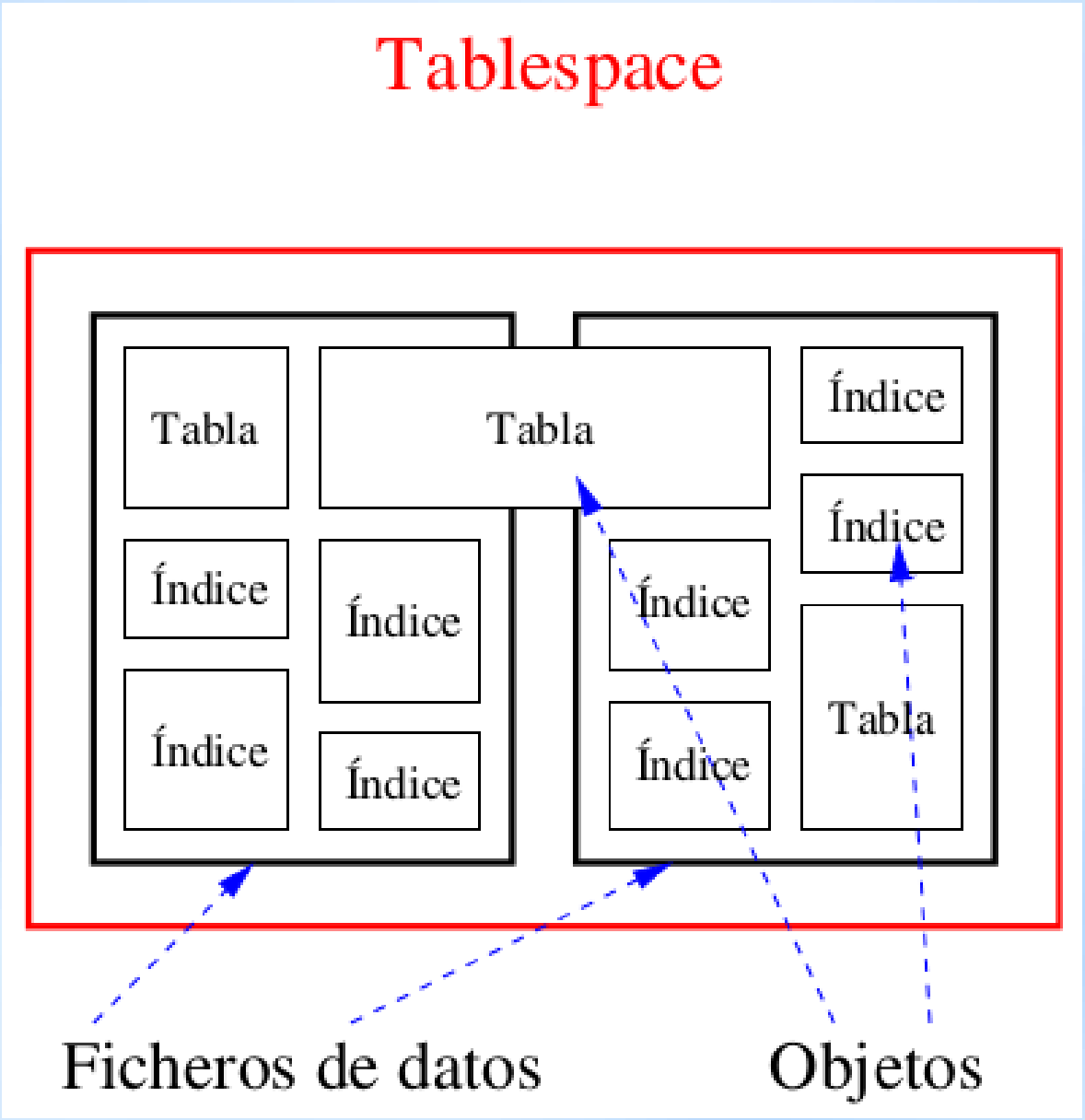


Relación lógica entre estructuras de la BD

Estructura física de la BD

- Un tablespace contiene uno o más ficheros de datos físicos, *ficheros base*
  - Cada fichero no puede pertenecer a más de un tablespace
- Oracle realiza la gestión del espacio de disco reservado para ese fichero
  - Producto transportable e independiente de los SO
- Un fichero puede almacenar diferentes segmentos de datos
  - Fichero mixtos
  - Registros de longitud variable
- Un segmento de datos está almacenado lógicamente en un único tablespace y físicamente entre sus ficheros

Estructura física de la BD



Creación de objetos

- Índices:
  - Primarios: se crean automáticamente con la clausula *constraint* al crear una tabla
  - Secundarios: requieren una sentencia específica para su creación
- Agrupamientos (clustering) de tablas:
  - Necesitamos indicar las tablas que van a formar parte de él y el campo o campos comunes a las tablas
  - Es un objeto propio de la BD y se crea de manera explícita
  - Soporta:
    - Índices: crear un índice
    - Funciones de dispersión: indicar la columna para generar la función

Diccionario de datos

- Conjunto de tablas para registrar información sobre la estructura de la BD
- El propietario del DD es el usuarios SYS
- Los usuarios pueden acceder directamente utilizando vistas
  - ALL
  - DBA
  - USER
- El DD es mantenido por el propio sistema y no puede ser modificado (solo lectura)
- Tablas de rendimiento dinámico:
  - No ocupan espacio
  - Existen mientras se esté ejecutando la instancia
  - Proporcionan información sobre el propio sistema

Conclusiones

- El diseño físico de una BBDD es clave para conseguir un buen rendimiento
  - Requisito NO FUNCIONAL que puede ser importante
  - Hay que ser realista (todo NO puede ser crítico)
- Es dependiente del SGBD
  - Aunque independiente de las capas superiores
  - Los SGBD competitivos suelen compartir soluciones
    - Pero no siempre las implementan igual a bajo nivel
- No existe una solución ideal
  - Y se evoluciona en el tiempo según necesidades
- La forma de consultar la tabla afecta

Referencias

- Transparencias ABBDD, Esther Gadeschi
- Libro Elmasri, 3ª Ed.
- http://webdiis.unizar.es/asignaturas/BD/transparenciasBD/PDFs_1x1/DisFisicSilarri.pdf
- http://ocw.uc3m.es/ingenieria-informatica/diseno-de-bases-de-datos
- http://dis.um.es/~jfernand/0405/dbd/

Gracias por la atención

¿Preguntas?