Tema 1. Introducción

En este tema se definen los conceptos básicos de las bases de datos, relacionándolos con los Sistemas de gestión de archivos. Concretamente, se hace especial énfasis en los conceptos de Base de datos, Sistema de gestión de bases de datos y Sistema de bases de datos. Al explicar estos términos, se describen los objetivos que persiguen los Sistemas de bases de datos, así como las implicaciones que supone su uso. Asimismo, se introduce el concepto de abstracción de datos, y se describe la arquitectura ANSI/SPARC de tres niveles como una forma de obtener independencia de datos. También se hace una breve introducción a los modelos de datos, aunque éstos serán tratados con profundidad en temas posteriores. Por último, se introducen otros términos de interés como la arquitectura de un Sistema de base de datos, explicando sus principales componentes y usuarios.

1.1 Datos e información

En primer lugar, vamos a establecer una diferencia entre dos términos que se suelen utilizar con bastante frecuencia en entornos de Bases de datos y Sistemas de información. Se trata de la diferencia entre datos e información.

Al hablar de *datos* nos referimos a hechos aislados, como por ejemplo, Juan tiene una cuenta corriente en Cajamar. En cambio, al hablar de *información* nos estamos refiriendo a datos procesados, organizados o resumidos. Por ejemplo, la respuesta a la pregunta *¿Cuál es el saldo de las cuentas corrientes de Juan?* sería información.

Por tanto, en el mundo empresarial, la información es un recurso vital, por lo que las empresas siempre han hecho uso de todos los medios que estaban a su alcance para el procesamiento de datos y la gestión de información. Entre estos medios se encuentran los *Sistemas de información*, que organizan los datos para producir información.

El desarrollo de los *Sistemas de bases de datos* se ha convertido en algo crucial para los Sistemas de Información, ya que son su núcleo: son los que le proporcionan los datos que necesitan para la elaboración de información.

1.2 Bases de datos, Sistemas de gestión de bases de datos y Sistemas de bases de datos

Una *base de datos* no es más que un conjunto de datos relacionados entre sí, creados para un propósito específico. Por tanto, vuelve a aparecer el concepto de datos, para hacer referencia a hechos conocidos que pueden guardarse y que tienen significado.

Nombre	Teléfono	Dirección
Juan	223344	Paseo de la Esperanza nº7
Luisa	224455	Calle de la Pereza nº5

Por tanto, este ejemplo constituiría una base de datos de un listín telefónico, pero los datos han de estar organizados de una forma lógica, no de una forma aleatoria. Es decir, consideramos que no todas las colecciones de datos son bases de datos.

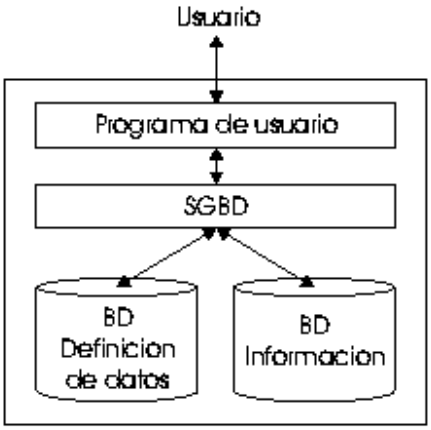
El tamaño y la complejidad de la bases de datos depende del problema que se esté resolviendo, por lo que podemos tener una base de datos personal para guardar las direcciones y teléfonos de nuestros mejores amigos, o bien tener una base de datos que almacene todos los datos clínicos de los usuarios de la Seguridad Social. Por tanto, tal cantidad de información, tiene que organizarse y controlarse para que la información pueda ser accedida y manipulada cuando sea necesario. Además, su almacenamiento ha de ser eficiente, así como las operaciones de manipulación que se llevan a cabo sobre los datos.

Un *SGBD (Sistema de Gestión de Bases de Datos)* es un conjunto de programas (en realidad, un sistema software) de propósito general que facilita el proceso de definición, construcción y manipulación de bases de datos para usos diversos.

- **Definición.** Para especificar tipos de datos, estructuras de datos y restricciones de los datos.
- **Construcción.** Para guardar los datos en un dispositivo de almacenamiento controlado por el SGBD.
- **Manipulación.** Para poder consultar y actualizar la información.

Un *Sistema de bases de datos* es el conjunto formado por una serie de programas que interactúan con el SGBD, por el propio SGBD y la base de datos.

Además, los Sistemas de bases de datos deben mantener la seguridad de la información ante caídas del sistema, y ante la posibilidad de accesos concurrentes.



Sistema de Bases de Datos

1.3 Aplicaciones de los Sistemas de bases de datos

Podemos distinguir dos grupos de aplicaciones de los Sistemas de bases de datos:

- Aplicaciones tradicionales: En ellas, la mayoría de la información que se maneja es alfanumérica, y las tareas básicas son la introducción y actualización de información, y la realización de consultas. Como ejemplos, podemos citar sistemas bancarios, sistemas de reserva de vuelos y habitaciones de hoteles, catálogos de bibliotecas o sistemas de gestión de supermercados.
- Nuevas aplicaciones: Los avances tecnológicos han favorecido la aparición de aplicaciones que manejan sonido y gráficos, como bases de datos multimedia o sistemas de información geográfica (GIS), y aplicaciones que permiten explotar los datos de un modo no tradicional, como los almacenes de datos y sistemas de procesamiento analítico en línea (OLAP). También se aplican bases de datos activas y en tiempo real en procesos de control y fabricación, y se aplican técnicas de búsqueda en bases de datos en la Web para mejorar la calidad de los resultados de los buscadores.

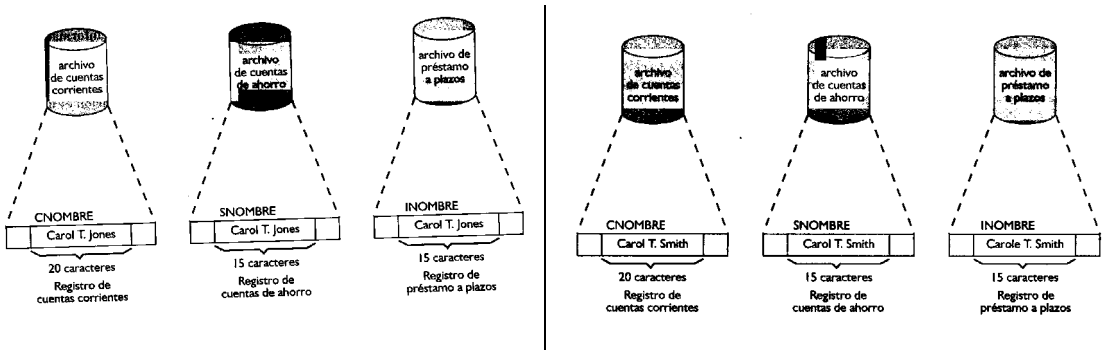
1.4 El enfoque de Bases de datos frente al de los Sistemas de gestión de archivos

Hay varios aspectos que distinguen al enfoque de Bases de datos del enfoque de los Sistemas de gestión de archivos tradicionales.

En la Gestión de archivos, los programadores definen e implementan los archivos requeridos para una aplicación específica. Si hay varias aplicaciones desarrolladas que utilizan datos comunes, cada programa utiliza archivos separados. Esta redundancia, además de suponer un aumento del espacio de almacenamiento, supone un aumento del esfuerzo necesario para el mantenimiento de dichas aplicaciones. En el enfoque de Bases de datos se define un único almacén para todos los datos, al cual pueden tener acceso muchos usuarios.

El enfoque de bases de datos viene a eliminar los problemas originados por los Sistemas de gestión de archivos, que son los siguientes:

- **Redundancia e inconsistencia de los datos.** Como los archivos de datos son creados por los programas de explotación durante mucho tiempo, es posible que los archivos tengan formatos diferentes y que haya información duplicada. Esta redundancia de los datos, además de suponer un coste de almacenamiento, puede llevar a la inconsistencia de los datos, debido a que no se hayan realizado las actualizaciones en todas las copias de los datos.



- **Dificultad para tener acceso a los datos.** Se debe a la llegada de peticiones nuevas que no estaban contempladas o no se han desarrollado a la hora de crear la aplicación, debido a que no se han creado rutinas para que proporcionen dicha información. Seguro que si se desarrollara dicha rutina, habría que modificarla en un futuro debido a nuevos requerimientos, lo que nos conduce a que tenemos que desarrollar sistemas de recuperación de información para su uso general.
- **Aislamiento de datos.** Como la información está repartida en varios archivos con varios formatos, es difícil crear nuevos programas para obtener la información deseada.
- **Anomalías en el acceso concurrente.** Para mejorar el funcionamiento global del sistema y obtener un tiempo de respuesta más rápido, muchos sistemas permiten que varios usuarios puedan acceder simultáneamente a la información. En entornos de este tipo, las interacciones pueden dar lugar a la inconsistencia de la información.

Saldo inicial
100.000

Sacar 10.000

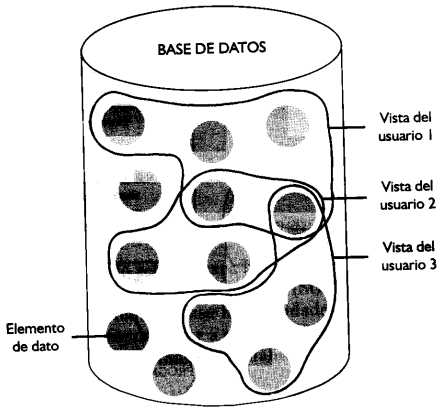
Sacar 20.000

Saldo final
90.000!!!

Saldo final
80.000!!!

Para prevenir esta situación, hay que disponer de mecanismos de supervisión que controlen el acceso concurrente a la información, y esto es difícil de hacer con aplicaciones que han sido desarrolladas de forma independiente y sin coordinación previa.

- **Problemas de seguridad.** Es conveniente que todos los usuarios no tengan acceso a toda la base de datos, sino sólo a la parte relativa a su trabajo. Para ello, se definen privilegios, de forma que todos los usuarios no puedan tener acceso a los mismos datos. De esta forma, se puede ver a la base de datos como una serie de parcelas lógicas divididas según los tipos de privilegios que se deseen definir, y creando tipos de usuarios para cada uno de estos privilegios. Por ejemplo, en la base de datos de una Universidad, los alumnos podrían tener acceso con privilegio de consulta a los datos relacionados con sus notas, pero no deberían tener acceso a modificar dichos datos, así como tampoco deberían tener acceso a información personal de los profesores.



- **Problemas de integridad.** Los valores almacenados en la base de datos deben satisfacer ciertos tipos de *restricciones de consistencia*, como por ejemplo, en una cuenta corriente el saldo tiene que ser mayor o igual que cero. Estas restricciones han de ser fáciles de definir y hacer que se cumplan, lo que resulta difícil de llevar a programas específicos.

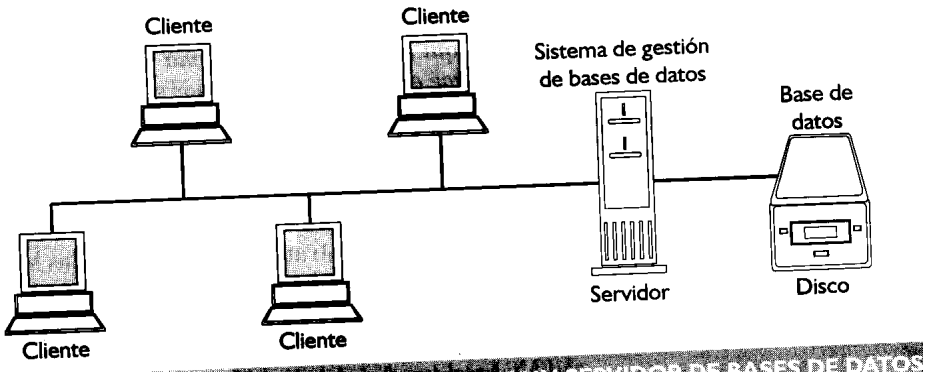
- **Almacenamiento persistente de objetos complejos.** En los lenguajes de programación actuales, es corriente disponer de estructuras de datos complejas, *clases en C++*. Si se desean guardar estos datos, hay que convertir estos formatos al formato de las estructuras de la base de datos, lo que se conoce como *falta de correspondencia* (*impedance mismatch*), y al volver a iniciar la aplicación, hay que recuperar dichas estructuras y convertirlas. Este problema lo resuelven los *SGBDOO* (*Sistemas de Gestión de Bases de Datos Orientados a Objetos*), en los que los objetos complejos se pueden guardar tal cual en la base de datos. Los SGBDOO eliminan la incompatibilidad entre las estructuras de datos ofrecidas por el SGBD y las estructuras de datos del lenguaje de programación.

- **Suministro de varias interfaces de usuario.** Debido a que los usuarios del sistema pueden tener gran variedad de conocimientos informáticos, sería interesante disponer de varias formas de acceder a la aplicación (por ejemplo, mediante una interfaz sencilla con acceso

limitados, o bien con una línea de órdenes para que el usuario pueda crear sus propias consultas).

1.5 Implicaciones del uso del enfoque de bases de datos

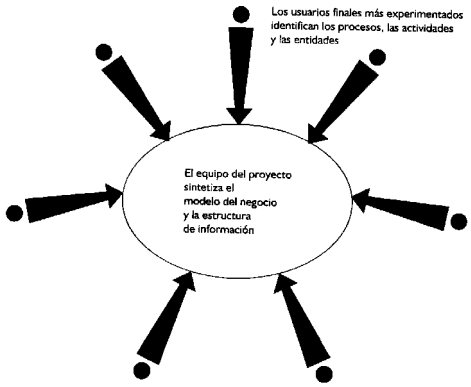
- **Potencial para la creación de normas y estilos.** En una organización grande, la comunicación es algo difícil, pero con un entorno centralizado de información, una vez que se haya definido la nomenclatura que se va a utilizar y los informes que se vayan a emplear, todas las personas que hagan uso del sistema obtendrán la información en el mismo formato y con los mismos términos.
- **Menor tiempo para la creación de nuevas aplicaciones.** Una vez que se ha creado la estructura de la base de datos (que podría necesitar más tiempo que la creación de una sola aplicación en el enfoque del Sistema de gestión de archivos), es fácil crear nuevas aplicaciones para la explotación de los datos de la base de datos.
- **Disponibilidad de información actualizada.** En entornos centralizados, los cambios que realice el usuario serán utilizados por los usuarios que vuelvan a acceder a dicha información.



1.6 Planificación estratégica de bases de datos

Debido a la función tan importante que tienen que realizar las Bases de datos (son el núcleo de los Sistemas de información), es necesario planificar su evolución, de forma que se determinen las necesidades del sistema para un periodo de tiempo largo.

La planificación de la base de datos ha de preceder al diseño e implementación de la base de datos, para que satisfaga las necesidades de información del sistema.



No se debe olvidar que la información sólo se puede obtener a partir de los datos, luego es necesario que estén disponibles todos los datos necesarios para producir la información. Esto sugiere que hay que hacer un análisis exhaustivo de la situación y determinar cuáles son los datos necesarios que hay que incluir en la base de datos, así como identificar cuáles son los procesos y actividades que hay que realizar sobre ellos.

1.7 Abstracción de datos

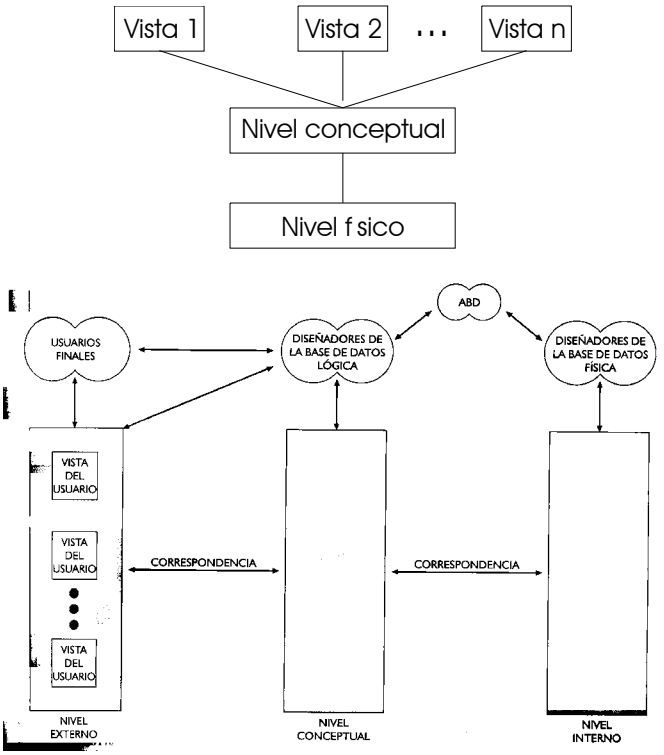
Un Sistema de bases de datos es un conjunto de archivos de datos interrelacionados junto con una serie de programas que permiten a los usuarios acceder y modificar los datos.

Un objetivo importante de los Sistemas de bases de datos es proporcionar a los usuarios una visión abstracta de los datos, es decir, el sistema debe ocultar los detalles sobre cómo se acceden y se manipulan los datos. Sin embargo, para que el sistema sea eficiente, tiene que disponer de estructuras de datos bien diseñadas que son las que se guardan en la base de datos.

Esto da origen a la creación de *niveles de abstracción*, que muestren distintas visiones de la complejidad de la representación de la información, es decir, oculten los detalles de almacenamiento. La *arquitectura ANSI/SPARC* permite ver una base de datos a tres niveles de abstracción denominados nivel físico, nivel lógico y nivel de visión.

- **Nivel físico.** Es el nivel de abstracción más bajo, y describe cómo se almacenan realmente los datos.
- **Nivel lógico.** Es el nivel que describe qué información se almacena en la base de datos, y cómo está relacionada dicha información. La definición de estructuras de datos a nivel lógico o conceptual puede suponer la creación de varias estructuras complejas a nivel físico (*creación de archivos indexados por varios campos*).
- **Nivel de visión.** Es el nivel de abstracción más alto, y en el que sólo se describen partes de la base de datos, ya que no todos los usuarios pueden acceder a la misma parte de la base de datos. Para facilitar la interacción del usuario con el sistema, se definen varios niveles de

visión, de forma que cada uno represente lo que cada usuario o grupo de usuarios necesita.



A partir de estos niveles de abstracción es posible establecer una analogía entre dichos niveles de abstracción y los tipos de datos de los lenguajes de programación, como se muestra a continuación.

La mayoría de los lenguajes de programación de alto nivel permiten la declaración de tipos:

```
typedef struct DatosAlumno{
   char nombre[50];
   char apellido1[50];
   char apellido2[50];
   char dni[8];
   char direccion[50];
   char localidad[50];
   char codigoPostal[50];
};
```

```
typedef struct NotasAlumno{
   char dni[8];
   char notas[5][10];
};
```

En el nivel físico, estos registros pueden describirse como un bloque de posiciones de memoria.

En el nivel lógico, que es el que suele utilizar el Administrador de la base de datos, estos registros quedan descritos mediante la declaración de tipos anterior, y además se definen las relaciones entre registros.

Finalmente, en el nivel de visión se definen varias *vistas* de la base de datos, ya que no todos los usuarios tienen acceso a la información sobre las notas de los alumnos. En general, al usuario no le interesa saber dónde está cada campo dentro del registro, ni en qué posición de memoria está guardada la información que está consultando, sino que lo que interesa, es que cuando quiera introducir las

calificaciones de los alumnos se realice en los campos y registros adecuados, y que cuando haga referencia a la nota de Bases de datos de 3º del alumno Xx Yy se devuelva la información correcta.

1.8 Esquemas e instancias

La información de la base de datos cambia continuamente a medida que se va modificando la información que contiene. Al conjunto de datos de la base de datos en un instante determinado se le denomina *instancia de la base de datos*, y cambia continuamente. Por otro lado, tenemos la propia estructura de la base de datos, denominada *esquema de base de datos*, la que rara vez cambia, o al menos no lo hace tan a menudo como el contenido de la base de datos.

Si volvemos a la declaración de tipos anterior, la declaración del tipo equivaldría al esquema de la base de datos, mientras que las variables de un cierto tipo de datos cambian continuamente, lo que equivaldría al concepto de instancia.

La división en tres niveles de abstracción que propone la arquitectura ANSI/SPARC también se puede expresar en términos de esquemas, ofreciendo tres tipos de esquemas, uno para cada nivel. Al nivel físico le corresponde el esquema interno, al nivel lógico le corresponde el esquema conceptual, y por último, al nivel de visión el corresponden los esquemas externos.

Nivel de abstracción	Tipo de esquema
Nivel físico	Esquema interno
Nivel lógico	Esquema conceptual
Nivel de visión	Esquemas externos

1.9 Independencia de datos

Es la capacidad de modificar un esquema de un nivel sin afectar a los esquemas de nivel superior. Hay dos niveles de independencia de datos:

- **Independencia física de datos.** Es la capacidad de modificar el esquema físico sin necesidad de modificar los programas de aplicación. *Modificar el tamaño de un campo o modificar índices*.
- **Independencia lógica de datos.** Es la capacidad de modificar el esquema conceptual sin necesidad de modificar los programas de aplicación.

La independencia lógica es más difícil de conseguir que la independencia física, ya que los programas de aplicación suelen ser muy dependientes de la estructura lógica de los datos a los que acceden (se han creado a partir de ella). No obstante, es posible realizar cambios mediante la definición de esquemas externos, de forma que en lugar de eliminar algunas partes del esquema conceptual, se podría definir un esquema externo a partir del esquema conceptual ofreciendo la visión deseada, sin afectar a los esquemas y aplicaciones existentes.

1.10 Modelos de datos

Una característica fundamental del enfoque de bases de datos es que proporciona cierto nivel de abstracción de los datos al ocultar detalles de almacenamiento que los usuarios no tienen por que conocer. Los modelos de datos son la herramienta principal para ofrecer esta abstracción.

Un modelo de datos es un conjunto de herramientas conceptuales para describir datos, las relaciones entre ellos, la semántica asociada a los datos y restricciones de consistencia, es decir un conjunto de conceptos útiles para describir la estructura de una base de datos.

Los modelos de datos que se han propuesto pueden clasificarse dentro de tres grupos: Modelos lógicos basados en objetos, Modelos lógicos basados en registros, y Modelos físicos de datos.

1.10.1 Modelos lógicos basados en objetos

Los modelos lógicos basados en objetos se usan para describir los datos en los niveles conceptual y de visión. Se caracterizan porque proporcionan una capacidad de estructuración bastante flexible y permiten especificar restricciones de datos explícitamente. A continuación describiremos brevemente el modelo entidad-relación y el modelo orientado a objetos.

El modelo Entidad-Relación (E-R)

El modelo de datos E-R (entidad-relación) se basa en una percepción del mundo real que consiste en una colección de objetos básicos denominado *entidades* y *relaciones* entre estos objetos.

Una entidad es un objeto que se diferencia de otros objetos mediante una serie de *atributos*. Por ejemplo, los atributos *numeroDeCuenta* y *saldo* describen una cuenta concreta de una banco.

Una *relación* es una asociación entre varias entidades. Por ejemplo, una relación *ClienteCuenta* asocia a cada cliente todas las cuentas que tiene en el banco.

Al conjunto de todas las entidades del mismo tipo se le denomina *conjunto de entidades*, y al conjunto de relaciones del mismo tipo, se le denomina *conjunto de relaciones*.

Además de las entidades y de las relaciones, el modelo E-R representa ciertas restricciones sobre el contenido de la base de datos. Una restricción importante es la *cardinalidad de asignación*, que expresa el número de entidades a las que puede asociarse una entidad a través de un conjunto de relaciones.

La estructura lógica global de una base de datos puede representarse gráficamente mediante un diagrama E-R, que consta de los siguientes componentes:

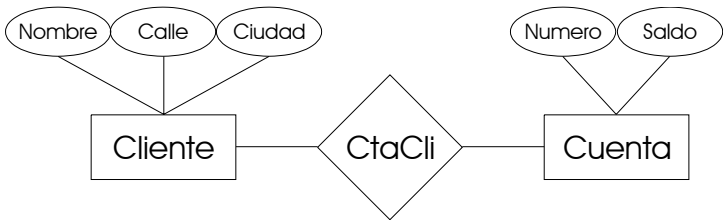
Rectángulos: Representan conjuntos de entidades

Elipses: Representan atributos

Rombos: Representan relaciones entre conjuntos de entidades

Líneas: Conectan atributos con conjuntos de entidades y conjuntos de entidades con relaciones.

Cada una de estas componentes se etiqueta con la entidad, relación o atributo que representa, de forma que los diagramas obtenidos tiene un aspecto como el de la figura siguiente.



El modelo orientado a objetos

Este modelo también se basa en la percepción de una colección de objetos.

Un *objeto* se caracteriza por tener un *estado* y un *comportamiento*. El estado corresponde a los valores que toman un conjunto de propiedades o *variables de instancia*, y el comportamiento es llevado a cabo mediante una serie de operaciones o funciones que operan sobre el objeto, y que se denominan *métodos*. Los objetos que tienen el mismo tipo de propiedades y el mismo comportamiento son agrupados en *clases*. Dichas clases se organizan en un *diagrama o jerarquía de clases*, en el que las clases pueden estar relacionadas mediante relaciones de asociación o mediante relaciones de herencia. La herencia permite la definición de clases a partir de clases existentes heredándose a las nuevas clases las propiedades y el comportamiento de las clases existentes, cumpliéndose también que todos los objetos de una subclases también es objeto de su superclase.

La única forma en la que un objeto puede acceder a los datos de otro objeto es a través de los métodos de este objeto. Esto se denomina *envío de mensajes* al objeto. De esta forma, la interfaz de llamada mediante los métodos de un objeto define la parte visible, mientras que la parte interna del objeto (variables y código de los métodos) no es visible externamente. De esta forma se tienen dos niveles de abstracción.

Por ejemplo, sea un objeto que representa a una cuenta corriente, y que dicho objeto contiene las variables de instancia *numeroDeCuenta* y *saldo*. Este objeto puede tener un método denominado Ingresar que añade una cantidad al saldo.

A diferencia del modelo E-R, en el modelo OO, cada objeto tiene su propia entidad que viene dado por un **OID (identificador del objeto)** asignado por el sistema.

1.10.2 Modelos lógicos basados en registros

Los modelos lógicos basados en registros se utilizan para describir datos en los modelos conceptual y de visión.

Los modelos basados en registros se llaman así porque la base de datos está estructurada en registros de formato fijo de varios tipos. Cada tipo de registro define un número fijo de campos o atributos, y cada campo normalmente es de longitud fija.

Los tres modelos de datos más ampliamente extendidos son el modelo relacional, el modelo en red y el modelo jerárquico

Modelo relacional

El modelo relacional representa los datos y las relaciones entre los datos mediante un conjunto de tablas, donde cada una de las tablas tiene una serie de columnas con nombres únicos.

Nombre	Calle	Ciudad
Angel	Caridad nº3	Almería
Luisa	Torreón nº7	Granada
Alberto	Cohete espacial nº1	Almería

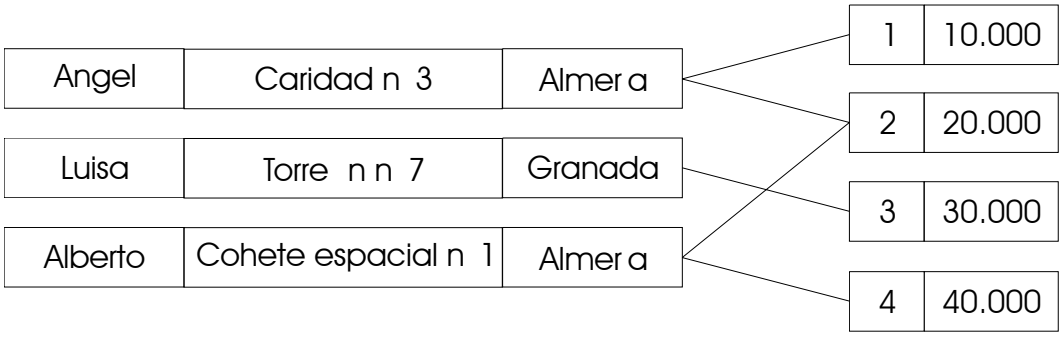
Numero	Saldo
1	10.000
2	20.000
3	30.000
4	40.000

Nombre	Numero
Angel	1
Angel	2
Luisa	3
Alberto	2
Alberto	4

Modelo en red

Se trata de un modelo en el que los datos se representan mediante conjuntos de registros y las relaciones mediante enlaces, los cuales pueden ser vistos como punteros.

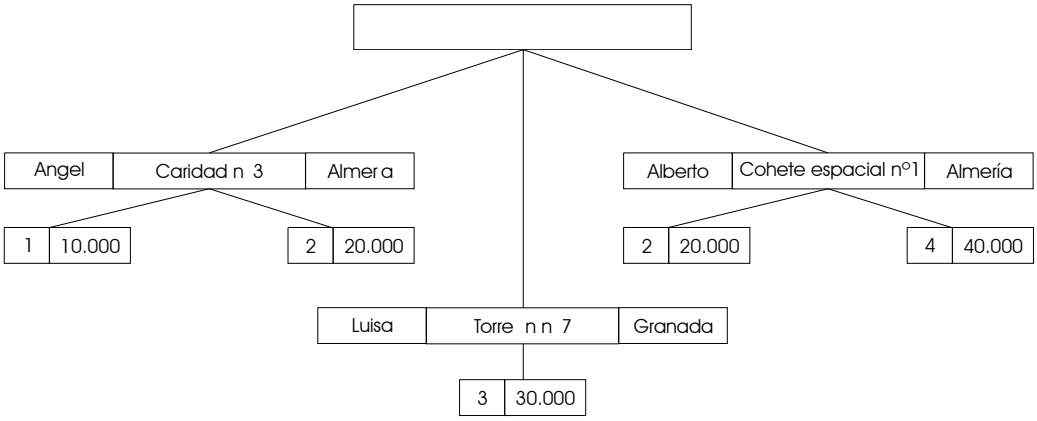
La organización de los registros se realiza mediante una serie de grafos, que interconecta los registros relacionados.



Modelo jerárquico

El modelo jerárquico es similar al modelo en red, en el sentido en que los datos y las relaciones entre los datos se representan mediante registros y enlaces.

La diferencia con el modelo en red es que la organización de los registros se representa mediante estructuras jerárquicas (árboles), por lo que los nodos hijo, sólo pueden tener un padre, tal y como se muestra a continuación.



1.10.3 Modelos físicos de datos

Los modelos físicos de datos se usan para la descripción de datos al más bajo nivel, pero en este curso no veremos nada acerca de ellos.

1.11 Lenguajes e interfaces de bases de datos

Como los usuarios de un SGBD pueden tener distintos privilegios y distintos conocimientos informáticos, es necesario disponer de diferentes lenguajes e interfaces para cada tipo de usuarios.

1.11.1 Lenguaje de definición de datos

Una vez que se ha finalizado la tarea de diseño de la base de datos, y que se ha seleccionado un SGBD para su implementación, el primer paso consiste en la especificación del esquema conceptual de la base de datos.

El esquema conceptual de la base de datos se especifica mediante una serie de definiciones expresadas en un Lenguaje de definición de datos (DDL, *Data Definition Language*). El SGBD contará con un compilador de DDL cuya función será procesar las sentencias en DDL para identificar las descripciones de los

elementos de los esquemas y guardar la descripción del esquema en un *diccionario de datos*.

El diccionario de datos es un archivo que contiene *metadatos*, es decir, datos acerca de los datos. Este archivo se consulta cada vez que se leen o modifican los datos del Sistema de base de datos.

1.11.2 Lenguaje de manipulación de datos

Una vez que se han compilado los esquemas de la base de datos, y que ya se han introducido datos en la base de datos, los usuarios necesitarán algún mecanismo para obtener información de la base de datos. Las operaciones más comunes de manipulación son la consulta, inserción, eliminación y modificación de datos. Para ello, el SGBD ofrece un Lenguaje de manipulación de datos (DML, *Data Manipulation Language*).

En general existen dos tipos de DML:

- **Procedimentales.** Requieren que el usuario especifique qué datos desea y cómo hay que obtenerlos.
- **No procedimentales.** Requieren que el usuario especifique qué datos desea sin tener que especificar cómo obtenerlos.

Los DML no procedimentales suelen ser más fáciles de utilizar para los usuarios, ya que no hay que especificar la forma en que se tienen que obtener los datos, pero esta ventaja se convierte en un inconveniente, puesto que el código que se genere puede que no sea tan eficiente como el producido por los lenguajes de consulta procedimentales.

Una *consulta* es una sentencia que solicita información de la base de datos, y un *lenguaje de consulta* es el subconjunto de un DML que se utiliza para la recuperación y actualización de información de la base de datos, pero nosotros obviaremos esta diferencia.

Por último, siempre que las sentencias del lenguaje de consulta se incluyan en un lenguaje de programación de propósito general, a este lenguaje se le denomina *lenguaje anfitrión*.

1.11.3 Interfaces para Sistemas de gestión de bases de datos

Normalmente, los usuarios de un Sistema de base de datos, utilizan un lenguaje de consulta de alto nivel, mientras que los programadores utilizan el DML para la creación de consultas. Para la mayoría de los usuarios se suelen definir interfaces de usuario amigables para la interacción con la base de datos. A continuación vamos a ver los tipos de interfaces que hay.

- **Interfaces basadas en menús.** Presentan al usuario una lista de opciones en forma de menús que guían al usuario en la petición de consultas. De esta forma no es necesario conocer la sintaxis de un lenguaje de consulta, pues permiten la creación de la consulta eligiendo las opciones que presenta la interfaz.

- **Interfaces gráficas.** Suelen presentar al usuario los esquemas en forma de diagrama, y las consultas se especifican manipulando el diagrama con el ratón.
- **Interfaces basadas en formularios.** Estas interfaces presentan un formulario al usuario en el que se rellenan los huecos del formulario para la modificación de los datos, o bien para especificar los parámetros de la consulta.
- **Interfaces de lenguaje natural.** Estas interfaces aceptan la especificación de una consulta descrita en términos de un idioma concreto y construyen expresiones DML a partir de dicha especificación.
- **Interfaces parametrizadas.** Se trata de interfaces para usuarios que siempre suelen realizar el mismo conjunto reducido de operaciones, reduciendo el número de pulsaciones para la creación de la consulta.

1.12 Usuarios y administradores de la base de datos

Uno de los objetivos primordiales de un Sistema de bases de datos es el proporcionar un entorno de recuperación de información y de almacenamiento de datos en la base de datos.

Podemos hacer una clasificación de los tipos de usuarios de una base de datos en función de la forma en que interaccionan con el sistema.

- **Administradores de la base de datos.** Persona que tiene centralizado el control del sistema.
- **Programadores de aplicaciones.** Se trata de los profesionales que interactúan con el sistema a través de llamadas en DML, las cuales están incorporadas en un lenguaje anfitrión. A estos programas se les denominan *programas de aplicación*, como por ejemplo, los programas para la generación de cargos, abonos, transferencias de un sistema bancario. Como la sintaxis DML suele ser diferente de la sintaxis del lenguaje anfitrión, las llamadas en DML suelen ir precedidas de un carácter especial, de forma que se genere el código apropiado en el lenguaje anfitrión, lo cual se hace mediante un *precompilador de DML*, que convierte las sentencias DML en sentencias del lenguaje anfitrión. Una vez precompilado el programa, se compilaría mediante el compilador del lenguaje anfitrión, que generaría el código objeto apropiado.
- **Usuarios sofisticados.** Son los que interactúan con el sistema sin escribir programas, escribiendo las consultas en el lenguaje de consulta de la base de datos.
- **Usuarios especializados.** Se trata de usuarios sofisticados que crean aplicaciones de bases de datos especializadas para el procesamiento de la información.

- **Usuarios ingenuos.** Son los usuarios que interactúan con el sistema llamando a uno de los programas desarrollados por los programadores de aplicaciones.

Como primer tipo de usuario hemos descrito la figura del administrador, un usuario vital en el enfoque de bases de datos, y que tiene unas funciones que merecen ser estudiadas más detalladamente. Estas son:

- **Definición del esquema conceptual.** El esquema original de la base de datos se crea escribiendo un conjunto de definiciones que son traducidas por el compilador de DDL a un conjunto de metadatos que se guardan en el diccionario de datos.
- **Definición del esquema físico.** Se trata de definir las estructuras de almacenamiento y los métodos de acceso adecuados (especificación de los tipos de índices)
- **Modificación del esquema y de la organización física.** Si bien las modificaciones tanto del esquema de la base de datos como de la organización física no son demasiado habituales, éstas se realizan modificando el esquema conceptual y físico.
- **Creación de permisos para el acceso a los datos.** El administrador de la base de datos es el encargado de definir los permisos que autorizan a los usuarios a acceder a ciertas partes de la base de datos.
- **Especificación de las restricciones de integridad.** Estas restricciones se guardan en el diccionario de datos para ser consultado cada vez que se realice una actualización.

1.13 Gestión de transacciones

Una transacción es un conjunto de operaciones sobre una base de datos que forman una unidad lógica de trabajo, como por ejemplo una transferencia de fondos de una cuenta a otra. Es indispensable que se produzca el cargo en la cuenta origen y el abono en la cuenta destino, o que no se produzca ninguna operación. A esto se le llama *atomicidad*. Otro de los requisitos es que se mantenga la *consistencia* de la base de datos (que la suma de las dos cuentas sea constante) , y un tercer requisito es que se garantize la *durabilidad* (que persistan los nuevos valores a pesar de la posibilidad de fallo del sistema).

La definición de las transacciones para que mantengan la consistencia es responsabilidad del programador de aplicaciones. El *módulo de gestión de transacciones* del SGBD es responsable de garantizar la atomicidad y la durabilidad. Necesita también un mecanismo de *recuperación de fallos*, para restaurar la base de datos al estado anterior al inicio de una transacción interrumpida por un fallo del sistema.

Cuando hay varias transacciones que actualizan la base de datos de forma concurrente, el *módulo de control de concurrencia* controla la interacción entre ellas para garantizar la consistencia.

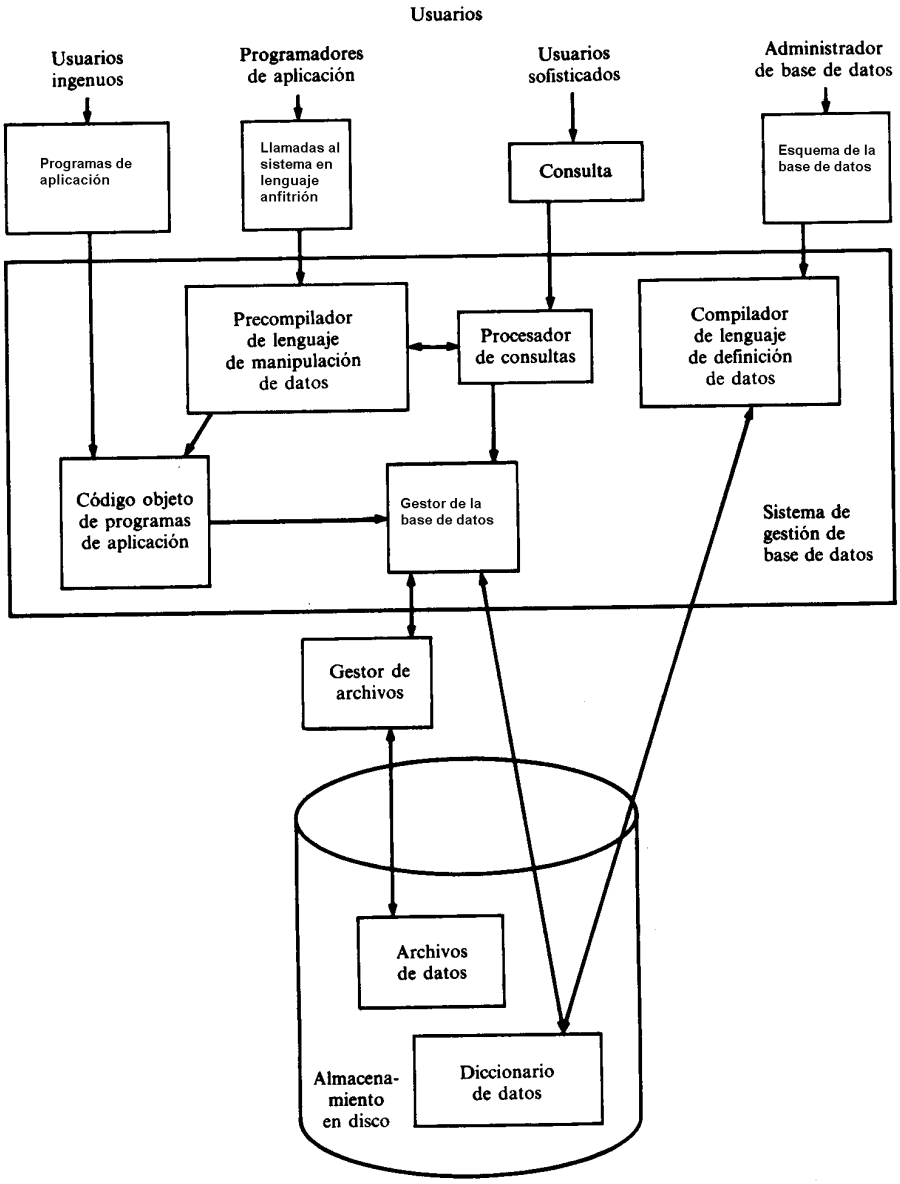
1.14 Estructura de un Sistema de bases de datos

El objetivo de los Sistemas de bases de datos es simplificar y facilitar el acceso a los datos. No obstante, hay que tener en cuenta que un factor importante en la satisfacción de los usuarios es el funcionamiento del Sistema de base de datos. Si el tiempo de respuesta es demasiado grande, el valor del sistema se reducirá, por lo que el funcionamiento del sistema depende de la eficiencia de las estructuras de datos que se utilicen para la representación de datos en la base de datos, y de la eficiencia del sistema para manipular dichas estructuras.

Por tanto es necesario que el gestor de la base de datos (una parte del Sistema de gestión de bases de datos) se encargue de las siguientes tareas:

- **Interacción con el gestor de archivos.** El gestor de la base de datos traduce las sentencias DML a órdenes del sistema de archivos proporcionado por el sistema operativo. Por tanto, el gestor de la base de datos es la parte del SGBD que se encarga del almacenamiento de los datos.
- **Implantación de la integridad.** Los valores de los datos que se almacenan en la base de datos deben satisfacer ciertas restricciones de consistencia (por ejemplo, el saldo de una cuenta corriente tiene que ser mayor o igual que cero). El Administrador de la base de datos es el que se encarga de la definición de estas restricciones, mientras que es el gestor de la base de datos el que se encarga de determinar si las actualizaciones de los datos infringen las restricciones de consistencia.
- **Implantación de la seguridad.** El gestor de la base de datos tiene que controlar el acceso al contenido de la base de datos por parte de usuarios no autorizados.
- **Copia de seguridad y recuperación.** Debido a que los sistemas informáticos están expuestos a fallos (fallos de disco, cortes de energía o de red y errores software), el gestor de la base de datos tiene que restaurar la base de datos al estado en que se encontraba antes de que se produjese el problema.
- **Control de concurrencia.** Si varios usuarios actualizan el contenido de la base de datos de forma concurrente, es posible que se pierda la consistencia de los datos. El control de la interacción de usuarios concurrentes también es tarea del gestor de la base de datos.

Para llevar a cabo todas estas funciones es necesario contar con una arquitectura específica. A continuación veremos como en un Sistema de bases de datos se divide en una serie de módulos donde cada uno de ellos tiene una serie de responsabilidades. En la mayoría de los casos, el sistema operativo sólo proporciona las operaciones más básicas, y el Sistema de bases de datos debe partir de este hecho, por lo que el diseño del SBD debe de incorporar una interfaz entre el propio SBD y el sistema operativo. A continuación se muestran cada uno de los componentes de un SBD.



Estructura del sistema.

- **Gestor de archivos.** Gestiona la asignación de espacio en el disco y las estructuras de datos que se utilizan para la representación de la información.
- **Gestor de base de datos.** Proporciona la interfaz entre los datos almacenados a bajo nivel y los programas de aplicación y las consultas que se realicen al sistema.
- **Procesador de consultas.** Traduce las sentencias DML a las instrucciones de bajo nivel que utiliza el gestor de la base de datos.
- **Precompilador de DML.** Convierte las sentencias DML escritas en un lenguaje anfitrión a instrucciones propias del lenguaje anfitrión.

- **Compilador de DDL.** Convierte la sentencias escritas en DDL a un conjunto de estructuras de datos que contienen metadatos.

Además, el SBD debe proporcionar estructuras de datos como parte del nivel físico, como estas:

- **Archivos de datos.** Almacenan la base de datos.
- **Diccionario de datos.** Almacena metadatos sobre la estructura de la base de datos.
- **Indices.** Proporcionan un acceso más rápido a los datos.