I.E.S. FIDIANA

Córdoba

Departamento de Informática

APUNTES MÓDULO BASES DE DATOS

UT2. ALMACENAMIENTO DE LA INFORMACIÓN

Profesor: **Samuel Barbero Hidalgo**

TEMA 1.- ALMACENAMIENTO DE LA INFORMACIÓN

1.- ALMACENAMIENTO DE LA INFORMACIÓN

Al ser la memoria RAM volátil y temporal, la información que se desee conservar debe grabarse en dispositivos externos y permanentes. Un *dispositivo de almacenamiento* está constituido por un *soporte* y la *unidad* o conexión que lo maneja (un pendrive y un puerto usb, por ejemplo).

Cuando guardamos un archivo, instalamos un programa, etc., el ordenador almacena la información en el disco duro en pequeñas áreas llamadas clústeres[1].

1.1.- Representación de la información

- **Bit** (Binary Digit o dígito binario) es la unidad mínima de información manipulada por el ordenador y está representada físicamente por un elemento biestable que solo puede tomar el valor on/off, encendido/apagado, 0/1 …

La representación de información se logra mediante la agrupación de bits para lograr un conjunto de valores mayor que permite manejar mayor información. Por ejemplo, la agrupación de ocho bits compone un byte que se utiliza para representar todo tipo de información, incluyendo las letras del alfabeto y los dígitos del 0 al 9.

[1] Un **clúster** (o **unidad de asignación)** es un conjunto contiguo de sectores que componen la unidad más pequeña de almacenamiento de un disco. Los archivos se almacenan en uno o varios clústeres, dependiendo de su tamaño de unidad de asignación. Sin embargo, si el archivo es más pequeño que un clúster, éste lo ocupa completo.

El tamaño de la unidad de asignación depende de la cantidad de fragmentos en que se divide un disco duro cuando se le da formato. Imaginemos un disco duro como un libro, y las unidades de asignación son las páginas, pero si el archivo o el fragmento restante es menor que la unidad de asignación se desperdicia el espacio sobrante. Por ejemplo, si la unidad es de 4096 y el archivo es de 512, la pérdida es de 3584. El tamaño de unidad de asignación de los clústeres así como el espacio de almacenamiento perdido debido a los archivos que ocupan menos que el tamaño del clúster depende del sistema de archivos que emplee el disco.

- **Código ASCII** (American Standard Code for Information Interchange o Código Estándar Americano para el Intercambio de Información): Es un código que asigna valores numéricos a las letras, dígitos numéricos y caracteres especiales (+-¿*/,-:….). Incluye 256 códigos divididos en dos conjuntos, estándar o básico y extendido, de 128 cada uno. Puedes ver la tabla completa y ampliar información en este enlace.

- **El sistema binario** desempeña un importante papel en la tecnología de los ordenadores. Los primeros 20 números en el sistema en base 2 son 1, 10, 11, 100, 101, 110, 111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111, 10000, 10001, 10010, 10011 y 10100. Cualquier número se puede representar en el sistema binario, como suma de varias potencias de dos.

1.2.- Medidas de almacenamiento de la información

El byte es una unidad de información que consta de 8 bits. En procesamiento y almacenamiento informático un byte es equivalente a un único carácter, como puede ser una letra, un número o un signo de puntuación. La siguiente tabla muestra las unidades de almacenamiento:

Tabla de equivalencias de informática			
Medida	**Simbología**	**Equivalencia**	**Equivalencia en Bytes**
byte	b	8 bits	1 byte
kilobyte	Kb	1024 bytes	1024 bytes
megabyte	MB	1024 KB	1 048 576 bytes
gigabyte	GB	1024 MB	1 073 741 824 bytes
terabyte	TB	1024 GB	1 099 511 627 778 bytes
petabyte	PB	1024 TB	1 125 899 906 842 624 bytes
exabyte	EB	1024 PB	1 152 921 504 606 846 976 bytes
zetabyte	ZB	1024 EB	1 180 591 620 717 411 303 424 bytes
yottabyte	YB	1024 ZB	1 280 925 819 614 629 174 706 176 bytes
brontobyte	BB	1024 YB	1 237 940 039 285 380 274 899 124 224 bytes
geopbyte	GB	1024 BB	1 267 650 600 228 229 401 496 703 205 376 bytes

2.- FICHEROS O ARCHIVOS DE DATOS

Para el almacenamiento permanente de la información, surge la necesidad de los **ficheros** o **archivos**, que son estructuras de datos externas, con elementos del mismo tipo almacenados secuencialmente.

Cada elemento de un archivo o fichero se denomina ***registro*** o ***registro lógico*** y contiene toda la información sobre algún ente o componente del fichero. A su vez, cada registro está formado por informaciones lógicas denominadas ***campos***.

Así, en un fichero de alumnos, toda la información correspondiente a un alumno o alumna constituye un registro, mientras que la información de alguna de sus características o atributos (nombre, fecha de nacimiento) por ejemplo, sería un campo. La noción de campo es subjetiva, así el campo fecha de nacimiento podría dividirse en tres ***subcampos*** correspondientes al día, mes y año.

Cuando se lee o se escribe en un fichero, el sistema economiza el número de lecturas/escrituras en disco de forma que lee una cantidad de bytes determinada (normalmente la capacidad de un sector de disco) que suele incluir más de un registro de información.

Esa cantidad de información que se lee o escribe en cada operación de lectura o escritura sobre un archivo se denomina ***registro físico*** y su tamaño dependerá del tipo de computadora y del sistema operativo. Un registro físico podrá contener uno o varios ***registros lógicos***. Se denomina ***factor de bloqueo*** al número de registros lógicos contenidos en un registro físico.

Algunas clasificaciones que suelen hacerse de los ficheros son:

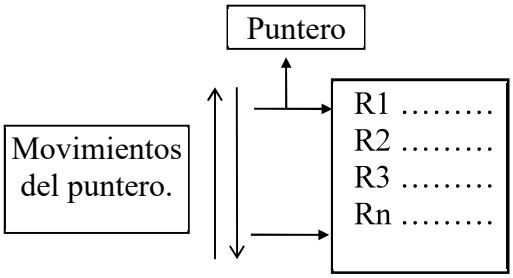
1.- Dependiendo de la dirección del flujo de datos:

- *De entrada*: los datos se leen por el programa desde el archivo.
- *De salida*: los datos se escriben por el programa hacia el archivo.
- *De entrada/salida*: los datos pueden se escritos o leídos

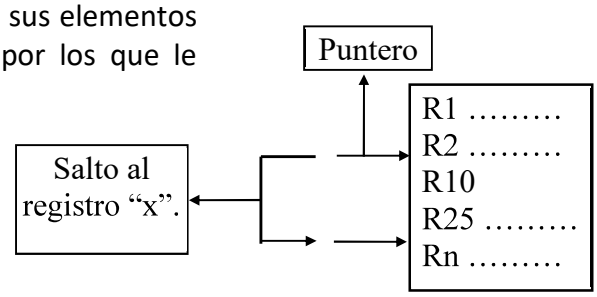
2.- Según la disposición física de los datos almacenados:

- *Ficheros Secuenciales*, que guardan los registros en el orden de escritura. La escritura se realizará al final del archivo y para leer un registro habrá antes que leer los que le preceden en orden. Cuando se abre un fichero secuencial, el puntero de lectura apuntará a la primera posición (primer registro).

  No obstante, podremos movernos por el fichero a cualquier posición del mismo. Tiene la desventaja de desaprovechar los huecos que se producen en el borrado de registros y el tiempo de acceso a un registro.



- *Ficheros Relativos o Directos*, que almacenan cada registro en una dirección, lo que permite acceder a sus elementos de forma inmediata sin tener que pasar por los que le preceden.



  Este acceso directo o inmediato se logra haciendo coincidir la clave del registro (nº natural) con su dirección relativa (direccionamiento directo), o mediante un algoritmo sobre la clave que calcule un valor para su dirección relativa (direccionamiento calculado).

  La necesidad de esta coincidencia lleva a un desaprovechamiento del soporte en ficheros con gran dispersión en sus claves. Piénsese por ejemplo que, si el fichero solo consta de un registro con clave 1000, en disco ocupará lo mismo que si constase de los 1000 primeros.

Además del acceso directo admiten el secuencial, pero es más lento que en los ficheros secuenciales ya que los registros no se encuentran consecutivos en el orden de escritura.

- *Ficheros Secuenciales Indexados*, con una organización tal que permite tanto el acceso secuencial como directo. Estos ficheros se organizan en dos áreas o zonas. Una de datos, que contiene la información almacenada. Otra de índices que guarda las referencias y direcciones de grupos de registros (similar a la estructura de un libro con su índice).

  El área de índices está organizada de forma secuencial ordenada por la clave de acceso, y junto a cada dirección del área de datos contiene la clave más almacenada en esa dirección. El área de datos contendrá la información accesible de forma directa por su dirección.

Área de índices:

20	d1	100	d2	200	d3	....	d4	....	d5	900	d6

Área de datos:

d1	1	Marina Pérez	2	Emilio García	20	Sara González
d2	21	Javier López	22	Ana B. Herrera	100	Patricia Rico
d3	101	David Rodríguez	175	José L. Robles	200	Ana Martínez
d4	.....		....		....	
d5	.....		....		....	
d6	884	Amparo Calvo	885	Mariano López	900	Julia Díez

Para localizar un registro por su clave, primero se realiza la búsqueda en la tabla de índices hasta encontrar la primera clave superior a la dada. Junto a esta clave, se encuentra la dirección inicial del grupo de registros que debe contener la información del registro a localizar. El puntero de lectura se colocará directamente en esa dirección del área de datos y leerá el registro físico completo que contiene el registro lógico buscado, al cual accederá desde memoria por su campo clave.

Puedes ampliar información sobre los tipos de organización de ficheros en el siguiente enlace.

3.- PASO DE LOS FICHEROS TRADICIONALES A LAS BASES DE DATOS

En ciertas ocasiones, la informatización de una empresa se realiza sin tener en cuenta los datos disponibles por las aplicaciones ya implementadas, por lo que nos encontramos con un gran número de ficheros con información repetida.

Ejemplo: Un Instituto de Enseñanza cuyo sistema de información se basa en los ficheros tradicionales. Los datos de los alumnos y alumnas matriculados se encontrarían en distintos ficheros asociados a distintas aplicaciones: por un lado, en el fichero que maneja la aplicación de Secretaría (datos de matrícula, notas, convalidaciones, etc..), por otro lado, en el fichero de la aplicación de biblioteca y en tercer lugar en los ficheros que cada tutor maneja para el control de faltas de asistencia del alumnado de su grupo.

Esta situación de aplicaciones tradicionales "estancas" padecía de ciertos inconvenientes:

- Desaprovechamiento del soporte debido a la redundancia de datos.
- Desincronización y pérdida de tiempo en la actualización de informaciones repetidas o relacionadas y pertenecientes a distintos ficheros. Así, por ejemplo, la matriculación de un nuevo alumno o alumna, la baja de una matrícula o simplemente la modificación de una dirección derivada de un cambio de domicilio, provocaría la actualización en cada uno de los ficheros.
- Dificultad para relacionar registros de distintos ficheros (campos clave de distinto tipo o formato, por ejemplo).
- Poca flexibilidad para adaptarse a otro contexto distinto al que fueron diseñados. Si fuera necesario introducir un nuevo campo (dni, por ejemplo), además de las estructuras de los ficheros habrían de modificarse los programas que manejasen los ficheros que hubiesen sufrido cambios en sus estructuras.

Lo ideal sería trabajar bajo un nuevo enfoque, que se apoye sobre una estructura de datos en la que **los datos son recogidos y almacenados una sola vez, con independencia de los tratamientos o programas que los utilizarán.** Este enfoque se le conoce como **Estructura de Base de Datos**.

Sus **ventajas** son las siguientes:

- Menor redundancia. No hace falta tanta repetición de datos. Aunque, sólo los buenos diseños de datos tienen poca redundancia.
- Menor espacio de almacenamiento. Gracias a una mejor estructuración de los datos.
- Acceso a los datos más eficiente. La organización de los datos produce un resultado más óptimo en rendimiento.
- Datos más documentados. Gracias a los metadatos que permiten describir la información de la base de datos.
- Independencia de los datos y los programas y procesos. Esto permite modificar los datos sin modificar el código de las aplicaciones.
- Integridad de los datos. Mayor dificultad de perder los datos o de realizar incoherencias con ellos.
- Mayor seguridad en los datos. Al limitar el acceso a ciertos usuarios.

Como contrapartida encontramos los siguientes **inconvenientes**:

- Instalación costosa. El control y administración de bases de datos requiere de un software y hardware potente.
- Requiere personal cualificado. Debido a la dificultad de manejo de este tipo de sistemas.
- Implantación larga y difícil. Debido a los puntos anteriores. La adaptación del personal es mucho más complicada y lleva bastante tiempo.

4.- CONCEPTO DE BASE DE DATOS

La expresión ***'base de datos'*** apareció en un simposio celebrado en Santa Mónica (USA) en 1.963, a partir de entonces ha adquirido un auge e implantación crecientes. Una B.D. es una ***colección de datos interelacionados y estructurados*** según un modelo. La B.D. debe ser capaz de ***almacenar las interelaciones entre esos datos***, al igual que almacena los propios datos, siendo ésta una diferencia esencial respecto a los ficheros.

La definición y la descripción del conjunto de datos contenidos en la B.D. deben ser únicas y estar integradas con los mismos datos. Su descripción, definición y documentación completas (metadatos) se almacenan junto a los datos, en su ***esquema*** de Base de Datos.

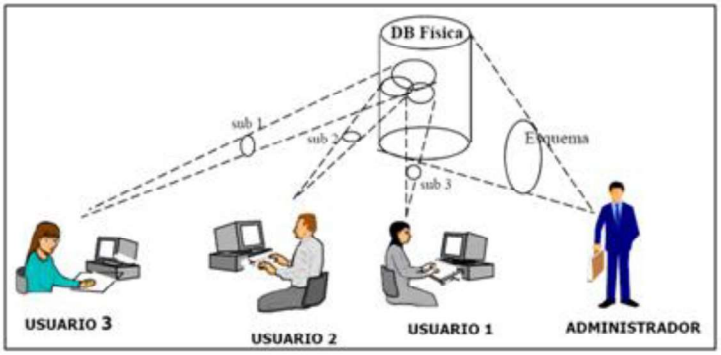
En las B.D. ***no debe existir redundancia lógica***, aunque es admitida cierta redundancia física por razones de eficiencia. Por lo que un dato se actualizará lógicamente por el usuario de forma única, corriendo a cargo del sistema la actualización física de los campos en los que el dato estuviera repetido (caso de redundancia física).

Las B.D. han de ***atender a múltiples usuarios y a diferentes aplicaciones***, a diferencia de los ficheros tradicionales, diseñados para una aplicación concreta. Es decir, debe existir ***independencia física y lógica entre datos y las aplicaciones que los usan***, independencia que constituye el objetivo fundamental de las B.D.

La manipulación de la B.D. debe realizarse con procesos bien determinados que permitan tres elementos fundamentales, como son la ***seguridad, integridad y confidencialidad*** de los datos.

Siguiendo con el ejemplo del Instituto de Enseñanza, podríamos diseñar una base de datos compuesta por los ficheros ALUMNOS, CURSOS, LIBROS, PRESTAMOS y FALTAS. Ahora no es necesario almacenar los datos del alumnado más que una vez, pues si la B.D. está bien diseñada los nombres de los alumnos y alumnas de un curso, o el del que tiene un libro prestado podrá obtenerse partiendo de las relaciones correspondientes y siempre del fichero ALUMNOS.

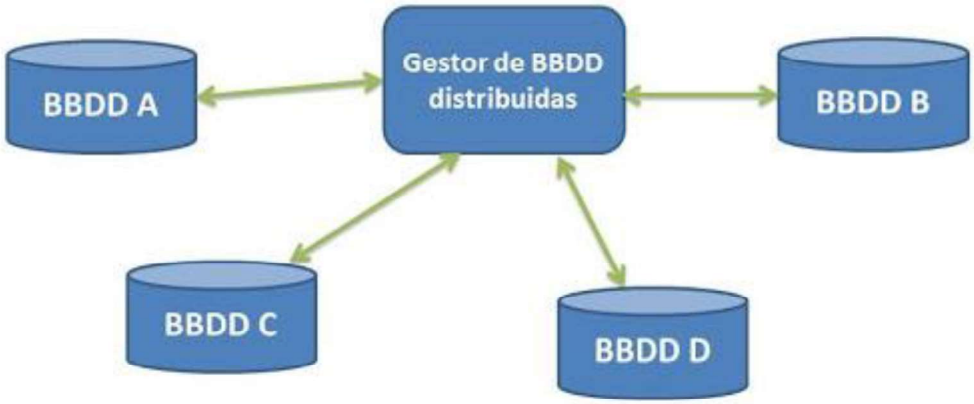
El estudio de estas características nos lleva a la siguiente definición de B.D.: "**Colección de datos integrados, con redundancia controlada, y con una estructura que refleje las interelaciones existentes en el mundo real; los datos, que han de ser compartidos por diferentes usuarios y aplicaciones, deben mantenerse independientes de éstas, y su definición y descripción, únicas para cada tipo de datos, han de estar almacenadas junto con los mismos. Los procedimientos de actualización y recuperación, habrán de ser capaces de conservar la integridad, seguridad y confidencialidad del conjunto de los datos.**"



5.- BASES DE DATOS DISTRIBUIDAS

La tecnología de bases de datos distribuidas es reciente. Una base de datos distribuida es, una base de datos no almacenada físicamente en un solo lugar, sino que se distribuye a lo largo de una red de ordenadores conectados y geográficamente distantes entre sí.

Por ejemplo: un sistema comercial internacional donde cada delegación nacional almacene físicamente la información de sus operaciones (clientes, ventas, ..) y en la Central se encuentren los datos de proveedores, resultados, históricos, etc. De esta forma, los datos se almacenan en el lugar donde más se usan, pero, a través de la red de comunicaciones, también están disponibles para los usuarios de otros lugares.



Las ventajas de tal distribución son, entre otras que combina la eficiencia del procesamiento local (sin excesivos costos de comunicación) y las ventajas de compartir los datos que ofrece un sistema centralizado. Pero también existen desventajas como son los costos de comunicación, elevados si el diseño no es el correcto y las dificultades técnicas para instrumentarla.

Un objetivo básico de un sistema distribuido es que el usuario lo perciba como un sistema centralizado, es decir, el usuario no necesita saber dónde se encuentran físicamente almacenados los datos (independencia física) por lo que el hecho de que la base de datos esté distribuida debe ser importante solo a nivel interno, y no a los niveles externo o conceptual.

Para manipular la información se utiliza un Sistema Administrador de Base de Datos Distribuida (SABDD), software que permite administrar la Base de Datos de forma transparente para el usuario, que la percibirá como si fuera local, de forma que cuando realiza una consulta que implica datos de varias localidades, la consulta se divide en subconsultas que se ejecutan en paralelo.

Para ampliar sobre el tema de la fragmentación puedes consultar este enlace:

6.- NIVELES DE ABSTRACCIÓN DE UNA BASE DE DATOS

En los ficheros tradicionales existen dos estructuras distintas, la **lógica de usuario** (vista de usuario) y la **física** (forma en que se encuentran los datos en el soporte). En las B.D. aparece un nuevo nivel de abstracción llamado **nivel conceptual**, o **esquema** que pretende ser una representación global de los datos intermedia entre las estructura lógica y física independiente, tanto del equipo como de cada usuario en particular.

La **estructura lógica de usuario** es la visión que cada usuario (usuario final o programador) tiene de la B.D. (vistas, en el modelo relacional). Sin embargo, **el esquema** se corresponde con el enfoque del conjunto de datos de la empresa (visión del administrador de la B.D.). La **estructura física** es la forma en que se organizan los datos en el soporte (Figura 3).

Niveles de abstracción de una B.D.

El nivel externo (lógica de usuario) es el más alto de abstracción y por ello el más cercano a los usuarios. Este nivel representa la percepción individual de cada usuario de la B.D., es decir las distintas vistas que los usuarios perciben de los datos, vistas que le habrá concedido el administrador en función de los programas que manejan, aunque físicamente no existen como ellos los ven.

Habrá tantos esquemas externos como exijan las diferentes aplicaciones. Un mismo esquema externo podrá ser utilizado por diversas aplicaciones y/o usuarios.

El nivel conceptual es un nivel medio de abstracción coincide con la percepción que tiene el administrador de todos los de datos que conforman la B.D. (ficheros y sus estructuras).

En este nivel, o esquema deberá incluirse la descripción de todos los datos (ficheros, campos, claves, ..) y sus interelaciones (campos de conexión, tipos de relación,..) además de las restricciones de integridad y confidencialidad (claves de acceso, algoritmos de comprobación de campos, rangos de validación, campos obligatorios, …). No siendo necesario especificar las rutas de acceso.

El nivel interno es el más bajo de abstracción. Representa la B.D. a nivel físico de almacenamiento y es particular para cada SGBD. En este nivel o esquema interno el administrador de la B.D. deberá especificar aspectos como: asignación de espacios de almacenamiento, rutas de acceso a los datos, claves de acceso (primarias, secundarias, etc.), control de acceso (confidencialidad y seguridad), etc…

7.- LOS SISTEMAS DE GESTION DE BASES DE DATOS

7.1.- Concepto.

Un **sistema gestor de bases de datos** (SGBD) consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a dichos datos. La colección de datos, normalmente denominada base de datos, contiene información relevante para una empresa. El objetivo principal de un SGBD es proporcionar el interfaz necesario entre los diferentes tipos de usuarios (usuarios finales, analistas, programadores y el administrador) y la Base de Datos.

El SGBD puede definirse como "***Un conjunto de programas, procedimientos, lenguajes, etc. que suministra a todos los usuarios los medios necesarios para describir, recuperar y manipular los datos almacenados en la base, manteniendo su integridad, confidencialidad y seguridad***."

La principal herramienta de un SGBD es la interfaz de programación con el usuario. Esta interfaz consiste en un lenguaje muy sencillo mediante el cual el usuario interactúa con el servidor. Este lenguaje comúnmente se denomina **SQL, Structure Query Language**, está estandarizado por la ISO 1, es decir, todas las BD que soporten SQL deben tener la misma sintaxis a la hora de aplicar el lenguaje.

7.2.- Funciones

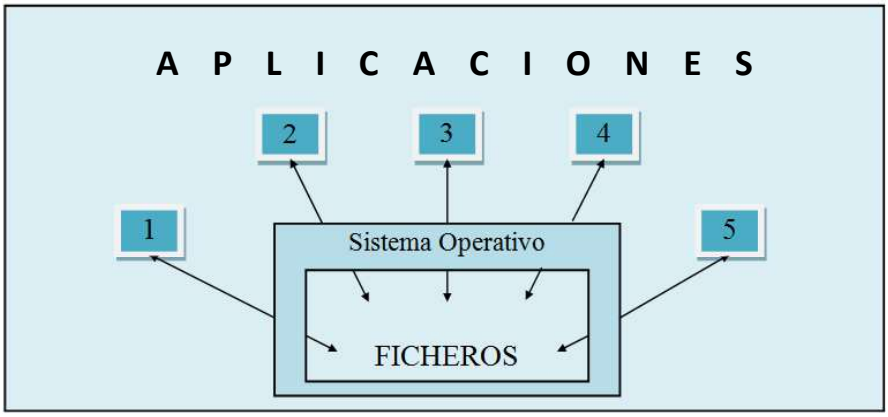
Las dos funciones esenciales de un SGBD son la de descripción y la de manipulación.

a) **La función de descripción o definición**, debe permitir al administrador de la B.D. especificar los elementos de datos, su estructura y las relaciones que existen entre ellos, las reglas de integridad, claves de acceso a la B.D., así como las características físicas y las vistas lógicas de los usuarios. Esta descripción se realiza con el Lenguaje de Descripción de Datos (LDD) propio de cada SGBD.

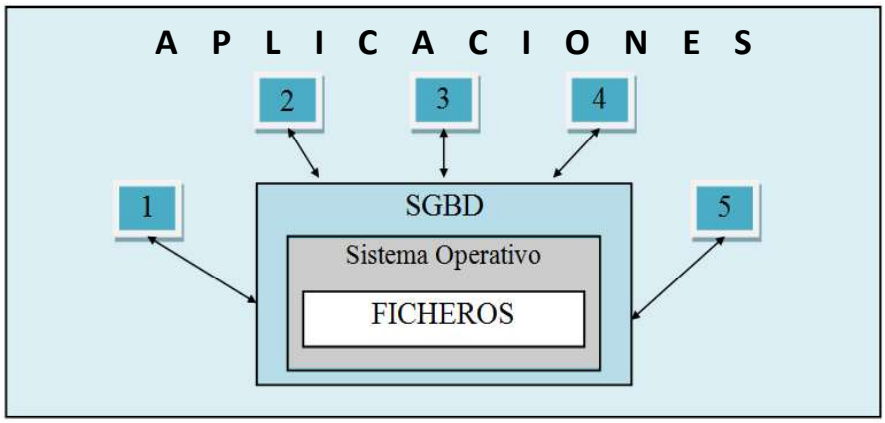
b) **La función de manipulación** permite a los usuarios buscar o actualizar datos de la B.D., de acuerdo con las especificaciones y normas de seguridad dadas por el administrador. Esta manipulación se realiza con el Lenguaje de Manipulación de Datos (LMD).

7.3.- Interfaz con el Sistema Operativo

La forma de trabajar con el sistema operativo y el almacenamiento físico de los datos que tiene una aplicación basada en un SGBD, difiere bastante con la forma de trabajo de una aplicación basada en ficheros tradicionales. Mientras que esta trabaja directamente con los ficheros de datos, a nivel físico, la primera trabaja contra el SGBD, y es él el que se encarga de interaccionar con el sistema operativo y el almacenamiento físico. Gráficamente podría representarse según las siguientes imágenes.



Interfaz entre aplicaciones y datos en un sistema orientado a ficheros.



Interfaz entre aplicaciones y datos en un Sistema orientado a Bases de Datos.

Podríamos comparar al S.I. orientado a B.D. respecto al orientado en ficheros tradicionales con el sistema de compra por Internet respecto al de autoservicio. Mientras que el autoservicio exige que sea el comprador el que localice el producto en las estanterías (sistema tradicional), en la compra por Internet el comprador se limita a demandar sus necesidades y será el vendedor el que deberá saber la ubicación física de los productos.

8.- ARQUITECTURA DE UN SGBD

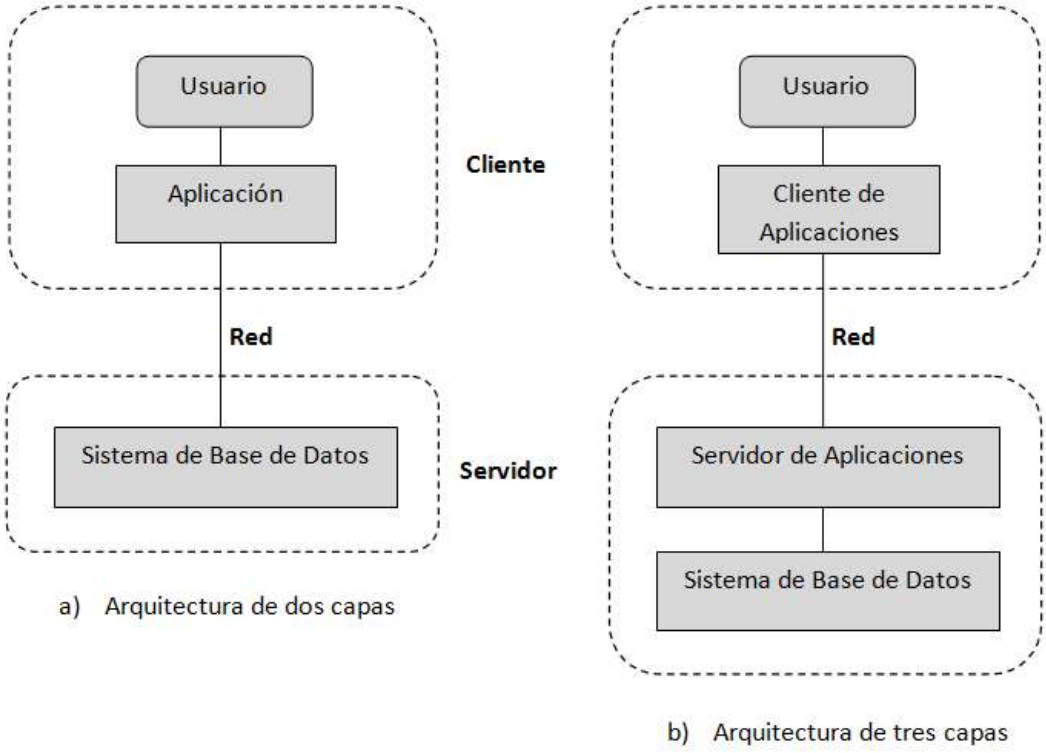
La mayoría de usuarios de un sistema de bases de datos no están situados actualmente junto al sistema de bases de datos, sino que se conectan a él a través de una red.

Se puede diferenciar entonces entre las **máquinas cliente**, en donde trabajan los usuarios remotos de la base de datos, y las **máquinas servidor**, en las que se ejecuta el sistema de bases de datos.

Las aplicaciones de bases de datos se dividen usualmente en dos o tres partes, como se ilustra en la siguiente imagen.

En una arquitectura de dos capas, la aplicación se divide en un componente que reside en la máquina cliente, que llama a la funcionalidad del sistema de bases de datos en la máquina servidor mediante instrucciones del lenguaje de consultas.

Los estándares de interfaces de programas de aplicación como ODBC y **JDBC** se usan para la interacción entre el cliente y el servidor.



Arquitectura de dos y tres capas.

En cambio, en una arquitectura de tres capas, la máquina cliente actúa simplemente como frontal y no contiene ninguna llamada directa a la base de datos. En su lugar, el cliente se comunica con un servidor de aplicaciones, usualmente mediante una interfaz de formularios.

El servidor de aplicaciones, a su vez, se comunica con el sistema de bases de datos para acceder a los datos.

La lógica de negocio de la aplicación, que establece las acciones a realizar bajo determinadas condiciones, se incorpora en el servidor de aplicaciones, en lugar de ser distribuida a múltiples clientes.

Las aplicaciones de tres capas son más apropiadas para grandes aplicaciones, y para las aplicaciones que se ejecutan en World Wide Web.

9.- EL ADMINISTRADOR DE LA BASE DE DATOS

Una de las principales razones de usar SGBDs (Sistemas Manejadores de Base de Datos) es tener un control centralizado tanto de los datos como de los programas que acceden a esos datos. La persona que tiene este control central sobre el sistema se llama **administrador de la base de datos** (ABD). Las funciones del ABD incluyen las siguientes:

- **Definición del esquema**. El ABD crea el esquema original de la base de datos escribiendo un conjunto de instrucciones de definición de datos en el LDD.
- **Definición de la estructura y del método de acceso**.
- **Modificación del esquema y de la organización física**. Los ABD realizan cambios en el esquema y en la organización física para reflejar las necesidades cambiantes de la organización, o para alterar la organización física para mejorar el rendimiento.
- **Concesión de autorización para el acceso a los datos**. La concesión de diferentes tipos de autorización permite al administrador de la base de datos determinar a qué partes de la base de datos puede acceder cada usuario. La información de autorización se mantiene en una estructura del sistema especial que el sistema de base de datos consulta cuando se intenta el acceso a los datos en el sistema.
- **Las especificaciones y vistas (o subesquemas)** para los programas en función de las necesidades de los usuarios sobre los datos almacenados. Aunque la programación no sea responsabilidad del administrador, si lo es dictar las normas y especificaciones que han de cumplir las aplicaciones en su acceso a la base de datos.
- Los **estándares** por los que se va a regir la organización en cuanto a documentación de la base de datos, metodologías de diseño de la misma, etc.
- **Mantenimiento rutinario**. Algunos ejemplos de actividades rutinarias de mantenimiento del administrador de la base de datos son:
    - Copia de seguridad periódica de la base de datos, bien sobre cinta o sobre servidores remotos, para prevenir la pérdida de datos en caso de desastres como inundaciones.
    - Asegurarse de que haya suficiente espacio libre en disco para las operaciones normales y aumentar el espacio en disco según sea necesario.
    - Supervisión de los trabajos que se ejecuten en la base de datos y asegurarse de que el rendimiento no se degrada por tareas muy costosas iniciadas por algunos usuarios.

Las herramientas de que dispone el ABD para cumplir sus funciones son:

Lenguaje de definición de datos (LDD), con el que especificará todos los objetos, atributos e interrelaciones que se almacenarán en la BD, su organización física en los soportes y las vistas de usuario, y por otro lado las restricciones de integridad y confidencialidad.

Utilidades del SGBD para copias de seguridad, arranque ante caídas, asignación de usuarios, importación y exportación de datos desde/a ficheros externos,…

Simuladores y monitores para estudiar distintas alternativas de instrumentación física al objeto de optimizar la BD. Especialmente importantes son las estadísticas de tiempos de acceso que permitirán ir afinando la BD.

Herramientas de ayuda al diseño o CASE (Computer Aided Software Engineering) que permiten automatizar en parte el laborioso proceso de diseño de la BD, desde el análisis de los requisitos hasta la instrumentación. Algunas de estas herramientas proporcionan también ayudas para la normalización, el dimensionamiento de la BD, etc..

Diccionarios de datos, que son "el arma más potente del arsenal del ABD" (Perkinson, 1984) y están evolucionando en los últimos años al concepto actual de diccionario de recursos de información (DRI).

10.- DICCIONARIO DE RECURSOS DE LA INFORMACIÓN

Conforme se aumenta la demanda de información son necesarios instrumentos capaces de ayudar a los diseñadores y demás usuarios a conocer y gestionar el contenido de la BD.

Frecuentemente los usuarios del Sistema de Información (SI) disponen de grandes volúmenes de datos, pero su utilidad es reducida al no estar debidamente documentados, lo que dificulta su localización para su posterior utilización. El **diccionario de recursos de la información** (DRI) pretende ser la herramienta para facilitar esa necesaria integración, y contendrá las descripciones de todos los datos que constituyen el SI.

Los recursos informativos de las empresas o instituciones se gestionan almacenando, administrando y controlando los denominados **metadatos**, esto es, los datos que definen y describen los datos propiamente dichos. En los últimos años han ido apareciendo almacenes de metadatos, siendo el término mas difundido el de *diccionario de datos* (DD).

Es preciso distinguir entre los conceptos de *diccionario y directorio de datos* que, aunque relacionados entre sí, cumplen distintos fines en el Sistema de Información.

Diccionario de datos: Reúne *la información sobre los datos* almacenados en la BD, como descripciones, significado, estructuras, consideraciones de seguridad, edición y uso de las aplicaciones, etc., es decir, lo que los usuarios necesitan para comprender el significado de los datos y poder recuperarlos y manejarlos adecuadamente. El diccionario tiene, por tanto, una finalidad de descripción lógica de los datos y está orientado hacia el usuario (programador y usuario final).

Directorio de datos: Es el subsistema del SGBD encargado de describir *dónde y cómo* se almacenarán los datos, el modo de acceso y otras características físicas de los datos atendiendo las peticiones de programas y procesos. Un directorio de datos contiene las especificaciones necesarias para pasar de la representación externa a la interna de los datos; su objetivo principal es transmitir al sistema la información necesaria para poder acceder a los datos de la base. El directorio, a diferencia del diccionario, es un instrumento del SGBD, y está orientado a facilitar a éste la información que necesita para su funcionamiento.

Las actuales herramientas CASE suelen contener un diccionario llamado ***enciclopedia*** o ***repositorio***, donde se almacenan los datos referentes a esquemas, grafos, información relativa a la gestión de proyectos y configuraciones, etc., los cuales deberán ser suministrados por el ABD para que la herramienta, a partir de ellos, genere las tablas.

11.- PROTECCIÓN DE DATOS: SEGURIDAD, INTEGRIDAD Y CONFIDENCIALIDAD

Los datos almacenados suelen tener un valor inestimable por lo que deben de estar protegidos contra fallos físicos, lógicos y humanos; fallos que alteran y corrompen los datos imposibilitando la finalidad para la que fueron creados, por lo que el ABD debe de conocerlos y remediarlos.

Los SGBD suelen ofrecer mecanismos para prevenir, detectar y corregir los fallos. El administrador deberá conocer y utilizar esos instrumentos para proteger la BD, siendo tres los ámbitos de esta protección: seguridad, integridad y confidencialidad:

11.1 Seguridad

La BD debe estar protegida contra fallos lógicos o físicos que puedan provocar la destrucción de los datos (catástrofes, incendios, sabotajes, rotura del disco duro, corte de suministro eléctrico, etc).

Lo importante, es asegurar que la BD quede en un estado consistente, para lo que se crean unidades de ejecución llamadas ***transacciones***, que pueden definirse como "*una secuencia de operaciones que han de ejecutarse de forma atómica, es decir, o se realizan todas o ninguna*".

El ejemplo clásico de transacción es una operación bancaria de transferencia de dinero entre dos cuentas, en la que o bien se carga el dinero en una cuenta y se abona en la otra, o no se realiza ninguna de esas dos operaciones.

Las transacciones pueden finalizar con éxito y ser grabadas, o por el contrario fracasar, en cuyo caso habría que restaurar el estado original de la BD antes del comienzo de la transacción. Los SGBD suelen gestionar automáticamente las transacciones tratando cada programa como una: su finalización anormal causa que se aborte la transacción y su terminación normal provoca su grabación, pero además suelen ofrecer sentencias para la gestión explícita de transacciones (COMMIT, ROLLBACK, etc.).

Además de la atomicidad, ya comentada, una transacción debe tener las siguientes propiedades:

- **Preservación de la consistencia**. La ejecución de una transacción debe dejar la BD en un estado consistente.
- **Aislamiento.** Una transacción no muestra los cambios que produce hasta que finaliza.
- **Persistencia**, pues los efectos de una transacción exitosa perduran en la BD.
- **Seriabilidad**, en el sentido de que el efecto de ejecutar transacciones concurrentemente debe ser el mismo que se produciría al ejecutarlas por separado en orden en el que van entrando en el sistema.

Para anular y recuperar transacciones, los SGBD suelen utilizar un fichero llamado diario o *log* en el que se guarda toda la información para deshacer (caso de fracasar) o rehacer (si hay que recuperar) las transacciones.

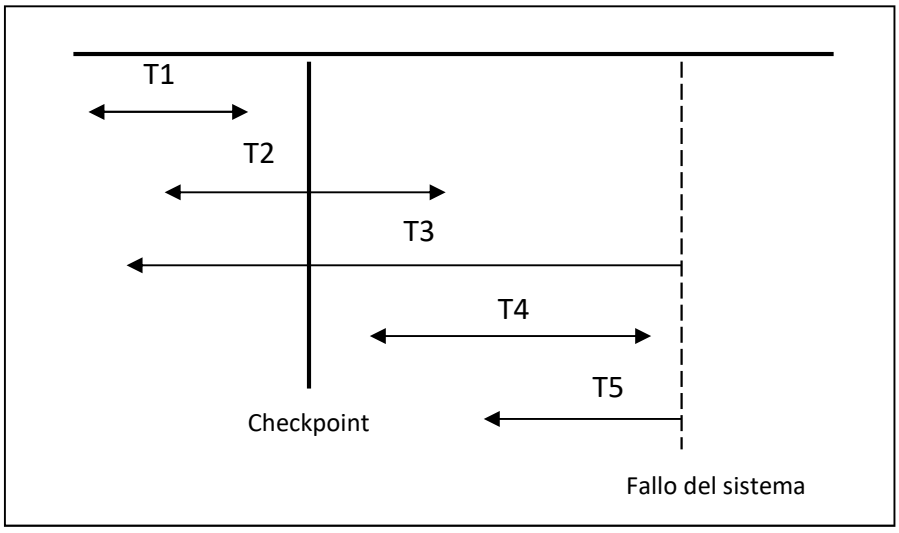
Un registro de ese fichero log suele constar de:

- identificador de la transacción
- hora de modificación
- identificador del registro afectado
- tipo de acción (actualización, borrado, inserción,..)
- valor anterior del registro
- nuevo valor del registro

El fichero diario suele implementarse en un fichero circular, es decir, una pila que una vez llena va eliminando registros conforme van entrando otros nuevos.

Cuando ocurre un fallo que implique la pérdida de memoria volátil, es preciso la llamada *recuperación en caliente* en la que el sistema consulta el fichero *log* para determinar las transacciones que hay que deshacer por que no han sido completadas y hay que rehacerlas porque si bien se han completado, no habían sido grabadas cuando el sistema falló.

Para no tener que recorrer todo el fichero *log* y ejecutar todas las transacciones ya grabadas en la BD, se introducen *puntos de verificación o de recuperación* (*checkpoint*), que se ejecutan periódicamente y que permiten realizar la recuperación desde ese punto. En la figura 7 se muestran varias transacciones, de entre ellas solo la T1 no se vería afectada por el proceso de recuperación por haberse completado antes del último *checkpoint*; las T2 y T4 deberían ser rehechas porque a pesar de haber finalizado no han sido grabadas en la BD, y las T3 y T5 deberán deshacerse al no haber concluido.



Estado de varias transacciones ante un fallo del sistema

Para mayor seguridad pueden obtenerse copias del fichero *log* en dispositivos independientes. Otra forma es con la técnica de páginas ocultas (shadow paging) que consiste en mantener dos tablas (o directorios de páginas) durante una transacción. Al comenzar ambas son idénticas, reflejándose los cambios solo en la tabla primaria; si la transacción tiene éxito se desecha la tabla secundaria y en caso contrario se desecha la primaria y se restablece la secundaria.

11.2 Integridad

La integridad consiste en proteger a la BD contra operaciones que introduzcan inconsistencias en los datos, por lo que nos referimos a corrección, validez o precisión de los datos de la base.

A pesar de que algunos errores no son detectables (año 76 en vez de 67 en una fecha de nacimiento, por ejemplo), existen dos tipos de operaciones que pueden atentar contra la integridad de los datos: las operaciones semánticamente inconsistentes y las interferencias debidas a accesos concurrentes.

A. Operaciones semánticamente inconsistentes

Son las que violan restricciones definidas por el administrador al diseñar la BD, bien sobre dominios (por ejemplo, la edad de un empleado debe ser un entero entre 16 y 65) o sobre los atributos (por ejemplo, la edad de un ingeniero debe ser mayor de 21).

Estas restricciones pueden ser estáticas, como los ejemplos apuntados o dinámicas (por ejemplo: el sueldo de un empleado no puede disminuir, o el sueldo de un empleado no puede ser mayor que el de su jefe inmediato).

Los SGBD ofrecen en su lenguaje de definición facilidades para describir las restricciones que están constituidas por tres elementos:

- la restricción propiamente dicha, que establece la condición que deben cumplir los datos.
- la respuesta a la violación, que especifica la acción o acciones a tomar, como rechazar la operación, informar al usuario, corregir el error con acciones complementarias, etc.
- la condición de disparo, que especifica cuándo debe desencadenarse la acción: antes, después o durante cierto evento.

Por tanto, las reglas de integridad pueden considerarse como un caso especial de los *disparadores* o *triggers*, que son procedimientos que se disparan al ocurrir un evento y que sirven para dar mensajes, calcular campos virtuales, depurar programas, etc..

Los SGBD deben poder almacenar las reglas de integridad en el diccionario, como parte integrante de la descripción de los datos, de forma que ya no habrán de incluirse en los programas, con lo que se consiguen las siguientes ventajas:

- Las reglas de integridad son más fáciles de entender, lo que facilita su mantenimiento.
- Se detectan mejor las inconsistencias.
- Se protege la integridad, pues ningún usuario podrá escribir un programa que las viole.

B. Interferencias por accesos concurrentes

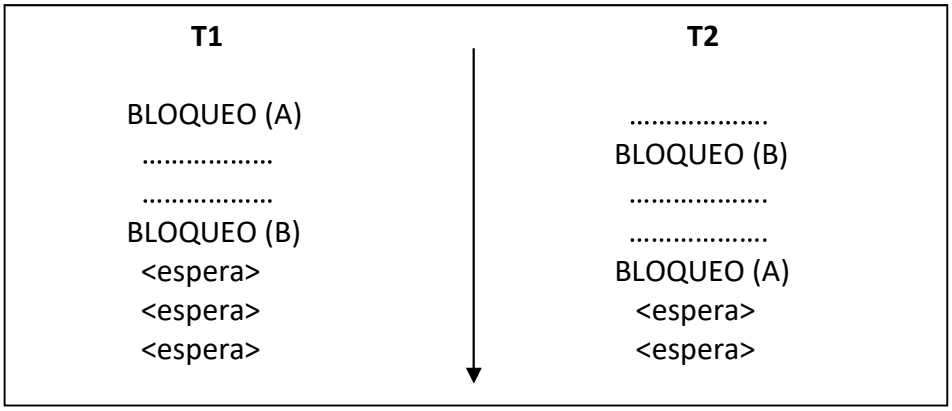
En sistemas multiusuario puede que varios usuarios accedan a la vez al mismo dato para su modificación, de forma que solo quede registrada una de las modificaciones como consecuencia del acceso concurrente. Las principales técnicas que permiten evitar estas interferencias son:

1.- Técnicas de bloqueo. El bloqueo es una técnica que restringe a otros usuarios ciertas operaciones (consulta o actualización) sobre un elemento (campo, registro, tabla o base de datos). Una transacción, por ejemplo, puede bloquear el registro que vaya a actualizar para impedir que otro usuario haga lo mismo, evitando la inconsistencia en el acceso concurrente.

Aunque el propio SGBD gestiona ciertos bloqueos para asegurar la consistencia, los usuarios también pueden hacerlo de forma explícita. Los bloqueos pueden ser:

- *Bloqueos exclusivos*, que impiden que ninguna otra transacción pueda acceder al objeto bloqueado ni realizar sobre él ningún tipo de bloqueo hasta que sea liberado por la transacción que lo había retenido. Se usa para actualizar algún objeto.
- *Bloqueos compartidos*, que permiten que otras transacciones retengan el mismo objeto en bloqueos compartidos, pero no exclusivos. Se usa en transacciones que no actualizan datos, pero que quieren impedir que se modifiquen mientras son consultados

Para prevenir el *interbloqueo* (*deadlock*), que es la situación que se da cuando dos o más transacciones están esperando que cada una de ellas libere algún objeto antes de seguir (figura 8), suele obligarse a que las transacciones bloqueen por adelantado todos los objetos que necesitan. En caso de no poder bloquearlos todos, no bloquea ninguno y se queda en espera hasta volver a intentarlo.



Ejemplo de "*interbloqueo*"

El tema de los bloqueos es importante porque afecta al rendimiento de la BD. En general si se bloquean objetos mayores (bases de datos o tablas) se disminuye el número de bloqueos, pero se retrasa la ejecución de muchas transacciones mientras no se liberen los objetos que éstas necesitan. Por el contrario, bloquear objetos menores (registros o campos) permite mayor concurrencia, pero aparecen más situaciones de interbloqueo.

2.- Técnicas de marcas de tiempo (*timestamping*). Las marcas de tiempo o estampillas son identificadores que se le asignan a las transacciones y que pueden considerarse como el tiempo de inicio de la transacción, de ahí su nombre.

Con esta técnica no existen bloqueos, si no que se controla que la ejecución de las transacciones se haga de forma ordenada

11.3 Confidencialidad

La confidencialidad o privacidad consiste en proteger la BD contra accesos no autorizados. Esta confidencialidad atañe a aspectos ajenos al propio SGBD como son:

- Aspectos legales (Ley de protección de datos).
- Cuestiones de política de la empresa para decidir qué información es pública y a qué niveles.
- Controles de tipo físico y acceso a las instalaciones.
- Controles de accesibilidad del sistema operativo.

Por lo que respecta al SGBD, éste debe mantener registrados tanto los identificadores de los usuarios y sus contraseñas, como los objetos a los que puede acceder un usuario y las operaciones que puede realizar sobre estos objetos.

Además de las vistas, que permiten controlar el acceso de los usuarios a los datos, los SGBD suelen ofrecer otros dos niveles para la asignación de permisos:

Tipo de usuario, distinguiendo entre:

- Administrador, al que le están permitidas todas las operaciones del sistema, incluida la de conceder privilegios y establecer usuarios.
- Usuario con derecho a crear, borrar o modificar objetos, y que además puede conceder privilegios a otros usuarios sobre los objetos que ha creado.
- Usuario con derecho a consultar la BD pero sin derecho a crear borrar o modificar objetos.

Privilegios sobre los objetos, que incluyen indexar, alterar la estructura de los objetos (añadir campos, por ejemplo), insertar, borrar y modificar ocurrencias, etc. (sentencias GRANT y REVOKE). Cuando se revoca un privilegio a un usuario, se le revoca también, en cascada, a los usuarios a los que éste haya podido conceder privilegios.

Para terminar, destacar las facilidades de auditoría que suelen tener los SGBD que permiten recoger en un fichero (audit trait) ciertas operaciones realizadas por determinados usuarios, pudiendo así detectar accesos no autorizados.

12.- SGBD COMERCIALES Y LIBRES

A continuación, mostramos una lista de los SGBD libres y comerciales más usados:

Libres	Comerciales	
• PostgreSQL • MySQL • MongoDB	• Oracle • DB2, Informix (IBM) • dBase (dBI) • Paradox (Borland)	• SQL-Server (MS) • Access (MS) • FoxPro (MS)