Tema 1

SISTEMAS DE ALMACENAMIENTO DE LA INFORMACIÓN

IES Francisco Romero Vargas
Departamento de Informática

1. Gestión de datos

En el mundo actual existe una cada vez mayor demanda de datos y, por tanto, más necesidad de gestionarlos. Esta demanda siempre ha sido patente en empresas y sociedades, pero en estos años la demanda todavía de ha disparado más debido al acceso multitudinario a las redes integradas en Internet y a la aparición de pequeños dispositivos (móviles y PDA) que también requieren esa información.

En informática se conoce como **dato** a cualquier elemento informativo que tenga relevancia para un usuario. Desde su nacimiento, la informática se ha encargado de proporcionar herramientas que faciliten la manipulación de los datos.

Antes de la aparición de las aplicaciones informáticas, las empresas tenían como únicas herramientas de gestión de datos a los cajones, carpetas y fichas en las que se almacenaban los datos. En este proceso manual, el tipo requerido para manipular estos datos eran enormes. Pero la propia informática ha adaptado sus herramientas para que los elementos que el usuario utiliza en cuanto a manejo de datos se parezcan a los manuales. Por eso en informática se sigue hablado de ficheros, formularios, carpetas, directorios,.... De esta forma, la clientela fundamental del profesional de la informática es la empresa. La empresa se puede entender como un sistema formado por diversos objetos: el capital, los recursos humanos, los inmuebles, los servicios que presta, etc. El sistema completo que forma la empresa se suele dividir en los siguientes subsistemas:

- **Subsistema productivo.** También llamado subsistema real o físico. Representa la parte de la empresa encargada de gestionar la producción de la misma.
- **Subsistema financiero.** Encargado de la gestión de los bienes económicos de la empresa.
- **Subsistema directivo.** Encargado de la gestión organizativa de la empresa.

Son los sistemas de información los encargados de manejar la gran cantidad de información que maneja cada uno de estos subsistemas empresariales.

Los sistemas de información actuales se basan en bases de datos y sistemas de bases de datos, que se han convertido en elementos imprescindibles de la vida cotidiana de la sociedad moderna.

Cada día, la mayoría de nosotros nos encontramos con actividades que requieren algún tipo de interacción con una base de datos (ingreso en un banco, reserva de una entrada para el teatro, solicitud de una suscripción a una revista, compra de productos, ...). Estas interacciones son ejemplos de lo que se llama aplicaciones tradicionales de bases de datos (básicamente información numérica o de texto), aunque los avances tecnológicos han permitido que también existan: *bases de datos multimedia*, *sistemas de información geográfica* (GIS), *almacenes de datos*, *sistemas de proceso analítico on-line*, ...

Sin embargo, para entender los fundamentos de las tecnologías de bases de datos debemos empezar desde las bases de las aplicaciones tradicionales.

Una base de datos será, por tanto, una colección de datos relacionados. Por datos queremos decir hechos conocidos que pueden registrarse y que tienen un significado implícito. Una agenda con los nombres y teléfonos de un conjunto de personas conocidas es una base de datos, puesto que es una colección de datos relacionados con un significado implícito.

2. Sistemas de información

Según la RAE un **Sistema** es un "Conjunto de reglas o principios sobre una materia racionalmente enlazados entre sí" o bien un "Conjunto de cosas que relacionadas entre sí ordenadamente contribuyen a determinado objeto".

Los sistemas que aglutinan los elementos que intervienen para gestionar la información que manejan los subsistemas empresariales es lo que se conoce como ***Sistemas de Información***. Se suele utilizar las siglas **SI** o **IS** (de *Information System)* para referirse a ello.

Un sistema de información es un sistema dentro de la empresa que permite el uso y las transferencias de informaciones entre unos subsistemas y otros de la empresa.

Realmente un sistema de información sólo incluye la información que nos interesa de la empresa y los elementos necesarios para gestionar esa información.

Un sistema de información genérico está formado por los siguientes elementos:

- **Recursos físicos**. Carpetas, documentos, equipamiento, discos,...
- **Recursos humanos.** Personal que maneja la información.
- **Reglas**. Normas que debe cumplir la información para que sea manejada (formato de la información, modelo para los documentos,...), es decir, las normas, métodos y protocolos determinados por la planificación de la empresa.

Las empresas necesitan implantar estos sistemas de información para obtener una mayor calidad en la organización de las actividades de los subsistemas empresariales.

Cuando parte o toda la gestión de un sistema de información se realiza con ordenadores se habla de sistema de información basado en ordenadores o **sistema informático**. En este caso a las reglas de la empresa se añaden las normas determinadas por el sistema operativo de los ordenadores y demás software instalado en ellos.

Componentes de un sistema informático

En el caso de una gestión electrónica de la información (lo que actualmente se considera un sistema de información), los componentes son:

- **Datos.** Se trata de la información relevante que almacena y gestiona el sistema de información. Esta información suele estar almacenada en bases de datos que cumplen con los requerimientos o normas de la empresa.
- **Hardware.** Equipamiento físico que se utiliza para gestionar los datos.
- **Software.** Aplicaciones que permiten el funcionamiento adecuado del sistema.
- **Recursos humanos.** Personal que maneja el sistema de información.

Tipos de sistemas informáticos

Según el propósito, los sistemas de información se pueden clasificar en:

- **Transaccionales.** Se ocupan de la automatización de las operaciones y transacciones que se realizan en la empresa, por ejemplo, las

actividades que realizan los empleados de manera cotidiana (fichar a la hora de entrada del trabajo, firmar, ...)

- **De Gestión.** Se ocupan de los datos que se manejan en la empresa y su almacenamiento. Por ejemplo controlan cómo se almacenan los datos en las bases de datos, cómo se recupera esa información, cómo se obtienen listados, informes, etc.

- **De soporte a la decisión.** Su misión es ayudar a los directivos y personal con responsabilidad dentro de la empresa en la toma de decisiones estratégicas. Estos sistemas están emparentados con las técnicas de inteligencia artificial y los sistemas expertos y son cada vez más comunes e importantes dentro de las empresas.

- **Ofimática, sistemas expertos y sistemas inteligentes** son otros tipos de sistemas de información informáticos.

Tipos de sistemas de información de gestión de datos

Según el enfoque dado a los datos, los sistemas de información pueden diferenciarse en orientados al proceso (sistema clásico de ficheros) u orientados a los datos (sistema de bases de datos).

- ### Sistemas de información orientados al proceso: FICHEROS

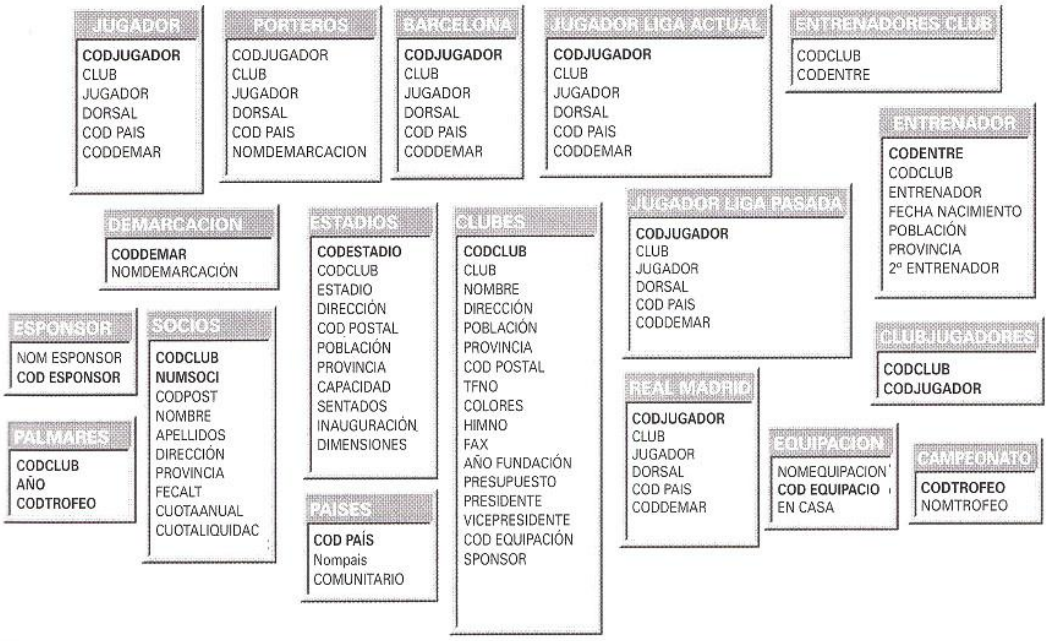
  En estos sistemas de información se crean diversas aplicaciones (software) para gestionar diferentes aspectos del sistema. Cada aplicación realiza unas determinadas operaciones.

  Los datos de dichas aplicaciones se almacenan en archivos digitales dentro de las unidades de almacenamiento del ordenador (a veces en archivos binarios, o en hojas de cálculo, ...).

  En estos sistemas, cada programa almacena y utiliza sus propios datos de forma un tanto caótica. La única ventaja que conlleva esto es que los procesos son independientes, por lo que la modificación de uno no afecta al resto. Pero tiene grandes inconvenientes:

- **Coste de almacenamiento elevado.** Al almacenarse varias veces el mismo dato en distintas apliocaciones, se requiere más espacio en los discos.
- **Datos redundantes.** Ya que se repiten continuamente.
- **Probabilidad alta de inconsistencia en los datos.** Ya que un proceso cambia sus datos y no el resto. Por lo que el mismo dato puede tener valores distintos según qué aplicación acceda a él.
- **Difícil modificación en los datos** Debido a la probabilidad de inconsistencia, que ocurre cuando se produce una pérdida o hay incoherencia de datos. Para que ésta no exista, cada modificación se debe repetir en todas las copias del dato (algo que normalmente es imposible).
- **Tiempos de procesamiento elevados**. Al no poder optimizar el espacio de almacenamiento.

En la siguiente figura se muestra un sistema de información basado en ficheros. En ella se ve que la información aparece inconexa y redundante.



- ## Sistemas de información orientados a los datos: BASES DE DATOS

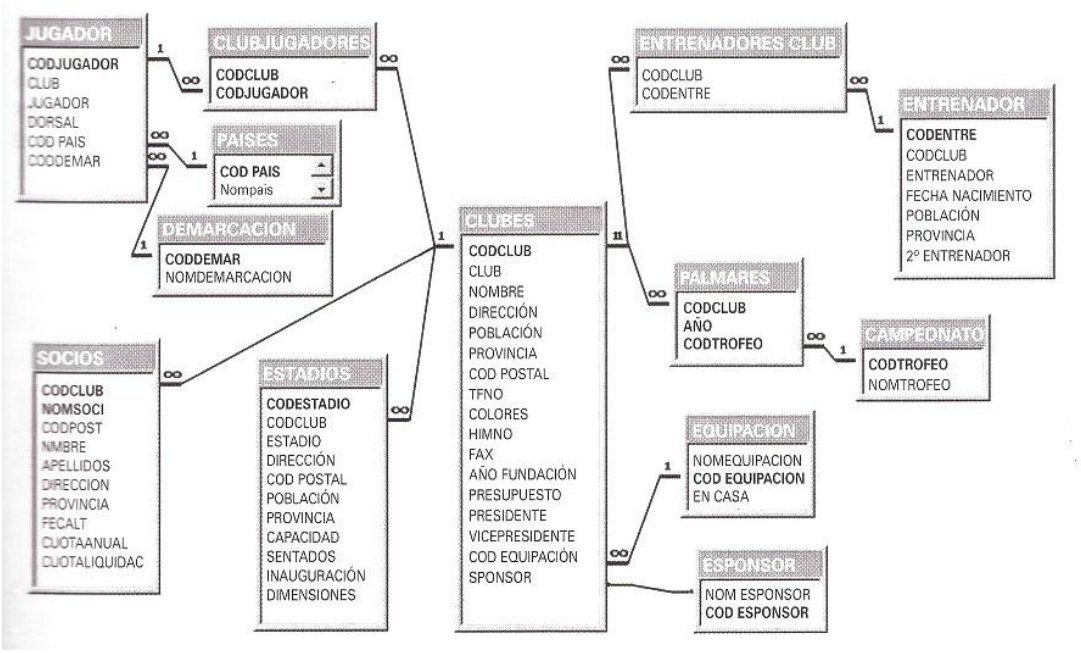
  En este tipo de sistemas los datos se centralizan en una **base de datos** común a todas las aplicaciones. Estos serán los sistemas que estudiaremos en este curso.

  En esos sistemas los datos se almacenan en una única estructura lógica que es utilizable por las aplicaciones. A través de esa estructura se accede a los datos que son comunes a todas las aplicaciones.

ventajas	inconvenientes
Independencia de los datos y los programas y procesos. Esto permite modificar los datos sin modificar el código de las aplicaciones. **Menor redundancia**. No hace falta tanta repetición de datos**.** Aunque, sólo los buenos diseños de datos tienen poca redundancia. **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. **Datos más documentados.** Gracias a los metadatos que permiten describir la información de la base de datos. **Acceso a los datos más eficiente.** La organización de los datos produce un resultado más óptimo en rendimiento. **Menor espacio de almacenamiento.** Gracias a una mejor estructuración de los	**Instalación costosa.** El control y administración de bases de datos requiere de un software y hardware poderoso **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. **Ausencia de estándares reales.** Lo cual significa una excesiva dependencia hacia los sistemas comerciales del mercado. Aunque hay una buena parte de esta tecnología aceptada como estándar de hecho.

datos.

En la siguiente figura se muestra un sistema de información basado en bases de datos. La información está relacionada y no es redundante.



3. Sistemas de ficheros

Un *sistema de ficheros* es un conjunto de programas que prestan servicio a los usuarios finales. Cada programa define y maneja sus propios datos.

Los sistemas de ficheros surgieron al tratar de informatizar el manejo de los archivadores manuales con objeto de proporcionar un acceso más eficiente a los datos.

En lugar de establecer un sistema centralizado en donde almacenar todos los datos de la organización o empresa, se escogió un modelo descentralizado en el que cada sección o departamento almacena y gestiona sus propios datos.

Tipos de operaciones sobre ficheros

Sobre los ficheros se pueden realizar las siguientes operaciones:

- **Abrir** (*open*). Prepara el fichero para su proceso.
- **Cerrar** *(close*). Cierra el fichero impidiendo su proceso inmediato.
- **Leer** *(read*). Obtiene información del fichero.
- **Escribir** *(write*). Graba información en el fichero.
- **Posicionarse** *(seek)*. Coloca el puntero de lectura en una posición concreta del mismo (no se puede realizar en todos los tipos de ficheros).
- **Detectar la marca de final de fichero** *(eof).* Indica si hemos llegado al final del fichero.

Tipos de ficheros según su estructura física

- **Ficheros secuenciales**
  En estos ficheros, los datos se organizan secuencialmente en el orden en el que fueron grabados. Para leer los últimos datos hay que leer los anteriores.

ventajas	inconvenientes
• Rápidos para obtener registros contiguos • No hay huecos en el archivo al grabarse los datos seguidos, datos más compactos	• Consultas muy lentas al tener que leer todos los datos anteriores al dato que queremos leer • Algoritmos de lectura y escritura más complejos • No se pueden eliminar registros del fichero (se pueden marcar de manera especial para que no sean tenidos en cuenta, pero no se pueden borrar) • La ordenación de los datos requiere volver a

	crearle de nuevo

- **Ficheros de acceso directo o aleatorio**

  Se puede leer una posición concreta del fichero, con saber la posición (normalmente en bytes) del dato a leer.

ventajas	inconvenientes
• Acceso rápido al no tener que leer los datos anteriores • Actualización más cómoda de programar • Se pueden borrar datos (aunque quedarán huecos) • Permiten acceso secuencial • Permiten leer y escribir a la vez	• En términos de base de datos no se utiliza la posición de los datos en bytes, sino respecto a una determinada clave. Por lo que habrá que convertir esa clave a bytes, lo que dificulta su manejo • No generan ficheros compactos ya que se crean huecos al borrar • Las consultas sobre multitud de registros son más lentas que en el caso anterior.

- **Ficheros secuenciales encadenados**

  Son ficheros secuenciales gestionados mediante punteros, datos especiales que contienen la dirección de cada registro del fichero. Cada registro posee ese puntero que indica la dirección del siguiente registro y que se puede modificar en cualquier momento.

ventajas	inconvenientes
• El fichero mantiene un determinado orden • La ordenación no requiere grabar nuevo fichero, sino modificar los punteros • Las mismas ventajas que	• No se borran los registros, sino que se marcan para ser ignorados. Por lo que se malgasta espacio • Más rápidos que los secuenciales, pero más

el acceso secuencial	lentos que los aleatorios

- **Ficheros secuenciales indexados**

  Se utilizan dos ficheros para los datos, uno posee los registros almacenados de forma secuencial, pero que permite su acceso aleatorio. El otro posee una tabla con punteros a la posición ordenada de los registros. Ese segundo fichero es el índice que es una tabla con la ordenación deseada para los registros y la posición que ocupan en el archivo.

  El archivo de índices posee unas cuantas entradas sólo en las que se indica la posición de ciertos valores claves en el archivo (cada 10, 15 ,20,... registros del archivo principal se añade una entrada en el de índices).

  El archivo principal tiene que estar siempre ordenado y así cuando se busca un registro, se busca su valor clave en la tabla de índices, la cual poseerá la posición del registro buscado. Desde esa posición se busca secuencialmente el registro hasta encontrarlo.

  Existe un archivo llamado de **desbordamiento** y **overflow** en el que se colocan los archivos que se van añadiendo los nuevos registros (para no tener que ordenar el archivo principal cada vez que se añade un nuevo registro) este archivo está desordenado. Se utiliza sólo si se busca un registro y no se encuentra en el archivo principal. En ese caso se recorre todo el archivo de overflow hasta encontrarlo.

  Para no tener demasiados archivos en overflow (lo que restaría velocidad), cada cierto tiempo se reorganiza el archivo principal.

ventajas	inconvenientes
• Se mantiene un orden concreto • La búsqueda de datos es rapidísima • Permite la lectura secuencial	• Para un uso óptimo hay que reorganizar el archivo principal y esta operación es muy costosa ya que hay que reescribir de nuevo y de forma ordenada todo el archivo. • La adición de registros

	requiere más tiempo que en los casos anteriores • No se pueden borrar los datos, se marcan para no ser leídos

- **Ficheros indexado-encadenados**

  Utiliza punteros e índices, es una variante encadenada del caso anterior. Hay un fichero de índices equivalente al comentado en el caso anterior y otro fichero de tipo encadenado con punteros a los siguientes registros. Cuando se añaden registros se añaden en un tercer registro llamado de desbordamiento u *overflow*. En ese archivo los datos se almacenan secuencialmente, se accede a ellos si se busca un dato y no se encuentra en la tabla de índices.

ventajas	inconvenientes
• Las mismas que los anteriores más la posibilidad de borrar registros (aunque se generan huecos) • La reorganización del archivo principal es más rápida al tener que cambiar sólo los punteros del encadenamiento	• Requieren compactar los datos a menudo para reorganizar índices y quitar el fichero de desbordamiento.

Tipos de ficheros según su uso

- **Permanentes.** Quedan grabados de forma permanente en los sistemas de disco.
- **Maestros.** Ficheros que se cambian muy poco a menudo. Son los principales en los sistemas de información. Contienen los datos fundamentales.
- **Constantes.** Contienen información que apenas varía a lo largo del tiempo.

- **Históricos.** Contienen los últimos cambios realizados sobre los datos. Después de un determinado tiempo esos cambios se llevan al fichero maestro.
- **De movimiento.** Almacenan cambios a realizar en los ficheros maestros. Se eliminan tras realizar esos cambios.
- **De maniobra.** Ficheros auxiliares utilizados por el software que gestiona los datos. Son destruidos en cuanto las aplicaciones finalizan.

4. Sistemas de Bases de Datos

Un **Sistema de Bases de Datos** es un sistema basado en ordenadores, cuyo propósito general es registrar y mantener datos mediante un sistema gestor de BD. Permite a los usuarios y aplicaciones la consulta y manipulación de estos datos - almacenados en bases de datos - usando un lenguaje de consulta estructurado: SQL (Structured Query Language).

Como se ha comentado anteriormente, cuando los datos de un sistema de información se almacenan en una única estructura, se llama base de datos. Se presentan a continuación dos definiciones:

Una **base de datos** es una **colección de datos almacenados en un soporte informático permanente de forma que sea posible obtener la relación entre los datos a través de un esquema conceptual que oculte la física real de los datos.**

Una base de datos es una **colección de datos estructurados según un modelo que refleje las relaciones y restricciones existentes en el mundo real. Los datos, que han de ser compartidos por diferentes usuarios y aplicaciones, deben mantenerse independientes de ésta, y su definición y descripción han de ser únicas estando almacenados junto a los mismos. Por último, los tratamientos que sufran estos datos tendrán que conservar la integridad y seguridad de éstos.**

El objetivo es que las aplicaciones puedan acceder a los datos sin necesidad de conocer exactamente cómo están almacenados los datos. Esto se consigue con un esquema conocido como **esquema conceptual**. Cualquier desarrollador que conozca ese esquema puede acceder a los datos desde cualquier aplicación.

Componentes de un sistema de base de datos

Los elementos de un sistema de base de datos son los mismos que los de un sistema de información. Se comentan a continuación para concretar un poco más la información que ya poseíamos:

- **Hardware.** Máquinas en las que se almacenan las bases de datos. Se compone de los volúmenes de almacenamiento secundario (discos, cintas, ...) donde reside la BD, junto con los dispositivos asociados como unidad de control, tarjetas, memoria, ...

- **Software.** Es el sistema gestor de bases de datos, es decir, la aplicación que permite el manejo de la base de datos. Entre la BD física y los usuarios del sistema, existe un nivel de software que recibe el nombre de ***SGBD***. Este maneja todas las solicitudes de acceso a la BD y registra y mantiene de forma controlada los datos almacenados. Asimismo pueden existir uno o varios programas para permitir elo acceso a las BD realizando consultas. Estas ***aplicaciones*** accederán a las BD usando el lenguaje de consultas SQL mediante los servicios ofrecidos por el SGBD.

- **Datos.** Incluyen los datos que se necesitan almacenar y los **metadatos** que son datos que sirven para describir lo que se almacena en la base de datos. Los datos almacenados en el sistema se dividen en una o más bases de datos. Una BD es ***integrada***, ya que puede considerarse como una unificación de varios archivos de datos independientes, donde se eliminan cualquier redundancia entre los mismos. Una BD es ***compartida***, en el sentido en que partes individuales de la BD pueden compartirse entre varios usuarios distintos, de tal forma que cada uno de ellos puede tener acceso a la misma parte de la BD. Hay que considerar que aunque dos usuarios compartan el mismo subconjunto de la BD, sus percepciones o vistas de ese conjunto pueden diferir mucho a nivel de detalle. La palabra "compartida" a menudo se amplia para abarcar no sólo al comportamiento entes descrito, sino también al acceso *concurrente*, es decir, la posibilidad de que varios usuarios accedan a la misma BD (tal vez incluso a la misma parte) al mismo tiempo. Un SBD que admite esta forma de acceso se llama ***Sistema de Usuarios Múltiples***.

- **Usuarios.** Personas que manipulan los datos del sistema. Hay cuatro grupos de personas que intervienen en el entorno de una base de datos: el administrador de la base de datos, los diseñadores de la base de datos, los programadores de aplicaciones y los usuarios.

- El ***administrador*** *de la base de datos* (ABD o DBA) se encarga del diseño físico de la base de datos y de su implementación, realiza el control de la seguridad y de la concurrencia, mantiene el sistema para que siempre se encuentre operativo y se encarga de que los usuarios y las aplicaciones obtengan buenas prestaciones. El administrador debe conocer muy bien el SGBD que se esté utilizando, así como el equipo informático sobre el que esté funcionando. El DBA, en resumen, autoriza el acceso a la BD, coordina y vigila su utilización, adquiere los recursos de software y hardware necesarios y es el responsable ante los problemas de violaciones de seguridad o respuesta lenta del sistema.

- Los **analistas o d*iseñadores*** *de la base de datos* realizan el diseño lógico de la base de datos, debiendo identificar los datos, las relaciones entre datos y las restricciones sobre los datos y sus relaciones. El diseñador de la base de datos debe tener un profundo conocimiento de los datos de la empresa y también debe conocer sus *reglas de negocio*. Las reglas de negocio describen las características principales de los datos tal y como las ve la empresa. Para obtener un buen resultado, el diseñador de la base de datos debe implicar en el desarrollo del modelo de datos a todos los usuarios de la base de datos, tan pronto como sea posible. El diseño lógico de la base de datos es independiente del SGBD concreto que se vaya a utilizar, es independiente de los programas de aplicación, de los lenguajes de programación y de cualquier otra consideración física. El diseñador, en resumen, identifica los datos que se van a almacenar en la BD, elije las estructuras apropiadas, se comunica con los futuros usuarios de la BD con el fin de comprender sus necesidades y requerimientos.

- Una vez se ha diseñado e implementado la base de datos, los ***programadores*** *de aplicaciones* se encargan de implementar los programas de aplicación que servirán a los usuarios finales. Estos programas de aplicación son los que permiten consultar datos, insertarlos, actualizarlos y eliminarlos. Estos programas se escriben mediante lenguajes de tercera generación o de cuarta generación. El programador, en resumen, implementa las especificaciones realizadas por los analistas/diseñadores en forma de programas y luego prueba, depura, documenta y mantiene estas transacciones programadas, debiendo conocer a la perfección toda la gama de capacidades del SGBD.

- Los ***usuarios finales*** son los clientes de la base de datos: la base de datos ha sido diseñada e implementada, y está siendo mantenida, para satisfacer sus requisitos en la gestión de su información. Podríamos separar en cuatro categorías a estos usuarios:
    - ***Usuarios ocasionales***. Acceden de vez en cuando a la BD, pero es posible que requieran información diferente en cada ocasión. Utilizan un lenguaje de consulta de BD avanzado para especificar sus solicitudes. Suelen ser gerentes de nivel medio o alto.
    - ***Usuarios simples***. Constituyen una porción apreciable de la totalidad de los usuarios finales. La función principal de su trabajo gira en torno a consultas y actualizaciones constantes de la BD, utilizando tipos estándar de consultas y actualizaciones, llamadas transacciones programadas. Por ejemplo, el cajero de un banco consultando saldos, el empleado de una compañía aérea revisando disponibilidad para una reserva, el trabajador de una oficina de correos introduciendo el código de barras de los paquetes enviados/recibidos, ...
    - ***Usuarios avanzados.*** Ingenieros, científicos, analistas de negocios que están familiarizados con los recursos del SGBD.
    - ***Usuarios autónomos.*** Mantienen BD personales mediante la utilización de paquetes de programas comerciales, que cuentan con interfaces de fácil uso, basados en menus o en gráficos. Un ejemplo es el usuario de un paquete fiscal que almacena diversos datos financieros personales para fines fiscales.

Hay que tener en cuenta que las necesidades de los usuarios son muy diferentes en función del tipo de usuario que sean: a los finales les interesa la facilidad de uso, a los desarrolladores la potencia y flexibilidad de los lenguajes incorporados del sistema de bases de datos y a los administradores herramientas de gestión avanzada para la base de datos.

Estructura de una base de datos

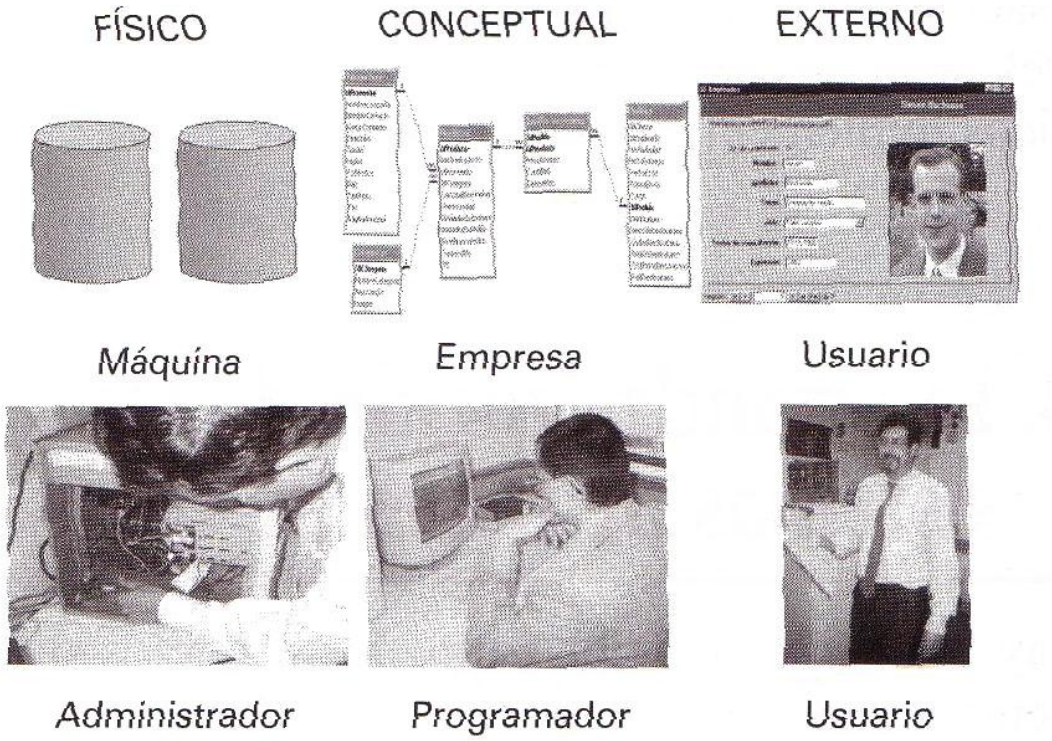
Las bases de datos están compuestas (como ya se han comentado), de **datos** y de **metadatos**. Los metadatos son datos (valga la redundancia) que sirven para especificar la estructura de la base de datos; por ejemplo qué tipo de

datos se almacenan (si son texto o números o fechas ...), qué nombre se le da a cada dato (nombre, apellidos,...), cómo están agrupados, cómo se relacionan,....

De este modo se producen dos visiones de la base de datos:

- **Estructura lógica o conceptual.** Indica la composición y distribución teórica de la base de datos. La estructura lógica sirve para que las aplicaciones puedan utilizar los elementos de la base de datos sin saber realmente cómo se están almacenando.

- **Estructura física**. Es la estructura de los datos tan cual se almacenan en las unidades de disco. La correspondencia entre la estructura lógica y la física se almacena en la base de datos (en los metadatos).



Niveles de descripción de datos

5. Modelos de datos

Una de las características fundamentales de los sistemas de bases de datos es que proporcionan cierto nivel de abstracción de datos, al ocultar las características sobre el almacenamiento físico que la mayoría de usuarios no necesita conocer. Los modelos de datos son el instrumento principal para ofrecer dicha abstracción.

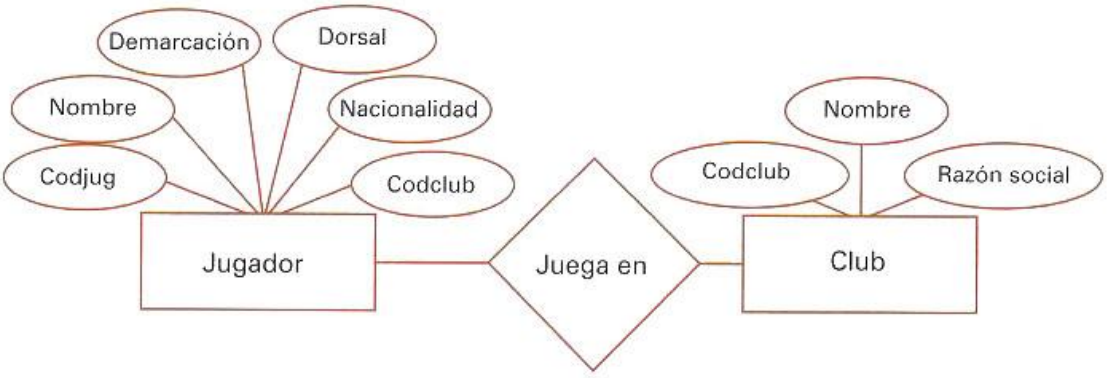
Un ***modelo de datos*** es un conjunto de conceptos que sirven para describir la estructura de una base de datos: los datos, las relaciones entre los datos y las restricciones que deben cumplirse sobre los datos.

Los modelos de datos contienen también un conjunto de operaciones básicas para la realización de consultas (lecturas) y actualizaciones de datos. Además, los modelos de datos más modernos incluyen conceptos para especificar comportamiento, permitiendo especificar un conjunto de operaciones definidas por el usuario.

Los modelos de datos se pueden clasificar dependiendo de los tipos de conceptos que ofrecen para describir la estructura de la base de datos. Los modelos de datos de alto nivel, o ***modelos conceptuales***, disponen de conceptos muy cercanos al modo en que la mayoría de los usuarios percibe los datos, mientras que los modelos de datos de bajo nivel, o ***modelos físicos***, proporcionan conceptos que describen los detalles de cómo se almacenan los datos en el ordenador. Los conceptos de los modelos físicos están dirigidos al personal informático, no a los usuarios finales. Entre estos dos extremos se encuentran los ***modelos lógicos***, cuyos conceptos pueden ser entendidos por los usuarios finales, aunque no están demasiado alejados de la forma en que los datos se organizan físicamente. Los modelos lógicos ocultan algunos detalles de cómo se almacenan los datos, pero pueden implementarse de manera directa en un ordenador.

- Los modelos conceptuales utilizan conceptos como entidades, atributos y relaciones. Una ***entidad*** representa un objeto o concepto del mundo real como, por ejemplo, un jugador de un club deportivo. Un ***atributo*** representa alguna propiedad de interés de una entidad como, por ejemplo, el nombre o el salario del empleado. Una ***relación*** describe una interacción entre dos o más entidades, por ejemplo, la relación

"jugar en" entre un jugador y su club. Un ejemplo de representación del modelo conceptual sería el siguiente:



Diseño conceptual: modelo entidad-interrelación

- Cada SGBD soporta un modelo lógico, siendo los más comunes el **relacional, el de red y el jerárquico**. Estos modelos representan los datos valiéndose de estructuras de registros, por lo que también se denominan *modelos orientados a registros*. Hay una nueva familia de modelos lógicos, son los *modelos* **orientados a objetos**, que están más próximos a los modelos conceptuales.

  Será el modelo relacional el que será caso de estudio durante este curso.

  Las siguientes figuras muestran tres modelos lógicos del mismo modelo conceptual.

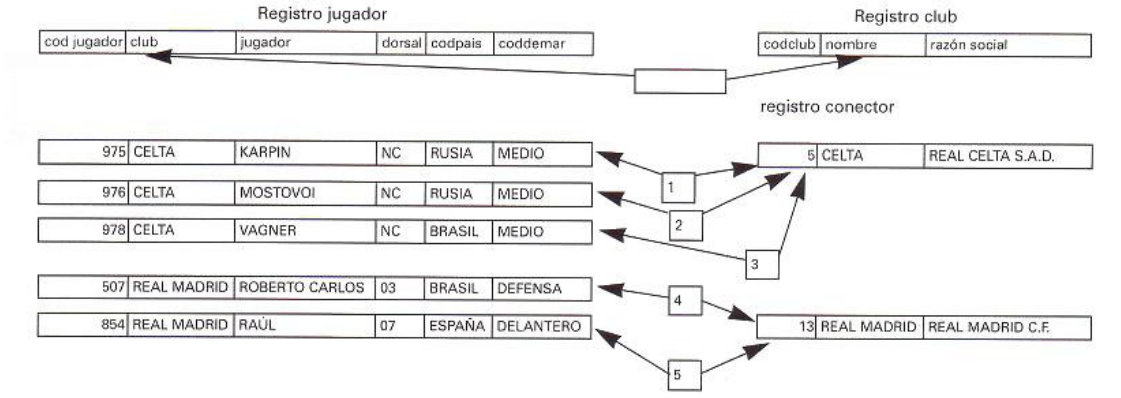
Jugador

CODJUGADOR	CLUB	JUGADOR	DORSAL	COD PAIS	CODDEMAR
799	RACING	ABELDON	06	ESPAÑA	MEDIO
536	OVIEDO	ABEL XABIER	15	COMUNITARIO	DEFENSA
610	BARCELONA	ABELARDO	03	ESPAÑA	DEFENSA
693	SPORTING	ABLANEDO	NC	ESPAÑA	PORTERO
702	SPORTING	ACEBAL	NC	ESPAÑA	PORTERO
932	ZARAGOZA	ACUÑA	NC	ESPAÑA	MEDIO
922	REAL SOCIEDAD	ADEPOJU	23	NIGERIA	MEDIO
951	ZARAGOZA	AGUADO	06	ESPAÑA	DEFENSA
636	ATLETICO MADRID	AGUILERA	15	ESPAÑA	DEFENSA
704	CELTA	AGUIRRETXU	NC	ESPAÑA	DEFENSA
665	SPORTING	AITOR	NC	ESPAÑA	DEFENSA
973	SALAMANCA	AIZPURUA	01	ESPAÑA	PORTERO
844	VALENCIA	ALBELDA	22	ESPAÑA	MEDIO
618	REAL SOCIEDAD	ALBERTO	01	ESPAÑA	PORTERO

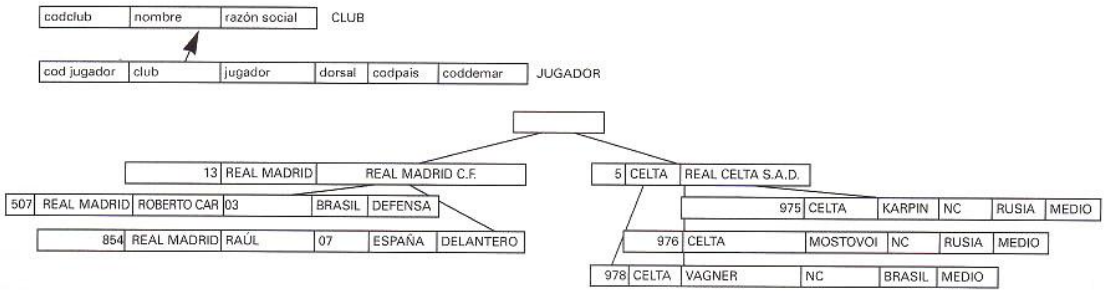
Club

CODCLUB	NOMBRE	RAZON SOCIAL
1	ATHLETIC	ATHLETIC CLUB DE BILBAO
2	ATLETICO MADRID	ATLETICO DE MADRID S.A.D.
3	BARCELONA	F.C. BARCELONA
4	BETIS	REAL BETIS BALOMPIE S.A.D.
5	CELTA	REAL CELTA S.A.D.
6	COMPOSTELA	COMPOSTELA S.A.D.
7	DEPORTIVO	DEPORTIVO DE LA CORUÑA S.A.D.
8	ESPANYOL	ESPANYOL DE BARCELONA S.A.D.
9	MALLORCA	REAL CLUB DEPORTIVO MALLORCA S.A.D.
10	MERIDA	CLUB POLIDEPORTIVO MERIDA S.A.D.
11	OVIEDO	REAL OVIEDO S.A.D.
12	RACING	RACING SANTANDER S.A.D.
13	REAL MADRID	REAL MADRID C.F.
14	REAL SOCIEDAD	REAL SOCIEDAD S.A.D.
15	SALAMANCA	UNIÓN DEPORTIVA SALAMANCA S.A.D.
16	SPORTING	REAL SPORTING DE GIJÓN S.A.D.
17	TENERIFE	CLUB TENERIFE S.A.D.
18	VALENCIA	VALENCIA C.F. S.A.D.
19	VALLADOLID	REAL VALLADOLID S.A.D.
20	ZARAGOZA	REAL ZARAGOZA S.A.D.
21	ALAVES	DEPORTIVO ALAVES S.A.D.

Modelo relacional



Modelo en red



Modelo jerárquico

- Los modelos físicos describen cómo se almacenan los datos en el ordenador: el formato de los registros, la estructura de los ficheros (desordenados, ordenados, etc.) y los métodos de acceso utilizados (índices, etc.).

A la descripción de una base de datos mediante un modelo de datos se le denomina ***esquema de la base de datos***. Este esquema se especifica durante el diseño, y no es de esperar que se modifique a menudo. Sin embargo, los datos que se almacenan en la base de datos pueden cambiar con mucha frecuencia: se insertan datos, se actualizan, etc. Los datos que la base de datos contiene en un determinado momento se denominan ***estado de la base de datos*** u ***ocurrencia de la base de datos***.

La distinción entre el esquema y el estado de la base de datos es muy importante. Cuando definimos una nueva base de datos, sólo especificamos su esquema al SGBD. En ese momento, el estado de la base de datos es el "estado vacío", sin datos. Cuando se cargan datos por primera vez, la base datos pasa al "estado inicial". De ahí en adelante, siempre que se realice una operación de actualización de la base de datos, se tendrá un nuevo estado. El SGBD se encarga, en parte, de garantizar que todos los estados de la base de datos sean estados válidos que satisfagan la estructura y las restricciones especificadas en el esquema. Por lo tanto, es muy importante que el esquema que se especifique al SGBD sea correcto y se debe tener muchísimo cuidado al diseñarlo. El SGBD almacena el esquema en su catálogo o ***diccionario de datos***, de modo que se pueda consultar siempre que sea necesario.

6. Arquitectura de los sistemas de bases de datos

Hay tres características importantes inherentes a los sistemas de bases de datos: la separación entre los programas de aplicación y los datos, el manejo de múltiples vistas por parte de los usuarios y el uso de un catálogo para almacenar el esquema de la base de datos.

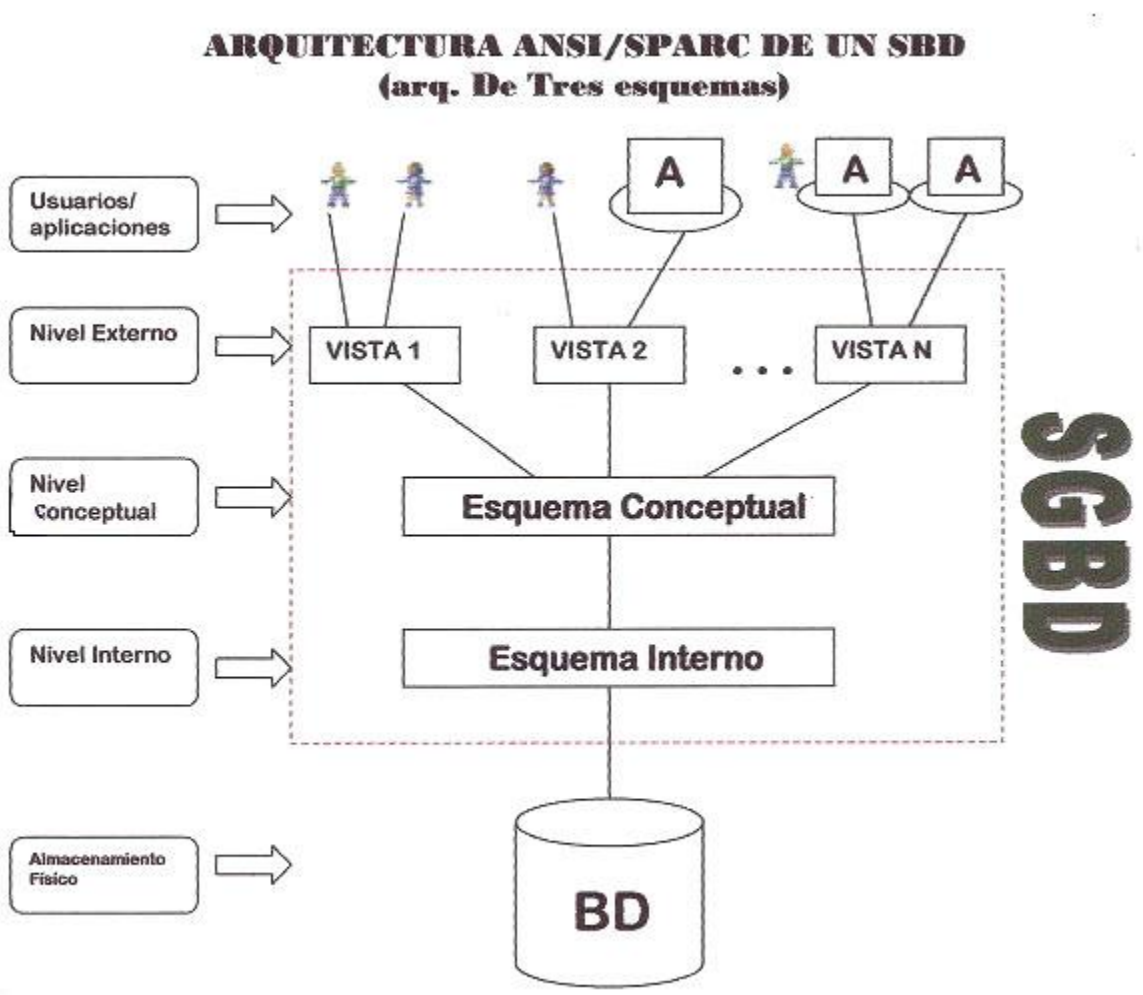
En 1975, el comité **ANSI-SPARC** (American National Standard Institute - Standards Planning and Requirements Committee) propuso una arquitectura de tres niveles para los sistemas de bases de datos, que resulta muy útil a la hora de conseguir estas tres características.

El objetivo de la arquitectura de tres niveles es el de separar los programas de aplicación de la base de datos física. En esta arquitectura, el esquema de una base de datos se define en tres niveles de abstracción distintos:

- En el ***nivel interno*** se describe la estructura física de la base de datos mediante un *esquema interno*. Este esquema se especifica mediante un modelo físico y describe todos los detalles para el almacenamiento de la base de datos, así como los métodos de acceso.

- En el ***nivel conceptual*** se describe la estructura de toda la base de datos para una comunidad de usuarios (todos los de una empresa u organización), mediante un *esquema conceptual*. Este esquema oculta los detalles de las estructuras de almacenamiento y se concentra en describir entidades, atributos, relaciones, operaciones de los usuarios y restricciones. En este nivel se puede utilizar un modelo conceptual o un modelo lógico para especificar el esquema.

- En el ***nivel externo*** se describen varios *esquemas externos* o *vistas de usuario*. Cada esquema externo describe la parte de la base de datos que interesa a un grupo de usuarios determinado y oculta a ese grupo el resto de la base de datos. En este nivel se puede utilizar un modelo conceptual o un modelo lógico para especificar los esquemas.



- La mayoría de los SGBD no distinguen del todo los tres niveles. Algunos incluyen detalles del nivel físico en el esquema conceptual. En casi todos los SGBD que se manejan vistas de usuario, los esquemas externos se especifican con el mismo modelo de datos que describe la información a nivel conceptual, aunque en algunos se pueden utilizar diferentes modelos de datos en los niveles conceptual y externo.

La arquitectura de tres niveles es útil para explicar el concepto de ***independencia de datos*** que podemos definir como la capacidad para

modificar el esquema en un nivel del sistema sin tener que modificar el esquema del nivel inmediato superior. Se pueden definir dos tipos de independencia de datos:

- La ***independencia lógica*** es la capacidad de modificar el esquema conceptual sin tener que alterar los esquemas externos ni los programas de aplicación. Se puede modificar el esquema conceptual para ampliar la base de datos o para reducirla. Si, por ejemplo, se reduce la base de datos eliminando una entidad, los esquemas externos que no se refieran a ella no deberán verse afectados.

- La ***independencia física*** es la capacidad de modificar el esquema interno sin tener que alterar el esquema conceptual (o los externos). Por ejemplo, puede ser necesario reorganizar ciertos ficheros físicos con el fin de mejorar el rendimiento de las operaciones de consulta o de actualización de datos. Dado que la independencia física se refiere sólo a la separación entre las aplicaciones y las estructuras físicas de almacenamiento, es más fácil de conseguir que la independencia lógica.

En los SGBD que tienen la arquitectura de varios niveles es necesario ampliar el catálogo o **diccionario de datos**, de modo que incluya información sobre cómo establecer la correspondencia entre las peticiones de los usuarios y los datos, entre los diversos niveles. El SGBD utiliza una serie de procedimientos adicionales para realizar estas correspondencias haciendo referencia a la información de correspondencia que se encuentra en el catálogo. La independencia de datos se consigue porque al modificarse el esquema en algún nivel, el esquema del nivel inmediato superior permanece sin cambios, sólo se modifica la correspondencia entre los dos niveles. No es preciso modificar los programas de aplicación que hacen referencia al esquema del nivel superior.

Por lo tanto, la arquitectura de tres niveles puede facilitar la obtención de la verdadera independencia de datos, tanto física como lógica. Sin embargo, los dos niveles de correspondencia implican un gasto extra durante la ejecución de una consulta o de un programa, lo cual reduce la eficiencia del SGBD. Es por esto que muy pocos SGBD han implementado esta arquitectura completa.

7. Estructuras operacionales

Actualmente casi todos los sistemas gestores de base de datos poseen también la idea operacional en la que se entiende que la base de datos se almacena en

un servidor y hay una serie de clientes que pueden acceder a los datos del mismo. Las posibilidades son:

- **Cliente-Servidor**. Estructura clásica, la base de datos y su SGBD están en un servidor al cual acceden los clientes. El cliente posee software que permite al usuario enviar instrucciones al SGBD en el servidor y recibir los resultados de estas instrucciones. Para ello el software cliente y el servidor deben utilizar software de comunicaciones en red.

- **Cliente Multi-servidor**. Ocurre cuando los clientes acceden a datos situados en más de un servidor. También se conoce esta estructura como **base de datos distribuida.** El cliente no sabe si los datos están en uno o más servidores, ya que el resultado es el mismo independientemente de dónde se almacenan los datos. En esta estructura hay un servidor de aplicaciones que es el que recibe las peticiones y el encargado de traducirlas a los distintos servidores de datos para obtener los resultados.

- **Cliente/Servidor Web/Servidor de datos**, el cliente se conecta a un servidor mediante un navegador web y desde las páginas de este ejecuta las consultas. El servidor web traduce esta consulta al servidor (o servidores) de datos.

8. Sistemas Gestores de Bases de Datos

Un *sistema gestor de bases de datos (SGBD)* es una aplicación que permite a los usuarios definir, crear y mantener una base de datos, y proporciona acceso controlado a la misma.

En general, un SGBD proporciona los siguientes servicios:

- Permite la definición de la base de datos mediante el *lenguaje de definición de datos (**DDL** – **Data Description Language**)*. Este lenguaje permite especificar la estructura y el tipo de los datos, así como las restricciones sobre los datos. Todo esto se almacenará en la base de datos.

- Permite la inserción, actualización, eliminación y consulta de datos mediante el *lenguaje de manejo de datos (**DML - Data Manipulation Language**)*.

- Proporciona un acceso controlado a la base de datos mediante:

    - un **sistema de seguridad**, de modo que los usuarios no autorizados no puedan acceder a la base de datos, mediante el lenguaje de control de datos (**DCL - Data Control Language**);
    - un **sistema de integridad** que mantiene la integridad y la consistencia de los datos;
    - un **sistema de control de concurrencia** que permite el acceso compartido a la base de datos;
    - un **sistema de control de recuperación** que restablece la base de datos después de que se produzca un fallo del *hardware* o del *software*;
    - un **diccionario de datos** o catálogo accesible por el usuario que contiene la descripción de los datos de la base de datos.

A diferencia de los sistemas de ficheros, el SGBD gestiona la estructura física de los datos y su almacenamiento. Con esta funcionalidad, el SGBD se convierte en una herramienta de gran utilidad. Sin embargo, desde el punto de vista del usuario, se podría discutir que los SGBD han hecho las cosas más complicadas, ya que ahora los usuarios ven más datos de los que realmente quieren o necesitan, puesto que ven la base de datos completa. Conscientes de este problema, los SGBD proporcionan un mecanismo de *vistas*, mediante el lenguaje de definición de vistas (**VDL – View Definition Language**), que permite que cada usuario tenga su propia vista o visión de la base de datos. El lenguaje de definición de datos permite definir vistas como subconjuntos de la base de datos.

Las **vistas**, además de reducir la complejidad permitiendo que cada usuario vea sólo la parte de la base de datos que necesita, tienen otras ventajas:

- Las vistas proporcionan un nivel de seguridad, ya que permiten excluir datos para que ciertos usuarios no los vean.
- Las vistas proporcionan un mecanismo para que los usuarios vean los datos en el formato que deseen.
- Una vista representa una imagen consistente y permanente de la base de datos, incluso si la base de datos cambia su estructura.

NOTA: Hay dos tipos de lenguajes de manejo de datos: los *procedurales* y los *no procedurales*. Estos dos tipos se distinguen por el modo en que acceden a los datos. Los lenguajes procedurales manipulan la base de datos registro a registro, mientras que los no procedurales operan sobre conjuntos de registros. En los lenguajes procedurales se especifica qué operaciones se deben realizar para obtener los datos resultado, mientras que en los lenguajes no procedurales se especifica qué datos deben obtenerse sin decir cómo hacerlo. El lenguaje no procedural más utilizado es el **SQL (Structured Query Language)** que, de hecho, es un estándar y es el lenguaje de los SGBD relacionales. SQL asume el papel, por tanto, de DDL, DML, DCL y VDL.

En general, los grandes SGBD multiusuario ofrecen todas las funciones que se acaban de citar y muchas más. Los sistemas modernos son conjuntos de programas extremadamente complejos y sofisticados, con millones de líneas de código y con una documentación consistente en varios volúmenes. Lo que se pretende es proporcionar un sistema que permita gestionar cualquier tipo de requisitos y que tenga un 100% de fiabilidad ante cualquier fallo *hardware* o *software*.

9. Historia de los SGBD

Como se ha visto en este capítulo, los predecesores de los sistemas de bases de datos fueron los sistemas de ficheros. No hay un momento concreto en que los sistemas de ficheros hayan cesado y hayan dado comienzo los sistemas de bases de datos. De hecho, todavía existen sistemas de ficheros en uso.

Se dice que los sistemas de bases de datos tienen sus raíces en el proyecto estadounidense Apolo de mandar al hombre a la luna, en los años sesenta. En aquella época, no había ningún sistema que permitiera gestionar la inmensa cantidad de información que requería el proyecto. La primera empresa encargada del proyecto, NAA (North American Aviation), desarrolló un *software* denominado GUAM (General Update Access Method) que estaba basado en el concepto de que varias piezas pequeñas se unen para formar una pieza más grande, y así sucesivamente hasta que el producto final está ensamblado. Esta estructura, que tiene la forma de un árbol, es lo que se denomina una ***estructura jerárquica***. A mediados de los sesenta, IBM se unió a NAA para desarrollar GUAM en lo que ahora se conoce como IMS (Information Management System). El motivo por el cual IBM restringió IMS al manejo de jerarquías de registros fue el de permitir el uso de dispositivos de almacenamiento serie, más exactamente las cintas magnéticas, ya que era un requisito del mercado por aquella época.

A mitad de los sesenta, se desarrolló IDS (Integrated Data Store), de General Electric. Este trabajo fue dirigido por uno de los pioneros en los sistemas de bases de datos, Charles Bachmann. IDS era un nuevo tipo de sistema de bases de datos conocido como ***sistema de red***, que produjo un gran efecto sobre los sistemas de información de aquella generación. El sistema de red se desarrolló, en parte, para satisfacer la necesidad de representar relaciones entre datos más complejas que las que se podían modelar con los sistemas jerárquicos, y, en parte, para imponer un estándar de bases de datos. Para ayudar a establecer dicho estándar, CODASYL (Conference on Data Systems Languages), formado por representantes del gobierno de EEUU y representantes del mundo empresarial, formaron un grupo denominado DBTG (Data Base Task Group), cuyo objetivo era definir unas especificaciones estándar que permitieran la creación de bases de datos y el manejo de los datos. El DBTG presentó su informe final en 1971 y aunque éste no fue formalmente aceptado por ANSI (American National Standards Institute), muchos sistemas se desarrollaron siguiendo la propuesta del DBTG. Estos sistemas son los que se conocen como sistemas de red, o sistemas CODASYL o DBTG.

Los sistemas jerárquico y de red constituyen la primera generación de los SGBD. Pero estos sistemas presentan algunos inconvenientes:

- Es necesario escribir complejos programas de aplicación para responder a cualquier tipo de consulta de datos, por simple que ésta sea.
- La independencia de datos es mínima.
- No tienen un fundamento teórico.

En los años 70 **Edgar Frank Codd**, de los laboratorios de investigación de IBM, escribió un artículo presentando el ***modelo relacional***. En este artículo, presentaba también los inconvenientes de los sistemas previos, el jerárquico y el de red. Entonces, se comenzaron a desarrollar muchos sistemas relacionales, apareciendo los primeros a finales de los setenta y principios de los ochenta. Uno de los primeros es System R, de IBM, que se desarrolló para probar la funcionalidad del modelo relacional, proporcionando una implementación de sus estructuras de datos y sus operaciones. Esto condujo a dos grandes desarrollos:

- El desarrollo de un lenguaje de consultas estructurado denominado SQL, que se ha convertido en el lenguaje estándar de los sistemas relacionales.

- La producción de varios SGBD relacionales durante los años ochenta, como DB2 y SLQ/DS de IBM, y ORACLE de ORACLE Corporation.

- El desarrollo de los sistemas relacionales de código aierto actuales.

Otros sistemas relacionales multiusuario son INGRES de Computer Associates, Informix de Informix Software Inc. y Sybase de Sybase Inc. Ejemplos de sistemas relacionales de microordenadores son Paradox y dBase IV de Borland, Access de Microsoft, FoxPro, PostgreSQL y MySQL .

Los SGBD relacionales constituyen la segunda generación de los SGBD. Sin embargo, el modelo relacional también tiene sus fallos, siendo uno de ellos su limitada capacidad al modelar los datos. Se ha hecho mucha investigación desde entonces tratando de resolver este problema. En 1976, Chen presentó el modelo entidad-relación, que es la técnica más utilizada en el diseño de bases de datos. En 1979, Codd intentó subsanar algunas de las deficiencias de su modelo relacional con una versión extendida denominada RM/T (1979) y más recientemente RM/V2 (1990). Los intentos de proporcionar un modelo de datos que represente al mundo real de un modo más fiel han dado lugar a los modelos de datos semánticos.

Como respuesta a la creciente complejidad de las aplicaciones que requieren bases de datos, han surgido dos nuevos modelos: el **modelo de datos orientado a objetos** y el **modelo relacional extendido**. Sin embargo, a diferencia de los modelos que los preceden, la composición de estos modelos no está clara. Esta evolución representa la tercera generación de los SGBD.

10. Funciones de los SGBD

Codd, el creador del modelo relacional, ha establecido una lista con los ocho servicios que debe ofrecer todo SGBD.

1. Un SGBD debe proporcionar a los usuarios la capacidad de almacenar datos en la base de datos, acceder a ellos y actualizarlos. Esta es la función fundamental de un SGBD y por supuesto, el SGBD debe ocultar al usuario la estructura física interna (la organización de los ficheros y las estructuras de almacenamiento).

2. Un SGBD debe proporcionar un catálogo en el que se almacenen las descripciones de los datos y que sea accesible por los usuarios. Este catálogo es lo que se denomina **diccionario de datos** y contiene

información que describe los datos de la base de datos (metadatos). Normalmente, un diccionario de datos almacena:

- Nombre, tipo y tamaño de los datos.
- Nombre de las relaciones entre los datos.
- Restricciones de integridad sobre los datos.
- Nombre de los usuarios autorizados a acceder a la base de datos.
- Esquemas externos, conceptual e interno, y correspondencia entre los esquemas.
- Estadísticas de utilización, tales como la frecuencia de las transacciones y el número de accesos realizados a los objetos de la base de datos.

Algunos de los beneficios que reporta el diccionario de datos son los siguientes:

- La información sobre los datos se puede almacenar de un modo centralizado. Esto ayuda a mantener el control sobre los datos, como un recurso que son.
- El significado de los datos se puede definir, lo que ayudará a los usuarios a entender el propósito de los mismos.
- La comunicación se simplifica ya que se almacena el significado exacto. El diccionario de datos también puede identificar al usuario o usuarios que poseen los datos o que los acceden.
- Las redundancias y las inconsistencias se pueden identificar más fácilmente ya que los datos están centralizados.
- Se puede tener un historial de los cambios realizados sobre la base de datos.
- El impacto que puede producir un cambio se puede determinar antes de que sea implementado, ya que el diccionario de datos mantiene información sobre cada tipo de dato, todas sus relaciones y todos sus usuarios.
- Se puede hacer respetar la seguridad.
- Se puede garantizar la integridad.
- Se puede proporcionar información para auditorías.

3. Un SGBD debe proporcionar un mecanismo que garantice que todas las actualizaciones correspondientes a una determinada transacción se realicen, o que no se realice ninguna. Una *transacción* es un conjunto de acciones que cambian el contenido de la base de datos. Una transacción en el sistema informático de la empresa inmobiliaria sería dar de alta a un empleado o eliminar un inmueble. Una transacción un poco más complicada sería eliminar un empleado y reasignar sus inmuebles a otro

empleado. En este caso hay que realizar varios cambios sobre la base de datos. Si la transacción falla durante su realización, por ejemplo porque falla el hardware, la base de datos quedará en un estado inconsistente. Algunos de los cambios se habrán hecho y otros no, por lo tanto, los cambios realizados deberán ser deshechos para devolver la base de datos a un estado consistente.

4. Un SGBD debe proporcionar un mecanismo que asegure que la base de datos se actualice correctamente cuando varios usuarios la están actualizando concurrentemente. Uno de los principales objetivos de los SGBD es el permitir que varios usuarios tengan acceso concurrente a los datos que comparten. El acceso concurrente es relativamente fácil de gestionar si todos los usuarios se dedican a leer datos, ya que no pueden interferir unos con otros. Sin embargo, cuando dos o más usuarios están accediendo a la base de datos y al menos uno de ellos está actualizando datos, pueden interferir de modo que se produzcan inconsistencias en la base de datos. El SGBD se debe encargar de que estas interferencias no se produzcan en el acceso simultáneo.

5. Un SGBD debe proporcionar un mecanismo capaz de recuperar la base de datos en caso de que ocurra algún suceso que la dañe. Como se ha comentado antes, cuando el sistema falla en medio de una transacción, la base de datos se debe devolver a un estado consistente. Este fallo puede ser a causa de un fallo en algún dispositivo hardware o un error del software, que hagan que el SGBD aborte, o puede ser a causa de que el usuario detecte un error durante la transacción y la aborte antes de que finalice. En todos estos casos, el SGBD debe proporcionar un mecanismo capaz de recuperar la base de datos llevándola a un estado consistente.

6. Un SGBD debe proporcionar un mecanismo que garantice que sólo los usuarios autorizados pueden acceder a la base de datos. La protección debe ser contra accesos no autorizados, tanto intencionados como accidentales.

7. Un SGBD debe ser capaz de integrarse con algún software de comunicación. Muchos usuarios acceden a la base de datos desde terminales. En ocasiones estos terminales se encuentran conectados directamente a la máquina sobre la que funciona el SGBD. En otras ocasiones los terminales están en lugares remotos, por lo que la comunicación con la máquina que alberga al SGBD se debe hacer a través de una red. En cualquiera de los dos casos, el SGBD recibe

peticiones en forma de mensajes y responde de modo similar. Todas estas transmisiones de mensajes las maneja el gestor de comunicaciones de datos. Aunque este gestor no forma parte del SGBD, es necesario que el SGBD se pueda integrar con él para que el sistema sea comercialmente viable.

8. Un SGBD debe proporcionar los medios necesarios para garantizar que tanto los datos de la base de datos, como los cambios que se realizan sobre estos datos, sigan ciertas reglas. La integridad de la base de datos requiere la validez y consistencia de los datos almacenados. Se puede considerar como otro modo de proteger la base de datos, pero además de tener que ver con la seguridad, tiene otras implicaciones. La integridad se ocupa de la calidad de los datos. Normalmente se expresa mediante **restricciones**, que son una serie de reglas que la base de datos no puede violar. Por ejemplo, se puede establecer la restricción de que cada empleado no puede tener asignados más de diez inmuebles. En este caso sería deseable que el SGBD controlara que no se sobrepase este límite cada vez que se asigne un inmueble a un empleado.

Además, de estos ocho servicios, es razonable esperar que los SGBD proporcionen un par de servicios más:

- Un SGBD debe permitir que se mantenga la independencia entre los programas y la estructura de la base de datos. La independencia de datos se alcanza mediante las vistas o subesquemas. La independencia de datos física es más fácil de alcanzar, de hecho hay varios tipos de cambios que se pueden realizar sobre la estructura física de la base de datos sin afectar a las vistas. Sin embargo, lograr una completa independencia de datos lógica es más difícil. Añadir una nueva entidad, un atributo o una relación puede ser sencillo, pero no es tan sencillo eliminarlos.

- Un SGBD debe proporcionar una serie de herramientas que permitan administrar la base de datos de modo efectivo. Algunas herramientas trabajan a nivel externo, por lo que habrán sido producidas por el administrador de la base de datos. Las herramientas que trabajan a nivel interno deben ser proporcionadas por el distribuidor del SGBD. Algunas de ellas son:
    - Herramientas para importar y exportar datos.
    - Herramientas para monitorizar el uso y el funcionamiento de la base de datos.

- Programas de análisis estadístico para examinar las prestaciones o las estadísticas de utilización.
- Herramientas para reorganización de índices.
- Herramientas para aprovechar el espacio dejado en el almacenamiento físico por los registros borrados y que consoliden el espacio liberado para reutilizarlo cuando sea necesario.

11. Ventajas

Los sistemas de bases de datos presentan numerosas ventajas que se pueden dividir en dos grupos: las que se deben a la integración de datos y las que se deben a la interface común que proporciona el SGBD.

Ventajas por la integración de datos

- *Control sobre la redundancia de datos.* Los sistemas de ficheros almacenan varias copias de los mismos datos en ficheros distintos. Esto hace que se desperdicie espacio de almacenamiento, además de provocar la falta de consistencia de datos. En los sistemas de bases de datos todos estos ficheros están integrados, por lo que no se almacenan varias copias de los mismos datos. Sin embargo, en una base de datos no se puede eliminar la redundancia completamente, ya que en ocasiones es necesaria para modelar las relaciones entre los datos, o bien es necesaria para mejorar las prestaciones.

- *Consistencia de datos.* Eliminando o controlando las redundancias de datos se reduce en gran medida el riesgo de que haya inconsistencias. Si un dato está almacenado una sola vez, cualquier actualización se debe realizar sólo una vez, y está disponible para todos los usuarios inmediatamente. Si un dato está duplicado y el sistema conoce esta redundancia, el propio sistema puede encargarse de garantizar que todas las copias se mantienen consistentes. Desgraciadamente, no todos los SGBD de hoy en día se encargan de mantener automáticamente la consistencia.

- *Más información sobre la misma cantidad de datos.* Al estar todos los datos integrados, se puede extraer información adicional sobre los mismos.

- *Compartición de datos.* En los sistemas de ficheros, los ficheros pertenecen a las personas o a los departamentos que los utilizan. Pero en los sistemas de bases de datos, la base de datos pertenece a la

empresa y puede ser compartida por todos los usuarios que estén autorizados. Además, las nuevas aplicaciones que se vayan creando pueden utilizar los datos de la base de datos existente.

- *Mantenimiento de estándares.* Gracias a la integración es más fácil respetar los estándares necesarios, tanto los establecidos a nivel de la empresa como los nacionales e internacionales. Estos estándares pueden establecerse sobre el formato de los datos para facilitar su intercambio, pueden ser estándares de documentación, procedimientos de actualización y también reglas de acceso.

Ventajas por la existencia del SGBD

- *Mejora en la integridad de datos.* La integridad de la base de datos se refiere a la validez y la consistencia de los datos almacenados. Normalmente, la integridad se expresa mediante restricciones o reglas que no se pueden violar. Estas restricciones se pueden aplicar tanto a los datos, como a sus relaciones, y es el SGBD quien se debe encargar de mantenerlas.

- *Mejora en la seguridad.* La seguridad de la base de datos es la protección de la base de datos frente a usuarios no autorizados. Sin unas buenas medidas de seguridad, la integración de datos en los sistemas de bases de datos hace que éstos sean más vulnerables que en los sistemas de ficheros. Sin embargo, los SGBD permiten mantener la seguridad mediante el establecimiento de claves para identificar al personal autorizado a utilizar la base de datos. Las autorizaciones se pueden realizar a nivel de operaciones, de modo que un usuario puede estar autorizado a consultar ciertos datos pero no a actualizarlos, por ejemplo.

- *Mejora en la accesibilidad a los datos.* Muchos SGBD proporcionan lenguajes de consultas o generadores de informes que permiten al usuario hacer cualquier tipo de consulta sobre los datos, sin que sea necesario que un programador escriba una aplicación que realice tal tarea.

- *Mejora en la productividad.* El SGBD proporciona muchas de las funciones estándar que el programador necesita escribir en un sistema de ficheros. A nivel básico, el SGBD proporciona todas las rutinas de manejo de ficheros típicas de los programas de aplicación. El hecho de

disponer de estas funciones permite al programador centrarse mejor en la función específica requerida por los usuarios, sin tener que preocuparse de los detalles de implementación de bajo nivel. Muchos SGBD también proporcionan un entorno de cuarta generación consistente en un conjunto de herramientas que simplifican, en gran medida, el desarrollo de las aplicaciones que acceden a la base de datos. Gracias a estas herramientas, el programador puede ofrecer una mayor productividad en un tiempo menor.

- *Mejora en el mantenimiento gracias a la independencia de datos.* En los sistemas de ficheros, las descripciones de los datos se encuentran inmersas en los programas de aplicación que los manejan. Esto hace que los programas sean dependientes de los datos, de modo que un cambio en su estructura, o un cambio en el modo en que se almacena en disco, requiere cambios importantes en los programas cuyos datos se ven afectados. Sin embargo, los SGBD separan las descripciones de los datos de las aplicaciones. Esto es lo que se conoce como independencia de datos, gracias a la cual se simplifica el mantenimiento de las aplicaciones que acceden a la base de datos.

- *Aumento de la concurrencia.* En algunos sistemas de ficheros, si hay varios usuarios que pueden acceder simultáneamente a un mismo fichero, es posible que el acceso interfiera entre ellos de modo que se pierda información o, incluso, que se pierda la integridad. La mayoría de los SGBD gestionan el acceso concurrente a la base de datos y garantizan que no ocurran problemas de este tipo.

- *Mejora en los servicios de copias de seguridad y de recuperación ante fallos.* Muchos sistemas de ficheros dejan que sea el usuario quien proporcione las medidas necesarias para proteger los datos ante fallos en el sistema o en las aplicaciones. Los usuarios tienen que hacer copias de seguridad cada día, y si se produce algún fallo, utilizar estas copias para restaurarlos. En este caso, todo el trabajo realizado sobre los datos desde que se hizo la última copia de seguridad se pierde y se tiene que volver a realizar. Sin embargo, los SGBD actuales funcionan de modo que se minimiza la cantidad de trabajo perdido cuando se produce un fallo.

12. Inconvenientes

- *Complejidad.* Los SGBD son conjuntos de programas muy complejos con una gran funcionalidad. Es preciso comprender muy bien esta funcionalidad para poder sacar un buen partido de ellos.

- *Tamaño.* Los SGBD son programas complejos y muy extensos que requieren una gran cantidad de espacio en disco y de memoria para trabajar de forma eficiente.

- *Coste económico del SGBD.* El coste de un SGBD varía dependiendo del entorno y de la funcionalidad que ofrece. Por ejemplo, un SGBD para un ordenador personal puede costar 500 euros, mientras que un SGBD para un sistema multiusuario que dé servicio a cientos de usuarios puede costar entre 10.000 y 100.000 euros. Además, hay que pagar una cuota anual de mantenimiento que suele ser un porcentaje del precio del SGBD.

- *Coste del equipamiento adicional.* Tanto el SGBD, como la propia base de datos, pueden hacer que sea necesario adquirir más espacio de almacenamiento. Además, para alcanzar las prestaciones deseadas, es posible que sea necesario adquirir una máquina más grande o una máquina que se dedique solamente al SGBD. Todo esto hará que la implantación de un sistema de bases de datos sea más cara.

- *Coste de la conversión.* En algunas ocasiones, el coste del SGBD y el coste del equipo informático que sea necesario adquirir para su buen funcionamiento, es insignificante comparado al coste de convertir la aplicación actual en un sistema de bases de datos. Este coste incluye el coste de enseñar a la plantilla a utilizar estos sistemas y, probablemente, el coste del personal especializado para ayudar a realizar la conversión y poner en marcha el sistema. Este coste es una de las razones principales por las que algunas empresas y organizaciones se resisten a cambiar su sistema actual de ficheros por un sistema de bases de datos.

- *Prestaciones.* Un sistema de ficheros está escrito para una aplicación específica, por lo que sus prestaciones suelen ser muy buenas. Sin embargo, los SGBD están escritos para ser más generales y ser útiles en muchas aplicaciones, lo que puede hacer que algunas de ellas no sean tan rápidas como antes.

- *Vulnerable a los fallos.* El hecho de que todo esté centralizado en el SGBD hace que el sistema sea más vulnerable ante los fallos que puedan producirse.

13. Componentes. Estructura genérica de un SGBD

Los SGBD son sistemas software muy complejos, compuestos de varios módulos software que se encargan de cada una de las responsabilidades del sistema completo. Algunas de estas funciones las puede proporcionar el sistema operativo (SO), pero en general los sistemas operativos sólo proporcionan los servicios más básicos y los SGBD deben construirse sobre esta base (por esto, en el diseño de un SGBD se debe considerar la interfaz entre el SGBD y el SO).

Procesamiento de consultas

El **procesador de consultas** se divide en:

- El **compilador de consultas.** Trata cada consulta de alto nivel (escrita en DML) que se introduce de forma interactiva, analiza su sintaxis, intenta optimizarla (transformarla en otra equivalente pero más eficiente) y genera una llamada al *motor de evaluación de consultas* para que la ejecute.

- El **precompilador de DML embebido** extrae las sentencias en DML de un programa escrito en un lenguaje host y las envía al **compilador de DML**, el cual intenta optimizarlas y las convierte en código objeto (instrucciones de bajo nivel que entiende el *motor de evaluación de consultas*) para el acceso a la BD. El resto del programa se envía al compilador del lenguaje host. El código objeto de las sentencias DML se enlaza (*link*) con el código objeto del resto del programa, formando una *transacción programada* cuyo código ejecutable incluye llamadas al *motor de evaluación de consultas* de la base de datos.

- El **compilador (o intérprete) de DDL** procesa las definiciones de esquema escritas en DDL, y almacena las descripciones de los esquemas (metadatos) en el catálogo del SGBD.

- El **motor de evaluación de consultas** en tiempo de ejecución se encarga de recibir solicitudes de recuperación o actualización, y las

ejecuta sobre la base de datos. El acceso a los datos (a disco) se realiza mediante el *gestor de almacenamiento*.

Gestión de almacenamiento

Los siguientes son los **componentes de gestión de almacenamiento**, que proporcionan la interfaz entre los datos almacenados y los programas de aplicación y envío de consultas al sistema.

- **Subsistema de control de concurrencia y recuperación** (o **gestor de transacciones**), que...

  · Asegura la consistencia y coherencia de los datos cuando varios usuarios actualizan a la vez la misma información en la BD.
  · Detecta fallos o caídas del sistema, en cuyo caso debe llevar a cabo la restauración de la basede datos a un estado consistente (correcto).

- **Subsistema de integridad y seguridad**, encargado de...

  · Determinar si las actualizaciones de los datos son correctas o por el contrario violan alguna restricción de integridad, en cuyo caso realiza la acción adecuada.
  · Asegurar que se cumplen las restricciones de seguridad en el acceso a la base de datos o a determinados datos.

- **Gestor de datos almacenados y de la memoria intermedia**, que controla el acceso a la información del SGBD almacenada en disco (datos o metadatos). Se encarga de la reserva de espacio de almacenamiento en disco y las estructuras de datos usadas para representar la información en disco. Este componente puede emplear los servicios básicos del SO para transferir datos de bajo nivel entre el disco y la memoria principal del ordenador. Es el responsable de otros aspectos de transferencia de datos, como por ejemplo el manejo de las áreas de almacenamiento intermedio (*buffers*) en la memoria principal, donde se llevan los datos desde el disco para que después otros módulos del SGBD puedan procesarlos. También se encarga de decidir qué datos tratar en la memoria caché.

Además de los componentes anteriores, se necesitan varias **estructuras de datos** como parte de la implementación física del sistema:

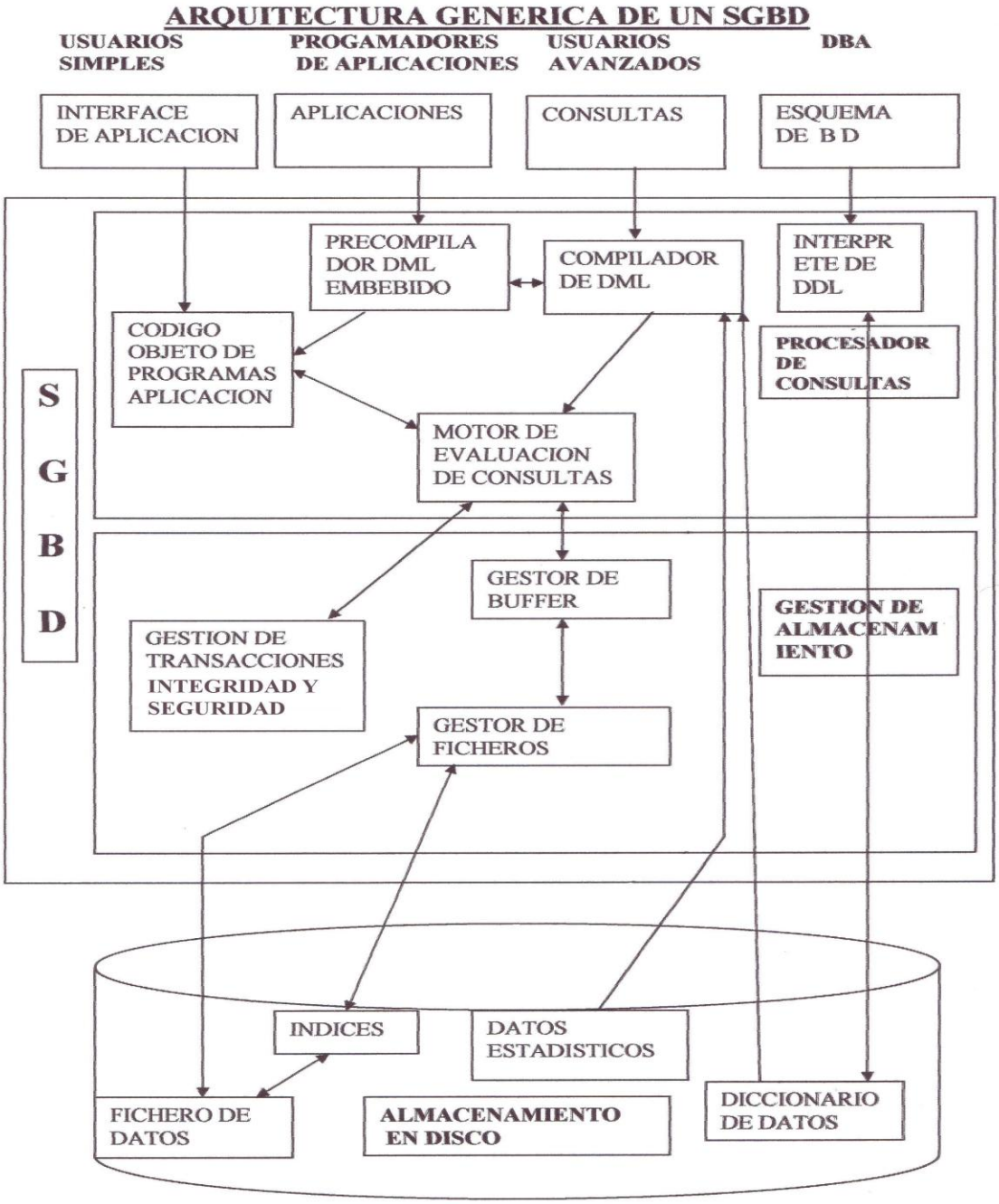
- **Ficheros de datos** en disco, que almacenan la base de datos en sí.

- El **catálogo** del SGBD, que es una mini-base de datos que almacena los esquemas (descripciones) de las bases de datos que mantiene (gestiona) el SGBD. De hecho, cada base de datos está descrita por los datos almacenados en el catálogo (llamados metadatos que describen su estructura, restricciones, autorizaciones, etc.). Contiene una descripción del esquema conceptual, del esquema interno, de los esquemas externos y de las correspondencias entre ellos. Además contiene la información necesaria para los componentes del SGBD relacionados con el procesamiento de consultas y los de seguridad y autorización. El catálogo del sistema es gestionado por el **sistema de diccionario de datos**.

  NOTA 1: **El sistema de diccionario de datos** es un mini-SGBD que gestiona el catálogo del sistema y además permite almacenar y controlar la siguiente información:

  - Información acerca del diseño físico de la BD: estructuras de almacenamiento (tipos de ficheros), caminos o estructuras de acceso a los datos, tamaño de los ficheros, registros, etc.
  - Descripción de los usuarios, sus responsabilidades y derechos de acceso.
  - Descripciones de alto nivel de las transacciones y aplicaciones de la BD, y de las relaciones entre los usuarios y las transacciones.
  - Relación entre las transacciones y la información a la que hacen referencia (consultan o modifican); disponer de este tipo de relaciones es útil para determinar qué transacciones son afectadas cuando se modifica la estructura de los datos.

- **Estructuras de acceso** (ficheros de **índices**, por ejemplo), que permiten acceso rápido a elementos de datos que tienen valores particulares.
- **Datos estadísticos** sobre los datos en la base de datos. Esta información es necesaria para seleccionar las formas eficientes de ejecutar una consulta (optimización).



14. Lenguajes de los SGBD

Los SGBD deben ofrecer lenguajes e interfaces apropiadas para cada tipo de usuario: administradores de la base de datos, diseñadores, programadores de aplicaciones y usuarios finales.

Lenguaje de definición de datos (DDL)

Una vez finalizado el diseño de una base de datos y escogido un SGBD para su implementación, el primer paso consiste en especificar el esquema conceptual y el esquema interno de la base de datos, y la correspondencia entre ambos. En muchos SGBD no se mantiene una separación estricta de niveles, por lo que el administrador de la base de datos y los diseñadores utilizan el mismo lenguaje para definir ambos esquemas, es el *lenguaje de definición de datos* (DDL). El SGBD posee un compilador de DDL cuya función consiste en procesar las sentencias del lenguaje para identificar las descripciones de los distintos elementos de los esquemas y almacenar la descripción del esquema en el catálogo o diccionario de datos. Se dice que el diccionario contiene *metadatos*: describe los objetos de la base de datos.

Cuando en un SGBD hay una clara separación entre los niveles conceptual e interno, el DDL sólo sirve para especificar el esquema conceptual. Para especificar el esquema interno se utiliza un *lenguaje de definición de almacenamiento* (ADL). Las correspondencias entre ambos esquemas se pueden especificar en cualquiera de los dos lenguajes. Para tener una verdadera arquitectura de tres niveles sería necesario disponer de un tercer lenguaje, el *lenguaje de definición de vistas* (VDL), que se utilizaría para especificar las vistas de los usuarios y su correspondencia con el esquema conceptual.

Lenguaje de manejo de datos (DML)

Una vez creados los esquemas de la base de datos, los usuarios necesitan un lenguaje que les permita manipular los datos de la base de datos: realizar consultas, inserciones, eliminaciones y modificaciones. Este lenguaje es el que se denomina *lenguaje de manejo de datos* (DML).

Hay dos tipos de DML: los procedurales y los no procedurales. Con un DML *procedural* el usuario (normalmente será un programador) especifica qué datos se necesitan y cómo hay que obtenerlos. Esto quiere decir que el usuario debe especificar todas las operaciones de acceso a datos llamando a los procedimientos necesarios para obtener la información requerida. Estos lenguajes acceden a un registro, lo procesan y basándose en los resultados obtenidos, acceden a otro registro, que también deben procesar. Así se va

accediendo a registros y se van procesando hasta que se obtienen los datos deseados. Las sentencias de un DML procedural deben estar embebidas en un lenguaje de alto nivel, ya que se necesitan sus estructuras (bucles, condicionales, etc.) para obtener y procesar cada registro individual. A este lenguaje se le denomina *lenguaje anfitrión*. Las bases de datos jerárquicas y de red utilizan DML procedurales.

Un DML *no procedural* se puede utilizar de manera independiente para especificar operaciones complejas sobre la base de datos de forma concisa. En muchos SGBD se pueden introducir interactivamente instrucciones del DML desde un terminal o bien embeberlas en un lenguaje de programación de alto nivel. Los DML no procedurales permiten especificar los datos a obtener en una consulta o los datos que se deben actualizar, mediante una sola y sencilla sentencia. El usuario o programador especifica qué datos quiere obtener sin decir cómo se debe acceder a ellos. El SGBD traduce las sentencias del LMD en uno o varios procedimientos que manipulan los conjuntos de registros necesarios. Esto libera al usuario de tener que conocer cuál es la estructura física de los datos y qué algoritmos se deben utilizar para acceder a ellos. A los DML no procedurales también se les denomina *declarativos*. Las bases de datos relacionales utilizan DML no procedurales, como SQL (Structured Query Language) o QBE (Query-By-Example). Los lenguajes no procedurales son más fáciles de aprender y de usar que los procedurales, y el usuario debe realizar menos trabajo, siendo el SGBD quien hace la mayor parte.

La parte de los DML no procedurales que realiza la obtención de datos es lo que se denomina un *lenguaje de consultas*. En general, las órdenes tanto de obtención como de actualización de datos de un DML no procedural se pueden utilizar interactivamente, por lo que al conjunto completo de sentencias del DML se le denomina lenguaje de consultas, aunque es técnicamente incorrecto.

Lenguaje de control de datos (DCL)

Los lenguajes de control de datos contienen elementos útiles para trabajar en entornos multiusuario en los que son importantes la protección de datos, la seguridad de las tablas y el establecimiento de restricciones en el acceso, así como elementos para coordinar la compartición de datos por parte de usuarios concurrentes, asegurando que no interfieren unos con otros.

15. Clasificación de los sistemas gestores de bases de datos

Según el modelo lógico utilizado

El criterio principal que se utiliza para clasificar los SGBD es el modelo lógico en que se basan. Los modelos lógicos empleados con mayor frecuencia en los SGBD comerciales actuales son el relacional, el de red y el jerárquico. Algunos SGBD más modernos se basan en modelos orientados a objetos.

El ***modelo relacional*** se basa en el concepto matemático denominado "relación", que gráficamente se puede representar como una tabla. En el modelo relacional, los datos y las relaciones existentes entre los datos se representan mediante estas relaciones matemáticas, cada una con un nombre que es único y con un conjunto de columnas.

En el modelo relacional la base de datos es percibida por el usuario como un conjunto de tablas. Esta percepción es sólo a nivel lógico (en los niveles externo y conceptual de la arquitectura de tres niveles), ya que a nivel físico puede estar implementada mediante distintas estructuras de almacenamiento.

En el ***modelo de red*** los datos se representan como colecciones de registros y las relaciones entre los datos se representan mediante conjuntos, que son punteros en la implementación física. Los registros se organizan como un grafo: los registros son los nodos y los arcos son los conjuntos. El SGBD de red más popular es el sistema IDMS.

El ***modelo jerárquico*** es un tipo de modelo de red con algunas restricciones. De nuevo los datos se representan como colecciones de registros y las relaciones entre los datos se representan mediante conjuntos. Sin embargo, en el modelo jerárquico cada nodo puede tener un solo padre. Una base de datos jerárquica puede representarse mediante un árbol: los registros son los nodos, también denominados segmentos, y los arcos son los conjuntos. El SGBD jerárquico más importante es el sistema IMS.

La mayoría de los SGBD comerciales actuales están basados en el modelo relacional, mientras que los sistemas más antiguos estaban basados en el modelo de red o el modelo jerárquico. Estos dos últimos modelos requieren que el usuario tenga conocimiento de la estructura física de la base de datos a la que se accede, mientras que el modelo relacional proporciona una mayor independencia de datos. Se dice que el modelo relacional es declarativo (se especifica qué datos se han de obtener) y los modelos de red y jerárquico son navegacionales (se especifica cómo se deben obtener los datos).

El ***modelo orientado a objetos*** define una base de datos en términos de objetos, sus propiedades y sus operaciones. Los objetos con la misma estructura y comportamiento pertenecen a una clase, y las clases se organizan en jerarquías o grafos acíclicos. Las operaciones de cada clase se especifican en términos de procedimientos predefinidos denominados métodos. Algunos SGBD relacionales existentes en el mercado han estado extendiendo sus modelos para incorporar conceptos orientados a objetos. A estos SGBD se les conoce como sistemas *objeto-relacionales.*

Según el número de usuarios a los que da servicio

Un segundo criterio para clasificar los SGBD es el número de usuarios a los que da servicio el sistema. Los sistemas ***monousuario*** sólo atienden a un usuario a la vez, y su principal uso se da en los ordenadores personales. Los sistemas ***multiusuario***, entre los que se encuentran la mayor parte de los SGBD, atienden a varios usuarios al mismo tiempo.

Según la distribución física de la base de datos

Un tercer criterio es el número de sitios en los que está distribuida la base de datos. Casi todos los SGBD son ***centralizados***: sus datos se almacenan en un solo computador. Los SGBD centralizados pueden atender a varios usuarios, pero el SGBD y la base de datos en sí residen por completo en una sola máquina. En los SGBD ***distribuidos*** la base de datos real y el propio software del SGBD pueden estar distribuidos en varios sitios conectados por una red. Los SGBD *distribuidos homogéneos* utilizan el mismo SGBD en múltiples sitios. Una tendencia reciente consiste en crear software para tener acceso a varias bases de datos autónomas preexistentes almacenadas en SGBD *distribuidos heterogéneos*. Esto da lugar a los SGBD *federados* o *sistemas multibase de datos* en los que los SGBD participantes tienen cierto grado de autonomía local. Muchos SGBD distribuidos emplean una arquitectura cliente-servidor.

Según el coste

Un cuarto criterio es el coste del SGBD. La mayor parte de los paquetes de SGBD cuestan entre 10.000 y 100.000 euros. Los sistemas monousuario más económicos para microcomputadores cuestan entre 100 y 3.000 euros. En el otro extremo, los paquetes más completos cuestan más de 100.000 euros.

Estamos hablando de los **sistemas propietarios**. Este mercado se reparte, principalmente, entre dos empresas: IBM, con DB2, y Oracle, con la veterana y prestigiosa serie xI. Las cuotas de mercado de ambos productos oscilan entre

el 35 y 40%, según las fuentes. Por su parte, Microsoft SQL Server ostenta el 11% de la cuota de este mercado, seguido de varias empresas, como Sybase, cuya participación es de un solo dígito.

Pero este panorama de dominio de las bases de datos de software propietario está comenzando a cambiar poco a poco con la introducción de iniciativas basadas en **sistemas de código abierto** (OSS). Entre los desarrolladores más conocidos figuran Borland, con Firebird, y PostgreSQL. Sin embargo, el líder indiscutido en el mercado de bases de datos de código abierto es MySQL, propiedad de la empresa sueca MySQL AB. Prueba de ello son los cuatro millones de instalaciones y las 30.000 descargas online diarias.

Según el propósito

Por último, los SGBD pueden ser de ***propósito general*** o de ***propósito específico***. Cuando el rendimiento es fundamental, se puede diseñar y construir un SGBD de propósito especial para una aplicación específica, y este sistema no sirve para otras aplicaciones. Muchos sistemas de reservas de líneas aéreas son SGBD de propósito especial y pertenecen a la categoría de *sistemas de procesamiento de transacciones en línea* (OLTP), que deben atender un gran número de transacciones concurrentes sin imponer excesivos retrasos.