ADMINISTRACION DE BASE DE DATOS

4. Operación y Mantenimiento

4. Operación y Mantenimiento

TEMAS

- 4.1 Archivos log del SGBD
- 4.2 Definición de los modos de operación de un SGBD. (alta, baja, recovery) y comandos de activación
- 4.3 Índices, reorganización y reconstrucción

COMPENTECIA

Crear y mantener bitácoras de operación para el diagnóstico del rendimiento del DBMS
Crear y mantener índices especializados

4.1 Archivos log del SGBD

Una **bitácora (log)** es una herramienta (archivos o registros) que permite registrar, analizar, detectar y notificar eventos que sucedan en cualquier sistema de información utilizado en las organizaciones.
La estructura más ampliamente usada para grabar las acciones que se llevan en la base de datos.

Cada registro de la bitácora escribe una única escritura de base de datos y tiene lo siguiente:

- Nombre de la Transacción
- Valor antiguo
- Valor Nuevo

4.1 Archivos log del SGBD

Una **transacción** consiste en una secuencia de instrucciones de consulta o de actualización. La norma SQL especifica que una transacción comienza implícitamente cuando se ejecuta una instrucción SQL.
Una de las siguientes instrucciones SQL debe finalizar la transacción:

Commit: compromete la transacción actual; es decir, hace que las actualizaciones realizadas por la transacción pasen a ser permanentes en la base de datos. Una vez comprometida la transacción, se inicia de manera automática una nueva transacción.

Rollback: provoca el retroceso de la transacción actual; es decir, deshace todas las actualizaciones realizadas por las instrucciones SQL de la transacción. Por tanto, el estado de la base de datos se restaura al que tenía antes de la ejecución de la primera instrucción de la transacción.

Permanencia (commit)

La operación **COMMIT** indica la finalización de una transacción satisfactoria: indica al administrador de transacciones que una unidad de trabajo lógica ha concluido satisfactoria mente, que la base de datos está o debería estar nuevamente en un estado consistente y que todas las actualizaciones efectuadas por esa unidad de trabajo ahora pueden ser "confirma das" o definitivas.

Para validar los cambios que se hagan en la base de datos tenemos que ejecutar la orden COMMIT:

SQL> **COMMIT**;
Validación terminada.

Recuperación (rollback)

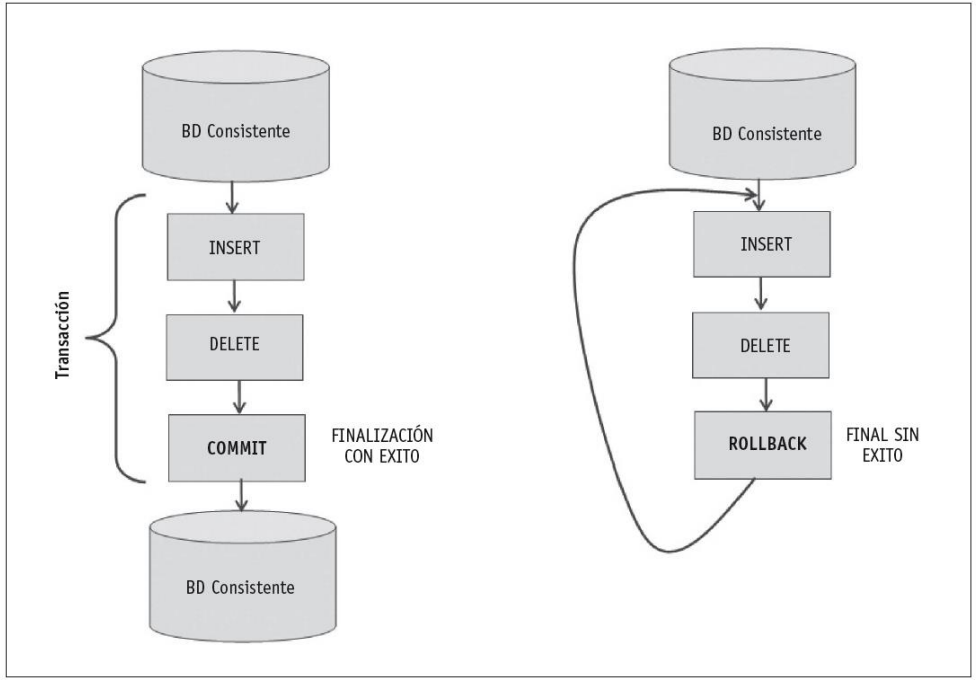
La operación ROLLBACK indica la finalización de una transacción no satisfactoria: indica al administrador de transacciones que algo ha salido mal, que la base de datos puede estar en un estado inconsistente y que todas las actualizaciones realizadas hasta este momento por la unidad de trabajo lógica deben ser “revertidas" o deshechas.

La orden **ROLLBACK** aborta la transacción volviendo a la situación de las tablas de la base de datos desde el último COMMIT:

SQL> **ROLLBACK**;
Rollback terminado.

Recuperación (rollback)

La imagen muestra transacciones típicas que ilustran las condiciones de COMMIT y ROLLBACK.



4.2 Definición de los modos de operación de un DBMS. (alta, baja, recovery)

El SGBD es esencial para el adecuado funcionamiento y manipulación de los datos contenidos en la base. Se puede definir como: "El Conjunto de programas, procedimientos, lenguajes, etc. que suministra, tanto a los usuarios no informáticos como a los analistas, programadores o al administrador, los medios necesarios para describir, recuperar y manipular los datos almacenados en la base, manteniendo su integridad, confidencialidad y seguridad".

4.2 Definición de los modos de operación de un DBMS. (alta, baja, recovery)

Las funciones esenciales de un SGDB son la descripción, manipulación y utilización de los datos.

Descripción: Incluye la descripción de: Los elementos de datos, su estructura, sus interrelaciones, sus validaciones. Tanto a nivel externo como lógico global e interno esta descripción es realizada mediante un LDD o Lenguaje de Descripción de Datos.

Manipulación: Permite: Buscar, Añadir, Suprimir y Modificar los datos contenidos en la Base de

Datos. La manipulación misma supone: Definir un criterio de selección, Definir la estructura lógica a recuperar, Acceder a la estructura física. Esta manipulación es realizada mediante un LMD o Lenguaje de Manipulación de Datos.

Utilización: permite acceder a la base de datos, no a nivel de datos sino a la base como tal, para lo cual: Reúne las interfaces de los usuarios y suministra procedimientos para el administrador. En términos ideales, un DBMS debe contar con estas funciones, sin embargo, no todos las poseen, así existen algunos manejadores que no cumplen la función de respaldo o de seguridad, dejándola al usuario o administrador; sin embargo un DBMS que sea completo y que deba manejar una base de datos multiusuario grande, es conveniente que cuente con todas estas operaciones.

4.2 Definición de los modos de operación de un DBMS. (alta, baja, recovery)

- **Recovery:**

Para poder recuperar un fallo de almacenamiento se debería realizar una copia de seguridad de los archivos de datos y archivos de control periódicamente. La frecuencia de la copia de seguridad determina el tiempo mayor de recuperación, puesto que lleva más tiempo la recuperación si la copia de seguridad es antigua.

4.2 Definición de los modos de operación de un DBMS. (alta, baja, recovery)

Propósito de Backup y Recuperación

Como administrador de copia de seguridad, la tarea principal es diseñar, implementar y gestionar una estrategia de backup y recuperación. En general, el propósito de una estrategia de recuperación de copia de seguridad y es para proteger la base de datos contra la pérdida de datos y reconstruir la base de datos después de la pérdida de datos. Normalmente, las tareas de administración de seguridad son las siguientes:

- Planificación y probar las respuestas a diferentes tipos de fallas.
- Configuración del entorno de base de datos de copia de seguridad y recuperación.
- La creación de un programa de copia de seguridad
- Seguimiento de la copia de seguridad y entorno de recuperación
- Solución de problemas de copia de seguridad
- Para recuperarse de la pérdida de datos en caso de necesidad

4.2 Definición de los modos de operación de un DBMS. (alta, baja, recovery)

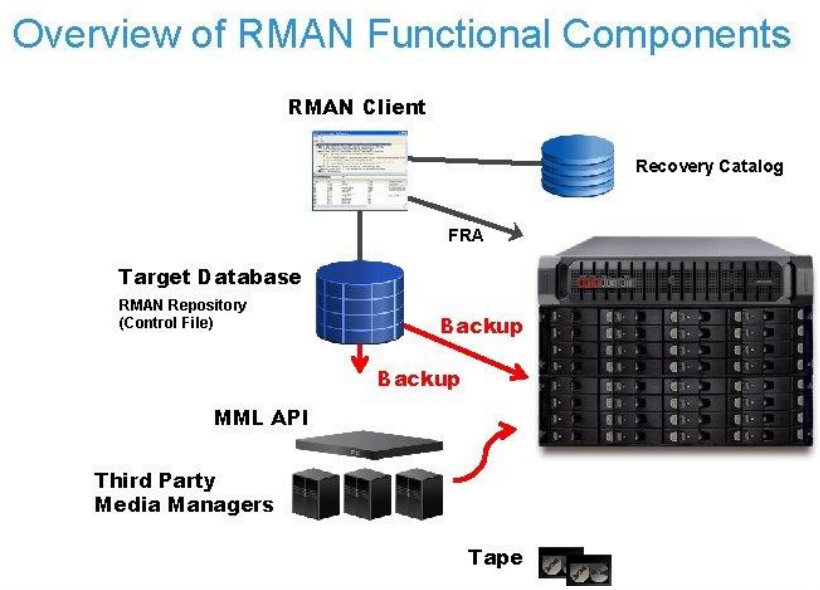
Propósito de Backup y Recuperación

Como administrador de copia de seguridad, es posible que se le pida que realice otros deberes que se relacionan con copia de seguridad y recuperación:

- La preservación de datos, lo que implica la creación de una copia de base de datos para el almacenamiento a largo plazo
- La transferencia de datos, lo que implica el movimiento de datos de una base de datos o un host a otro.

4.2 Definición de los modos de operación de un DBMS. (alta, baja, recovery)

Oracle dispone de la herramienta RMAN (Recovery Manager) para hacer copias de seguridad, tanto en caliente como en frío (incluyendo increméntales).
Esta herramienta es el método preferido Oracle para hacer copias de seguridad eficientes y restauraciones fiables. Se caracteriza principalmente por la optimización que ha ce del espacio en dis y del rendimiento en tiempo durante el backup. Se integra fácilmente con Oracle Secure Backup y con otros productos de control de backups en cintas y dispositivo especiales de salvaguarda.



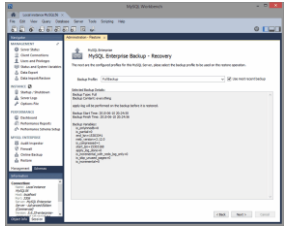
4.2 Definición de los modos de operación de un DBMS. (alta, baja, recovery)

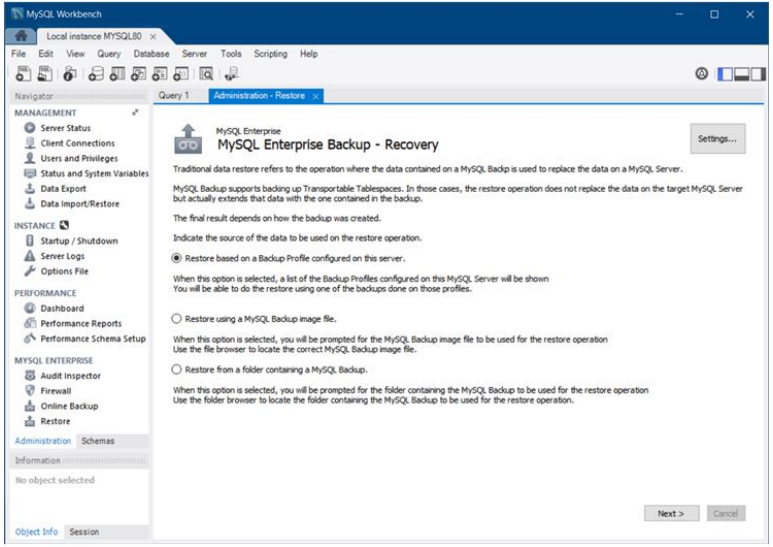
RMAN es una herramienta por línea de comandos que permite realizar copias de seguridad y restauraciones de una base de datos llamada base de datos de destino (target database). RMAN utiliza un repositorio (repository) para almacenar la información de su configuración, las copias de seguridad realizadas, la estructura de la base de datos de destino, los archivos de traza almacenados, etc.
Este repositorio siempre se almacena en el archivo de control de la base de datos de destino. La duración de conservación de la información en el archivo de control se determina mediante el argumento de inicialización CONTROL_FILE_RECORD_KEEP_TIME (7 días, por defecto).

El repositorio también se puede almacenar en un catálogo de recuperación (recovery catalog) que se materializa por un esquema en otra base de datos. Es posible utilizar un único catálogo de recuperación para centralizar los repositorios RMAN de varias bases de datos de destino. Emplear un catálogo de recuperación separado resulta interesante porque la información de copia de seguridad se conserva si todos los archivos de control de la base de datos de destino se pierden.

4.2 Definición de los modos de operación de un DBMS. (alta, baja, recovery)

MySQL ofrece una variedad de estrategias de copia de seguridad desde el que puede elegir los métodos que mejor se adapten a los requisitos de la instalación. El asistente de recuperación de copia de seguridad le permite restaurar las copias de seguridad de las carpetas, archivos de imágenes





Perfil de copia de seguridad: Elija entre los perfiles disponibles de copia de seguridad de MySQL Enterprise en su sistema.

Copia de seguridad de archivo de imagen: Abre el explorador de archivos del sistema; elegir un archivo de imagen de copia de seguridad para restaurar.

Carpeta de respaldo: Abre el explorador de archivos del sistema; Elija una carpeta de copia de seguridad para restaurar.

4.2 Definición de los modos de operación de un DBMS. (alta, baja, recovery)

Para recuperar una base de datos SQL Server desde un fracaso, un administrador de base de datos tiene que restaurar un conjunto de copias de seguridad de SQL Server en una secuencia lógica correcta y significativa de restauración. SQL Server restauración y recuperación soporta la restauración de datos de copias de seguridad de toda una base de datos, un archivo de datos, o una página de datos, como sigue:

La base de datos (una base de datos completa de restauración): Toda la base de datos se restaura y se recuperó, y la base de datos está en línea para la duración de las operaciones de restauración y recuperación.

El archivo de datos (un archivo de restauración): Un archivo de datos o un conjunto de archivos se restauran y se recuperaron. Durante una restauración de archivos, los grupos de archivos que contienen los archivos son automáticamente fuera de línea durante la duración de la restauración. Cualquier intento de acceder a un grupo de archivos sin conexión se produce un error.

La página de datos (una página de restauración): Bajo el modelo de recuperación completa o el modelo de recuperación por medio de registros, puede restaurar las bases de datos individuales. restauraciones de página se pueden realizar en cualquier base de datos, sin importar el número de grupos de archivos.

4.2 Definición de los modos de operación de un DBMS. (alta, baja, recovery)

Existen varias opciones para crear copias de seguridad de bases de datos PostgreSQL.

Los comandos básicos para hacer la copia de seguridad y restauración en PostgreSQL de una base de datos completa son lo siguientes

```
$ pg_dump database > database.sql

$ psql -d database -f database.sql
```

postgresql-backup.sh

PosgreSQL usando un formato de copia de seguridad personalizado con la opción *-Fc* permite restaurar posteriormente partes de la base de datos, por ejemplo, una tabla o un índice en concreto.

```
$ pg_dump -Fc database > db.dump

$ pg_restore -d database db.dump
$ pg_restore -U $username --dbname=$dbname --table=$tablename
```

postgresql-backup-format-custom.sh

4.2 Definición de los modos de operación de un DBMS. (alta, baja, recovery)

Existen varias opciones para crear copias de seguridad de bases de datos PostgreSQL.

También es posible separar la definición y los datos de las copias de seguridad con las opciones *–schema-only* y *–data-only*.

```
$ pg_dump -Fc --schema-only database > database-ddl.dump
$ pg_dump -Fc --data-only database > database-data.dump

$ psql -d database -f database-ddl.sql
$ psql -d database -f database-data.sql
```

postgresql-schema-data.sh

Como las bases de datos pueden llegar a ser de varias decenas de GiB generar las copias de seguridad comprimidas que reducirán su tamaño notablemente. Los volcados con la opción *-Fc* ya está comprimidos con lo que esto solo es necesario si generamos volcados en archivos *.sql*.

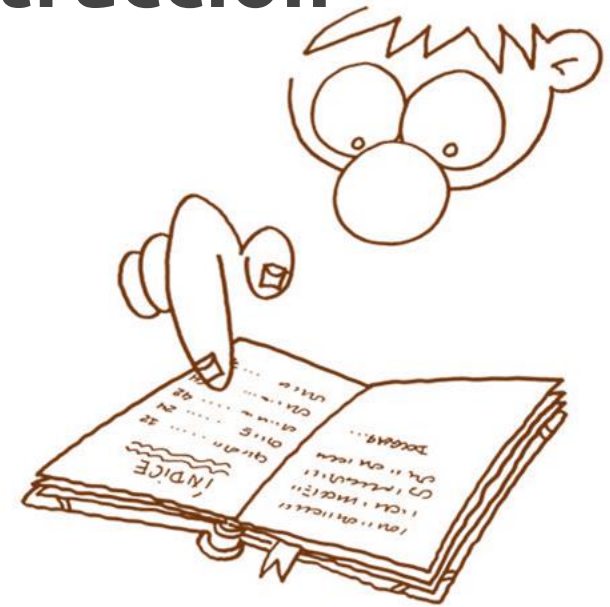
```
$ pg_dump database | gzip > database-backup.sql.gz

$ gunzip < database-backup.sql.gz | psql -d database
```

postgresql-compress.sh

ACTIVIDAD 1. PRACTICA COPIA DE SEGURIDAD Y RESTAURACIÓN

- **Realizar una copia de seguridad completa de una BD en su SGBD**
- **Realizar la recuperación de la BD.**
- **Documentar el proceso y presentar evidencia de la practica.**

4.3 Índices, reorganización y reconstrucción

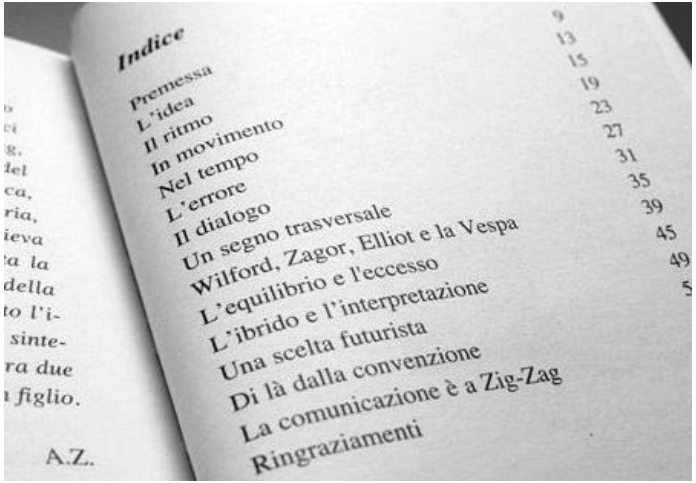


4.3 Índices, reorganización y reconstrucción

Muchas consultas hacen referencia sólo a una pequeña parte de los registros de un archivo.
Por ejemplo, la pregunta «Buscar todas las cuentas de la sucursal Pamplona» o «Buscar el saldo del número de cuenta C-101» hace referencia solamente a una fracción de los registros de la relación cuenta. No es eficiente para el sistema tener que leer cada registro y comprobar que el campo nombre-sucursal contiene el nombre «Pamplona» o el valor C-101 del campo número-cuenta. Lo más adecuado sería que el sistema fuese capaz de localizar directamente estos registros.
Para facilitar estas formas de acceso se diseñan estructuras adicionales que se asocian con archivos.

4.3 Índices, reorganización y reconstrucción

Los índices de los sistemas de bases de datos juegan el mismo papel que los índices de los libros o los catálogos de fichas de las bibliotecas. Por ejemplo, para recuperar un registro cuenta dado su número de cuenta, el sistema de bases de datos buscaría en un índice para encontrar el bloque de disco en que se encuentra el registro correspondiente, y entonces extraería ese bloque de disco para obtener el registro cuenta.



4.3 Índices, reorganización y reconstrucción

Un índice puede estar compuesto por una o varias columnas.
El número de posibles valores que un índice puede contener se llama cardinalidad. Esto determina el número de comprobaciones que hara falta realizar para encontrar un dato concreto
Es el DBA el que debe elegir Ia combinacion óptima que permita los accesos mas rápidos. La máxima al crear índices es que aumente Ia velocidad al buscar datos pero disminuye significativamente Ia velocidad al modificarlos, puesto que hay que regenerar el índice de tal manera que siga estando actualizado.
En el ejemplo de la biblioteca todos los libros se encuentran en estanterías. En un sistema informático esto es mucho mas complejo y los datos ni siguiera necesitan estar ordenados físicamente en el disco duro.

4.3 Índices, reorganización y reconstrucción

Lo importante es contar con un sistema que permita determinar donde se encuentra un dato sin necesidad de buscar uno a uno.

Diferentes SGBD proporcionan diferentes tipos de índice más importante es la clave primaria. La clave primaria siempre se encuentra indexada. Como se ha apuntado, cada SGBD gestiona los índices de forma diferente, en algunos casos incluso los datos se ordenan físicamente según el valor de la clave primaria de forma correlativa, por lo que acceder a un dato sabiendo su clave primara es la manera más rápida posible de hacerlo.

Cabe destacar que es posible indexar por el valor completo de un campo o agrupar. Es decir, si ay un campo nombre, pueden indexarse solo los primeros 3 caracteres. De esta manera se obtiene un acceso bastante radio sin perjudicar demasiado la gestión del índice.

4.3 Índices, reorganización y reconstrucción

A continuación, se muestra un ejemplo de como un índice podría ayudar a buscar datos en una tabla.

Cliente	Empresa	Ciudad	Fecha alta
José	ACME	Ciudad Real	2013-05-20
Marcos	ACME	Málaga	2013-05-10
Luis	ACME	Valencia	2010-05-10
Margarita	Toys Harris Ltd.	Málaga	2012-12-20
María	Toys Harris Ltd.	Málaga	2010-01-08
José María	Toys Harris Ltd.	Madrid	2009-09-20

4.3 Índices, reorganización y reconstrucción

Suponiendo que el único índice se encuentra en {Cliente} (la clave primaria).

Se quiere acceder a todos los clientes de la empresa ACME:

- Se necesitaría hacer un escaneo completo de la tabla. Se necesitarían 6 registros y solo se devolverían 3.
- Si se añadiera un índice en {Empresa} podrían devolverse los 3 registros directamente.

Se quiere acceder a todos los clientes de “Toys Harris Ltd.” que sean de Maga.

- Si se contara con el índice en {Empresa), se podria ir a los tres registros que son de esa empresa y luego filtrar. Se escanearían 3 registros y se devolverían 2.
- Si se creara un índice en {Empresa, Ciudad} se podría ir directamente a los registros de nuestro interés.

Cliente	Empresa	Ciudad	Fecha alta
José	ACME	Ciudad Real	2013-05-20
Marcos	ACME	Málaga	2013-05-10
Luis	ACME	Valencia	2010-05-10
Margarita	Toys Harris Ltd.	Málaga	2012-12-20
María	Toys Harris Ltd.	Málaga	2010-01-08
José María	Toys Harris Ltd.	Madrid	2009-09-20

4.3 Índices, reorganización y reconstrucción

Este es un ejemplo con una tabla diminuta, no se aprecia rendimiento alguno al utilizar índices, pero una tablas de millones de registros o que ocupen muchos megabytes de almacenamiento en disco, Ia diferencia seria enorme.
El DBA debe conocer como va a ser utilizada su base de datos, decidir
que índices son necesarios y crearlos. Deber, además, decidir Ia cardinalidad de los mismos para encontrar el equilibrio entre el rendimiento de consulta y el rendimiento de modificación.

Cliente	Empresa	Ciudad	Fecha alta
José	ACME	Ciudad Real	2013-05-20
Marcos	ACME	Málaga	2013-05-10
Luis	ACME	Valencia	2010-05-10
Margarita	Toys Harris Ltd.	Málaga	2012-12-20
María	Toys Harris Ltd.	Málaga	2010-01-08
José María	Toys Harris Ltd.	Madrid	2009-09-20

4.3.1 Tipos de índices

Hay dos tipos básicos de índices:

Índices ordenados. Estos índices están basados en una disposición ordenada de los valores.

La idea tras la estructura de acceso de un índice ordenado es parecida a la que hay tras el índice de un libro, que enumera al final de la obra los términos importantes ordenados alfabéticamente, junto con las páginas en las que aparecen. Podemos buscar en un índice en busca de una lista de direcciones (números de páginas en este caso) y utilizar esas direcciones para localizar un término en el libro, buscando en las páginas especificadas.

4.3.1 Tipos de índices

Índices asociativos *(hash indices)*. Estos índices están basados en una distribución uniforme de los valores a través de una serie de cajones (buckets). El valor asignado a cada cajón está determinado por una función, llamada función de asociación *(hash function)*.

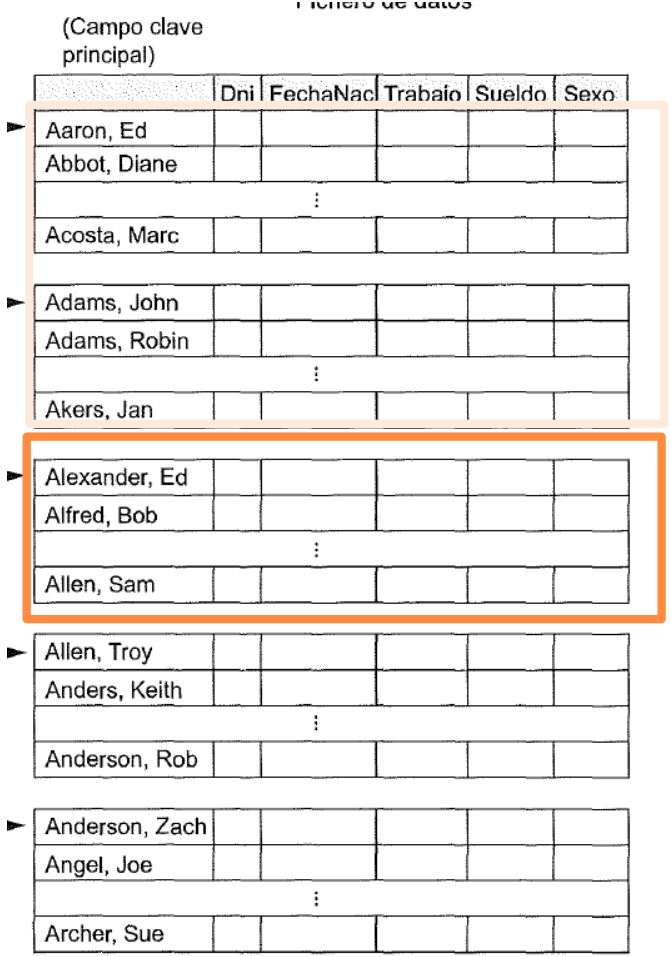
4.3.1 Tipos de índices

Archivos ordenados.

Ejemplo:

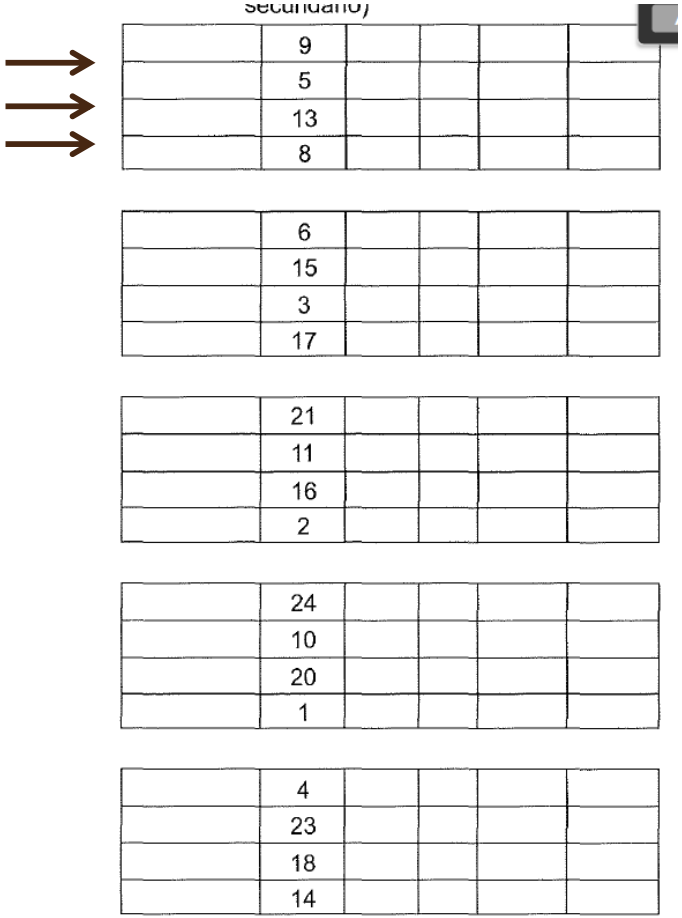
- Buscar Al

- Buscar AD



4.3.1 Tipos de índices

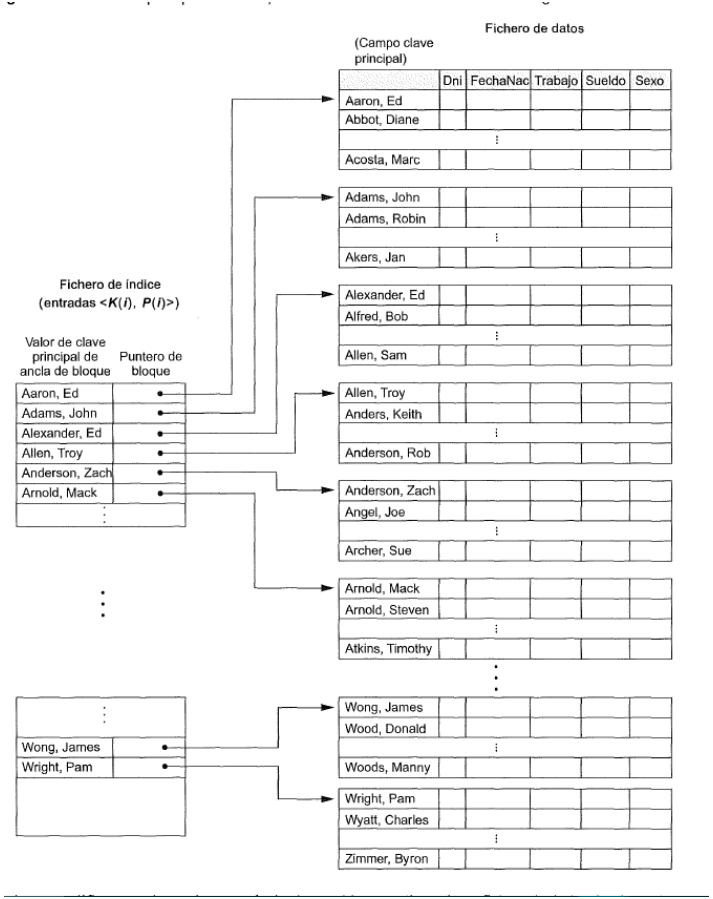
Archivos desordenados



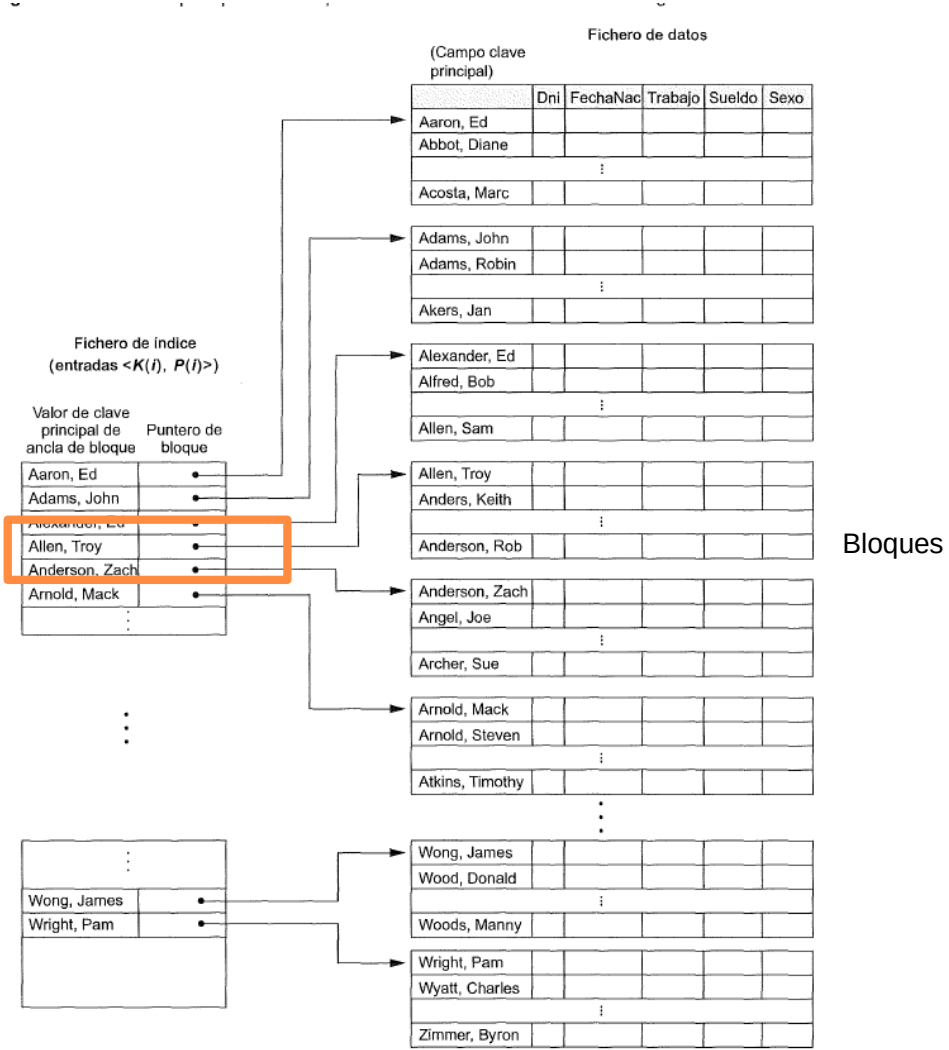
4.3.1 Tipos de índices

Índice Principal

- Un **índice principal** es un fichero ordenado cuyos registros son de longitud fija con dos campos.
- El primer campo es del mismo tipo de datos que el campo clave de ordenación (denominado **clave principal)** del fichero de datos, y el segundo campo es un puntero a un bloque del disco (una dirección de bloque).
- En el fichero índice hay una **entrada de índice (o registro de índice)** por cada *bloque* del fichero de datos. Cada entrada del índice tiene dos valores: el valor del **campo clave principal** para el *primer* registro de un bloque, y **un puntero** a ese bloque.



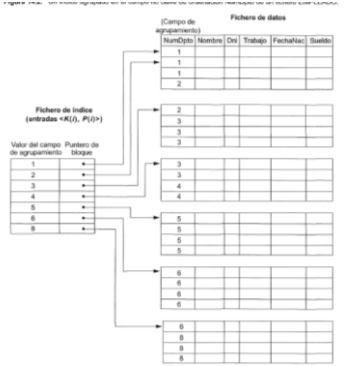
Índice Principal



Bloques

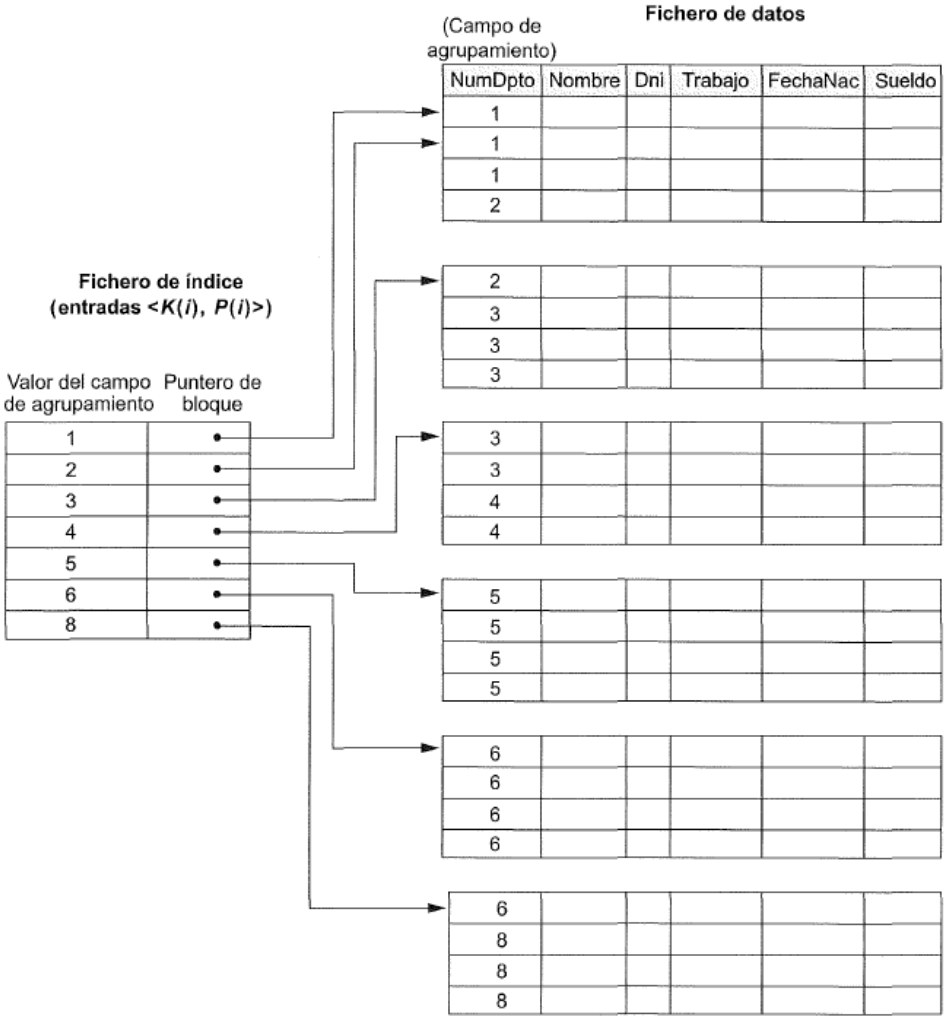
4.3.1 Tipos de índices

Índices agrupados



Si los registros del fichero están ordenados fisicamente por un campo no clave (es decir, un campo que no tiene un valor distinto para cada registro), este campo se denomina campo agrupado.
Podemos crear un tipo de índice diferente, denominado índice agrupado, para acelerar la recuperación de los registros que tienen el mismo valor para el campo agrupado.
Esto difiere de un Índice principal, que requiere que el campo de ordenación del fichero de datos tenga un valor distinto para cada registro.
Un Índice agrupado también es un fichero ordenado con dos campos; el primero es del mismo tipo que el campo agrupado del fichero de datos, y el segundo es un puntero a un bloque.
En el Índice agrupado hay una entrada por cada valor distinto del campo agrupado, que contiene el valor y un puntero al primer bloque del fichero de datos que tiene un registro con ese valor para su campo agrupado.

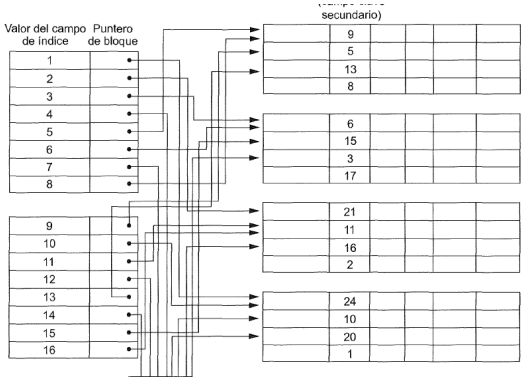
Índices agrupados



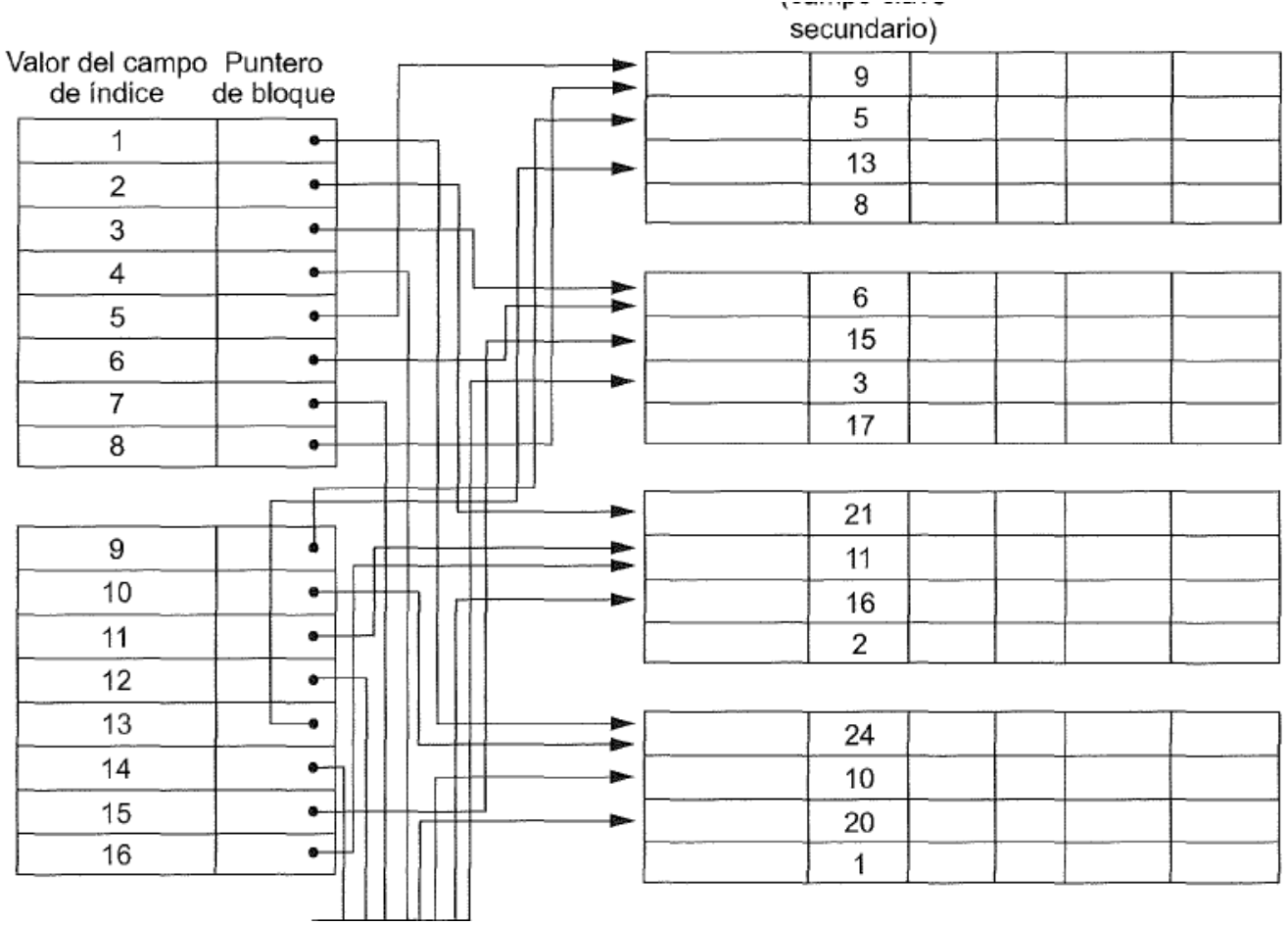
4.3.1 Tipos de índices

índices secundarios

Un índice secundario proporciona un medio secundario de acceso a un fichero para el que ya existe algún acceso principal.
El Índice secundario puede ser un campo que es una clave candidata y tiene un valor único en cada registro, o puede ser una "no clave" con valores duplicados.
El Índice es un fichero ordenado con dos campos. El primero es del mismo tipo de datos que algún campo no ordenado del fichero de datos que es un campo de indexación. El segundo campo es un puntero de bloque o un puntero de registro.
Puede haber muchos Índices secundarios (y, por tanto, campos de indexación) para el mismo fichero.



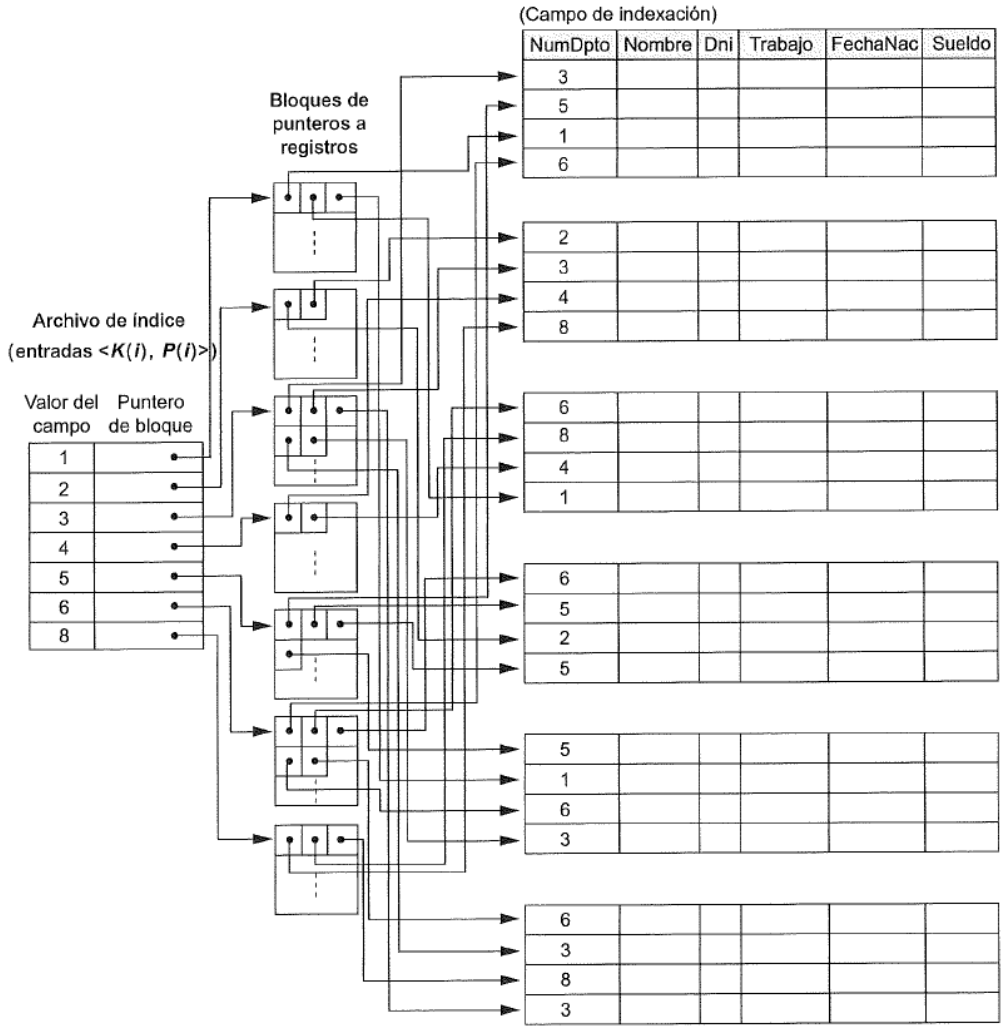
índices secundarios



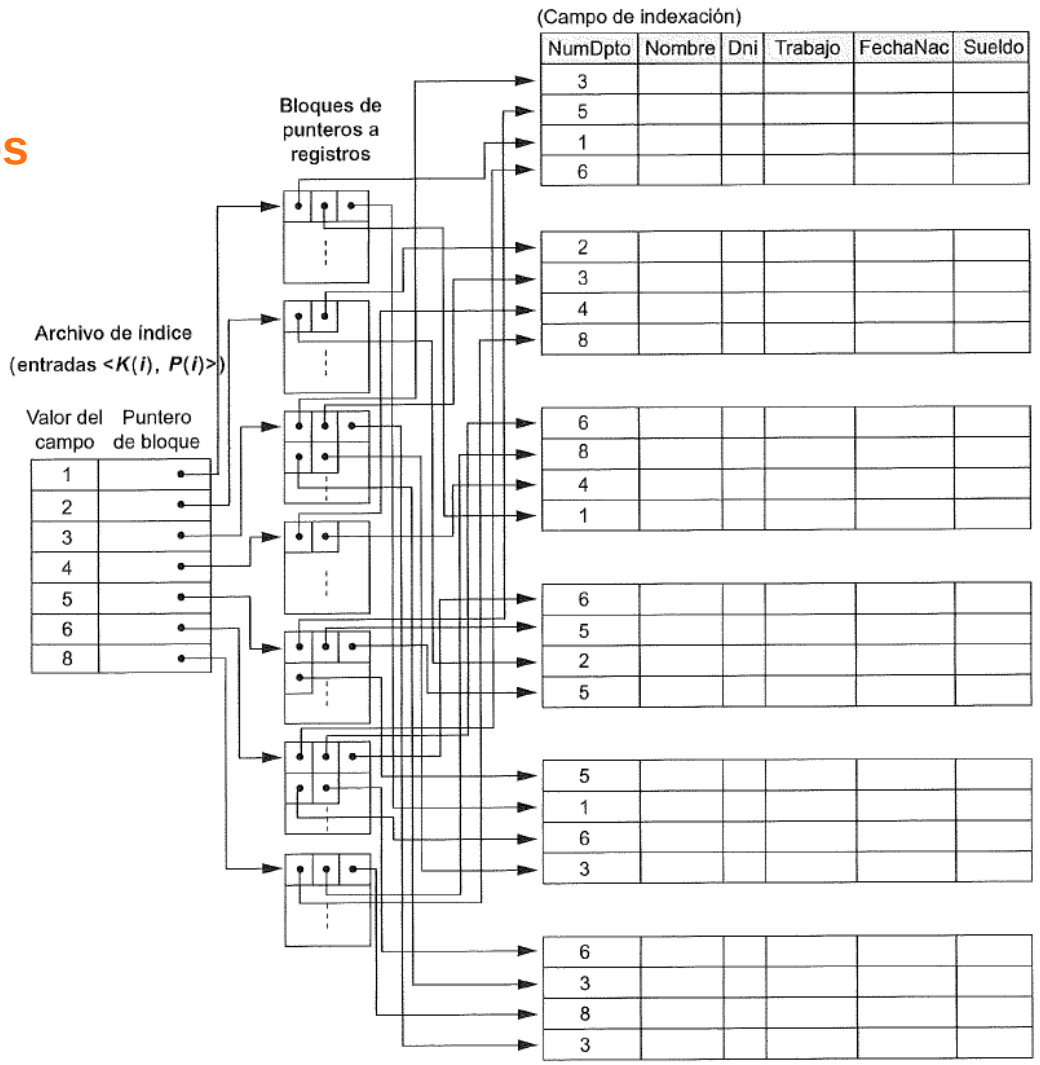
4.3.1 Tipos de índices

índices secundarios

- El campo de valor de índice es una campo No ordenados del fichero
- Este campo no ordenado puede tener valores repetidos
- Habrá una entrada en el índice por cada registro en el fichero
- El campo de dirección del índice apunta aun bloque de punteros a cada registro que contienen el valor indexado.



índices secundarios



4.3.1 Tipos de índices

Se considerarán varias técnicas de indexación y asociación.
Ninguna de ellas es la mejor. Sin embargo, cada técnica es la más apropiada para una aplicación específica de bases de datos. Cada técnica debe ser valorada según los siguientes criterios:

- **Tipos de acceso.** Los tipos de acceso que se soportan eficazmente. Estos tipos podrían incluir la búsqueda de registros con un valor concreto en un atributo, o buscar los registros cuyos atributos
contengan valores en un rango especificado.
- **Tiempo de acceso.** El tiempo que se tarda en buscar un determinado elemento de datos, o conjunto de elementos, usando la técnica en cuestión.
- **Tiempo de inserción.** El tiempo empleado en insertar un nuevo elemento de datos. Este valor incluye el tiempo utilizado en buscar el lugar apropiado donde insertar el nuevo elemento de datos, así como el tiempo empleado en actualizar la estructura del índice.
- **Tiempo de borrado.** El tiempo empleado en borrar un elemento de datos. Este valor incluye el tiempo utilizado en buscar el elemento a borrar, así como el tiempo empleado en actualizar la estructura del índice.

4.3.1 Tipos de índices

Espacio adicional requerido. El espacio adicional ocupado por la estructura del índice. Como
normalmente la cantidad necesaria de espacio adicional suele ser moderada, es razonable sacrificar el espacio para alcanzar un rendimiento mejor.

Amenudo se desea tener más de un índice por archivo.
Volviendo al ejemplo de la biblioteca, nos damos cuenta de que la mayoría de las bibliotecas mantienen varios catálogos de fichas: por autor, por materia y por título.

Los atributos o conjunto de atributos usados para buscar en un archivo se llaman **claves de búsqueda**. Hay que observar que esta definición de *clave* difiere de la usada en *clave primaria*, *clave candidata* y *superclave*.
Este doble significado de *clave* está (por desgracia) muy extendido en la práctica.
Usando nuestro concepto de clave de búsqueda vemos que, si hay varios índices en un archivo, existirán varias claves de búsqueda.

4.3.1 Tipos de índices

Actualización del índice

Sin importar el tipo de índice que se esté usando, los índices se deben actualizar siempre que se inserte o borre un registro del archivo

4.3.1 Tipos de índices

En la tabla siguiente se indican los tipos de índice disponibles en SQL Server y se proporcionan vínculos a información adicional.

Hash	Directrices para usar índices en las tablas con optimización para memoria
Índices no clúster con optimización para memoria	Directrices para usar índices en las tablas con optimización para memoria
Agrupado	Índices agrupados y no agrupados descritos Crear índices clúster
No agrupado	Índices agrupados y no agrupados descritos Crear índices no clúster
Único	Crear índices únicos
Almacén de columnas	Descripción de los índices de almacén de columnas Usar índices no clúster de almacén de columnas Usar índices clúster de almacén de columnas
Índice con columnas incluidas	Crear índices con columnas incluidas
Índice en columnas calculadas	Índices en columnas calculadas
Filtrados	Crear índices filtrados
Espacial	Información general sobre los índices espaciales
XML	Índices XML (SQL Server)
Texto completo	Rellenar índices de texto completo

4.3.1 Tipos de índices

En MySQL hay cinco tipos de índices:

PRIMARY KEY: Este índice se ha creado para generar consultas especialmente rápidas, debe ser único y no se admite el almacenamiento de NULL.

KEY o INDEX: Son usados indistintamente por MySQL, permite crear indices sobre una columna, sobre varias columnas o sobre partes de una columna.

UNIQUE: Este tipo de índice no permite el almacenamiento de valores iguales.

FULLTEXT: Permiten realizar búsquedas de palabras. Sólo pueden usarse sobre columnas CHAR, VARCHAR o TEXT

SPATIAL: Este tipo de índices solo puede usarse sobre columnas de datos geométricos (spatial) y en el motor MyISAM

4.3.1 Tipos de índices

Para crear un índice, se empleará la siguiente estructura:

“CREATE [UNIQUE | FULLTEXT | SPATIAL] INDEX index_name ON table_name (index_col_name…) index_type;”

Donde:

index_name: es el nombre del índice.
table_name: es el nombre de la tabla donde se va a crear el índice.
index_col_name: nombre de la columna (o columnas) que formarán el índice.
index_type: es el tipo del índice. Se emplea con USING [BTREE | HASH].

4.3.1 Tipos de índices

Un índice de base de datos Oracle provee una vía rápida de acceso a la información de las tablas. Oracle dispone de varios esquemas de indización para complementar la funcionalidad de desempeño. Estos son:

Indices B-tree: los más usados.
Índices B-tree cluster: definidos específicamente para clusters.
Índices Hash cluster: definidos específicamente para un hash cluster.
Índices globales y locales: relacionados a tablas e índices particionados.
Índices bitmap: para trabajar mejor con columnas que contienen un pequeño conjunto de valores.
Índices basados en funciones: contienen el valor precalculado de una función o expresión.
Índices de dominio: específicos para una aplicación

4.3.1 Tipos de índices

La creación de un índice en Oracle se realiza mediante el comando ***create index***. Cuando se define una clave primaria o una columna unívoca (UNIQUE) durante la creación de una tabla o su mantenimiento, Oracle creará automáticamente un índice de tipo UNIQUE que gestione dicha restricción,La sintaxis completa de ***create index*** es la siguiente:

create [bitmap | unique] index nombre_indice on
nombre_tabla (nombre_columna [, nombre_columna2] ...) [reverse];

- *bitmap* indica que se cree un índice de mapa de bits que permite crear índices en columnas con muy pocos valores diferentes.
- *unique* indica que el valor de la o las columnas indexadas debe ser único, no puede haber duplicidades.
- *nombre_indice* debe ser un nombre unívoco (no debe existir otro nombre de objeto en Oracle) que siga los convenios de denominación de Oracle para nombrar columnas.
- *nombre_tabla* será el nombre de la tabla donde se creará el índice.
- *nombre_columna* (o columnas) será la columna de la tabla *nombre_tabla* en la que se creará el índice. Se puede crear un índice para varias columnas.
- *reverse* indica a Oracle que invierta los bytes del valor indexado, lo que puede mejorar la distribución del procesamiento y de los datos cuando se insertan muchos valores de datos secuenciales

4.3.1 Tipos de índices

Estructuras de datos que utiliza su motor:
Sus estructuras trabajan con archivos auxiliares para manejar los índices y asi acceder directamente a los registros**,** los cuales son:
Arboles B. (<, <=, =, >=, > y modificadores)

Arboles R. *GiST* (<<, &<, &>, >>, <<|, &<|, |&>, |>>, @>, <@, ~=, &&; estos son operadores para datos geométricos)

Hash (= y sin soporte de NULL data)

GIST. (<@, @>, =, &&; estos son operadores para datos tipo array y para “Full Text Searching”)

SP-GiST, GIN and BRIN

4.3.1 Tipos de índices

Índices de una sola columna

Un índice de una sola columna es el que se crea en función de una sola columna de la tabla. La sintaxis básica es la siguiente:

```
CREATE INDEX index_name
ON table_name (column_name);
```

Índices de varias columnas Un índice de varias columnas se define en más de una columna de una tabla. La sintaxis básica es la siguiente:

```
CREATE INDEX index_name ON table_name
(column1_name, column2_name);
```

El comando DROP INDEX

```
DROP INDEX index_name;
```

ACTIVIDAD 2. PRACTICA ÍNDICES

- **Implementar Índices en una BD en su SGBD**
- **Realizar 3 tipos de índices.**
- **Documentar el proceso y presentar evidencia de la practica.**
- **Elaborar un Video en equipo documentado el proceso creación de todos los tipos de índices del SGBD asignado..**