

## - Tecnicatura Superior en Programación -

# DISEÑO Y ADMINISTRACIÓN DE BASES DE DATOS

**Docente:** Ing. Darío Haspert

**Curso:** 2º



# Unidad I: Introducción a las Bases de Datos

## UNIDAD 1. INTRODUCCIÓN A LAS BASES DE DATOS.

### Contenidos:

Introducción. Concepto de base de datos. Definición de Sistema de base de datos. Componentes de un Sistema de Base de Datos. El Administrador de Base de Datos. Objetivos de las Bases de Datos. Características. Ventajas de enfoque de Base de datos. Beneficios del enfoque de Base de datos. Independencia de datos. Desventajas de enfoque de Base de datos. Síntesis cronológica.

### Índice

1.	Introducción .....	3
	¿Cómo se trabajaba anteriormente? .....	3
2.	Concepto de base de datos .....	5
3.	Sistema de base de datos .....	6
4.	Componentes .....	8
a.	Hardware .....	8
b.	Software .....	8
c.	Datos.....	8
d.	Procedimientos .....	9
e.	Personas .....	9
4.1	El administrador de Bases de Datos .....	11
5.	Objetivos de las bases de datos .....	12
6.	Características de la metodología de Bases de Datos .....	13
7.	Ventajas del enfoque de Base de datos .....	14
8.	Beneficios del enfoque de Base de datos.....	17
8.1.	Independencia de datos .....	18
9.	Desventajas de enfoque de Base de datos.....	19
10.	Síntesis cronológica .....	19
	Referencias Bibliográficas .....	21

## 1. Introducción

Las bases de datos hoy en día forman una parte integrante de nuestra vida cotidiana, hasta tal punto que muchas veces no somos conscientes de estar usando una base de datos. Ejemplos de esto son las compras en el supermercado, las compras utilizando una tarjeta de crédito, la reserva de un programa de vacaciones en una agencia de viajes, la utilización de una biblioteca, la contratación de un seguro, la utilización de Internet, el estudio en una universidad; entre muchos otros.

Para comenzar, consideraremos que una base de datos es una colección de datos relacionados y que el Sistema de Gestión de Bases de Datos (SGBD) es el software que gestiona y controla el acceso a la base de datos. Una aplicación de bases de datos es simplemente un programa que interactúa con la base de datos en algún punto de su ejecución. También utilizaremos el término más inclusivo sistema de base de datos para referirnos a una colección de programas de aplicación que interactúan con la base de datos, junto con el SQL y la propia base de datos.

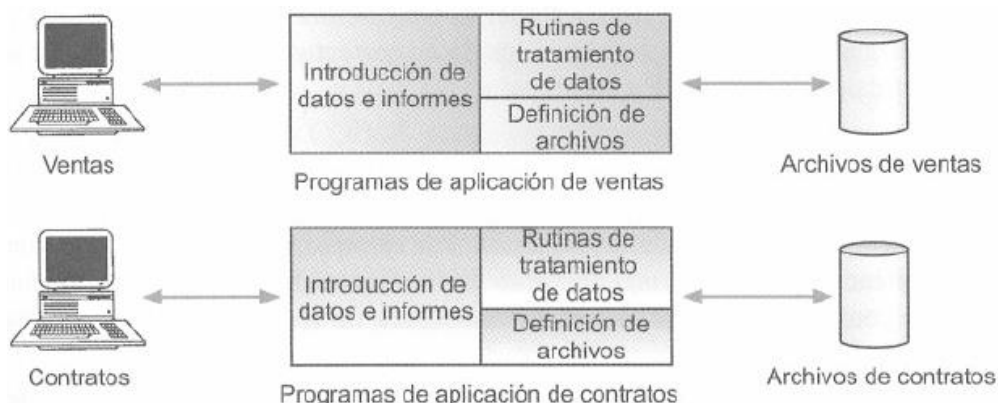
### ¿Cómo se trabajaba anteriormente?

Los **sistemas basados en archivos** fueron uno de los primeros intentos para informatizar los sistemas de archivo manual con los que todos nosotros estamos familiarizados. Por ejemplo, puede crearse un archivo manual en una organización para albergar toda la correspondencia externa e interna relativa a un proyecto, a un producto, a una tarea, a un cliente o a un empleado. Normalmente, existen muchos de dichos archivos, los cuales es preciso etiquetar y almacenar en una o más cajas o contenedores por cuestiones de seguridad. Los lugares donde se almacenen esos archivos pueden disponer de llave, también por cuestiones de seguridad, o pueden estar ubicados en áreas seguras del edificio. En nuestra propia casa, probablemente dispongamos de algún tipo de sistema de archivo en el que depositamos los recibos, las facturas, la información bancaria, los contratos de seguros, etc. Si necesitamos consultar algo, vamos a nuestro archivo personal y buscamos en él, de principio a fin, hasta encontrar la información deseada. Alternativamente, puede que dispongamos de un sistema de indexación que nos ayude a localizar más rápidamente lo que queremos. Por ejemplo, puede que tengamos subdivisiones en el sistema de archivo o carpetas separadas para los diferentes elementos que están *relacionadas de alguna manera lógica* entre sí.

Los sistemas de archivo manual funcionan bien cuando el número de elementos almacenados es pequeño. También pueden funcionar de forma adecuada cuando hay un gran número de elementos y lo único que necesitamos es almacenarlos o extraerlos. Sin embargo, los sistemas manuales de archivo dejan de ser útiles cuando tenemos que establecer referencias cruzadas o procesar la información contenida en los documentos.

Claramente, los sistemas manuales no resultan adecuados para acceder rápidamente a la información y generar informes. Los sistemas basados en archivos fueron desarrollados para dar respuesta a la necesidad que las empresas tenían de acceder de forma más eficiente a los datos. Sin embargo, en lugar de establecer un sistema centralizado de gestión de los datos operacionales de las organizaciones, lo que se hizo fue adoptar un enfoque descentralizado, en el que cada departamento, con la ayuda de personal especializado en **procesamiento de datos**, almacenaba y controlaba sus propios datos.

Así, los sistemas basados en archivo se veían como una colección de programas de aplicación que realiza diversos servicios para los usuarios finales, como por ejemplo la producción de informes. Cada programa define y gestiona sus propios datos.



En la terminología utilizada en los sistemas basados en archivos, un **archivo** es simplemente una colección de registros, que contienen datos lógicamente relacionados. Cada registro contiene un conjunto lógicamente relacionado de uno o más **campos**, donde cada campo representa alguna característica del objeto del mundo real que se quiere modelar.

Claramente, existía gran cantidad de datos duplicados, en los diferentes departamentos. Esta era una entre varias limitaciones que presentaban los sistemas basados en archivos:

- Separación y aislamiento de los datos: cada programa mantiene su propio conjunto de datos. Los usuarios de un programa pueden no ser conscientes de los datos potencialmente útiles en poder de otros programas.
- Duplicación de los datos: debido al enfoque descentralizado se promueve una duplicación incontrolada de los datos. La misma información es mantenida por diferentes programas. Desperdicio de espacio y valores potencialmente diferentes y/o diferentes formatos para el mismo ítem.
- Dependencias entre los datos: resulta difícil realizar cambios a una estructura física y el almacenamiento de los archivos y registros de datos.
- Formatos de archivos incompatibles: la estructura de los archivos dependen del lenguaje de programación de aplicaciones que se utilice.
- Consultas fijas/proliferación de programas de aplicación: los sistemas basados en archivos son muy dependientes del desarrollador de aplicaciones, que es quien tiene que escribir todas las consultas e informes requeridos. Entonces, no existía ninguna posibilidad de solicitar consultas no planificadas (*ad hoc*) y se contribuía a la proliferación de archivos y de programas de aplicación, que terminaban siendo inadecuados o ineficientes a la hora de satisfacer las demandas de los usuarios.
- Otras limitaciones eran: no se incluían mecanismos de seguridad o integridad; la recuperación ante fallos de hardware o software, era limitada o inexistente; el acceso a los archivos estaba restringido, no era compartido.

Todas estas limitaciones de los sistemas basados en archivos pueden atribuirse a dos factores distintos:

- I. la definición de los datos está incluida en los programas de aplicación, en lugar de almacenarse de forma separada e independiente;
- II. no existe ningún control sobre el acceso y manipulación de los datos, más allá del que imponen los propios programas de aplicación.

## 2. Concepto de base de datos



Una primera definición muy general de Base de Datos (BD) es la que se presentó anteriormente:

“Una base de datos es una colección de datos relacionados”. [1]

Con la palabra datos nos referimos a los hechos (datos) conocidos que se pueden grabar y que tienen un significado implícito. Por ejemplo, los nombres, números de teléfono y direcciones de las personas que conoce. Puede tener todos estos datos grabados en un libro de direcciones indexado o los puede tener almacenados en el disco rígido de una computadora mediante una aplicación como *Microsoft Access* o *Excel*. Esta colección de datos relacionados con un significado implícito es una base de datos.


La definición anterior de base de datos es muy genérica; por ejemplo, podemos pensar que la colección de palabras que compone esta página de texto es una colección de datos relacionados y que, por tanto, constituye una base de datos. No obstante, el uso común del término base de datos es normalmente más restringido.

Una base de datos tiene las siguientes propiedades implícitas:

- Una base de datos representa algún aspecto del mundo real, lo que en ocasiones se denomina universo de discurso (UoD, *Universe of discourse*). Los cambios introducidos en el UoD se reflejan en la base de datos.
- Una base de datos es una colección de datos lógicamente coherente con algún tipo de significado inherente. No es correcto denominar base de datos a un surtido aleatorio de datos.
- Una base de datos se diseña, construye y rellena con datos para un propósito específico. Dispone de un grupo pretendido de usuarios y algunas aplicaciones preconcebidas en las que esos usuarios están interesados.

En otras palabras, una base de datos tiene algún origen del que se derivan los datos, algún grado de interacción con eventos del mundo real y un público que está activamente interesado en su contenido.

Cabe destacar que una base de datos puede ser de cualquier tamaño y complejidad, y que la misma se puede generar y mantener manualmente o estar computarizada. En esta cátedra sólo nos ocuparemos de las bases de datos computarizadas.

 Entonces, una base de datos se define como una colección de datos que representan algún aspecto del mundo real, relacionados entre sí, y que se encuentran organizados y estructurados de manera tal que faciliten las tareas de consulta, manipulación y administración de los mismos y que están destinados a ser compartidos entre muchos usuarios para una diversidad de aplicaciones.

Otra definición de Bases de Datos es la que proponen *Thomas Connolly* y *Carolyn Begg*: [5]

*Una colección compartida de datos lógicamente relacionados, junto con una descripción de estos datos, que están diseñados para satisfacer las necesidades de información de una organización.*

La descripción de los datos se conoce con el nombre de catálogo del sistema (o diccionario de datos o metadatos, es decir, 'datos acerca de los datos'). Es esta naturaleza auto-descriptiva de la Base de Datos la que proporciona la independencia entre programas y datos.

Datos lógicamente relacionados comprende entidades, atributos, y relaciones de la información de una organización.

### 3. Sistema de base de datos

Las bases de datos no son tan sólo una colección de archivos. Más bien, una base de datos es una fuente central de datos destinados a compartirse entre muchos usuarios para una diversidad de aplicaciones. El corazón de una base de datos lo constituye el Sistema de Administración de Base de Datos (**DBMS**, DataBase Management System), también conocido como Sistema de Gestión de Base de Datos o **SGBD**.

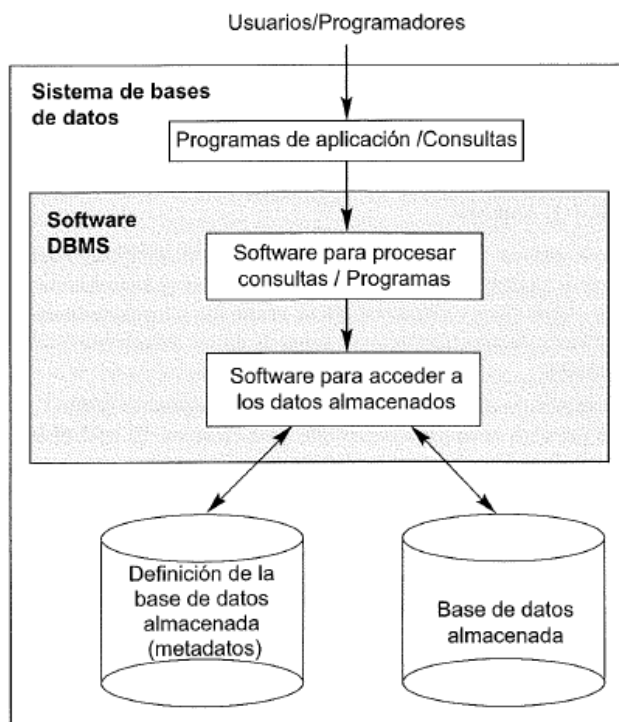


Un sistema de administración de base de datos es una colección de programas que permite a los usuarios definir, crear, mantener y controlar el acceso a una base de datos. El DBMS es un sistema de software de propósito general que facilita los procesos de definición, construcción, manipulación y compartición de bases de datos entre varios usuarios y aplicaciones. Definir una base de datos implica especificar los tipos de datos, estructuras y restricciones de los datos que se almacenarán en la base de datos. La definición o información descriptiva de una base de datos también se almacena en esta última en forma de catálogo o diccionario de la base de datos; es lo que se conoce como metadatos. La construcción de la base de datos es el proceso consistente en almacenar los datos en algún medio de almacenamiento controlado por el DBMS. La manipulación de una base de datos incluye funciones como la consulta de la base de datos para recuperar datos específicos, actualizar la base de datos para reflejar los cambios introducidos en el mini-mundo y generar informes a partir de los datos. Compartir una base de datos permite que varios usuarios y programas accedan a la base de datos de forma simultánea.

Una aplicación accede a la base de datos enviando consultas o solicitudes de datos al DBMS. Una consulta normalmente provoca la recuperación de algunos datos; una transacción puede provocar la lectura o la escritura de algunos datos en la base de datos.

Otras funciones importantes ofrecidas por el DBMS son la protección de la base de datos y su mantenimiento durante un largo período de tiempo. La protección incluye el resguardo del sistema contra el funcionamiento defectuoso del hardware o el software (caídas) y la protección de la seguridad contra el acceso no autorizado o malintencionado. Una gran base de datos típica puede tener un ciclo de vida de muchos años, por lo que el DBMS debe ser capaz de mantener el sistema de bases de datos permitiendo que el sistema evolucione según cambian los requisitos con el tiempo.

Así, *Elmasri & Navathe* [1], denominan sistema de bases de datos a la combinación de base de datos y software DBMS. La Figura a la derecha muestra el entorno de un sistema de bases de datos simplificado.



Por su parte, *Silberschatz & Korth* [2] expresan que:




Un sistema gestor de bases de datos (SGBD) consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a dichos datos. La colección de datos, normalmente

denominada base de datos, contiene información relevante para una empresa. El objetivo principal de un SGBD es proporcionar una forma de almacenar y recuperar la información de una base de datos de manera que sea tanto práctica como eficiente.

Los sistemas de bases de datos se diseñan para gestionar grandes cantidades de información.

La gestión de los datos implica tanto la definición de estructuras para almacenar la información como la provisión de mecanismos para la manipulación de la información. Además, los sistemas de bases de datos deben proporcionar la fiabilidad de la información almacenada, a pesar de las caídas del sistema o los intentos de acceso sin autorización. Si los datos van a ser compartidos entre diversos usuarios, el sistema debe evitar posibles resultados anómalos. [2]

A su vez, *C. J. Date*, dice que un Sistema de Base de Datos es un sistema computarizado cuya finalidad general es almacenar información y permitir a los usuarios recuperar y actualizar esa información con base a peticiones. La información en cuestión puede ser cualquier cosa que sea de importancia para el individuo u organización; en otras palabras, todo lo que sea necesario para auxiliarle en el proceso general de su administración. [3]

 Por su parte *Connolly y Begg* [5] manifiestan que un sistema de gestión de base de datos (SGBD) proporciona la siguiente funcionalidad:

- ✓ Permite a los usuarios definir la base de datos, usualmente mediante un lenguaje de definición de datos (DDL, *Data Definition Language*).
- ✓ Permite a los usuarios insertar, actualizar, borrar y extraer datos de la base de datos, usualmente mediante un lenguaje de manipulación de datos (DML, *Data Manipulation Language*).
- ✓ Proporciona un acceso controlado a la base de datos. Puede proporcionar:
  - un sistema de seguridad
  - un sistema de integridad
  - un sistema de control de concurrencia
  - un sistema de control de recuperación
  - un catálogo accesible por el usuario

El DBMS (sistema de administración de base de datos) es el software que maneja todo acceso a la base de datos. Conceptualmente, lo que sucede es lo siguiente: [*C. J. Date*][3]

- a. Un usuario emite una petición de acceso, utilizando algún sub-lenguaje de datos específico (por lo general SQL).
- b. El DBMS intercepta esa petición y la analiza.
- c. El DBMS inspecciona, en su momento, (las versiones objeto de) el esquema externo para ese usuario, la transformación externa/conceptual correspondiente, el esquema conceptual, la transformación conceptual/interna y la definición de la estructura de almacenamiento.
- d. El DBMS ejecuta las operaciones necesarias sobre la base de datos almacenada.

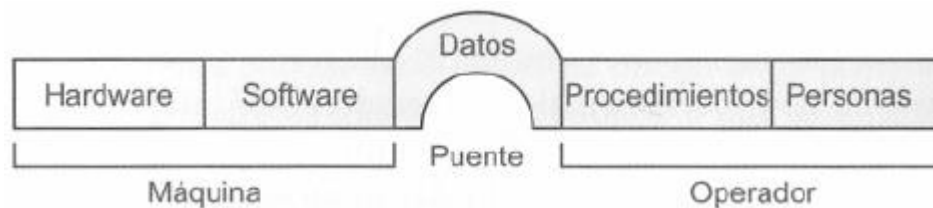




## 4. Componentes



Un sistema de base de datos comprende cinco componentes principales: datos, hardware, software, procedimientos y personas.



### a. Hardware

Un SDB está compuesto básicamente por:

- Los volúmenes de almacenamiento, dispositivos asociados de E/S, controladores de dispositivos, los canales de E/S, entre otros; y
- Los procesadores de hardware y la memoria principal asociada usados para apoyar la ejecución del software del SDB.

### b. Software

Comprende el propio software SGBD (Sistema de Gestión de Base de Datos) y los programas de aplicación, junto con el sistema operativo, que incluye el software de red si el SGBD se está utilizando en una red. El SGBD; o bien, DBMS, por sus siglas en inglés, funciona de intermediario entre el usuario y el hardware del SDB y es el encargado de manejar todas las solicitudes de acceso a la DB.

Una función general que ofrece el DBMS es ocultar a los usuarios del sistema los detalles de bajo nivel. En otras palabras, el DBMS ofrece a los usuarios una percepción de la base de datos que está, en cierto modo, por encima del nivel del hardware y que maneja las operaciones del usuario expresadas en términos de ese nivel más alto de percepción.

### c. Datos

Quizá el componente más importante de un entorno SGBD, al menos desde el punto de vista de los usuarios finales, sean los datos. En la Figura anterior podemos observar que los datos actúan como una especie de “puente” entre los componentes ligados a la máquina y los componentes ligados al operador humano. Como se mencionó anteriormente, los datos contenidos en una BD pueden hacer referencia a cualquier cosa importante para la persona u organización. Los datos constituyen uno de sus activos más valiosos con los que cuenta toda organización.

Dentro de una BD, los datos se agrupan en archivos. Estos, a su vez, están compuestos por registros. Finalmente, los registros están compuestos por campos. Visualmente, podemos relacionar los archivos con tablas, los registros con filas de esa tabla y los campos con las columnas de esa fila. Cabe aclarar que en este apunte usaremos esos términos como sinónimos. Se muestran luego ejemplos de tablas de una base de datos.

Los datos de una DB pueden ser:

- Integrados: se observa a la base de datos como una unificación de varios archivos que de otro modo serían distintos, con una redundancia entre ellos eliminada al menos parcialmente.



- **Compartidos:** los datos son compartidos de manera individual entre diferentes usuarios, y cada uno de ellos puede tener acceso al mismo conjunto de datos, probablemente con fines diferentes.

En general, en la actualidad los datos de las DB son tanto integrados como compartidos.

Cód. Cliente	Nombre y Apellido	Domicilio	Teléfono
356642	Fernández Alejandro	San Martín 123	577009
234569	Jiménez Luis Alberto	Av. Santa Fe 1789	503456
370184	García María Laura	Sarmiento 456	435662
299004	López Fernando Carlos	Las Heras 321	426694

Serie de TV	Temporadas	Género	Cadena Original	Año aparición
La Casa de Papel	3	Acción/Crimen/Misterio	Antena 3	2017
The Walking Dead	9	Suspense/Drama/Terror	AMC	2010
Breaking Bad	5	Drama/Thriller	AMC	2008
Lost	6	Acción/Aventura/Misterio/Fantasía	ABC	2004
Prison Break	4	Suspense/Drama/Acción	FOX	2005
Game of thrones	8	Fantasía/Drama/Aventura	HBO	2011

#### d. Procedimientos

Los procedimientos son las instrucciones y reglas que gobiernan el diseño y utilización de la base de datos. Los usuarios del sistema y el personal que gestiona la base de datos requieren una serie de procedimientos documentados que les permitan saber cómo utilizar o ejecutar el sistema. Estos procedimientos pueden estar compuestos de instrucciones que les digan cómo:

- iniciar una sesión en el SGBD;
- utilizar una funcionalidad concreta del SGBD o un programa de aplicación;
- iniciar y detener el SGBD;
- realizar copias de seguridad de la base de datos;
- gestionar los fallos de hardware o de software. Esto puede incluir procedimientos para identificar el componente fallido, para reparar el componente fallido y, después de la reparación del fallo, para recuperar la base de datos;
- cambiar la estructura de una tabla, reorganizar la base de datos entre múltiples discos, mejorar el rendimiento o archivar los datos en un almacenamiento secundario.

#### e. Personas

Son las personas que se relacionan con el sistema. Podemos identificar cinco tipos distintos de personas que pueden participar en un entorno SGBD:

- administradores de datos
- administradores de la base de datos
- diseñadores de bases de datos
- desarrolladores de aplicaciones
- usuarios finales.



Describimos cada uno de ellos:

- El **Administrador de Datos (DA)**: es un administrador, no técnico, que decide qué datos se deben almacenar en la BD, así como también de establecer políticas para mantener y manejar esos datos. El administrador de datos (DA, *Data Administrator*) es responsable de gestionar los recursos de datos, lo que incluye la planificación de la base de datos, el desarrollo y mantenimiento de estándares, políticas y procedimientos y el diseño procedimental/lógico de la base de datos. El DA consulta con los gerentes de mayor nivel y les aconseja, para garantizar que la dirección seguida por el desarrollo de la base de datos permita soportar los objetivos corporativos.
- El **Administrador de Base de Datos (DBA)**: es un técnico, con conocimientos de IT, que implementa las decisiones del DA. Es el encargado de crear la base de datos real e implementar los controles técnicos necesarios para hacer cumplir las diversas decisiones de las políticas hechas por el administrador de datos. Es responsable de la materialización física de la base de datos, incluyendo la implementación y diseño físico de la base de datos, el control de la seguridad y de la integridad, el mantenimiento de la fiabilidad del sistema y la garantía de que las aplicaciones exhiban un rendimiento satisfactorio para los usuarios. El papel de un DBA requiere un conocimiento detallado del SGBD de destino y del entorno de sistema en el que está implementado.
- **Diseñadores de bases de datos**: En los grandes proyectos de diseño de bases de datos, podemos distinguir entre dos tipos de diseñador: los diseñadores lógicos de la base de datos y los diseñadores físicos de la base de datos. Las responsabilidades del diseñador lógico de la base de datos son identificar los datos (es decir, las entidades y atributos), las relaciones entre los datos y las restricciones que hay que aplicar a los datos que se almacenen en la base de datos. El diseñador lógico de la base de datos debe tener una comprensión profunda y completa de los datos de la organización y de las restricciones aplicables (las restricciones se denominan en ocasiones reglas de negocio). Estas restricciones describen las principales características de los datos.

El diseñador físico de la base de datos decide cómo materializar físicamente el diseño lógico de la base de datos. Esto implica:

- establecer la correspondencia entre el diseño lógico de la base de datos y un conjunto de tablas y restricciones de integridad;
- seleccionar estructuras de almacenamiento y métodos de acceso específicos para los datos con el fin de conseguir unas buenas prestaciones;
- diseñar las medidas de seguridad que los datos requieran.

Muchas partes del diseño físico de una base de datos dependen en gran medida del SGBD de destino y puede haber más de una forma de implementar cada mecanismo concreto. Por tanto, el diseñador físico de la base de datos debe conocer a la perfección la funcionalidad del SGBD de destino y debe entender las ventajas y desventajas de cada alternativa para cada implementación concreta. El diseñador físico de la base de datos debe ser capaz de seleccionar una estrategia de almacenamiento adecuada que tenga en cuenta el uso de la base de datos. Mientras que el diseño conceptual y lógico de la base de datos estén relacionados con el *qué*, el diseño físico de la base de datos se preocupa de

cómo. Se requieren capacidades y conocimientos diferentes, lo que implica en muchas ocasiones utilizar personas distintas.

- **Desarrolladores de aplicaciones:** son los encargados de escribir programas de aplicación que acceden a la BD. Estos programas pueden estar escritos en cualquier lenguaje de programación y se comunican con la BD emitiendo la solicitud adecuada al DBMS, y su objetivo es poder permitir al usuario final acceder a la BD. Normalmente, los desarrolladores de aplicaciones trabajan a partir de una especificación producida por los analistas de sistemas. Cada programa contiene enunciados que exigen al SGBD realizar algún tipo de operación sobre la base de datos. Esto incluye extraer datos, insertarlos, actualizarlos o borrarlos.
- Los **usuarios finales:** son aquellos que acceden a la DB desde sus estaciones de trabajo utilizando los programas antes mencionados; o bien, usando una interfaz proporcionada como parte integral del software del sistema de base de datos. Estas interfaces controladas por menús o por formularios tienden a facilitar el uso a personas que no cuentan con una capacitación formal en IT. En contraste, las interfaces controladas por comandos (por ejemplo, los lenguajes de consulta) tienden a requerir cierta experiencia profesional en IT. Así, podríamos clasificar a los usuarios en:
  - Usuarios **inexpertos**, que normalmente no son conscientes de la existencia de un SGBD. Acceden a la base de datos mediante programas de aplicación que intentan que las operaciones sean lo más simples posibles. Estos usuarios invocan las operaciones sobre la base de datos introduciendo comandos simples o seleccionando una serie de opciones en un menú. Esto quiere decir que no necesitan conocer ningún detalle ni sobre la base de datos, ni sobre el SGBD.
  - Usuarios **avanzados**. están familiarizados con la estructura de la base de datos y con las funcionalidades ofrecidas por el SGBD. Los usuarios finales avanzados pueden utilizar un lenguaje de consulta de alto nivel, como SQL, para llevar a cabo las operaciones requeridas. Algunos usuarios finales avanzados pueden incluso escribir sus propios programas de aplicación para su uso personal.

#### 4.1 El administrador de Bases de Datos

El DBA es el responsable del control general del SBD a nivel técnico. Ahora podemos describir con un poco más de detalle algunas de las tareas del DBA. En general, estas tareas comprenden:

✓ *Definir el esquema conceptual*

Es trabajo del administrador de datos decidir exactamente qué información contendrá la base de datos; en otras palabras, identificar las entidades de interés para la empresa e identificar la información que hay que registrar acerca de dichas entidades. Por lo general a este proceso se le conoce como diseño lógico —en ocasiones conceptual— de la base de datos. Una vez que el administrador decidió el contenido de la base de datos a un nivel abstracto, entonces el DBA creará el esquema conceptual correspondiente, utilizando el DDL conceptual.

✓ *Definir el esquema interno*

El DBA también debe decidir la forma en que van a ser representados los datos en la base de datos almacenada. A este proceso se le conoce comúnmente como diseño físico de la base de datos. Una

vez realizado el diseño físico, el DBA deberá crear la definición de la estructura de almacenamiento correspondiente (es decir, el esquema interno), utilizando el DDL interno. Además, también deberá definir la transformación conceptual/interna asociada.

✓ *Establecer un enlace con los usuarios*

Es asunto del DBA enlazarse con los usuarios para asegurar que los datos necesarios estén disponibles y para escribir (o ayudar a escribir) los esquemas externos necesarios, utilizando el DDL externo aplicable. (Recordar que un sistema dado podría manejar varios DDLs externos distintos.) Además, también es necesario definir las transformaciones externas/conceptual correspondientes. Otros aspectos de la función de enlace con los usuarios incluyen la asesoría sobre el diseño de aplicaciones; una capacitación técnica; ayuda en la determinación y resolución de problemas; así como otros servicios profesionales similares.

✓ *Definir las restricciones de seguridad y de integridad*

El DDL conceptual debe incluir facilidades para especificar dichas restricciones.

✓ *Definir las políticas de vaciado y recarga*

Una vez que una empresa se compromete con un sistema de base de datos, se vuelve drásticamente dependiente del funcionamiento exitoso de dicho sistema. En el caso de que se produzca un daño en cualquier parte de la base de datos —ocasionado, por ejemplo, por un error humano o por una falla en el hardware o en el sistema operativo— resulta esencial poder reparar los datos afectados con el mínimo de demora y con tan poco efecto como sea posible sobre el resto del sistema. El DBA debe definir e implementar un esquema apropiado de control de daños que comprenda realizar backups de la BD para poder recuperar los datos cuando sea necesario.

✓ *Supervisar el rendimiento y responder a los requerimientos cambiantes*

Organizar el sistema de tal manera que se obtenga el rendimiento "ideal para la empresa" y de hacer los ajustes apropiados, —es decir, afinar— conforme las necesidades cambien.

## 5. Objetivos de las bases de datos



Entre los objetivos de efectividad de la base de datos están los siguientes [4]:

1. Asegurar que los datos se puedan compartir entre los usuarios para una diversidad de aplicaciones.
2. Mantener datos que sean exactos y consistentes.
3. Asegurar que todos los datos requeridos por las aplicaciones actuales y futuras se podrán acceder con facilidad.
4. Permitir a la base de datos evolucionar conforme aumenten las necesidades de los usuarios.
5. Permitir a los usuarios construir su vista personal de los datos sin preocuparse por la forma en que los datos se encuentren almacenados físicamente.

Esta lista de objetivos proporciona un recordatorio de las ventajas del enfoque de base de datos. Primero, la **compartición** de los datos significa que éstos deben almacenarse una sola vez. Como consecuencia, esto ayuda a lograr la **integridad** de los datos, debido a que los cambios en los datos se realizan con mayor facilidad y confiabilidad si éstos aparecen sólo una vez en lugar de en muchos archivos diferentes.

Cuando un usuario necesita datos específicos, una base de datos bien diseñada anticiparía dicha necesidad (o quizás ya se habrían usado en otra aplicación). Por lo tanto, es más probable que los datos estén disponibles en una base de datos que en un sistema de archivos convencional. Una base de datos bien diseñada también puede ser más **flexible** que los archivos separados; es decir, una base de datos puede evolucionar conforme cambien las necesidades de los usuarios y las aplicaciones.

Además, el enfoque de base de datos tiene la ventaja de permitir a los usuarios obtener su propia **vista** de los datos. Los usuarios no tienen que preocuparse por la estructura real de la base de datos o su almacenamiento físico.

## 6. Características de la metodología de Bases de Datos

Unas cuantas características distinguen la metodología de bases de datos de la metodología tradicional de programación con archivos. En el procesamiento tradicional de archivos, cada usuario define e implementa los archivos necesarios para una aplicación concreta como parte de la programación de esa aplicación. Por ejemplo, un usuario, la oficina de notificación de calificaciones, puede encargarse del mantenimiento de un archivo con los estudiantes y sus calificaciones. Los programas encargados de imprimir el certificado de estudios de un estudiante e introducir nuevas calificaciones en el archivo se implementan como parte de la aplicación.

Un segundo usuario, la oficina de contabilidad, puede encargarse del seguimiento de las cuotas de los estudiantes y sus pagos. Aunque ambos usuarios están interesados en datos relacionados con los estudiantes, cada uno mantiene archivos separados (y programas para la manipulación de esos archivos), porque cada uno requiere algunos datos que no están disponibles en los archivos del otro. Esta redundancia en la definición y el almacenamiento de datos da como resultado un derroche de espacio de almacenamiento y unos esfuerzos redundantes por mantener al día datos comunes.

En la metodología de bases de datos se mantiene un único almacén de datos, que se define una sola vez, y al que acceden varios usuarios. En los sistemas de archivos cada aplicación tiene libertad para asignar un nombre independientemente a los elementos de datos. Por el contrario, en una base de datos, los nombres o etiquetas de los datos se definen una vez, y son utilizados por consultas, transacciones y aplicaciones. Las principales características de la metodología de bases de datos frente a la metodología de procesamiento de archivos son las siguientes:

- Naturaleza auto-descriptiva de un sistema de bases de datos.

Una característica fundamental de la metodología de bases de datos es que el sistema de bases de datos no sólo contiene la propia base de datos, sino también una completa definición o descripción de la estructura de la base de datos y sus restricciones. Esta definición se almacena en el catálogo DBMS, que contiene información como la estructura de cada archivo, el tipo y el formato de almacenamiento de cada elemento de datos, y distintas restricciones de los datos. La información almacenada en el catálogo se denomina **metadatos** y describe la estructura de la base de datos principal.

- Aislamiento entre programas y datos, y abstracción de datos.

En el procesamiento de archivos tradicional, la estructura de los archivos de datos está incrustada en las aplicaciones, por lo que los cambios que se introducen en la estructura de un archivo pueden obligar a realizar cambios en todos los programas que acceden a ese archivo. Por el contrario, los programas que acceden a un DBMS no necesitan esos cambios en la



mayoría de los casos. La estructura de los archivos de datos se almacena en el catálogo DBMS, independientemente de los programas de acceso. Esta propiedad se conoce como *independencia programa-datos*.

La característica que permite la independencia programa-datos y la independencia programa-operación (cuando las aplicaciones de usuario pueden operar sobre los datos invocando operaciones por sus nombres y argumentos, independientemente de cómo estén implementadas tales operaciones) se denomina abstracción de datos.

- Soporte de varias vistas de los datos.

Normalmente una base de datos tiene muchos usuarios, cada uno de los cuales puede necesitar una perspectiva o vista diferente de la base de datos. Una vista puede ser un subconjunto de la base de datos o puede contener datos virtuales derivados de los archivos de la base de datos pero que no están explícitamente almacenados.

Algunos usuarios no tienen la necesidad de preocuparse por si los datos a los que se refieren están almacenados o son derivados. Un DBMS multiusuario cuyos usuarios tienen variedad de diferentes aplicaciones debe ofrecer facilidades para definir varias vistas.

- Compartición de datos y procesamiento de transacciones multiusuario.

Un DBMS multiusuario, debe permitir que varios usuarios puedan acceder a la base de datos al mismo tiempo. El DBMS debe incluir software de control de la concurrencia para que esos varios usuarios que intentan actualizar los mismos datos, lo hagan de un modo controlado para que el resultado de la actualización sea correcto. Estos tipos de aplicaciones se denominan, por lo general, aplicaciones de procesamiento de transacciones en línea (OLTP, *OnLine Transaction Processing*). Un papel fundamental del software DBMS multiusuario es garantizar que las transacciones concurrentes operan correcta y eficazmente.

Otras características generales que deben tener los DBMS son:

- Debe poseer un lenguaje de definición de datos (DDL) que permita fácilmente la creación de nuevas bases de datos, así como la modificación de su estructura.
- Debe poseer un lenguaje de manipulación de datos (DML), que permita la inserción, eliminación, modificación y consulta de los datos de la base, de la forma más eficiente y conveniente posible.
- Debe permitir el almacenamiento de enormes cantidades de datos (miles de millones de caracteres), sin que el usuario perciba una degradación en cuanto al rendimiento global del sistema.
- Debe permitir la gestión segura de los datos, con respecto a accesos no autorizados, y a accidentes producidos por los dispositivos mecánicos o electrónicos que soportan los datos almacenados.

## 7. Ventajas del enfoque de Base de datos

Algunas de las ventajas de utilizar un DBMS y las capacidades que un buen DBMS debe poseer se mencionan a continuación.

---

✓ **Control de la redundancia**

En el desarrollo tradicional de software que hace uso del procesamiento de archivos, cada grupo de usuarios mantiene sus propios archivos para manipular sus aplicaciones de procesamiento de datos. En la metodología de bases de datos, las vistas de los diferentes grupos de usuarios se integran durante el diseño de la base de datos. Idealmente, debemos tener un diseño que almacene cada elemento de datos lógico *sólo en un lugar* de la base de datos. Este hecho garantiza la coherencia y ahorra espacio de almacenamiento. Sin embargo, en la práctica, a veces es necesario recurrir a una **redundancia controlada** para mejorar el rendimiento de las consultas.

✓ **Coherencia de los datos**

Al eliminar o controlar la redundancia, reducimos el riesgo de que se produzcan incoherencias. Si un elemento de datos sólo se almacena una vez en la BD, las actualizaciones de su valor sólo tienen que hacerse una vez y el nuevo valor estará disponible de forma inmediata para todos los usuarios.

✓ **Restricción del acceso no autorizado**

Cuando varios usuarios comparten una base de datos grande, es probable que la mayoría de los mismos no tengan autorización para acceder a toda la información de la base de datos. Por ejemplo, los datos financieros se consideran a menudo confidenciales, y sólo las personas autorizadas pueden acceder a ellos. Además, algunos usuarios sólo pueden recuperar datos, mientras que otros pueden recuperarlos y actualizarlos. Un DBMS debe proporcionar seguridad y un subsistema de autorización.

✓ **Almacenamiento persistente para los objetos del programa**

Las bases de datos se pueden utilizar para proporcionar almacenamiento persistente a los objetos de programa y las estructuras de datos. Normalmente, los lenguajes de programación tienen estructuras de datos complejas, aunque los valores de las variables de un programa se descartan una vez que termina ese programa, a menos que el programador los almacene explícitamente en archivos permanentes. Los sistemas de bases de datos orientados a objetos son compatibles con los lenguajes de programación, y el software DBMS realiza automáticamente las conversiones necesarias. Se dice que dichos objetos son persistentes, porque sobreviven a la terminación de la ejecución del programa y otro programa lo puede recuperar más tarde.

✓ **Suministro de estructuras de almacenamiento para un procesamiento eficaz de las consultas**

Los sistemas de bases de datos deben proporcionar capacidades para *ejecutar eficazmente consultas y actualizaciones*. Como la base de datos normalmente se almacena en el disco, el DBMS debe proporcionar estructuras de datos especializadas para acelerar la búsqueda en el disco de los registros deseados. Con este fin se utilizan unos archivos auxiliares denominados *índices*.

El *módulo de procesamiento y optimización de consultas* del DBMS es el responsable de elegir un plan eficaz de ejecución de consultas para cada consulta basándose en las estructuras de almacenamiento existentes.

✓ **Copia de seguridad y recuperación**

Un DBMS debe ofrecer la posibilidad de recuperarse ante fallos del hardware o del software. El *subsistema de copia de seguridad y recuperación* del DBMS es el responsable de la recuperación, es decir de garantizar la restauración de la base de datos al estado anterior a que comenzase la ejecución de la transacción. Como alternativa, el subsistema de recuperación podría asegurarse de retomar la transacción en el punto en que se interrumpió para que todo su efecto se grabe en la base de datos.



---

✓ **Suministro de varias interfaces de usuario**

Como una base de datos la utilizan muchos tipos de usuarios con distintos niveles de conocimiento técnico, un DBMS debe proporcionar distintas interfaces de usuario, entre las que podemos citar los lenguajes de consulta para los usuarios casuales, las interfaces de lenguaje de programación para los programadores de aplicaciones, formularios y códigos de comando para los usuarios paramétricos, e interfaces por menú y en el idioma nativo para los usuarios independientes.

✓ **Representación de relaciones complejas entre los datos**

Una base de datos puede incluir numerosas variedades de datos que se interrelacionan entre sí de muchas formas.

✓ **Implementación de las restricciones de integridad**

La mayoría de las aplicaciones de bases de datos tienen ciertas restricciones de integridad que deben mantenerse para los datos. Un DBMS debe proporcionar servicios para definir e implementar esas restricciones. El tipo de restricción de integridad más simple consiste en especificar un tipo de datos por cada elemento de datos. Otro tipo de restricción especifica la unicidad en los valores del elemento de datos. Estas restricciones se derivan del significado o la **semántica** de los datos y del UoD que representan. Algunas restricciones pueden especificarse en el DBMS e implementarse automáticamente. Otras restricciones pueden tener que ser comprobadas por los programas de actualización o en el momento de introducir los datos. En las aplicaciones grandes es costumbre denominar estas restricciones como **reglas de negocio**.

✓ **Inferencia y acciones usando reglas**

Algunos sistemas de bases de datos ofrecen la posibilidad de definir *reglas de deducción* para *inferir* información nueva a partir de los hechos guardados en la base de datos. Estos sistemas se denominan sistemas de bases de datos deductivos. En los sistemas de bases de datos relacionales actuales es posible asociar *triggers* a las tablas. Un *trigger* es una forma de regla que se activa con las actualizaciones de la tabla, lo que conlleva la ejecución de algunas operaciones adicionales sobre otras tablas, el envío de mensajes, etcétera.

✓ **Implicaciones adicionales de utilizar la metodología de bases de datos**

- Potencial para implementar estándares: podrían ser estándares de la organización, nacionales, internacionales, como por ejemplo determinados formatos para facilitar el intercambio de datos entre sistemas, estándares de documentación, reglas de acceso, etc.
- Tiempo de desarrollo de aplicación reducido: una vez construida la base de datos se requerirá mucho menos tiempo para crear nuevas aplicaciones.
- Flexibilidad: modificar la estructura de la bases de datos debido a requerimientos cambiantes es muy simple, sin afectar a los datos almacenados y programas de aplicación existentes.
- Disponibilidad de la información actualizada: en el momento en que un usuario actualiza la base de datos, todos los demás usuarios pueden ver inmediatamente esta actualización.
- Economías de escala: al tener una fuente centralizada de datos, pueden reducirse enormemente los costos.
- Más información a partir de la misma cantidad de datos.
- Mejor nivel de concurrencia.
- Mantenimiento más sencillo gracias a la independencia de los datos.

## 8. Beneficios del enfoque de Base de datos

### ❖ Los datos pueden compartirse

Compartir no sólo significa que las aplicaciones existentes puedan compartir la información de la base de datos, sino también que sea posible desarrollar nuevas aplicaciones para operar sobre los mismos datos. En otras palabras, es posible satisfacer los requerimientos de datos de aplicaciones nuevas sin tener que agregar información a la base de datos.

### ❖ Es posible reducir la redundancia

En sistemas que no son de bases de datos, cada aplicación tiene sus propios archivos exclusivos. A menudo este hecho puede conducir a una redundancia considerable de los datos almacenados, con el consecuente desperdicio de espacio de almacenamiento. No pretendemos sugerir que toda la redundancia pueda o deba necesariamente ser eliminada. En ocasiones hay razones sólidas, tácticas o del negocio, para mantener varias copias distintas de los mismos datos. Sin embargo, sí pretendemos sugerir que cualquier redundancia de este tipo debe ser controlada cuidadosamente.

### ❖ Es posible (hasta cierto grado) evitar la inconsistencia

Éste es en realidad el corolario del punto anterior. Suponga que un hecho más real —digamos que el empleado E3 trabaja en el departamento D8— está representado por dos entidades distintas en la base de datos. Suponga también que el DBMS no está al tanto de esta duplicidad (es decir, la redundancia no está controlada). Entonces necesariamente habrá ocasiones en las que las dos entidades no coincidan: digamos, cuando una de ellas ha sido actualizada y la otra no. En esos momentos, decimos que la base de datos es inconsistente. Resulta claro que una base de datos en un estado inconsistente es capaz de proporcionar a sus usuarios información incorrecta o contradictoria.

Por supuesto, si el hecho anterior es representado por una sola entrada (es decir, si se elimina la redundancia), entonces no puede ocurrir tal inconsistencia. Como alternativa; si no se elimina la redundancia pero se controla (haciéndola del conocimiento del DBMS), entonces el DBMS puede garantizar que la base de datos nunca será inconsistente a los ojos del usuario, asegurando que todo cambio realizado a cualquiera de las dos entidades será aplicado también a la otra en forma automática. A este proceso se le conoce como propagación de actualizaciones.

### ❖ Es posible brindar un manejo de transacciones

Una transacción es una unidad de trabajo lógica, que por lo regular comprende varias operaciones de la base de datos (en particular, varias operaciones de actualización). El ejemplo común es el de transferir una cantidad de efectivo de una cuenta A a otra cuenta B. Es claro que aquí se necesitan dos actualizaciones, una para retirar el efectivo de la cuenta A y la otra para depositarlo en la cuenta B. Si el usuario declara que las dos actualizaciones son parte de la misma transacción, entonces el sistema puede en efecto garantizar que se hagan ya sea ambas o ninguna de ellas, aun cuando el sistema fallara (digamos por falta de suministro eléctrico) a la mitad del proceso.

### ❖ Es posible mantener la integridad

El problema de la integridad es el de asegurar que los datos de la base de datos estén correctos. La inconsistencia entre dos entradas que pretenden representar el mismo "hecho" es un ejemplo de la falta de integridad desde luego, este problema en particular puede surgir sólo si existe redundancia en los datos almacenados. No obstante, aun cuando no exista redundancia, la base de datos podría seguir conteniendo información incorrecta. Por ejemplo, un empleado podría aparecer con 400 horas laboradas durante la semana, en lugar de 40; o como parte de un departamento que no existe. El

control centralizado de la base de datos puede ayudar a evitar estos problemas (en la medida de lo posible) permitiendo que el administrador de datos defina y el DBA implemente las restricciones de integridad (también conocidas como reglas del negocio) que serán verificadas siempre que se realice una operación de actualización.

#### ❖ Es posible hacer cumplir la seguridad

Al tener la completa jurisdicción sobre la base de datos, el DBA (por supuesto, bajo la dirección apropiada del DA) puede asegurar que el único medio de acceso a la base de datos sea a través de los canales adecuados y por lo tanto puede definir las reglas o restricciones de seguridad que serán verificadas siempre que se intente acceder a datos sensibles. Es posible establecer diferentes restricciones para cada tipo de acceso (recuperación, inserción, eliminación, etcétera) para cada parte de la información de la base de datos. Sin embargo, observe que sin dichas restricciones la seguridad de los datos podría de hecho estar en mayor riesgo que en un sistema de archivos tradicionales (dispersos); es decir, la naturaleza centralizada de un sistema de base de datos requiere, en cierto sentido, que también sea establecido un buen sistema de seguridad.

#### ❖ Es posible equilibrar los requerimientos en conflicto

Al conocer los requerimientos generales de la empresa (a diferencia de los requerimientos de los usuarios individuales), el DBA puede estructurar los sistemas de manera que ofrezcan un servicio general que sea "el mejor para la empresa" (de nuevo bajo la dirección del DA). Por ejemplo, es posible elegir una representación física de los datos almacenados que proporcione un acceso rápido para las aplicaciones más importantes (posiblemente a costa de un acceso más lento para otras aplicaciones).

#### ❖ Es posible hacer cumplir los estándares

Con el control central de la base de datos, el DBA (una vez más, bajo la dirección del DA) puede asegurar que todos los estándares aplicables en la representación de los datos sean observados. Estos estándares podrían incluir alguno o todos los siguientes: departamentales, de instalación, corporativos, de la industria, nacionales e internacionales. Es conveniente estandarizar la representación de datos, en particular como un auxiliar para el intercambio de datos o para el movimiento de datos entre sistemas. En forma similar, los estándares en la asignación de nombres y en la documentación de los datos también son muy convenientes como una ayuda para compartir y entender los datos.

#### ❖ Independencia de datos

La independencia de datos constituye no solo una ventaja sino también uno de los objetivos de los SBD. Se desarrolla en el apartado siguiente.

### 8.1. Independencia de datos

Una aplicación es **dependiente de los datos**, cuando no es posible modificar la representación física de los mismos (la forma en que los datos están físicamente representados en el almacenamiento) o la técnica de acceso a ellos (la forma en que son accedidos físicamente) sin afectar a la aplicación de manera drástica. En un SBD significaría un gran inconveniente permitir que algunas aplicaciones fuesen dependientes de los datos. En primer lugar, porque las distintas aplicaciones podrían requerir visiones diferentes de los mismos datos; por otro lado, el DBA debe tener la libertad de cambiar las representaciones físicas o la técnica de acceso en respuesta a los requerimientos cambiantes, sin tener que modificar las aplicaciones existentes.

De aquí que dar independencia a los datos sea un objetivo principal de los sistemas de base de datos. Podemos definir la independencia de los datos como **la inmunidad de las aplicaciones a cambios en la representación física y en la técnica de acceso**; lo que implica desde luego que las aplicaciones involucradas no dependan de ninguna representación física o técnica de acceso en particular.

Entre otras cosas, la independencia de los datos requiere que se haga una clara distinción entre el **modelo de datos y su implementación**.

## 9. Desventajas de enfoque de Base de datos

Como desventajas de la técnica de base de datos, se pueden mencionar las siguientes: [5]

- **Complejidad:** Para que un buen SGBD pueda proporcionar la funcionalidad esperada, el SGBD tiene que ser un programa de software de gran complejidad. Los desarrolladores y diseñadores de base de datos, los administradores de datos y de base de datos y los usuarios finales deben ser capaces de comprender esta funcionalidad para poder aprovecharla al máximo.
- **Tamaño:** La complejidad y el amplio rango de funcionalidades hacen que el SGBD sea un programa de software de gran tamaño, que ocupa muchos megabytes de espacio de disco y requiere una gran cantidad de memoria para poder ejecutarse de manera eficiente.
- **Costo del SGBD:** El costo del SGBD varía considerablemente, dependiendo del entorno y de la funcionalidad proporcionada. Ejemplos: SGBD monousuario para PC o multiusuario para mainframe, para cientos de usuarios.
- **Costos de hardware adicional:** los requisitos de almacenamiento en disco para el SGBD y la base de datos pueden imponer la compra de espacio de almacenamiento adicional. Para obtener las prestaciones requeridas puede que sea necesario comprar una plataforma de hardware mayor.
- **Costos de conversión:** El costo del SGBD y del hardware adicional puede ser insignificante si lo comparamos con el costo de convertir las aplicaciones existentes para que se ejecuten sobre el nuevo SGBD y sobre la nueva plataforma hardware.
- **Prestaciones:** Normalmente los sistemas basados en archivos se escriben para una aplicación específica como por ejemplo una prestación de facturación. Por el contrario, un SGBD se escribe para constituir una solución más general, con el fin de que puedan utilizarlo muchas aplicaciones y no solo una aplicación concreta.
- **Mayor impacto de los fallos:** La centralización de los recursos implementa la vulnerabilidad del sistema. Puesto que todos los usuarios y aplicaciones dependen de la disponibilidad de SGBD el fallo de ciertos componentes puede hacer que se detengan todas las operaciones.

## 10. Síntesis cronológica

Se expone a continuación una breve historia de las aplicaciones de bases de datos.

Muchas de las primeras aplicaciones de bases de datos almacenaban registros en grandes organizaciones, como corporaciones, universidades, hospitales y bancos. En muchas de esas aplicaciones había muchos registros de estructura parecida. También había muchos tipos de registros y muchas interrelaciones entre ellos.

Uno de los principales problemas que se presentaban era la mezcla de relaciones conceptuales con el almacenamiento físico y la ubicación de los registros en el disco. Aunque esto ofrecía un acceso muy eficaz para las consultas y las transacciones originales para las que fue diseñada la base de datos, no



proporcionaba suficiente flexibilidad para acceder eficazmente a los registros cuando se identificaban consultas y transacciones nuevas, que requerían una organización diferente del almacenamiento. También era muy laborioso reorganizar la base de datos cuando había cambios en los requisitos de la aplicación. Otro defecto de los primeros sistemas era que sólo proporcionaban interfaces de lenguaje de programación. La mayoría de esos sistemas de bases de datos se implantaron en grandes y costosos computadores *mainframe* a mediados de la década de 1960, y a lo largo de las décadas de 1970 y 1980. Los principales tipos de esos sistemas estaban basados en tres paradigmas principales: sistemas *jerárquicos*, sistemas basados en un *modelo de red* y sistemas de *archivos inversos*.

Las *bases de datos relacionales* se propusieron originalmente para separar el almacenamiento físico de los datos de su representación conceptual, así como para proporcionar una base matemática para el almacenamiento de contenidos. El modelo de datos relacional también introdujo lenguajes de consulta de alto nivel que proporcionaban una alternativa a las interfaces de lenguaje de programación; por tanto, era mucho más rápido escribir consultas nuevas. Los sistemas relacionales estaban pensados para ofrecer flexibilidad en el desarrollo de nuevas consultas y para reorganizar la base de datos cuando cambiaran los requisitos. Los sistemas relacionales experimentales desarrollados a finales de la década de 1970 y los sistemas de administración de bases de datos relacionales (RDBMS) comerciales que aparecieron a principios de la década de 1980 eran muy lentos. Su rendimiento mejoró con el desarrollo de nuevas técnicas de almacenamiento e indexación y unas técnicas mejores de procesamiento y optimización. Eventualmente, las bases de datos relacionales se convirtieron en el tipo de sistema de bases de datos predominante para las aplicaciones de bases de datos tradicionales.

El surgimiento de los lenguajes de programación orientados a objetos en la década de 1980 y la necesidad de almacenar y compartir objetos estructurados complejos induce al desarrollo de las bases de datos orientadas a objetos (OODB). Inicialmente, las OODB estaban consideradas como competidoras de las bases de datos relacionales, porque proporcionaban más estructuras de datos generales. También incorporaban muchos de los útiles paradigmas de la orientación a objetos, como los tipos de datos abstractos, la encapsulación de operaciones, la herencia y la identidad de objeto. No obstante, la complejidad del modelo y la carencia de un estándar contribuyeron a su limitado uso.

Más adelante, surgió la World Wide Web como una gran red de computadoras interconectadas, en la cual los usuarios pueden crear documentos utilizando un lenguaje de publicación web, como HTML o similares, y almacenar esos documentos en servidores web desde los que otros usuarios (clientes) pueden acceder a ellos. Los documentos se pueden enlazar mediante **hipervínculos**, que son punteros a otros documentos. En la década de 1990 apareció el comercio electrónico (*e-commerce*) como una aplicación trascendental en la Web. Parte de la información que aparecía en las páginas web frecuentemente eran datos que se extraían dinámicamente de DBMSs. Se desarrollaron varias técnicas que permitían el intercambio de datos en la Web. Ya en el siglo XXI, XML (eXtensible Markup Language - lenguaje de marcas extensible) fue considerado como el principal estándar para el intercambio de datos entre varios tipos de bases de datos y páginas web. XML combina conceptos de los modelos utilizados en los sistemas de documentación con conceptos de modelado de bases de datos.

Con la llegada de la Web y la proliferación de las páginas HTML ejecutándose por miles de millones, es necesario aplicar otras técnicas para procesar los datos en la Web, que incluyen ahora nuevos tipos como imágenes, documentos y objetos que se activan y modifican dinámicamente.



## Referencias Bibliográficas

- [1] Fundamentos de Sistemas de Bases de Datos. Ramez Elmasri & Shamkant Navathe. Addison-Wesley. Tercera edición. 2002. [Capítulo 1]
- [2] Fundamentos de Bases de Datos. Abraham Silberschatz, Henry F. Korth, S. Sudarshan. Mc Graw-Hill. Quinta Edición. 2006. [Capítulo 1]
- [3] Introducción a los sistemas de bases de datos. Date, C. J. Addison-Wesley. Séptima edición. 2001. [Capítulo 1]
- [4] Análisis y Diseño de sistemas. E. Kendall, Kenneth & E. Kendall, Julie. Pearson Educación. Sexta Edición. [Capítulo 13]
- [5] Sistemas de Bases de Datos - Un enfoque práctico para diseño, implementación y gestión. Thomas Connolly, Carolyn Begg. Pearson - Addison Wesley. Cuarta edición. 2005. [Capítulos 1 y 2]