BASES DE DATOS curso 2000-2001

1. CONCEPTOS BÁSICOS DE LOS SISTEMAS DE BASE DE DATOS

1.1 BASES DE DATOS Y SUS USUARIOS.

Definición de los conceptos *base de datos, SGBD y sistema de BD*

En la actualidad, las bases de datos tienen una importancia decisiva en la práctica totalidad de las áreas de aplicación de la informática, como la ingeniería, la medicina, la educación, la biblioteconomía, la empresa, etc. Esto ha fomentado el desarrollo de una gran cantidad de conceptos y técnicas para la gestión eficiente de los datos.

Una primera definición sería esta:

"una base de datos es un conjunto de datos relacionados entre sí"

Un ejemplo sería el conjunto de nombres y números de teléfono de nuestros amigos, que tenemos registrados en una agenda.

Pero esta definición resulta demasiado general, puesto que una **base de datos** (BD) tiene las siguientes **propiedades**:

- Representa algún aspecto del mundo real, llamado **universo de discurso** (UoD, *Universe of Discourse*) del cual provienen los datos. Los cambios en el UoD se reflejan en la base de datos.
- Es un conjunto de datos lógicamente coherente, con significado implícito. Un montón de datos sin relación entre sí, agrupados de forma aleatoria, no se considera una base de datos.
- Toda base de datos se diseña, se crea y se carga con datos, con un objetivo determinado, y está dirigida a un grupo de usuarios, interesados en el contenido de la base de datos.

Las grandes organizaciones o empresas (bancos, hospitales, universidades, multinacionales, etc.) suelen utilizar dos bases de datos: una base de datos *operacional* y otra *de apoyo a la toma de decisiones*. Una BD operacional contiene la información necesaria para la gestión diaria de la organización (facturación, gestión de nóminas, etc.). La BD de apoyo a decisiones contiene información resumida, que se extrae periódicamente de la BD operacional (cálculos estadísticos de ventas por sector, resumen de producción o gastos, etc.) y que ayuda a los directivos de la empresa a la hora de tomar decisiones estratégicas y / o tácticas.

Las bases de datos pueden ser de cualquier tamaño y complejidad. Cuando la cantidad de información es grande y las interacciones entre los diferentes datos son muchas, es necesario organizar y controlar toda esta información almacenada, para que los usuarios puedan buscar, obtener y actualizar los datos cuando les sea necesario.

Un **sistema gestor de bases de datos** (SGBD, o *Database Management System DBMS*) es un conjunto de programas que permite a los usuarios crear y mantener una base de datos. Es un sistema software de propósito general, que facilita el proceso de **definir, construir** y **manipular** bases de datos para diversas aplicaciones.

Para **definir** una base de datos, hay que especificar los *tipos* de los datos, las *estructuras* de los datos y las *restricciones* de los datos. **Construir** una BD es el proceso de almacenar los datos (*reales*) en algún medio de almacenamiento controlado por el SGBD. **Manipular** la BD es *consultar* los datos para obtener cierta información, o *actualizar* la base de datos (*modificar* o *eliminar* datos, o *introducir nuevos*) para reflejar los cambios ocurridos en el UoD.

El primer objetivo de un SGBD es proporcionar un entorno que sea tanto práctico como eficiente de usar en la recuperación y el almacenamiento de la información de la base de datos.

Al conjunto formado por la base de datos y el software (del SGBD y el de los programas de aplicación) lo llamaremos **sistema de base de datos** (**SBD**). Véase la figura 1.

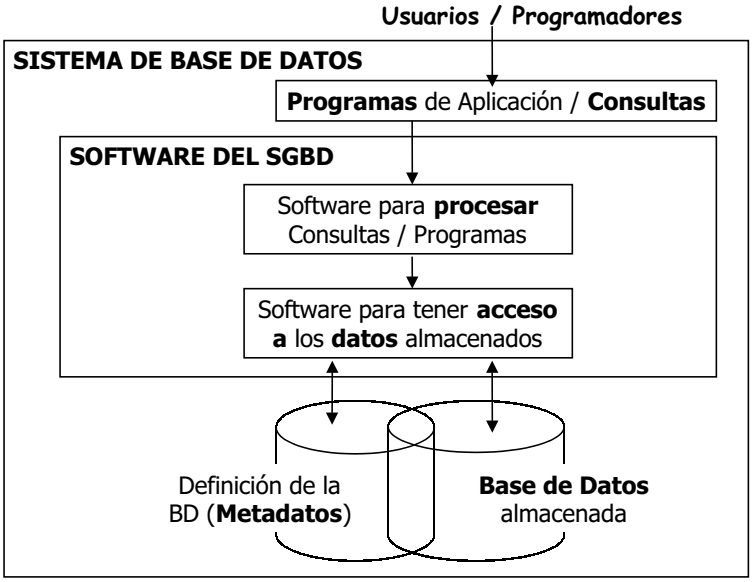


Figura 1. Entorno simplificado de un Sistema de Base de Datos

Características del enfoque de bases de datos

Son varias las características que distinguen este enfoque del clásico de los *sistemas de procesamiento de ficheros*, en los cuales las empresas almacenaban su información antes de la llegada de los SGBD.

✓ *Naturaleza autodescriptiva de los sistemas de base de datos*

El sistema contiene, además de la BD misma, una descripción completa de la base de datos. Esta descripción se almacena en el *catálogo del sistema* y consiste en información sobre la estructura de cada fichero, el tipo y formato de almacenamiento y las restricciones que se aplican a los datos. La información contenida en el catálogo se llama metainformación[1].

El catálogo es usado por el software del SGBD y a veces, por usuarios que necesiten información sobre la estructura de la BD.

El catálogo es necesario porque el SGBD no está escrito para una determinada aplicación, sino para *cualquier* aplicación de bases de datos, de forma que el SGBD tiene que consultar el catálogo para conocer la estructura de los archivos de cada BD (la de una universidad, o la de un banco).

En cambio, en el procesamiento de ficheros clásico, la definición de los datos es parte del código de los programas de aplicación, así que un programa "sólo puede trabajar con una base de datos" específica, cuya estructura se describe en el propio código (un ejemplo es un programa escrito en lenguaje C que contenga declaraciones "*struct*").

✓ *Separación entre los programas y los datos*

En el procesamiento de ficheros tradicional, como ya hemos indicado, la estructura de los ficheros de datos está integrada en los programas, así que cualquier cambio en la estructura de un fichero implica modificar todos los programas que acceden al mismo.

En cambio, los programas de acceso del SGBD se escriben para que sean independientes de cómo y dónde estén almacenados los datos. La estructura de los ficheros se guarda en el catálogo del SGBD, separada de los programas de acceso.

Esta propiedad es la *independencia con respecto a los programas y los datos*.

[1] Los *metadatos* son datos acerca de los datos.

Por ejemplo, es posible escribir un programa que sólo pueda tener acceso a *registros de* ACTOR *de 62 caracteres de longitud*. Si añadimos otro dato (campo) a cada registro de ACTOR, por ejemplo el *lugar de nacimiento*, ese programa no podrá seguir funcionando: habrá que modificarlo. Sin embargo, en un entorno de SGBD, basta modificar la descripción en el catálogo de los registros de ACTOR, y no se cambia ningún programa. La próxima vez que el programa del SGBD consulte el catálogo, tendrá acceso a la nueva estructura de los registros de ACTOR y la utilizará de forma adecuada.

La flexibilidad que proporciona esta independencia de los datos y los programas es crucial para conseguir sin esfuerzo excesivo, la adaptación continua del sistema de información a la evolución de las organizaciones.

✓ *Compartición de datos y procesamiento de transacciones multiusuario*

Un SGBD debe permitir el acceso simultáneo a la base de datos por parte de varios usuarios. Debe incluir software de control de concurrencia para asegurar que, cuando varios usuarios intenten actualizar los mismos datos, lo hagan de manera controlada, de forma que el resultado final sea correcto.

Un ejemplo sería el caso de varios encargados de realizar reservas de asientos numerados en una sala de cine: el SGBD debe asegurar que sólo un empleado tenga acceso a un asiento específico en un momento dado, para asignarlo a un cliente, y que en cuanto un empleado reserve un asiento, los demás lo vean inmediatamente. Cada operación de reserva sería una *transacción* y las aplicaciones de este tipo (de consulta y actualización *masiva* de la BD) son llamadas *aplicaciones de procesamiento de transacciones*. Una función fundamental del SGBD es asegurar que las transacciones concurrentes se realizan de manera correcta, sin interferencias entre ellas.

✓ *Manejo de múltiples vistas de los datos*

Como ya hemos indicado, un sistema de BD suele tener muchos usuarios. Algunos de ellos no deberían poder acceder a todos los datos (por cuestiones de seguridad), o simplemente no necesitan acceder más que a una parte de ellos.

Por ejemplo, en un *sistema de gestión de una productora de películas de cine*, el personal de nóminas necesita ver sólo la parte de la base de datos que contiene información acerca de los empleados de la productora; y no necesita saber nada acerca de la recaudación de las películas proyectadas en diferentes salas de cine.

Por tanto, cada usuario (o grupo de usuarios) puede necesitar una *vista* o perspectiva diferente de la BD. Una vista puede ser un subconjunto de la base de datos, o puede contener *datos virtuales* (no almacenados, sino que se derivan o calculan a partir de otros datos).

Los usuarios normalmente no necesitan saber si ven y utilizan todos o sólo parte de los datos, y tampoco si son datos derivados o no.

Un SGBD multiusuario debe proporcionar mecanismos para definir muchas vistas. Por ejemplo, es posible que a un usuario de la base de datos de nuestro ejemplo, sólo le interesen los datos relacionados con el presupuesto y gastos de los rodajes de las películas dirigidas por cada director; mientras que otro esté dedicado a realizar estudios de recaudación en taquilla de las películas en cartelera.

GASTOS	Director	Nacionalidad Director	Películas Rodadas			
			Título de Película	Presupuesto	Año de Rodaje	Gasto Realizado
	Vicente Aranda	Española	Celos	*XXXX*	1999	*XXXX*
	Fernando León	Española	Familia	*XXXX*	1996	*XXXX*
			Barrio	*XXXX*	1998	*XXXX*

Figura 2.a. Una vista de la base de datos.

TAQUILLA	Película	Películas en Cartelera			
		Ciudad	Nº de Salas	Fecha de Estreno	Recaudación Actual
	Barrio	Murcia	3	*d-m-1998*	*XXXX*
	Celos	Yecla	1	*d-m-1999*	*XXXX*
		Archena	1	*d-m-1999*	*XXXX*
		Cieza	2	*d-m-1999*	*XXXX*

Figura 2.b. Otra vista de la BD. [nota: lo que aparece como *XXXX* (cantidad de dinero) o *d-m-* (día-mes), correspondería a datos reales]

Actores en un sistema de BD

Son muchas las personas que participan en el **diseño**, **mantenimiento** y **uso** de una base de datos, sobre todo si ésta es grande. Por un lado, están los que emplean de forma cotidiana la base de datos (la información que ésta contiene) y, por otro lado, las personas ocupadas de mantener el entorno del sistema de base de datos.

Administrador de la Base de Datos ***(****DBA* **Data Base Administrator*****)***

Una de las principales razones para usar SGBD es tener un control centralizado de los datos, así como de los programas que acceden a dichos datos. La persona que tiene ese control central sobre el sistema es el *administrador de la base de datos*. Es el responsable de administrar los recursos del SBD, es decir la propia base de datos, el SGBD y el software relacionado con éste.

Las funciones del DBA incluyen las siguientes:

- Definir y modificar el esquema de la base de datos y las restricciones de los datos.
- Crear y modificar las estructuras de almacenamiento y definir los métodos de acceso.
- Autorizar el acceso a la BD, y coordinar y controlar tales accesos.
- Garantizar el funcionamiento correcto del sistema y prestar servicio técnico: se ocupa de los problemas de violación de la seguridad del sistema de BD, o de respuesta lenta del sistema.
- Definir y poner en práctica planes adecuados de copias de seguridad (*backups*) del contenido de la BD, etc.
- Adquirir los recursos necesarios de software y hardware.

Diseñador de la Base de Datos

Identifica los datos que se almacenarán y eligen las estructuras adecuadas, para representar y guardar dichos datos. Estas tareas se realizan antes de que se implemente la base de datos.

Interactúan con los futuros usuarios o grupos de usuarios de una BD, para comprender sus necesidades, desarrollar una vista de la base de datos que satisfaga los requisitos de cada grupo, y construir un diseño final que cumpla con las necesidades de todos los grupos.

Usuarios finales

Son los que necesitan tener acceso a la base de datos, para consultar sus datos o modificarlos.

- *Usuarios finales ocasionales*. Acceden a la BD eventualmente, posiblemente para obtener información diferente cada vez. Usan un lenguaje de consulta de BD para especificar sus solicitudes. Usuarios de este tipo suelen pertenecer a la plantilla de la organización en la que se ha implantado el SBD por ejemplo, al que se le dan algunas nociones de un lenguaje de consultas; no tienen por qué conocer con qué recursos cuenta el SGBD.
- *Usuarios finales paramétricos*. En su trabajo realizan consultas y actualizaciones constantes de la BD, utilizando operaciones que se han programado y probado (el personal de caja de un banco, el encargado de reservas de hotel, empleados en una empresa de reparto a domicilio). No necesitan saber con qué recursos cuenta el SGBD, sino las operaciones diseñadas para que ellos las usen.

- *Usuarios finales avanzados*. Ingenieros, científicos, analistas de empresas. Conocen los recursos del SGBD para satisfacer sus complejas necesidades. Hacen consultas a la BD desde una terminal utilizando un lenguaje de consulta (sin programas escritos) para explorar los datos en la base de datos.
- *Usuarios autónomos*. Usan BD personalizadas mediante una aplicación comercial o paquete software específico. Un usuario autónomo sería el de una aplicación de contabilidad que gestiona los datos contables de su propio negocio.

Analistas de sistemas y programadores de aplicaciones

Son profesionales informáticos que deben conocer perfectamente las capacidades y recursos del SGBD.

Los analistas determinan los requisitos de los usuarios finales (sobre todo de los paramétricos) y desarrollan especificaciones de conjuntos de operaciones (transacciones programadas) que satisfagan esos requisitos.

Los programadores implementan esas especificaciones en forma de programas de aplicación, las prueban, depuran, documentan y mantienen.

Los siguientes usuarios son las personas encargadas del diseño, creación y operación del software y entorno del sistema. No les suele interesar demasiado la BD en sí misma, es decir apenas utilizan el contenido de la base de datos para sus propios propósitos.

Diseñadores e implementadores del SGBD

Diseñan e implementan los módulos e interfaces del SGBD. Un SGBD es un sistema software compuesto de diversos componentes o módulos (que veremos más adelante). Además debe disponer de interfaces que le permitan comunicarse con otros programas, como el sistema operativo o compiladores de lenguajes de programación.

Creadores de herramientas

Son los encargados de diseñar e implementar herramientas, es decir paquetes software que facilitan el diseño y uso de los sistemas de base de datos, y que permiten aumentar el rendimiento de los mismos. Suelen adquirirse por separado. Incluyen paquetes para diseñar esquemas de bases de datos, supervisar el rendimiento, proporcionar interfaces para el usuario (de lenguaje natural o gráficos), crear prototipos, realizar simulaciones y generar datos de prueba.

Operadores y personal de mantenimiento

Personal de administración encargado del funcionamiento y mantenimiento del entorno hardware y software del sistema de base de datos.

Características deseables de un SGBD.

Un SGBD debe ofrecer una serie de (ventajosas) capacidades, que el DBA debe aprovechar para conseguir los objetivos de diseño, de administración y de uso de una BD multiusuario. Entre ellas, destacamos:

1. *Disminución y control de la redundancia de datos.*

En un sistema de procesamiento de ficheros, para cada (grupo de) usuario(s) se define e implementa los ficheros necesarios para sus propias aplicaciones (programas). De este modo, cuando varios grupos de usuarios necesitan acceder a la misma información, ésta deberá estar almacenada en varios sitios (ficheros) simultáneamente.

Por ejemplo, un grupo de usuarios sería el personal de contabilidad encargado de la gestión de las nóminas de los directores cinematográficos (interesado por tanto en los datos personales y económicos), mientras que otro grupo sería el personal de publicidad (interesado en los datos personales y profesionales –películas realizadas, etc.) de los mismos directores; los datos personales de los directores se duplicarían: se almacenarían una vez en el fichero del primer grupo y otra en el fichero del segundo. Esta situación se conoce como *redundancia de datos.*

La redundancia de datos provoca varios problemas:

- Duplicación del trabajo, pues al introducir nuevos datos en el sistema (un nuevo director) es necesario copiarlos en varios sitios (en cada archivo en el que se guarden datos de directores).
- Desperdicio del espacio de almacenamiento (mayor coste de almacenamiento).
- La obligación de controlar que, cada vez que el dato cambie, todas sus copias sean actualizadas correctamente. En caso contrario (si las copias no se actualizan, o se comete algún error al actualizarlas) se incurriría en una *inconsistencia de datos* (ver siguiente apartado).

2. Evitar inconsistencias en los datos

La inconsistencia surge cuando existen varias copias del mismo dato y tras la modificación de una de ellas, las demás no son actualizadas, o sí lo son pero de forma incorrecta.

Es posible evitar la inconsistencia de dos maneras:

- Si se elimina la redundancia. Esto se consigue si se diseña la BD de forma que *a)* se integren las vistas de los diferentes grupos de usuarios y *b)* cada dato lógico se almacene en un *único* lugar.
- Si existe *redundancia controlada.* Por ejemplo, si ocurre que siempre que se necesita obtener los datos asociados a una película, interesa obtener la nacionalidad del director que la ha rodado y el nombre del autor de su guión, podríamos almacenar de manera redundante los campos *nacioDire* y *nombreGuionista* en el fichero PELICULA. Si almacenamos juntos todos los datos, al solicitar un listado de películas ya no será necesario buscar la información en varios ficheros (PELICULA, DIRECTOR, GUION), sino únicamente en PELICULA.

  En casos de este tipo, el SGBD aplicará automáticamente al resto de copias, cualquier modificación realizada sobre un dato. Por ejemplo debe verificar que cada valor de *nacioDire* de todo registro almacenado en PELICULA, coincide con algún valor del campo *nacionalidad* en el fichero DIRECTOR, y que si se modifica el nombre de un guionista en GUION, tal cambio se lleva a los correspondientes registros en PELICULA (películas cuyo guión esté escrito por dicho guionista). Esto es la *propagación de actualizaciones.*

  Pocos sistemas de bases de datos actuales soportan la redundancia controlada.

3. Mantenimiento de la integridad

Mantener la integridad es asegurar que la información (utilizada por una aplicación de bases de datos) es correcta (es decir, refleja fielmente la realidad, el UoD).

Por tanto, no existe integridad de datos (se viola la integridad) cuando:

a) Existe *inconsistencia*.

   Esto sólo puede darse cuando existe redundancia de datos (ver apartado anterior).

b) Existe *información imposible* (40/MAY/1972 como fecha de fin de un rodaje, una película sin director) o *información falsa*, que no se ajusta a la realidad (una película con 523 actores protagonistas, cuando en la realidad son 5).

Estas situaciones se evitan si los datos cumplen las llamadas *restricciones de integridad* (RI).

Las RI más sencillas son las restricciones sobre los *tipos de datos* de los elementos de información, como por ejemplo:

- El nombre propio de un actor debe ser una cadena de hasta 25 caracteres.
- La duración de una película ha de ser un nº entero entre 1 y 180 (minutos).

Otras restricciones de integridad son las llamadas *semánticas*, pues tienen que ver con el *significado* de los datos y el UoD.

- Toda película debe estar relacionada con, al menos, un director.
- Cada película ha de tener un título diferente del resto (título único).
- Ningún actor o actriz en una película puede percibir más dinero que el protagonista.

Los diseñadores deben identificar las restricciones de integridad durante el diseño de la BD.

Algunas RI (como las que indican cuándo una fecha es válida) pueden ser verificadas de forma automática por el SGBD; así pues, el SGBD debe permitir la definición de dichas RI.

Pero para otras restricciones de integridad es necesario crear programas que verifiquen su cumplimiento; en este caso, el SGBD debe permitir la creación y ejecución de tales programas.

Evitar las violaciones de reglas de integridad es crucial en los sistemas de BD multiusuario, en los que muchos usuarios acceden a la misma información: el que uno de ellos modifique un dato o introduzca información errónea, afecta al resto de usuarios del sistema. Será necesario verificar el cumplimiento de las restricciones de integridad en cada actualización (introducción, modificación o eliminación) de datos.

Sin embargo, puede darse el caso de que se introduzca información errónea (por ejemplo escribir *Jines*, en lugar de *Ginés*) **sin** violar restricciones de integridad. En casos como este, el SGBD no es capaz de detectar el error de forma automática, pues el nombre, aunque mal escrito, es del tipo de datos correcto (una cadena de menos de 25 caracteres).

4. Aplicación de restricciones de seguridad

El que en un SBD los datos estén centralizados supone mayor peligro de accesos no autorizados, que si estuvieran almacenados en un sistema de ficheros. Es imprescindible que sólo tengan acceso al SBD los usuarios autorizados. Además, es muy habitual (y conveniente) que no todos los usuarios puedan acceder a toda la información almacenada, pues existen datos confidenciales, que sólo ciertas personas pueden ver o utilizar. Incluso es posible que ciertos usuarios sólo estén autorizados para obtener (consultar o ver) los datos, mientras que otros puedan actualizarlos además de consultarlos.

El SGBD debe disponer de un robusto *subsistema de seguridad y autorización*, mediante el cual el DBA pueda:

- Crear cuentas de usuario protegidas con contraseñas (para asegurar que sólo acceden a los datos, los usuarios que tengan permiso para ello).

- Crear restricciones para cada cuenta, de forma que se controle *a)* a qué datos tiene acceso el usuario y *b)* el tipo de operaciones que puede realizar sobre esos datos (es decir, si puede verlos o modificarlos o crear nuevos o eliminarlos).

El SGBD obligará (de forma automática) el cumplimiento de estas restricciones.

Otros controles de seguridad son, por ejemplo, que sólo el DBA pueda usar el software de administración y monitorización de la BD (para crear nuevas cuentas de usuario, por ejemplo), o bien que los usuarios paramétricos sólo puedan acceder a la BD mediante los programas que se crearon para ellos.

5. Suministro de múltiples interfaces de usuario

Puesto que las BD son usadas por muchos usuarios, con variados niveles de conocimientos técnicos, el SGBD debe ofrecer diferentes interfaces, para todos ellos.

- Lenguajes de consulta [usuarios ocasionales]
- Interfaces de Lenguajes de Programación [programadores de aplicaciones]
- Formularios (*forms* o pantallas) y comandos (órdenes) [usuarios paramétricos]
- Interfaces controladas por menús y lenguaje natural [usuarios autónomos]

6. Representación de interrelaciones complejas entre los datos

Los datos en la BD están relacionados entre sí de diversas formas. Por ejemplo, el registro de *Ernesto Alterio* en el fichero ACTOR, puede estar relacionado con varios registros del fichero PELICULA (*Insomnio, Cuarteto de La Habana, Los lobos de Washington*). Por otro lado, cada registro de PELICULA (*Celos*) se relaciona con un registro de DIRECTOR (*Vicente Aranda*) y con varios de ACTOR (*Aitana Sanchez-Gijón, Daniel Giménez Cacho, Maria Botto, ...* uno por cada actor participante en la película en cuestión).

El SGBD debe permitir la representación de estas interrelaciones entre los datos, así como la obtención y actualización de datos que estén relacionados (obtener todas las películas y sus directores en las que actúe).

7. Respaldo y recuperación

Todo SGBD debe contar con recursos para recuperarse de fallos de hardware o de software. De esto se encargará el *subsistema de respaldo y recuperación* del SGBD. Si el fallo ocurre mientras estaba en marcha un programa que actualizaba gran cantidad de datos, el subsistema de recuperación debe asegurar (una vez el sistema ha sido reiniciado tras el fallo) que:

a) la base de datos se restaura al estado en que estaba justo antes de comenzar el programa,

o bien que

b) el programa continúa su ejecución por el punto en donde la dejó cuando se produjo el fallo, y finaliza su trabajo correctamente.

Otras ventajas del enfoque de bases de datos

1. *Datos compartidos actualizados*

Los datos están disponibles para todos los usuarios y cuando alguno actualiza la información, los demás ven los cambios inmediatamente. Esto es posible gracias a los subsistemas de control de concurrencia y recuperación del SGBD.

2. *Flexibilidad*

Cuando los requisitos del sistema varían o surgen nuevas necesidades de datos, normalmente es necesario modificar la estructura de la base de datos, como ocurre cuando es necesario añadir un nuevo fichero (para almacenar animales actores) o ampliar un fichero ya existente (un nuevo campo en el fichero PELICULA para indicar si ha salido a la venta en vídeo). Algunos SGBD permiten realizar estos cambios en la estructura de la BD sin afectar ni a los datos ya almacenados, ni a los programas de aplicación ya existentes.

3. *Rápida creación de nuevas aplicaciones*

Diseñar e implementar una BD desde cero cuesta bastante más que crear una sola aplicación de ficheros tradicional; pero una vez que la BD está creada y en funcionamiento, crear una aplicación nueva (como la obtención de ciertos datos para imprimir un informe nuevo) necesita de mucho menos tiempo.

4. *Cumplimiento de reglas o normas de empresa*

El DBA puede establecer normas a los usuarios de la BD de una gran organización, de forma que se potencie el intercambio de información y la cooperación entre departamentos, proyectos, etc. Es posible establecer un estándar para los nombres y formatos de los elementos de datos, para la estructura de la documentación, etc.

5. *Proporcionar independencia de datos*

(se verá más adelante)

Anexo: Cronología de los Sistemas de BD

1.2 CONCEPTOS Y ARQUITECTURA DEL SISTEMA DE BASES DE DATOS.

Modelos de datos y esquemas

Una característica fundamental (y un objetivo importante) del enfoque de bases de datos es proporcionar al usuario una visión abstracta de los datos, es decir, ocultarle detalles de almacenamiento y mantenimiento de los datos, que no necesita conocer.

Para conseguir la abstracción de datos se utilizan los modelos de datos. Un **modelo de datos** sirve para describir la estructura de una BD, es decir los tipos de datos, las interrelaciones entre ellos y las restricciones que deben cumplir.

Tipos de modelos de datos

Una forma de clasificar los modelos de datos es la basada en los conceptos que ofrecen para describir la estructura de la BD.

- ✓ Los ***modelos** de alto nivel o **conceptuales*** constan de conceptos muy cercanos al modo en que el usuario percibe la realidad, y describen ésta como un conjunto de entidades y las relaciones entre ellas. Usan conceptos como:

  Entidad: representa una cosa, objeto o concepto del mundo real (un director, una película) almacenado en la base de datos.

  Atributo: representa una propiedad interesante de alguna entidad (nombre del director, título de la pelicula).

  Interrelación: describe una asociación entre varias entidades (vínculo que existe entre una película y su director).

  · Un modelo de alto nivel es el Modelo Entidad / Interrelación (MER, *entity-relationship*) que estudiaremos con profundidad más adelante.

  · Los *modelos de datos orientados a objetos* también suelen usarse como modelos conceptuales de alto nivel, sobre todo en el área de la Ingeniería del Software, pero este tipo de modelos caen fuera del ámbito de esta asignatura.

- ✓ Los ***modelos** de implementación* o ***lógicos***, proporcionan conceptos que pueden ser entendidos por los usuarios finales, aunque no están muy alejados de cómo los datos se organizan dentro del ordenador. Ocultan algunos detalles de almacenamiento, pero sus conceptos pueden implementarse directamente en un sistema informático.

  Son los más utilizados en los SGBD comerciales actuales (como Oracle).

  · Los más comunes son el *modelo relacional* (que estudiaremos con detalle más adelante), el de *red* y el *jerárquico*. Son *modelos de datos basados en registros*, porque usan estructuras de registros para representar los datos.

  · Los *modelos de datos orientados a objetos* pueden ser considerados modelos de implementación, aunque de un nivel próximo a los modelos conceptuales.

- ✓ Los ***modelos** de bajo nivel o **físicos*** disponen de conceptos que describen los detalles de almacenamiento de los datos en el ordenador. Estos conceptos no están dirigidos a los usuarios finales, sino a usuarios más especializados. Describen cómo se almacenan los datos, indicando el formato y el ordenamiento de los registros y los caminos de acceso. Un *camino de acceso* es una estructura que permite realizar búsquedas de datos de forma eficiente (por ejemplo, un *fichero índice*). Más adelante, estudiaremos técnicas de almacenamiento y estructuras de acceso.

Esquemas de base de datos

En un modelo de datos es preciso diferenciar entre *descripción de base de datos* y la *base de datos* en sí.

La descripción es el **esquema** de la base de datos (los metadatos), el cual se especifica en el diseño y rara vez es modificado. El **diagrama del esquema** es la representación de un esquema utilizando la notación de algún modelo de datos.

DIRECTOR

nombreDirector	nacionalidad	fechaNacimiento	numPeliculas

PELICULA

tituloPelicula	director	género	guión	añoRodaje	nacionalidad	duración

Figura 3. Una posible representación (diagrama del esquema) de los esquemas de dos tipos de registro.

El diagrama del ejemplo representa la estructura de los **tipos de registro** DIRECTOR y PELICULA, pero no su contenido (los datos sobre directores y películas reales, almacenados en la BD). Cada tipo de registro (DIRECTOR, o PELICULA) es un **objeto** o **elemento del esquema**.

Otra posible representación sería:

Director (nombreDirector, nacionalidad, fechaNacimiento, numPeliculas)

Pelicula (tituloPelicula, director, género, guión, añoRodaje, nacionalidad, duración)

Un diagrama de esquema sólo muestra *parte* del esquema. El del ejemplo muestra los nombres de los tipos de registro y de los elementos de información (características o atributos de cada objeto), pero no describe restricciones tales como los tipos de datos de cada atributo, o la interrelación entre cada película y su director. Los diagramas del Modelo Entidad/Interrelación sí permiten la representación de este tipo de restricciones.

Los datos reales cambian muy a menudo (cada vez que introducimos un nuevo director o se elimina una película, o aumenta el número de películas rodadas por un director). El **estado de la base de datos** es el conjunto de datos que contiene en un momento determinado, es decir el conjunto de *instancias* (ejemplares, ocurrencias) que contiene. En un estado de la BD, cada elemento del esquema tiene su propio conjunto actual de instancias. Por ejemplo, en cierto instante, el elemento PELICULA podría contener estas dos instancias, correspondientes a las películas tituladas *Torrente* y *The Matrix*:

Torrente	Santiago Segura	Comedia Casposa	Santiago Segura	1997	España	110
The Matrix	Andy Wachowski	Ciencia-ficción	Andy Wachowski	1999	EEUU	138

Mientras que en un instante posterior puede contener tres instancias más: *Episodio I : la amenaza fantasma*, *Celos* y *Locos en Alabama*:

Torrente	Santiago Segura	Comedia Casposa	Santiago Segura	1997	España	110
The Matrix	Andy Wachowski	Ciencia-ficción	Andy Wachowski	1999	EEUU	138
Episodio I: la amenaza fantasma	**George Lucas**	**Ciencia-ficción**	**George Lucas**	**1999**	**EEUU**	**133**
Celos	**Vicente Aranda**	**Drama**	**Álvaro del Amo**	**1999**	**España**	**122**
Locos en Alabama	**Antonio Banderas**	**Comedia**	**Mark Childress**	**1999**	**España**	**108**

Más tarde, podemos eliminar la instancia *Torrente* y cambiar el valor del campo *nacionalidad* de la película *Locos en Alabama* para introducir su valor correcto:

The Matrix	Andy Wachowski	Ciencia-ficción	Andy Wachowski	1999	EEUU	138
Episodio I: la amenaza fantasma	George Lucas	Ciencia-ficción	George Lucas	1999	EEUU	133
Celos	Vicente Aranda	Drama	Álvaro del Amo	1999	España	122
Locos en Alabama	Antonio Banderas	Comedia	Mark Childress	1999	**EEUU**	108

Los tres estados anteriores corresponden a un mismo esquema de la base de datos en la que está PELICULA.

Para definir una nueva base de datos, es necesario especificar su esquema en el SGBD, el cual lo almacena en su catálogo. En ese momento, la base de datos está en el *estado vacío*: no contiene datos. Cuando se introducen (se cargan) los datos iniciales (aquellos necesarios para que comiencen a trabajar las aplicaciones), la BD pasa al *estado inicial*. Cada vez que insertamos o eliminamos una instancia, o modificamos el valor de alguno de sus elementos de información, la base de datos cambia de un estado a otro.

El SGBD asegura que todo estado de BD es válido o consistente, es decir que satisface la estructura y restricciones especificadas en el esquema.

Abstracción de datos: arquitectura de tres esquemas de un SGBD

Para que el sistema sea útil, debe recuperar los datos de forma eficiente. Esta necesidad ha llevado al diseño de estructuras de datos complejas, para la representación de los datos en la base de datos. Como muchos usuarios de sistemas de BD no están familiarizados con ordenadores, los desarrolladores ocultan la complejidad a los usuarios a través de varios niveles de abstracción y así simplificar la interacción de los usuarios con el sistema.

La arquitectura de tres esquemas para sistemas de bases de datos ayuda a la consecución de dos de los objetivos (características) del enfoque de bases de datos: la separación entre los programas y los datos y el manejo de múltiples vistas de usuario.

Arquitectura ANSI/X3/SPARC [2]

El objetivo de esta arquitectura es separar las aplicaciones del usuario de la base de datos física. Los **esquemas** pueden ser definidos en **tres niveles**:

1. **Nivel Interno**.

   Es el nivel más bajo de abstracción. Tiene un **esquema interno** (EI) que describe *cómo* se almacenan realmente los datos, utilizando un **modelo físico de datos**, y muestra detalles de la organización física de los ficheros (estructuras físicas de almacenamiento, orden de secuencia de los registros físicos, tamaño de página, de bloque, etc.) y caminos de acceso (tipos de índice, etc.).

2. **Nivel Conceptual**.

   Tiene un **esquema conceptual** (EC), que describe la estructura de *toda* la BD para el conjunto de usuarios. El EC oculta los detalles físicos y describe *qué* datos se almacenan en la base de datos y *qué* relaciones existen entre ellos, es decir entidades, tipos de datos, interrelaciones, operaciones de los usuarios y restricciones (seguridad, integridad).
   En este nivel puede utilizarse un **modelo de datos lógico** o uno de **alto nivel** para describir el esquema conceptual.

3. **Nivel Externo o de Vistas**.

   Es el nivel más alto de abstracción y describe sólo parte de la base de datos. A muchos usuarios no les preocupara toda la información almacenada en la base de datos, sino que necesitan acceder a sólo una parte. Para simplificar su interacción con el sistema, se define el nivel de abstracción de vistas, que está compuesto de varios **esquemas externos** (EE) o **vistas** de usuario. Cada EE describe la parte de la BD que interesa a un grupo de usuarios determinado (la porción de la realidad que perciben), ocultándoles el resto de la BD. Para el grupo de usuarios, su vista *es* la base de datos.
   En este nivel puede usarse un **modelo de datos de alto nivel** o uno **lógico** para describir cada esquema externo.

Nota: los tres esquemas son descripciones de datos, puesto que los *datos reales* están (sólo) en el nivel físico.

[2] ANSI/X3/SPARC es un grupo de estudio sobre sistemas de administración de bases de datos, del *Standard Planning And Requirements Committee* (SPARC) del ANSI (*American National Standards Institute*), dentro del comité X3 que se encarga de informática y ordenadores.

Los *SGBD comerciales actuales* no distinguen del todo los tres niveles. Algunos de ellos incluyen detalles del nivel físico en el esquema conceptual. Los esquemas externos (vistas) suelen especificarse mediante el mismo modelo de datos que se usa para describir el esquema conceptual.

Correspondencia entre esquemas (transformación o mapping)

En un SGBD con esta arquitectura de tres esquemas, cada grupo de usuarios trabaja únicamente con su esquema externo, así que el SGBD debe traducir una solicitud expresada en términos de un esquema externo a una petición en un esquema conceptual, y luego a una solicitud expresada en el esquema interno, la cual se procesará sobre la BD física. Si la solicitud era la obtención de ciertos datos, será preciso que el SGBD modifique el formato de la información extraída de la BD física para que coincida con la vista del usuario.

Independencia respecto a los datos

La independencia con respecto a los datos es la *capacidad para modificar el esquema en un nivel del SBD sin tener que modificar el esquema del nivel inmediato superior*. Existen dos tipos de independencia de datos:

1. *Independencia lógica con respecto a los datos.*

   Capacidad de modificar el esquema conceptual (su estructura) sin alterar los esquemas externos (lo que ven los usuarios), ni (el código de) los programas de aplicación.

   Las modificaciones en el nivel conceptual son necesarias siempre que la estructura lógica de la base de datos se altere. Por ejemplo puede modificarse el esquema conceptual para ampliar la base de datos al añadir un nuevo tipo de registro ANIMAL_ACTOR, o al añadir un elemento de información, como el campo *nacioDire* (nacionalidad del director) o el campo *enVideo* para PELICULA.

PELICULA

tituloPelicula	director	género	guión	añoRodaje	nacionalidad	duración	**nacioDire**	**enVideo**

Figura 4. Modificación del esquema del tipo de registro PELICULA: adición de dos nuevos campos de información.

   También puede reducirse la base de datos, por ejemplo eliminando un tipo de registro, o un campo de información; en este caso, la modificación sólo debe afectar a los esquemas externos que utilicen los elementos que desaparecen (es decir, *esos* sí deberán ser modificados, claro está).

   Por ejemplo, el esquema externo de la figura 2.a. no debería alterarse si se modificara el esquema PELICULA (figura 3) para convertirlo en el esquema de la figura 4.

   Sólo será necesario modificar la *definición* de la vista, o sea, indicar al SGBD de dónde debe obtener los datos que debe mostrar en la vista *Gastos*: antes tomaba la nacionalidad de cada director a partir del contenido de DIRECTOR, y ahora debe obtenerla directamente de PELICULA (del campo *nacioDire*).

   Es decir, se modifica la correspondencia entre el esquema externo y el esquema conceptual (de este modo no cambian las vistas, pues siguen mostrando la misma información, ni las aplicaciones que usan directamente dichas vistas sin importar de dónde extraen éstas los datos). Las aplicaciones que utilizan los elementos del EE funcionarán igual que antes de modificar la estructura del esquema conceptual (por ejemplo, una consulta como "*listar todas las películas de Alex de la Iglesia rodadas antes de 1997*").

   Además, también pueden modificarse las restricciones en el esquema conceptual, sin afectar a los esquemas externos.

La independencia de datos lógica es más difícil de conseguir que la independencia de datos física (que vemos a continuación), ya que los programas de aplicación son fuertemente dependientes de la estructura lógica de los datos a los que acceden.

2. *Independencia física con respecto a los datos*.

Es la *capacidad de modificar el esquema interno sin alterar el esquema conceptual (o los esquemas externos), ni los programas de aplicación*.

Las modificaciones en el nivel interno suelen ser necesarias para mejorar el funcionamiento, por ejemplo es posible que haya que añadir una nueva estructura de acceso al esquema interno, para mejorar el rendimiento de operaciones de obtención y actualización; si la base de datos sigue conteniendo los mismos datos, no habrá que modificar el esquema conceptual ni los programas de aplicación.

Este tipo de independencia es más fácil de lograr que el anterior (independencia lógica), puesto que consiste en la separación entre las aplicaciones y las estructuras físicas de almacenamiento. Por ejemplo, si se añade un camino de acceso nuevo para incrementar la eficiencia de la consulta "*obtener los registros de PELICULA agrupados por director y año*", no será necesario modificar ninguna consulta (ni aplicación) del tipo "*listar todas las películas rodadas por Francis F. Coppola en 1972*", aunque (por supuesto) el SGBD la ejecutará con mayor rapidez si utiliza el nuevo camino de acceso.

El concepto de *independencia de datos* es similar al concepto de *tipos abstractos de datos* en los lenguajes de programación modernos: ambos ocultan los detalles de implementación a los usuarios, de forma que pueden concentrarse en la estructura general, más que en los detalles de implementación de nivel más bajo.

En todo SGBD de múltiples niveles de abstracción, se amplía el *catálogo* para incluir información sobre cómo hacer corresponder entre los diferentes niveles, tanto las operaciones como los datos.

Se consigue la independencia de datos porque, al modificar el esquema en algún nivel, el esquema del nivel inmediato superior no varía y sólo cambia la correspondencia entre los niveles. Además, no hace falta modificar los programas de aplicación que usan el esquema.

De este modo, la arquitectura de tres niveles ayuda a conseguir la independencia de los datos, tanto física como lógica. Pero las correspondencias entre niveles suponen un gasto extra durante la compilación o ejecución de una consulta o programa, lo cual disminuye la eficiencia del SGBD. Por esto, pocos SGBD comerciales han implementado la arquitectura de tres esquemas completa y en muchos de ellos no existe una rigurosa separación entre los niveles.

Lenguajes e interfaces de bases de datos

El SGBD debe proporcionar lenguajes e interfaces apropiados para cada tipo de usuarios.

Lenguajes del SGBD

Cuando se ha terminado el diseño de la BD y se ha elegido el SGBD concreto en el que implementar la BD, hay que especificar los esquemas conceptual e interno de la base de datos, así como las correspondencias entre ellos.

En muchos SGBD en los que no existe una separación estricta entre niveles, el DBA y los diseñadores usan un mismo lenguaje, llamado **lenguaje de definición de datos** (**DDL**: *data definition language*) para especificar los dos esquemas.

El SGBD dispone de un **compilador del DDL** encargado de procesar sentencias escritas en DDL para identificar las descripciones de los elementos de los esquemas y almacenar dicha descripción del esquema en el catálogo del SGBD.

En caso de que el SGBD distinga claramente entre los niveles conceptual e interno, el DDL sólo se utilizará para especificar el esquema conceptual (entidades, interrelaciones entre ellas y restricciones). Y se emplea el **lenguaje de definición del almacenamiento** (**SDL**: *storage definition language*) para especificar el esquema interno (las estructuras de almacenamiento y los métodos de acceso). Las correspondencias entre los dos esquemas puede escribirse en cualquiera de los dos lenguajes.

Para conseguir una verdadera arquitectura de tres esquemas, sería necesario un tercer **lenguaje de definición de vistas** (**VDL**: *view definition language*) para especificar las vistas de usuario y sus correspondencias con el esquema conceptual.

Después de compilar los esquemas de la base de datos y de introducir datos en la misma, los usuarios necesitan disponer de algún medio para acceder a dichos datos. Las operaciones más comunes son la *obtención* (recuperación o consulta), la *inserción* (introducción), la *eliminación* y la *modificación* de datos. Para esto, el SGBD proporciona un **lenguaje de manipulación de datos** (DML: *data manipulation language*).

Los SGBD comerciales actuales no disponen de varios lenguajes, sino que ofrecen un **único lenguaje** que cuenta con elementos para *definir* esquemas conceptuales y vistas, *manipular* datos y definir su estructura de almacenamiento. Por ejemplo el sistema gestor de bases de datos relacionales (SGBDR) **Oracle** proporciona un lenguaje mezcla de DDL, DML, VDL y SDL.

Existen dos **tipos de DML**: de alto nivel (no procedimentales o declarativos) y de bajo nivel (procedimentales).

a. DML no procedimental

- Requiere que el usuario especifique QUÉ datos necesita obtener o actualizar, SIN especificar CÓMO obtenerlos o modificarlos (es un lenguaje *declarativo*). Un ejemplo es el lenguaje SQL.
- Puede utilizarse de forma independiente para realizar operaciones complejas de base de datos.
- Es posible usarlo de dos formas:
  - · Interactivamente (desde una terminal),
  - · Incorporado en un programa escrito en un lenguaje de programación de propósito general, como C o Pascal (*DML embebido*).
- Puede recuperar y actualizar muchos registros con una sola sentencia (*DML orientado a conjuntos*).

b. DML procedimental

- Requiere que el usuario especifique QUÉ datos necesita obtener o modificar y CÓMO obtener y actualizar tales datos.
- Siempre debe estar incrustado (*DML embebido*) en el código de un programa escrito en un lenguaje de programación de propósito general.
- Normalmente sólo permite obtener uno a uno los registros de la BD para procesarlos por separado (*DML orientado a registros*)
- Así pues, necesita usar los elementos de dicho lenguaje como *los bucles*, para poder obtener cada uno de los registros del conjunto de los que interesan almacenados en la BD, y para procesarlo individualmente.

Siempre que las sentencias en DML se incorporen en un programa escrito en cierto lenguaje, a este último se le llama ***lenguaje host*** (o anfitrión), y al DML sublenguaje de datos o *lenguaje embebido*[3].

Cuando el DML no procedimental se usa de forma independiente e interactiva, se llama **lenguaje de consulta** (aunque sea usado tanto para consultar (obtener) datos como para actualizarlos).

[3] También suele denominarse lenguaje *incrustado*, o *empotrado*.

Los *usuarios ocasionales* utilizan un lenguaje de consulta para especificar sus solicitudes de información. Los *programadores* suelen utilizar el DML embebido en algún lenguaje de programación (Cobol, PL/I, Pascal, C). Los *usuarios paramétricos* usan interfaces *amigables* que les permiten interactuar con la base de datos; también pueden usarlas los *usuarios ocasionales* y aquellos no interesados en aprender un lenguaje de consulta.

Interfaces del SGBD

1. *Interfaces basadas en menús*.

   Presentan al usuario una ventana con una lista de opciones de menú que lo guían para formular solicitudes. Así no debe memorizar órdenes ni aprender la sintaxis de un lenguaje de consulta.

2. *Interfaces gráficas*.

   Presentan al usuario los esquemas de la BD en forma de diagramas y el usuario puede formular consultas manipulando el diagrama. Suelen combinarse con menús.

3. *Interfaces basadas en formularios en pantalla (forms).*

   Presentan un formulario que muestra espacios en blanco que puede rellenar el usuario para introducir registros nuevos, o bien el usuario puede rellenar sólo algunos campos para recuperar todos los registros cuyo contenido coincida con los datos especificados. Los formularios suelen ser diseñados y programados para usuarios paramétricos.

4. *Interfaces de lenguaje natural*.

   Aceptan solicitudes escritas en inglés u otro idioma y tratan de “entenderlas”. Si la traducción tiene éxito, la interfaz genera la correspondiente consulta de alto nivel y la envía al SGBD para que la procese; si no se puede interpretar la solicitud, se dialoga con el usuario para aclarar la solicitud.

5. *Interfaces para usuarios paramétricos.*

   Este tipo de usuarios dispone de un pequeño conjunto de operaciones que realizan muchas veces (como ocurre con los empleados de banco que atienden al público en las ventanillas). Los analistas de sistemas y los programadores diseñan e implementan una interfaz especial para cada clase de usuarios paramétricos: esta interfaz incluye un conjunto de órdenes abreviadas, para reducir al mínimo el número de teclas necesarias que es necesario presionar, o de pulsaciones de ratón, para invocar cada operación y ganar rapidez.

6. *Interfaces para el DBA*.

   Estas interfaces permiten al DBA invocar órdenes privilegiadas del SGBD, que sólo él puede ejecutar, por ejemplo a través de ellas el DBA creará nuevas cuentas de usuario, establecerá los parámetros de ajuste del rendimiento del sistema, concederá permisos de acceso a las cuentas, modificará los esquemas de la base de datos o las correspondencias entre ellos, accederá al catálogo, reorganizará la estructura de almacenamiento de la BD, realizará o restaurará copias de seguridad.

1.3 ESTRUCTURA GENERAL DEL SISTEMA DE BASE DE DATOS.

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, pero en general los sistemas operativos sólo proporcionan los servicios más básicos y los SGBD deben construirse sobre esta base (por esto el diseño de un SGBD debe considerar la interfaz entre el SGBD y el SO).

Módulos componentes de un SGBD

Los siguientes son los componentes de procesamiento de consultas:

- ✓ El **compilador de consultas** trata las consultas de alto nivel (escritas en DML) que se introducen de forma interactiva, analiza su sintaxis y genera llamadas al *procesador de consultas* para que las ejecute.
- ✓ El **precompilador de DML embebido** extrae las sentencias en DML de un programa escrito en un lenguaje host y las envía a un *compilador de DML* para convertirlas en código objeto para el acceso a la BD; el resto del programa se envía al *compilador del lenguaje host*. El código objeto de las sentencias en DML se enlaza (*link*) con el código objeto del resto del programa, formando una *transacción programada* cuyo código ejecutable incluye llamadas al *procesador de consultas* de la base de datos.
- ✓ El **compilador de DDL** procesa las definiciones de esquema escritas en DDL, y almacena las descripciones de los esquemas (metadatos) en el catálogo del SGBD. Otros módulos del SGBD necesitan conocer la información contenida en el catálogo.
- ✓ El **procesador de consultas** en tiempo de ejecución se encarga de recibir solicitudes de recuperación o actualización, las optimiza (encontrando así una buena estrategia para ejecutar la consulta) y las ejecuta sobre la base de datos. El acceso a los datos (a disco) se realiza mediante el *gestor de datos almacenados*.

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 base de 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.

  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*) de 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, 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 almacena metadatos acerca de la estructura de la base de datos.
- **Estructuras de acceso** (ficheros 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. El procesador de consultas utiliza esta información para seleccionar las formas eficientes para ejecutar una consulta.

Utilidades del sistema de base de datos

El SGBD es el componente software más importante en un sistema de base de datos, pero no el único. Existen otras aplicaciones o utilidades que ayudan al DBA a manejar el sistema. Estas utilidades realizan las siguientes funciones:

- *Utilidades para Carga*. Se utiliza para cargar ficheros de datos ya existentes (de texto, por ejemplo) en la base de datos. Normalmente se indica el formato de los datos del fichero fuente y el que deben tener en la base de datos destino, y la utilidad de carga (herramienta de conversión) convierte los datos de un formato a otro para almacenarlos en la BD. Esto permite el intercambio de información entre diferentes SGBD comerciales (por ejemplo de *Oracle* a *Access*).
- *Utilidades para Respaldo*. Crean una copia de seguridad (*backup*) del contenido de la base de datos. Esta copia puede utilizarse para restaurar la BD después de un fallo general del sistema.
- *Utilidades para Reorganización de ficheros*. Permiten modificar la organización de los ficheros de la BD, para mejorar el rendimiento. Pueden incluir utilidades para ordenar y comprimir ficheros.
- *Utilidades para Monitorización* o vigilancia del rendimiento. Permiten supervisar el uso de la BD y proporcionan datos estadísticos al DBA, que los utiliza para tomar decisiones con el objetivo de mejorar el rendimiento o funcionamiento del sistema de base de datos.
- *Utilidades para control de accesos* de los usuarios de la base de datos.
- *Utilidades para acceso al Diccionario de Datos* (ver más adelante).

Recursos de comunicaciones

El SGBD interactúa con el software de comunicaciones, el cual permite que usuarios remotos[4] accedan al sistema de base de datos mediante

- terminales de ordenador,
- estaciones de trabajo (*workstation*),
- ordenadores personales (PCs)

los cuales se conectan al ordenador en el que reside la base de datos, a través de

- redes de larga distancia
  - cable (teléfono, fibra óptica...)
  - dispositivos de comunicación por satélite
- redes de área local - LAN - *Local Area Network* (cable coaxial, fibra óptica...)

Muchos SGBD comerciales disponen de paquetes (software) de comunicaciones, que funcionan conjuntamente con el SGBD. El sistema integrado por el SGBD y el sistema de comunicaciones de datos suele denominarse sistema DB / DC (*data base/data communication*).

[4] Usuarios NO locales, es decir, los puestos desde los que acceden a los datos almacenados están localizados físicamente lejos del equipo en el que reside la base de datos.

Anexo 1. Clasificación de los SGBD.

a) Según el **modelo de datos** en el que está basado.

1. *Relacional*
2. En *Red*
3. *Jerárquico*
4. *Orientado a Objetos*
5. Otros: *Objeto-Relacional*

b) Según el **número de usuarios** a los que da servicio simultáneamente.

1. *Monousuario*
2. *Multiusuario*

c) Según el **número de lugares** en que se **almacenan** los datos.

1. *Centralizado*. La base de datos y el SGBD residen en un único ordenador.
2. *Distribuido* (SGBDD). La base de datos y el software del SGBD pueden estar repartidos en varios sitios conectados en red. La base de datos distribuida (BDD) es una colección de datos que pertenece *lógicamente* al mismo sistema, pero que *físicamente* está dispersa en varios sitios de una red de ordenadores.

   2.a. *SGBDD Homogéneo*. Usa el mismo software de SGBD en todos los sitios.

   2.b. *SGBDD Heterogéneo*. Cada sitio puede tener un *software* de SGBD distinto. En particular, el llamado *SGBD Federado o Multi-Base de datos* suele construirse a partir de varios sistemas de base de datos ya existentes (diferentes o no entre sí).

   Este es un tipo de SGBD híbrido entre centralizado y distribuido. Por un lado, un usuario de cierto sistema de BD puede acceder al éste de forma local, como si de un sistema centralizado se tratase. Por otro lado, el sistema visto globalmente es un sistema distribuido, puesto que un usuario puede acceder a los datos almacenados en cualquier otro sitio (ser cliente de cualquier sistema componente de la multi-base de datos).

   Así, los SGBD no son del mismo tipo, son independientes entre sí, están débilmente acoplados y tienen cierto grado de autonomía local (es decir, se permite a las transacciones locales tener acceso directo a su propio SGBD).

d) Según su **propósito**

1. *Propósito General*. Cualquier aplicación puede comunicar con él para acceder a la información de la base de datos.
2. *Propósito Específico*. Es decir, construido para un *tipo determinado* de aplicaciones cuyo rendimiento es muy importante, como ocurre con las denominadas OLTP (*online transaction processing*), que son aplicaciones que consisten en un gran número de transacciones de actualización de datos, que deben ejecutarse de forma concurrente y sin retrasos excesivos. Un ejemplo sería una aplicación encargada de la reserva de plazas en vuelos de diferentes líneas aéreas.

Anexo 2. El Catálogo y el Diccionario de Datos del Sistema.

El *diccionario de datos* almacena información más amplia y variada que el *catálogo* de la BD. Aunque en esta asignatura nos centraremos en el estudio del catálogo del sistema y no los sistemas de diccionario de datos generales, veamos algunas cuestiones interesantes acerca de ambos términos.

El catálogo del sistema

El catálogo del sistema es una mini-base de datos que almacena los esquemas y descripciones de las bases de datos que mantiene (gestiona) el SGBD.

Cada BD está descrita por los datos almacenados en el catálogo (llamados metadatos, es decir, datos sobre los datos, que describen su estructura, restricciones, autorizaciones, etc.).

Contiene una descripción del esquema conceptual (EC), del esquema interno (EI) y de los esquemas externos (EE), y de las correspondencias entre ellos. Además contiene la información necesaria para el procesador de consulta y los subsistemas de seguridad y autorización.

El sistema de diccionario de datos

Un sistema de diccionario de datos (SDD) es un mini-SGBD que gestiona los **metadatos** del SBD (es decir, la información contenida en el catálogo, relativa a los esquemas, restricciones, autorizaciones, etc.), **junto con** otro tipo de información:

- **decisiones de diseño** y resultados de cada fase del diseño de la BD
- **normas** de uso
- **descripción de programas** de aplicación, y transacciones
- información sobre los **usuarios** y **documentación** existente

Un SDD útil permitirá almacenar y controlar:

a. Descripciones de los *esquemas* del SBD
b. 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.
c. Descripción de los *usuarios*, sus *responsabilidades* y *derechos de acceso*, etc.
d. Descripciones de alto nivel de las transacciones y aplicaciones de la BD, y de las interrelaciones entre los usuarios y las transacciones.
e. 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.
f. Cifras estadísticas de uso, frecuencias de consultas, transacciones, nº de accesos a los datos

Esta información está disponible para el administrador de la base de datos (DBA), los programadores de aplicaciones y los usuarios autorizados, de forma que permite:

- el control del sistema de base de datos, por parte del DBA
- la mayor comprensión (entendimiento) y aprovechamiento, por parte de los programadores y usuarios.

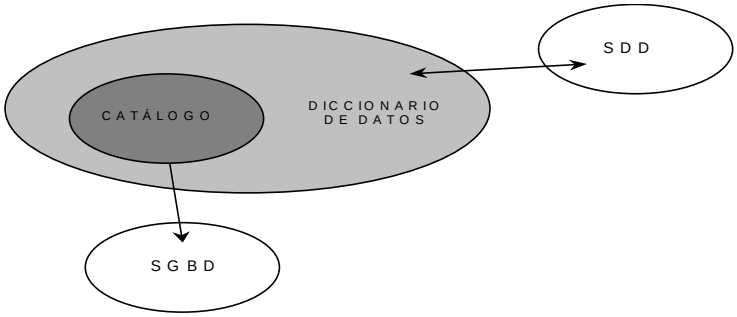
Es importante destacar por un lado que el catálogo está fuertemente acoplado al SGBD. Proporciona información a los usuarios y (sobre todo) al DBA, pero fundamentalmente es utilizado por ciertos módulos del SGBD. El catálogo se usa con mucha frecuencia, por lo que es muy importante que sea diseñado e implementado de forma tal que el acceso al mismo sea muy eficiente.

Los módulos software componentes del SGBD que utilizan el catálogo son:

- compiladores y precompiladores de DDL y DML
- procesador (optimizador) de consultas
- subsistema de integridad y seguridad.
- gestor de transacciones

Y, por otro lado, el sistema de diccionario de datos es un paquete de software autónomo. Puede interactuar con los módulos del SGBD citados anteriormente, así como con programas de aplicación y generadores de informes, pero es utilizado principalmente por el DBA, los usuarios finales autorizados y los programadores de aplicaciones.

En grandes organizaciones, el sistema de diccionario de datos se considera tan importante como un SGBD.



BIBLIOGRAFÍA BÁSICA

[EN97] Elmasri, R.; Navathe, S.B.: **Sistemas de bases de datos. Conceptos fundamentales**. 2ª Edición. Addison-Wesley Iberoamericana, 1997.

[dMP93] De Miguel, A.; Piattini, M.: **Concepción y diseño de bases de datos: del Modelo E/R al Modelo Relacional**. RA-MA, 1993.

[Korth98] Korth, H; Silberschatz, A., Sudarshan, S.:**Fundamentos de bases de datos.** 3ª Edición. McGraw-Hill, 1998.