Introducción a las bases de datos

1. Sistemas de información

Actualmente las organizaciones manejan multitud de datos, procedentes de diversas fuentes: sus aplicaciones de gestión (compras, ventas, nóminas, gestión de stocks,…), información pública del sector, utilización de sus sistemas por parte de los usuarios, etc.

Todos estos datos conforman un activo muy importante para la organización, y transformar estos datos en información puede suponer un valor añadido muy importante para la misma.

Un sistema de información es un conjunto de elementos orientados al tratamiento y administración de datos e información, es decir, un sistema por medio del cual se recopilan datos, se homogeneizan y se transforman en información relevante.

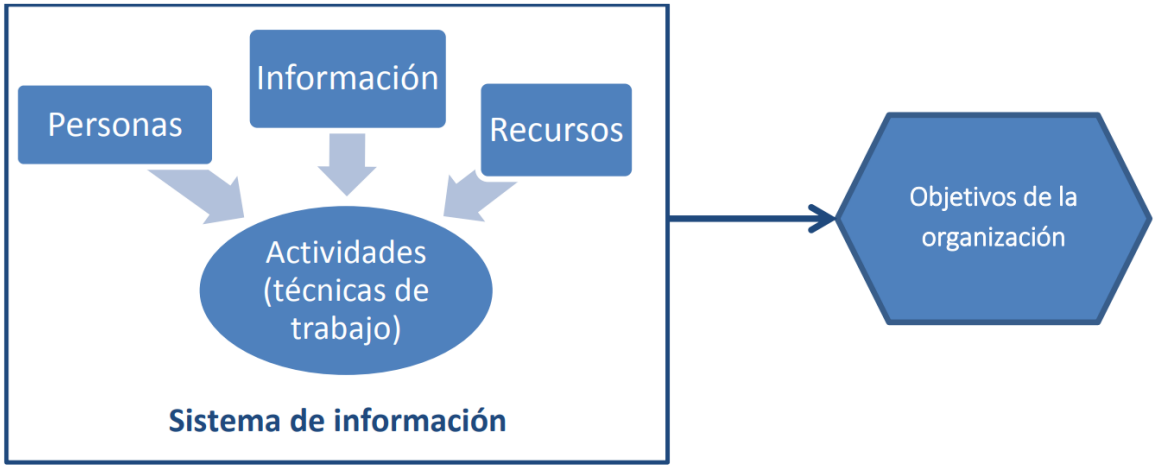
Un sistema de información es especialmente adecuado en entornos complejos, competitivos o muy cambiantes. En estos entornos los cambios son continuos, y la rapidez en la toma de decisiones y la implementación de los cambios es crítica, por lo que los gestores necesitan acceder de forma rápida y eficaz a las fuentes de información que les permitan tomar las decisiones estratégicas más adecuadas.

1.1 Componentes

Un sistema de información no son solamente las bases de datos o herramientas informáticas por medio de las que se maneja la información, sino que tiene los siguientes componentes:

- **Personas**: todo el personal involucrado en la recolección, mantenimiento y análisis de la información.
- **Datos**: hace referencia a toda la información existente en la organización, no solamente la almacenada por medios informáticos (ficheros, bases de datos, etc) sino toda aquella información que pueda ser relevante de cara a la toma de decisiones.
- **Actividades**: son los procedimientos de trabajo, prácticas habituales, protocolos y políticas existentes en la empresa.
- **Recursos materiales**: no solamente informáticos, sino cualquier tipo de recurso utilizado para manejar la información (papel, sistemas de comunicación, etc.).

Todos estos elementos interactúan para procesar los datos y dan lugar a información más elaborada, que se distribuye de la manera más adecuada posible en una determinada organización, en función de sus objetivos.



1.2 Objetivos

El objetivo de un sistema de información será proveer a la organización de herramientas que le permitan:

- Automatizar los procesos operativos (cobros, pagos, gestión de entradas y salidas, etc.).
- Proporcionar información oportuna y exacta que sirva de apoyo para la toma de decisiones.
- Obtener ventajas competitivas derivadas de su implantación y uso.

1.3 Funciones

Las principales funciones de un sistema de información son las siguientes:

- Recolección de información: recoger la información, representarla en un formato adecuado y almacenarla.
- Tratamiento de la información: fundamentalmente, aplicar tres operaciones: integración, realización de cálculos y transferir información entre diversas fuentes. La información recogida debe ser clara, precisa, coherente, oportuna y completa.

- Difusión de la información: proporcionar a cada usuario la información que necesite. Será necesario identificar qué, cómo, cuándo y a quién se distribuirá la información.

1.4 Organización

Los sistemas de organización se organizan en varios niveles, cada uno de los cuales se encarga de una función determinada:

- **Nivel operativo**: sistemas operacionales utilizados en el día a día de la organización (sistema de ventas, sistema de nóminas,…), que son la principal fuente de información relevante para la organización ya que proporcionan información acerca de la propia empresa. Cada departamento o estructura de la organización puede tener su propio sistema operacional, y los usuarios de estos sistemas suelen ser los propios trabajadores de la empresa y los directores operativos.
- **Nivel de conocimiento**: a este nivel corresponden los sistemas orientados a apoyar las operaciones diarias de control. Estos sistemas permiten explotar la información generada por los sistemas de nivel operativo. Pertenecen a este nivel los sistemas de inteligencia de negocio (Business Intelligence), de minería de datos (Data Mining), de “ciencia de los datos” (Data Science) y herramientas de elaboración de informes (Reporting). Los usuarios de estos sistemas tienen un perfil de analistas con un alto conocimiento de algún área específica de la organización.
- **Nivel administrativo**: Los usuarios de estos sistemas son los mandos intermedios de la organización, encargados de definir las líneas a seguir en el medio plazo por su unidad organizativa. La información que se maneja en este nivel es más comparativa que descriptiva, y es tanto interna (normalmente procedente del data warehouse) como externa, ya que además de mostrar información relevante es importante contextualizar esa información para poder definir objetivos realistas y medibles. Las herramientas principales a nivel táctico son las herramientas de elaboración de informes y especialmente los cuadros de mando: sistemas orientados a la toma de decisiones en un área concreta de la organización a partir del seguimiento de una serie de indicadores.
- **Nivel estratégico**: Los usuarios de estos sistemas son la alta dirección, que trabajan con una visión a medio-largo plazo, definiendo las líneas maestras que debe seguir la organización. La herramienta principal de la dirección en este ámbito son los cuadros de mando integrales, que son cuadros de mando “resumidos” que incorporan información de todas las unidades de la organización de forma precisa y resumida, para facilitar la toma de decisiones estratégicas

La información se distribuye por la organización de forma vertical (entre los distintos niveles de la jerarquía) y de forma horizontal (en un mismo nivel entre los distintos departamentos).

2. Sistemas de almacenamiento

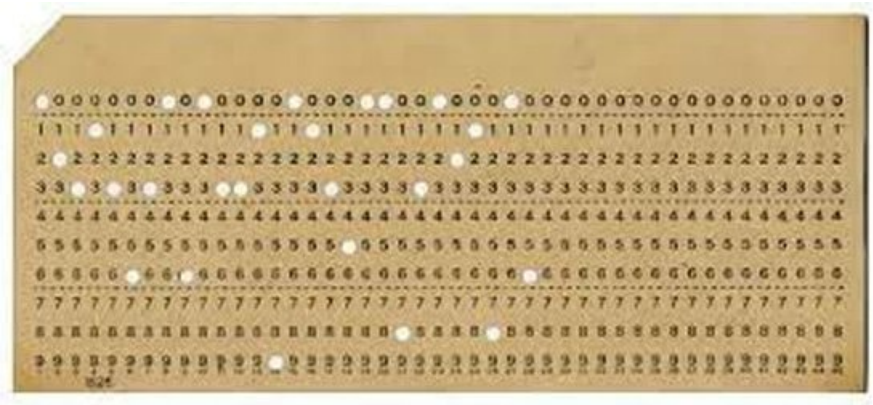
Un sistema de almacenamiento es el soporte físico y lógico de un sistema de información, que permite, guardar, mantener, actualizar y buscar los datos de nuestro sistema para generar nueva información. Además, un sistema de almacenamiento crea una abstracción de como se almacenan los datos para poder ser consultados de una forma más sencilla.

Un sistema de almacenamiento tiene dos partes, la primera, un soporte físico donde la información es almacenada de forma física. Y un sistema lógico que define como se almacena la información dentro del soporte físico. La combinación de estos dos soportes determinará muchos aspectos de eficiencia del sistema de almacenamiento. El estudio y correcta combinación de estos será vital en el diseño de bases de datos o diferentes sistemas de la información.

2.1 Soporte físico

Los elementos del soporte físico son aquellos dispositivos que permiten almacenar la información de forma no volátil en un sistema informático. Habitualmente solemos referirnos a ellos como memoria secundaria o memoria externa, y aunque existen múltiples opciones tecnológicas, las decisiones de diseño se suelen reducir a aquellas tecnologías más recientes o en su defecto las más asequibles económicamente.

- **Mecánicos**: La información se almacenaba por agujeros en papel o cartón y se usaban normalmente para introducir información al ordenador. Es una tecnología obsoleta como las tarjetas perforadas.



Tarjeta perforada.

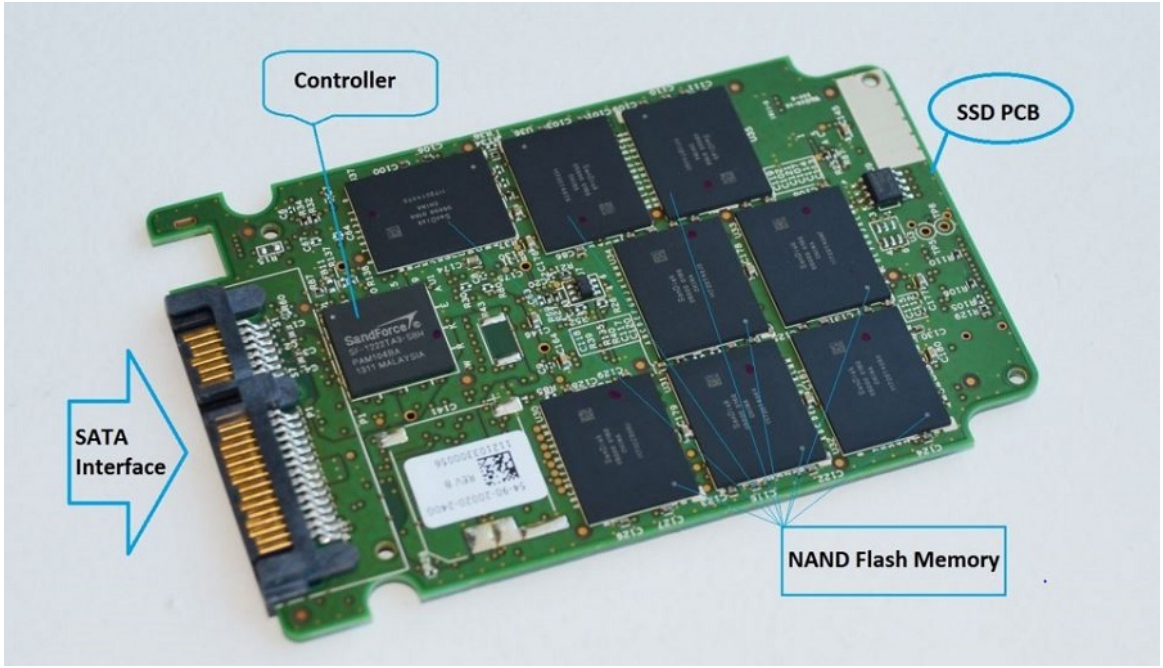
- **Magnéticos**: Emplean la energía magnética para almacenar la información, produciendo un campo electromagnético modifican el estado del soporto que almacena la polaridad, directa o inversa que se interpreta como un uno o un cero. Los soportes más habituales son los discos duros magnéticos o los sistemas de cintas. Aunque las cintas magnéticas puedan parecer una solución antigua son muy interesantes en aquellos casos que tenemos que almacenar muchos datos durante un largo periodo de tiempo sin necesidad de realizar lecturas como el caso de administraciones públicas u hospitales debido a su reducido coste a día de hoy.
- **Ópticos**: utilizan la energía lumínica para almacenar la información en las capas de un disco. Fueron muy populares pero el abaratamiento de los semiconductores ligado a la necesidad de una unidad óptica ha hecho que poco a poco vayan desapareciendo. CD-ROM, DVD, Blu-ray.

CD-ROM, DVD y blue-ray

- **Magneto-ópticos**: Combinan ambas tecnologías como ZIP o Jaz. Totalmente descatalogadas a nivel de usuario.

Dispositivos Zip y Jaz

- **De estado sólido**: Es la tecnología más reciente en llegar al mercado y utiliza energía eléctrica almacenada mediante semiconductores como las memorias USB, SSD. Actualmente son las más eficientes pero hasta hace poco aún eran muy caras y con capacidades limitadas. Actualmente se encuentran en el punto de comenzar a ser el estándar. La tecnología de estado sólido tiene un problema de vida útil según el uso del dispositivo, es importante establecer buenos protocolos de lectura y escritura así como utilizar algoritmos de equilibrado de celdas para poder hacer un buen uso de ellos.



SSD sin carcasa.

2.2 Soporte lógico

El soporte lógico es el método en el que se organizan los datos, existen diferentes formas de organizar los datos, cada una con sus ventajas e inconvenientes. En esta unidad veremos tres principales.

- Sistemas de ficheros.
- Bases de datos.
- Sistemas gestores de bases de datos.

2.3 Características y funciones

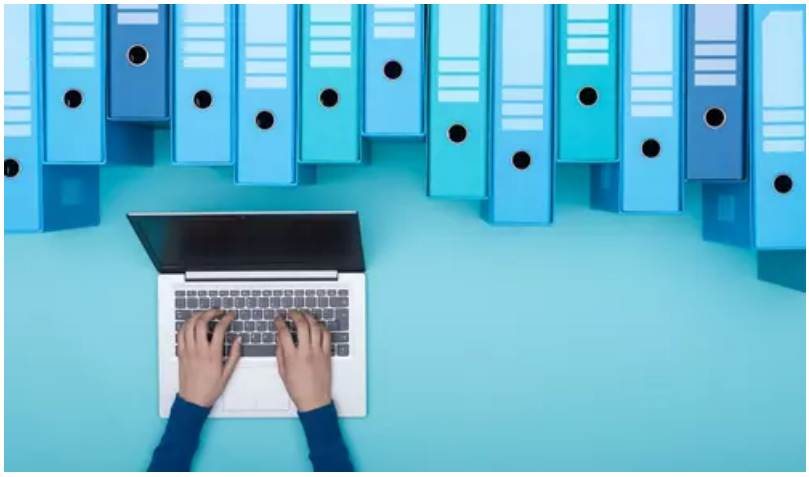
El **almacenamiento de datos** es el **proceso tecnológico** por el cual se **archiva, organiza y comparten los bytes de información que componen los archivos** que se utilizan en el día a día como documentos de texto, imágenes, configuraciones, vídeos, sonidos y cualquier otra información en formato digital. Las principales características que definen a un sistema de **almacenamiento de datos** son:

- **Capacidad.** Mide la cantidad de datos que puede almacenar el sistema de almacenamiento, y es medida en bytes (Gigabytes o Terabytes, habitualmente, aunque con el Big Data se manejan incluso Petabytes).
- **Rendimiento.** Cómo de rápido y eficiente es el sistema de almacenamiento de datos.
- **Fiabilidad.** Es la disponibilidad de los datos cuando son solicitados, así como el hecho de disponer de una baja tasa de errores o fallos (por ejemplo, utilizando una configuración RAID).
- **Recuperabilidad.** Mide la capacidad del sistema para recuperar datos tras una pérdida, borrado, corrupción o cualquier otro incidente que impida el acceso a los mismos.

De esta forma, definir las características en base al soporte físico y lógico de los datos permite al sistema de almacenamiento ser más eficiente en sus principales funciones; que son:

- **Asegurar la integridad de la información**: Si el proceso de almacenamiento no se realiza correctamente, es muy probable que parte de esta información se dañe, pierda o altere, comprometiendo su integridad.
- **Mayor accesibilidad a los datos**; Los buenos sistemas de almacenamiento de datos también lo tienen en cuenta, implantando herramientas que facilitan al personal el acceso a datos críticos, así como análisis y comparaciones más profundas. Recursos de alto valor para planificar una estrategia de marketing digital, por ejemplo.
- **Flexibilidad en los puntos de acceso**: Los principales tipos de almacenamiento de datos en la actualidad ya favorecen este tipo de organización, permitiendo recuperar fácilmente la misma información desde diferentes puntos de acceso. De esta manera, puedes realizar un seguimiento de los procesos desde tu casa o desde una computadora nueva con mayor facilidad.

3. Sistemas de ficheros



Los dispositivos de almacenamiento se dividen en estructuras lógicas llamadas particiones sobre las que se implementan los sistemas de archivos. Un sistema de archivos es una organización de los datos del sistema en forma de ficheros y directorios que gestiona el SO.

3.1 Ficheros

Un fichero es una abstracción de la información que identifica en el SO un conjunto de información para independizar al usuario de saber como y donde se encuentra almacenada. Las características principales de un fichero son:

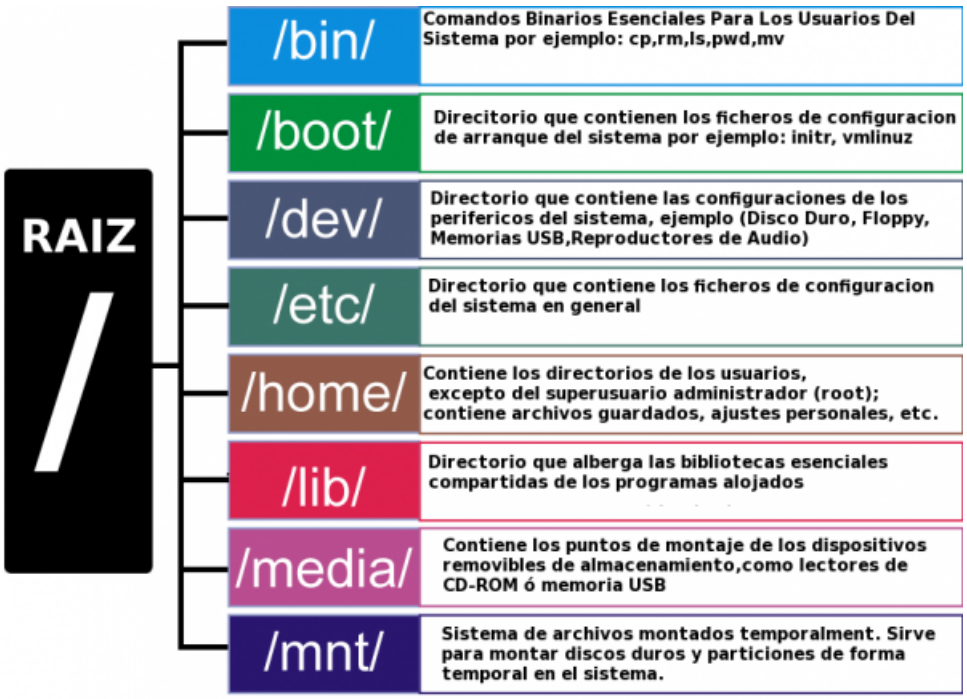
- **Nombre**: Es una forma de identificar el fichero y depende del sistema de archivos pero se suele dividir en dos partes denominadas nombre y extensión que se separan por un punto. La extensión hace referencia al tipo de archivo. Por ejemplo, un .txt sería un archivo de texto plano o un .exe un ejecutable.
- **Estructura**: Es la forma en la que se organiza el fichero dentro del sistema de archivos. Lo más común y lo implementado en UNIX y Windows es describir los ficheros como secuencias de Bytes que son las estructuras más flexibles. Pero existen otras en forma de registros o árboles.
- **Tipo**: En diferentes SO existen diferentes tipos de archivos como los directorios, archivos de bloque, archivos especiales de caracteres, etc.
- **Acceso**: Existen dos formas de acceso tradicionales a los ficheros:
  - Secuencial: Lee todos los Bytes en orden.
  - Aleatorio: Permite acceder a los datos en cualquier orden, es el sistema estándar y permite una mejora del acceso.

  Los sistemas operativos aspiran a la independencia del dispositivo, es decir a hacer que el acceso sea el mismo sin importar dónde esté el archivo
- **Atributos**: Además del nombre y sus datos los ficheros almacenan una gran cantidad de información en forma de metadata para mejorar el uso y eficiencia en el sistema de archivos. Los datos pueden ser de distintos tipos como fechas de creación modificación, permisos, claves de seguridad, etc.
- **Operaciones**: La finalidad de los ficheros es almacenar información para luego recuperarla. Para esto el SO dispone de una serie de operaciones diferentes entre sistemas operativos pero que cumplen los mismos propósitos generales: Crear, buscar, eliminar, abrir, cerrar, leer, escribir, etc.

3.2 Directorios

Un directorio es un contenedor que almacena ficheros y otros subdirectorios y en muchos SO están implementados como otro tipo de ficheros. Un directorio contiene una entrada por cada uno de los ficheros que se encuentra en su interior con el nombre, tamaño, tipo, propietario y otros datos. La forma más flexible de mantener un sistema de directorios es permitir almacenar otros subdirectorios creando una estructura jerárquica de árbol. De esta forma se pueden mantener espacios de almacenamiento para cada usuario donde puede crear sus propios directorios para tener su espacio de trabajo organizado. Habitualmente también existen dos entradas genéricas en los directorios, el "." que hace referencia al directorio de trabajo actual y ".." que hace referencia al directorio del que cuelga el directorio actual. De esta forma se puede interactuar en la jerarquía fácilmente.

Para especificar los nombres de los archivos en estas estructuras de árbol se puede realizar de dos formas. Mediante rutas absolutas, que indican la ruta completa desde la raíz del sistema de archivos o la ruta relativa al espacio de trabajo actual.



Árbol de directorios en Linux

3.3 Gestión de ficheros

Existen dos problemas principales en la implantación de un sistema de ficheros, cómo diseñar los archivos y como asignarles el espacio de forma que la memoria secundaria se utilice de una forma eficiente y el acceso a los archivos sea rápido. Para ello el sistema debe diseñar como se asigna el espacio en disco y como se gestionan el espacio libre.

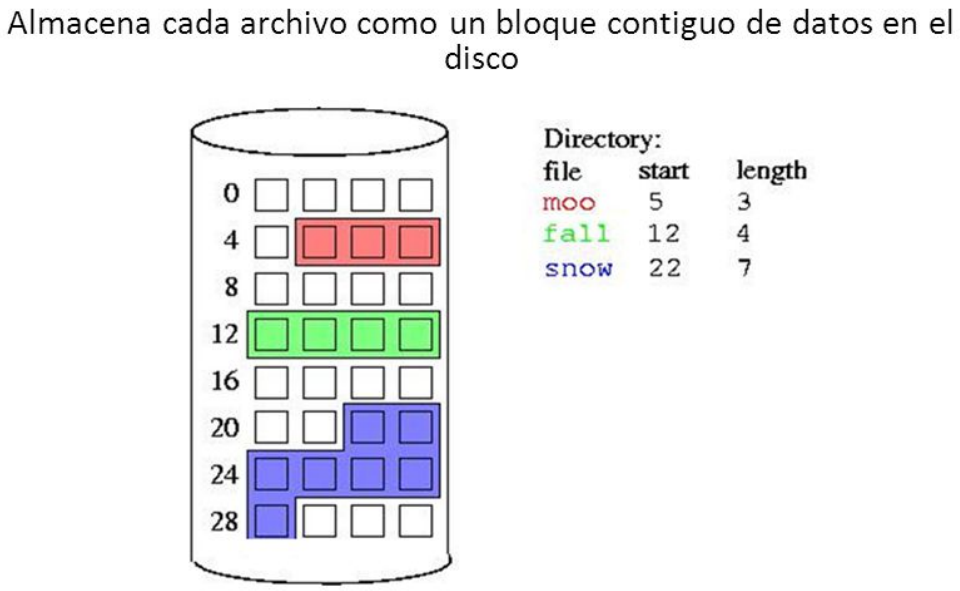
3.3.1 Asignación de espacio en disco

La forma estándar de almacenar los archivos es mediante la división en bloques de información esto facilita la gestión y la operativa. La asignación de disco puede llevarse a cabo mediante dos estrategias en función de si los bloques están contiguos en memoria o no:

- **Asignación contigua**: Toda la información del archivo se almacena en bloques contiguos en memoria de forma que la lectura se hace mediante bloques adyacentes. Es muy fácil de implementar

pero tiene el problema asociado de que los archivos no pueden crecer y no suele ser viable. Este motivo es el mismo por el que se dividieron en bloques en un inicio.

Asignación contigua (Contiguous Allocation)

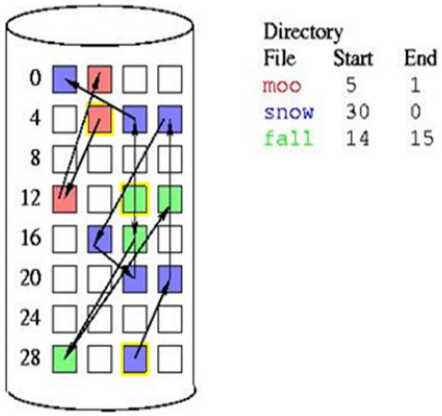


Asignación de bloques de disco de forma contigua.

- **Asignación no contigua o dispersa**: Almacena los distintos bloques en diferentes posiciones del disco y mantiene una referencia desde el archivo a los diferentes bloques que lo componen. Esto soluciona el problema anterior y existen diferentes estrategias de implementación:
    - **Asignación mediante listas enlazadas**: En este caso cada bloque reserva un espacio de 4 Bytes para almacenar un puntero al siguiente bloque creando así una lista. Esta implementación presenta dos problemas graves principales. En primer lugar, el bloque deja de ser potencia entera de dos lo que a la larga produce una gran pérdida de eficiencia en sistemas informáticos (más fallos de páginas, fallos caché 2 accesos a DMA, etc). En segundo lugar, el acceso aleatorio se vuelve extremadamente lento porque para acceder al bloque N se debe acceder al primer bloque y recorrer n-1 bloques.

2 Asignación por lista enlazada (Linked Allocation)

Guardar cada uno como una lista enlazada de bloques de disco. La primera palabra de cada bloque se emplea como apuntador al siguiente. El resto del bloque se destina a datos

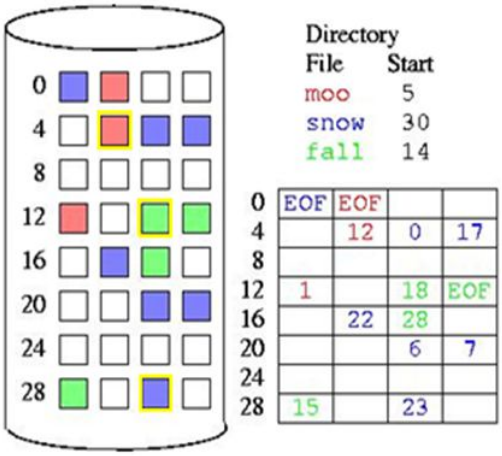


Asignación de bloques de disco mediante lista enlazada.

- **Asignación indexada**: Es una mejora de la lista enlazada en la que se almacena una tabla de memoria con una entrada para cada bloque en la que se almacena el puntero al siguiente bloque del fichero. De esta forma se solucionan los dos problemas anteriores. El problema principal de esta solución es que desperdicia mucho espacio de memoria para mantener la tabla, por ejemplo un disco de 200GB con bloques de 1K serían 800MG de memoria principal. Además es necesario tener toda la tabla en RAM aunque se vaya a utilizar solo uno de los ficheros. Es el sistema que utiliza FAT de MS-DOS y posteriormente Windows, aunque actualmente se ha enriquecido el sistema para solventar estos problemas.

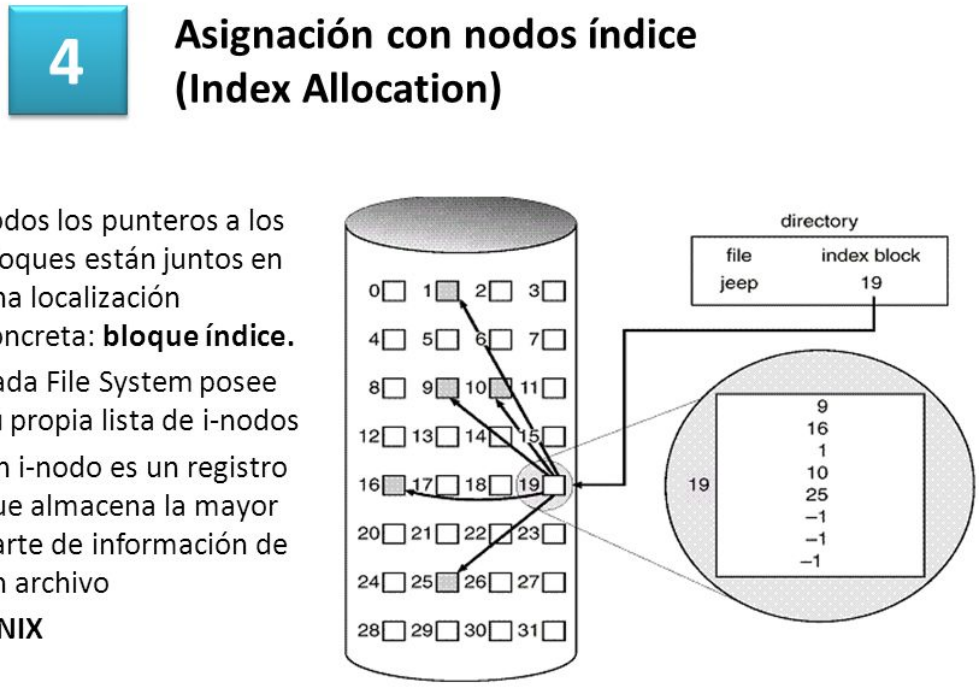
3 Asignación por lista enlazada usando tabla (Linked Allocation using file allocation table (FAT))

Se toma la palabra de apuntador de cada bloque y se le coloca en una tabla o índice en la **memoria Principal (caché).** La cadena está por completo en la memoria, y puede seguirse sin tener que consultar el disco. ***Windows* y *OS/2***



Asignación de bloques de disco mediante lista enlazada usando tabla indexada.

- **Asignación por nodos-i**: Los nodos índice son una mejora de la asignación de lista indexada en la que la información de los siguientes bloques se guarda en unos nodos ubicados en diferentes lugares, de esta forma, no hay que mantener toda la tabla en memoria principal, solo hay que mantener el bloque correspondiente al nodo-i del fichero abierto. El nodo-i también almacena la información referente al fichero. Es el sistema de asignación utilizado en sistemas UNIX.



Asignación de bloques de disco mediante nodos índice.

3.3.2 Gestión de espacio libre

Para un correcto uso de la memoria es necesario también tener un control del espacio libre para realizar las asignaciones. Existen dos estrategias principales:

- **Lista enlazada de bloques a disco**: Consiste en almacenar una lista de todos los huecos libres en memoria y cuando se precisa uno se accede al primero de la lista. Es un método bastante bueno dado que los bloques siempre tienen el mismo tamaño.
- **Mapa de bits**: Se almacena un mapa de bits donde los espacios libres están identificados por un 1 y en caso contrario por un cero. Este método es bastante bueno porque ocupa poco espacio y es fácil encontrar bloques libres contiguos. Sin embargo, a medida que se va llenando el disco es más eficiente el método de listas enlazadas.

3.4 Tipos de sistemas de ficheros

3.4.1 Sistemas de ficheros en disco

Son los implementados sobre los discos de la máquina y que dependen directamente del sistema operativo. Los más importantes son:

- **FAT**: Es el sistema tradicionalmente usado en Windows, a día de hoy se utiliza en unidades extraíbles no ópticas. Se basa en la división del espacio del disco en clusters y la creación de una tabla donde para cada fichero se almacenan los clusters utilizados. Esta tabla se almacena en las primeras posiciones del disco y es necesario cargarla toda en memoria principal para poder usarla. Es un sistema muy sencillo pero que desperdicia mucha memoria principal dado que hace falta

acceder a toda la tabla aunque solo se quiera usar un fichero. Existen otras versiones de FAT que intentan solucionar estos u otros problemas como FAT12 FAT16 FAT32 FATX.

- **NTFS**: Sistema usado por los sistemas Windows en la actualidad, implementa mejoras frente a FAT como la inclusión de metadatos, listas de control de acceso, etc. Se mantiene gracias a una tabla que almacena todo el sistema de ficheros con una estructura de árboles B+ junto a toda la metadata. Es un sistema mucho mejor que FAT pero la tabla principal ocupa mucho y hace que no sea recomendable para sistemas pequeños dado que desperdiciaría memoria de más.
- **HFS**: Es el sistema de ficheros de OSX, muy similar a NTFS almacenando un catálogo de árboles B+ aunque tenía el problema de que la tabla era de acceso exclusivo y no existía el acceso simultáneo entre procesos. Este sistema evoluciono posteriormente a HFS+ que corregía este error e incluía otras mejoras como soporte de UNICODE y soporte para ficheros con nombres más largos.
- **EXT**: Es el sistema de ficheros de los sistemas LINUX. Al igual que los anteriores se basa en una estructura donde guardan información de los archivos en nodos-i y además los directorios también se guardan como un fichero con información especial. Existen diferentes implementaciones ext3 y ext4.
- **F2FS**: Flash-friendly file system, es un sistema destinado a mejorar la eficiencia de las memorias flash. Muy típico en sistemas android y normalmente es el que implementan la gran mayoría de smartphones.

3.4.2 Sistemas de ficheros en red

Son sistemas de ficheros diseñados para poder acceder a través de internet al los sistemas de almacenamiento. Estos sistemas se implementan como una capa transparente de tal forma de que para el ordenador el acceso funciona de la misma forma que si accediese a una partición propia. Existen diferentes tipos de sistemas de ficheros en red.

- **Sistemas distribuidos**: Estos sistemas no permiten la E/S en paralelo y funcionan como otros sistemas de ficheros pero a través de internet. Los más comunes son:
  - **NFS**: Network file system, diseñado para sistemas distribuidos en una red local posibilitando que distintos ordenadores a través de la misma red puedan acceder a los distintos espacios como si fuesen carpetas locales.
  - **HDFS** (*Haddop distributed file system*): Es un sistema de ficheros usado en entornos big data que gestiona miles de nodos y PetaBytes de datos. Es un sistema muy tolerante a fallos y que implementan la tecnología map reduce para buscar a través de todos los diferentes nodos. Un ejemplo es la implementación de Amazon S3
- **Sistemas de archivos paralelos**: Son sistemas de archivos menos habituales, que permiten el acceso concurrente de diferentes procesos o hilos. Están diseñados para la implementación de aplicaciones con programación paralela como por ejemplo MPI y que se usan sobre todo en entornos de altas prestaciones.

3.4.3 Sistemas de ficheros de propósito especial

Son los sistemas de ficheros diseñados para implementar alguna aplicación o protocolo particular y específica como son los sistemas de ficheros de ftp (ftpfs), de acceso a pistas de audio (cdfs) o de control de dispositivos (devfs).

4. Bases de datos

una base de datos es un conjunto de datos estructurados que pertenecen a un mismo contexto y, en cuanto a su función, se utiliza para administrar de forma electrónica grandes cantidades de información. En este sentido; una biblioteca puede considerarse una base de datos compuesta en su mayoría por documentos y textos impresos en papel e indexados para su consulta. Actualmente, y debido al desarrollo tecnológico de campos como la informática y la electrónica, la mayoría de las bases de datos están en formato digital.

La construcción de una base de datos tiene la información almacenada en ficheros junto una descripción a la que llamaremos **metadatos** que permiten operar con la información de una forma mucho más eficiente.

Las bases de datos más comunes son las bases de datos relacionales, que están compuestas de tablas y relaciones entre las tablas. Además, emplean igualmente ficheros para almacenar la información, pero hay una diferencia fundamental entre los sistemas de ficheros y los sistemas de bases de datos; en los sistemas de ficheros, las aplicaciones acceden directamente a los ficheros, por lo que deben conocer su estructura, contenido, y por tanto son muy dependientes de los ficheros, mientras que en las bases de datos las aplicaciones no se conectan a los ficheros, sino que habrá un software intermedio que se encargará de gestionar las conexiones y peticiones de las aplicaciones, acceder a los ficheros a buscar la información, y enviársela a quien la solicitase. Este software intermedio es lo que se denomina el Sistema Gestor de Bases de Datos (SGBD).

4.1 Elementos básicos de una base de datos

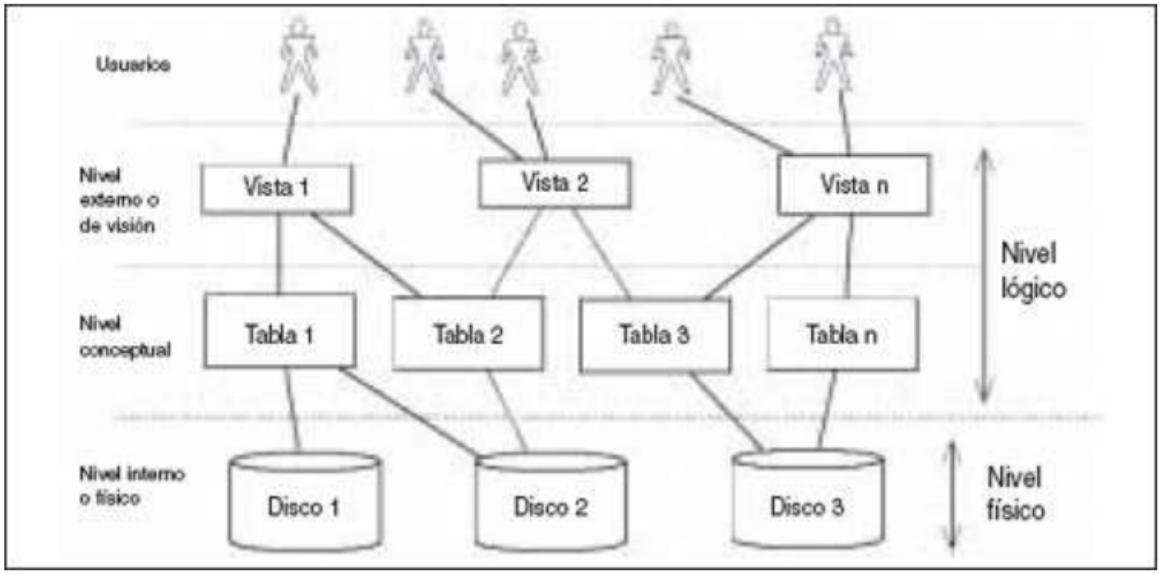
- **Tabla**: se refiere al tipo de modelado de datos donde se guardan los datos recogidos por un programa. Su estructura general se asemeja a la vista general de un programa de tablas.
- **Registro o fila**: Corresponde a cada fila que compone la tabla. Ahí se componen los datos y los registros. Eventualmente pueden ser nulos en su almacenamiento.
- **Campo o atributo**: Corresponde al nombre de la columna. Debe ser único y además de tener un tipo de dato asociado.
- **Dato**: Es el elemento almacenado en una fila y una columna.
- **Tipo de dato**: Se refiere a la codificación utilizada para almacenar los datos de una columna.
- **Clave primaria**: Restricción de base de datos en una columna que permite identificar de forma inequívoca los registros de esa tabla.
- **Índice**: es una estructura de datos que mejora la velocidad de las operaciones, por medio de identificadores y algoritmos más eficientes.
- **Vista**: En una base de datos, una vista es el conjunto de resultados de una consulta almacenada en los datos. Es una consulta que se presenta como una tabla (virtual) a partir de un conjunto de tablas en una base de datos relacional.
- **Script**: Es un conjunto de sentencias en un lenguaje de manipulación de datos para automatizar tareas.
- **Procedimiento**: Un procedimiento almacenado (stored procedure en inglés) es un programa almacenado físicamente en una base de datos. Su implementación varía de un gestor de bases de datos a otro. La ventaja de un procedimiento almacenado es que al ser ejecutado, en respuesta a una petición de usuario, es ejecutado directamente en el motor de bases de datos, el cual usualmente corre en un servidor separado

4.2 Niveles de abstracción una base de datos

Una de las principales funciones del SGBD es proveer una capa de abstracción entre la información y las aplicaciones que permita gestionar los datos de forma independiente a los programas que acceden a ellos. Para conseguir esta abstracción, se utiliza una arquitectura en 3 niveles, definida en 1975 por la ANSI-SPARC (American National Standard Institute – Standards Plannings and Requeriments Committee) para conseguir la separación entre aplicaciones y datos, el manejo de múltiples vistas y el uso de catálogos para el almacenamiento de los esquemas de las BD.

Los tres niveles en que se organiza el modelo son los siguientes:

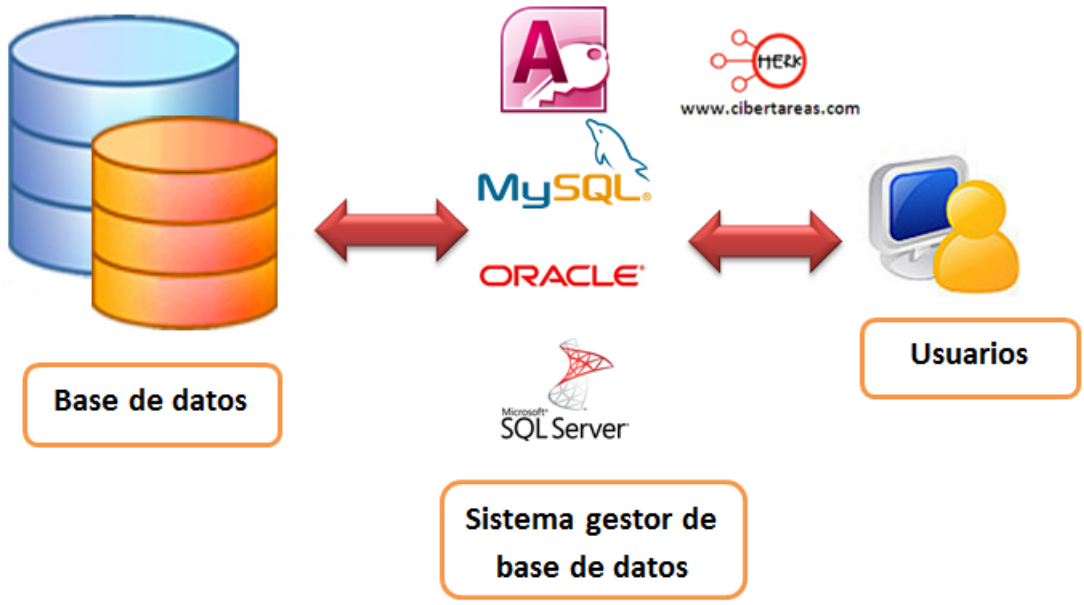
- **interno o físico**: el más cercano al almacenamiento físico. Determina cómo se va a guardar la información. Define la estructura de la BD mediante un esquema interno, donde se definirán los ficheros donde se guardará cada componente de la BD (datafiles), índices, estructura física de las tablas (pctfree, tamaño del bloque, …).
- **conceptual**: determina cómo se va a organizar la información. Define la estructura de la BD mediante un esquema conceptual. Define las entidades, atributos, relaciones, etc
- **externo o de visión**: determina cómo se va a mostrar la información a los usuarios. Define los usuarios de BDs y lo que ve cada uno de ellos (sinónimos, vistas, …)



Esta arquitectura en 3 niveles garantiza dos tipos de independencias:

- **Independencia lógica**: capacidad de modificar el esquema conceptual sin tener que alterar los esquemas externos ni las aplicaciones.
- **Independencia física**: la capacidad de modificar el esquema interno sin tener que alterar ni el esquema conceptual, ni los externos (ej: reorganizar los archivos físicos con el fin de mejorar el rendimiento de las operaciones de consulta o de actualización, o se pueden añadir nuevos archivos de datos porque los que había se han llenado). La independencia física es más fácil de conseguir que la lógica.

5. Sistemas gestores de bases de datos



Un sistema gestor de base de datos (SGBD) es un conjunto de programas cuya finalidad es establecer la interfaz necesaria entre los diferentes tipos de usuarios y la base de datos, permitiendo el almacenamiento, modificación y extracción de la información en una base de datos, además de proporcionar herramientas para explotar, administrar y gestionar las bases de datos.

En estos sistemas de gestión de archivos, la definición de los datos se encuentra codificada dentro de los programas de aplicación en lugar de almacenarse de forma independiente, y además son los programas quienes deben controlar el acceso y la manipulación de los datos. Los sistemas gestores de bases de datos aparecen con el objetivo de solucionar todos estos problemas que presentaban los sistemas de ficheros y añadir nuevas funcionalidades a mayores.

5.1 Funciones

Las principales funciones que debe proveer un SGBD son las siguientes:

- **Definición de datos**: Mediante el Lenguaje de Definición de datos (DDL) el S.G.B.D. permite describir y definir los esquemas de la base de datos. Este lenguaje debe permitir la creación de objetos y su descripción, la modificación y el borrado de los objetos existentes. Un DDL está compuesto por un conjunto de comandos que actúan sobre los objetos. El conjunto de las descripciones de objetos de una base de datos se le conoce con el nombre de diccionario de datos.
- **Manipulación de datos**: La función de manipulación de datos se encarga de todas las operaciones de gestión de los de la base de datos. Esta función se hace con la ayuda del Lenguaje de Manipulación de datos (DML), que está compuesto por un conjunto de comandos que permiten la consulta o actualización (inserción, modificación y borrado) de los datos de una base de datos. Cada SGBD tiene sus propios DDL y DML, que funcionan de forma diferente. Por ejemplo, los modelos en red y jerárquicos utilizan DMLs procedimentales (es decir el programador debe indicar el camino a seguir para acceder a los datos solicitados), mientras que en el modelo relacional se utiliza un lenguaje declarativo, SQL, con el que únicamente es necesario indicar qué datos se quieren, y no la forma de obtenerlos.
- **Seguridad e integridad de los datos**: garantizar la coherencia de los datos, comprobando que sólo los usuarios autorizados puedan efectuar las operaciones correctas sobre la base de datos. Esto se consigue mediante:

    - Gestión de autorizaciones y permisos: control sobre los usuarios que acceden a la base de datos y los tipos de operaciones que están autorizados a realizar.
    - Restricciones de integridad: validación de las operaciones realizadas con los datos.
    - Protección contra accesos malintencionados y fallos: los accesos malintencionados se suelen evitar por medio de la autentificación de usuarios, la definición de vistas y la protección física de los datos (encriptado de los datos). Con respecto a los fallos causados por manipulaciones incorrectas, o accidentes lógicos o físicos, los S.G.B.D. suelen disponer de utilidades de recuperación de lo datos después de un fallo.
- **Gestión de acceso concurrente**: Los SGBDs deben ser capaces de habilitar el acceso a los datos simultáneamente para gran número de usuarios, por lo que deben ser capaces de gestionar el acceso concurrente de estos usuarios asegurando la consistencia de los datos.
- **Gestión de las transacciones**: Una transacción se define como una unidad lógica de tratamiento (conjunto de órdenes) que aplicada a un estado coherente de la base de datos la deja, de nuevo, en un estado coherente, después de hacer las modificaciones. Una transacción solo se puede ejecutar completamente o ser anulada. La gestión de transacciones debe asegurar que la ejecución simultánea de transacciones por parte de diferentes usuarios da el mismo resultado que una ejecución secuencial. Esto se consigue habitualmente por medio de la utilización de bloqueos. Si bien esta función es muy importante en los sistemas relacionales, pero no en todos los SGBD. En los sistemas de bases de datos NoSQL, ésta característica no solo no es prioritaria sino que algunos ni siquiera la implementan, ya que el objetivo principal de estos sistemas se centra en el rendimiento y la disponibilidad de datos en tiempo real, para lo que sacrifica características como la atomicidad o la consistencia.
- **Auditoria**: Para el administrador de la base de datos es muy importante conocer quien accede a la base de datos y que operaciones realiza, información que debe proporcionar el SGBD
- **Garantizar un tiempo de respuesta idóneo**: para el diálogo entre los usuarios y la base de datos en los procesos en línea.
- **Abstracción de la información**: La arquitectura de las BDs creadas por medio de los SGBDs sigue el modelo en tres niveles definido por ANSI-SPARC. En este modelo se definen tres niveles para una base de datos: interno (indica cómo se almacenará la información, por medio del esquema interno de la BD), conceptual (cómo se organizará la información, por medio del esquema conceptual de la BD) y externo (cómo se mostrará la información a los usuarios, utilizando un esquema externo por usuario). La incorporación de estos tres niveles permite abstraer al usuario los detalles acerca del almacenamiento físico de los datos o su organización.
- **Independencia física y lógica**: La forma de almacenar los datos (esquema físico), no debe afectar a cómo se ha definido su estructura (esquema lógico), y la modificación de su esquema lógico no debe afectar a las aplicaciones que hacen uso de la BD (que hacen uso del esquema externo)
- Monitorización de la BD y elaboración de estadísticas
- **Conectividad**: proveen de herramientas de conectividad como los drivers odbc o jdbc que facilitan el acceso de las aplicaciones a los datos.
- **Respaldo y recuperación**: Incorporan herramientas para la realización de copias de seguridad y restauración, y para la recuperación ante caídas del sistema.

5.2 Tipos de SGBDs

5.2.1 Por en número de usuarios.

- **Monousuarios**: sistemas de un solo usuario. Estos sistemas son muy básicos y carecen de muchas de las posibilidades de los sistemas multiusuario (monitorización, auditoría, gestión de copias de seguridad, etc), pero ofrecen algunas ventajas como una mayor seguridad de los datos y no necesitan realizar un control de concurrencia.
- **Multiusuarios**: Un sistema multiusuario se encarga de dar servicio a un gran número de usuarios que están conectados al sistema a través de terminales. Habitualmente estos sistemas disponen de algún mecanismo para priorizar tareas y controlar el uso que los usuarios hacen del sistema, pudiendo limitarlo para que no copen los recursos.

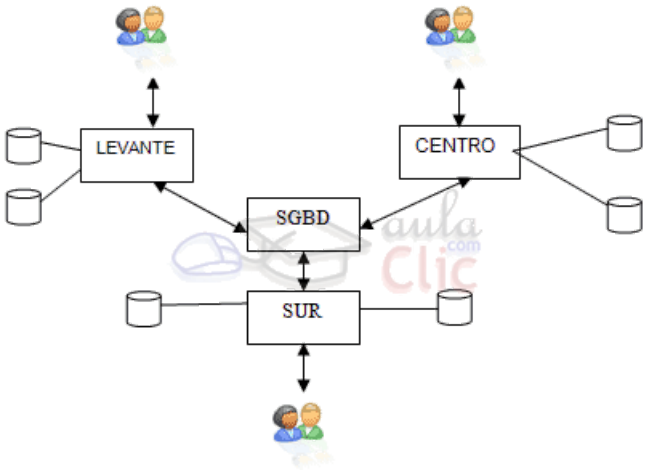
5.2.2 Por la localización de la información

- **Centralizados**: Los sistemas de bases de datos centralizados son aquellos que se ejecutan en un único sistema informático, que gestiona todos los recursos. Esto no implica que sea una máquina física, sino que aunque físicamente el sistema conste de un clúster de servidores, lógicamente se verá como un solo servidor. Estos sistemas tienen varias ventajas: evitan la redundancia (normalmente en los sistemas distribuidos cada nodo maneja sus propios maestros locales), evitan inconsistencias (al tener una sola versión de los datos), es más sencillo aplicar restricciones de seguridad. Entre sus inconvenientes: las prestaciones del servidor central tienen que ser mucho más elevadas que en servidores distribuidos, por lo que económicamente suele ser más costoso, en caso de fallo del sistema no existe posibilidad de acceder a los datos, y son poco escalables

Sistema gestor de bases de datos centralizado.

- **Distribuidos**: Un sistema de bases de datos distribuido es un sistema en el cual múltiples nodos de bases de datos están ligados por un sistema de comunicaciones, de tal forma que un usuario en cualquier nodo puede acceder a los datos en cualquier parte de la red exactamente como si los datos estuvieran almacenados en su propio nodo. Cada nodo es autónomo, y no hay dependencia de un

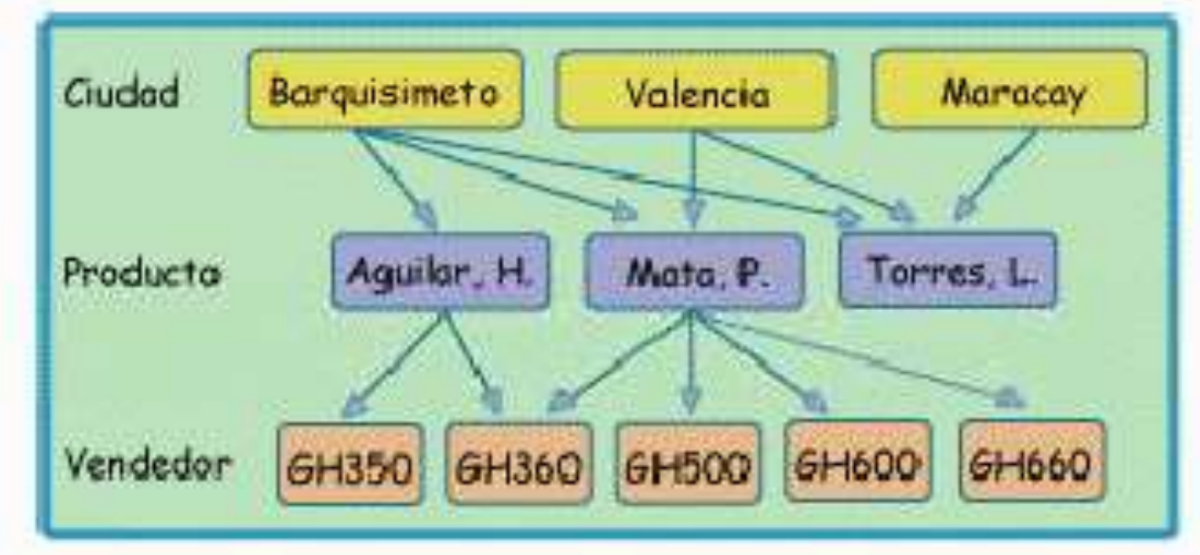
nodo central. La operación de los nodos es continua (no es necesario parar los nodos para realizar operaciones como añadir nuevos nodos a la red o instalar nuevas versiones), y existe una independencia de localización (los usuarios no tienen que saber dónde están almacenados los datos a los que están accediendo). Entre las ventajas de los sistemas distribuidos: son más adaptables a una estructura organizacional, tienen autonomía local (cada departamento controla sus propios datos), tienen mayor disponibilidad (no dependen de un solo nodo), son sistemas más baratos que los centralizados y son fácilmente escalables.



Sistema gestor de base de datos distribuido.

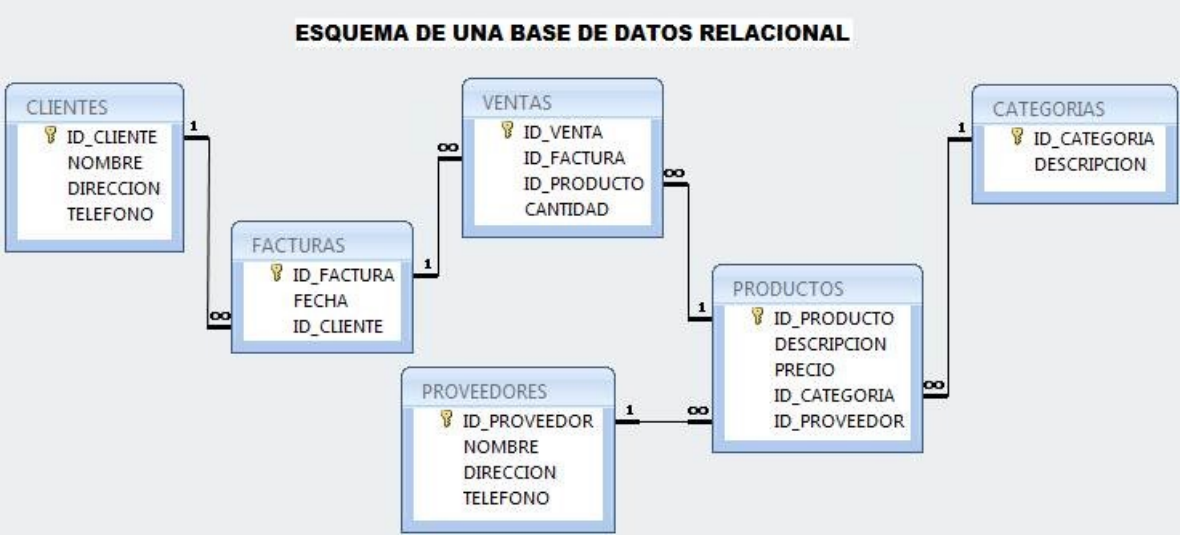
5.2.3 Por su estructura

- **Sistemas navegacionales**: A mediados de los 60 fueron apareciendo los primeros sistemas de bases de datos de propósito general. En 1971 se definió el estándar de CODASYL (el grupo responsable de la creación y estandarización de COBOL), y en breve aparecieron algunos productos basados en esta línea. La estrategia de CODASYL estaba basada en la navegación manual por un conjunto de datos enlazados en red. Cuando se arrancaba la base de datos, el programa devolvía un enlace al primer registro de la base de datos, el cual a su vez contenía punteros a otros datos. Para encontrar un registro concreto el programador debía ir siguiendo punteros hasta llegar al registro buscado. Este estándar se implementó tanto en sistemas en árbol como es sistemas en red, y evolucionó incorporando incluso un sistema de consulta, pero todos estos sistemas resultaban muy complejos y requerían de mucho esfuerzo y práctica para producir una aplicación útil.



Sistema navegacional

- **Sistemas relacionales**: A principios de los años 70 surge el modelo relacional, planteado por Edgar Codd. Este modelo basado en la utilización de tablas de registros de tamaño fijo relacionadas entre ellas. De esta manera se solucionaba el problema de mantenibilidad de las relaciones del modelo anterior, ya que las relaciones se implementarán como nuevas tablas y se definirán restricciones para asegurar la integridad de los datos.



Modelo relacional gráfico.

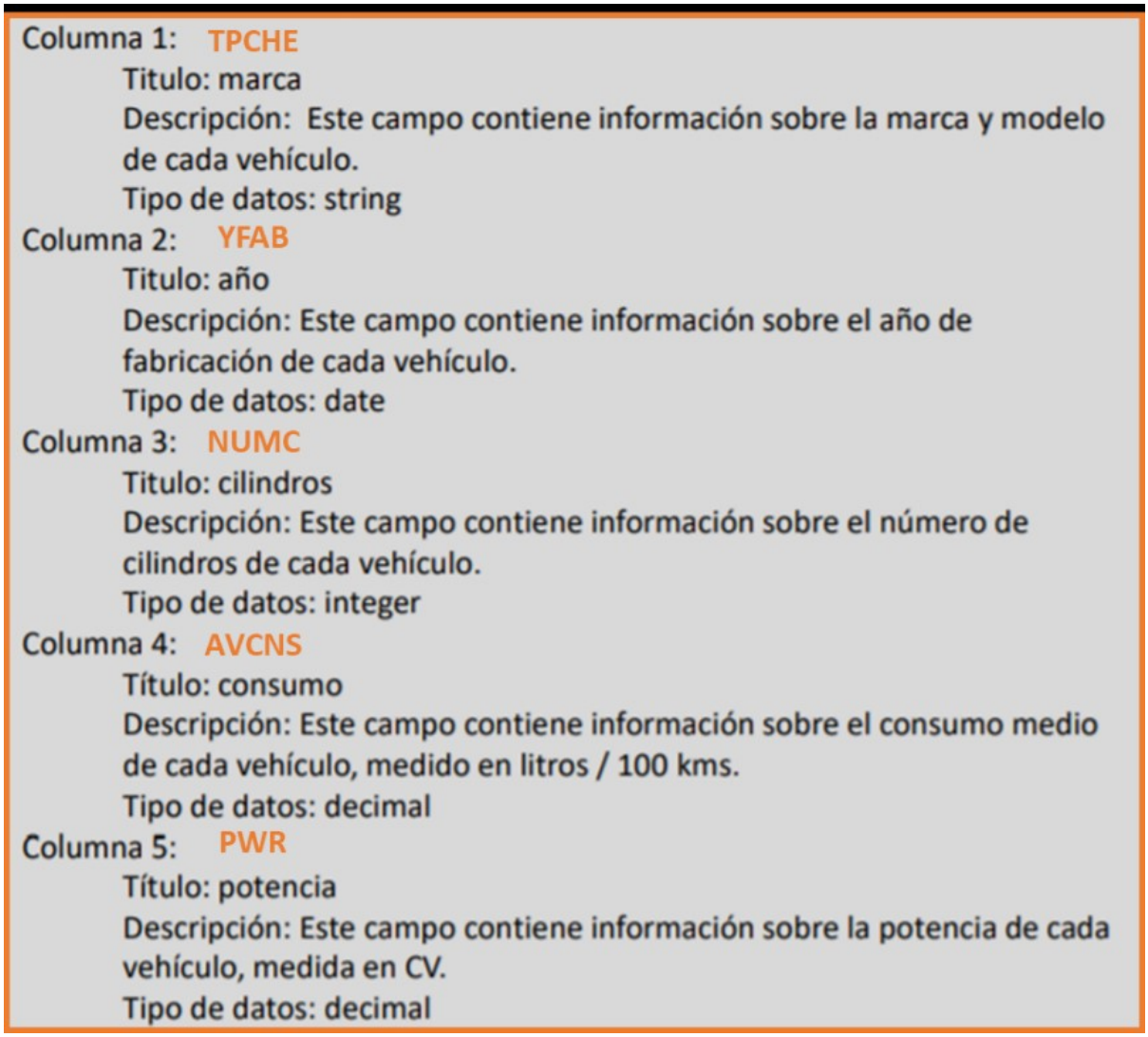
A lo largo de la década de los 70 se crearon varias soluciones basadas en el modelo propuesto por Codd, pero estos sistemas tenían un problema: no estaban preparados para la consulta de la información almacenada de forma sencilla (los lenguajes tradicionales no estaban pensados para recuperar la información en un solo registro).

Para solventar este problema Codd propuso una solución basada en un lenguaje orientado a conjuntos, que más tarde cristalizaría en el SQL. Entre los SGBDs más importantes de este tipo destacan Informix, Oracle, SQL Server o MySQL.

- **Sistemas orientados a objetos**: Con el auge de la programación orientada a objetos aparece el modelo orientado a objetos, que permite establecer relaciones entre objetos y atributos, en lugar de hacerlo entre campos individuales. Estos sistemas tienen una integración transparente con los programas escritos en un lenguaje OO, por lo que facilitan la persistencia de los objetos sin la necesidad de una capa adicional de abstracción para gestionar el almacenamiento como sucede con las bases de datos relacionales. Son de este tipo los SGBDs ObjecStore y O2.
- **Sistemas NoSQL**: Los sistemas NoSQL surgen para paliar dos de las problemáticas fundamentales de los sistemas relacionales: los problemas de escalado y la poca flexibilidad para el tratamiento de información no estructurada. Estos sistemas no requieren por lo general esquemas fijos, evitan las operaciones join almacenando datos desnormalizados y están diseñadas para escalar horizontalmente tanto como sea necesario. Estos sistemas están basados en el uso de la técnica “Map reduce”. Realizan mucho trabajo en memoria, y poco en disco, por lo que son muy rápidos pero menos consistentes que los sistemas relacionales. Este tipo de sistemas son los más adecuados para dar soporte a los sistemas big-data. Existen 4 tipos de bases de datos NoSQL: orientadas a documentos (MongoDB), columnares (HBase de Facebook o Vertica de HP), de clave- valor (DynamoDB de Amazon) y de grafos (Neo4j de Cisco)

5.3 Elementos de un SGBD

- **Lenguaje** -> SQL (structured query language). 4 componentes:
  - DDL (lenguaje de definición de datos: create table, alter table, …)
  - DML (lenguaje de manipulación de datos: select, insert, update)
  - DCL (lenguaje de control de datos: grant, revoke)
  - TCL (lenguaje de control de transacciones: commit, rollback)
  - Lenguajes de 4º generación: lenguajes de programación que permiten definir programas sobre el SGBD
- **Diccionario de datos**: Los SGBD suelen guardar los esquemas en una BD propia para ello, de forma que se puede consultar la información del esquema por medio de consultas SQL.



Ejemplo de diccionario de datos.

- **Optimizador de consultas**: determina la estrategia óptima para la ejecución de las consultas.
- **Gestor de las transacciones**: El gestor de transacciones debe asegurar que las operaciones que forman parte de una transacción se ejecutan completamente, o en caso de error que se deshagan los cambios realizados previamente.
- **Gestor de acceso concurrente**: Los SGBDs deben ser capaces de habilitar el acceso a los datos simultáneamente para gran número de usuarios, por lo que deben ser capaces de gestionar el acceso concurrente de estos usuarios asegurando la consistencia de los datos. Esto se consigue habitualmente por medio de la utilización de bloqueos.
- **Planificador (scheduler)**: para programar y automatizar la realización de ciertas operaciones y procesos.
- **Copias de seguridad**: para garantizar que la base de datos se puede devolver a un estado consistente en caso de que se produzca algún fallo o error grave.
- **Otras herramientas**:
  - Seguridad: control de acceso de los usuarios, encriptación de los datos, auditoría, etc.
  - Integridad: que mantiene la integridad y la consistencia de los datos.

- Control de recuperación: que restablece la base de datos después de que se produzca un fallo.
- Programación de aplicaciones.
- Distribución de datos.