



DIVISIÓN ACADÉMICA DE INGENIERÍA

DEPARTAMENTO ACADÉMICO DE
COMPUTACIÓN

**Notas del curso de
Desarrollo de Aplicaciones
Informáticas**

Autor:

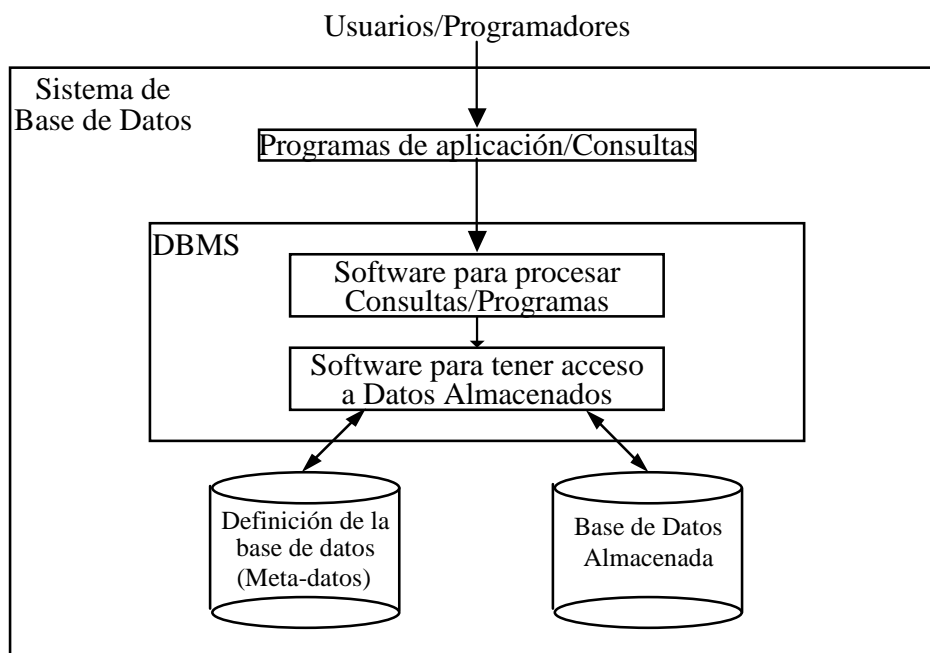
Dr. Felipe López G.

Agosto 2017

1 Introducción al procesamiento con bases de datos

1.1 Conceptos de bases de datos

1.1.1 Sistema de base de datos



Una **base de datos** es una **colección de datos relacionados** la cual tiene las siguientes propiedades implícitas:

- representa algún aspecto del mundo real, llamado el **minimundo** o el **Universo de Discurso (UD)**. Cambios en el minimundo se reflejan en la base de datos;
- su colección de datos es lógicamente coherente con un significado inherente; esto es, un conjunto de datos al azar normalmente no constituye una base de datos;
- se diseña, se construye y se llena con datos para un propósito específico;
- puede ser usada desde múltiples aplicaciones por diversos grupos de usuarios.

Un **DBMS (DataBase Management System)** (o SABD- Sistema de Administración de Bases de Datos) es una colección de programas que permite a usuarios crear y mantener bases de datos. Es un software de *propósito general* que facilita los procesos de:

- definición (especificación de estructuras, tipos de datos y restricciones),
- construcción (almacenamiento de los datos en algún medio) y
- manipulación (consultas y actualización)

de bases de datos para diversas aplicaciones.

Un DBMS puede ser de *propósito especial* para manipular bases de datos que tienen un propósito específico.

Un **lenguaje de 4a. generación (4GL)** es un lenguaje de programación con un conjunto poderoso de instrucciones el cual permite crear aplicaciones que manipulan bases de datos (y que llamaremos **aplicaciones de bases de datos**). Actualmente estos lenguajes incluyen elementos para manejar en forma gráfica la información de una base de datos (Visual Basic y Java son ejemplos). Con un 4GL se pueden definir aplicaciones constituidas por formas, reportes, menús, etc., que utilicen el contenido de una base de datos a fin de satisfacer las necesidades de usuarios de la misma.

A la conjunción de base de datos con el software que la manipula le llamaremos **Sistema de Base de Datos**.

1.1.2 Características del enfoque de base de datos

En estas notas se usará el término **enfoque de bases de datos** para indicar que el manejo de la base de datos se hará con un DBMS. A continuación se detallan las características principales de este enfoque:

- **Naturaleza auto-descriptiva de un sistema de base de datos**

El sistema de base de datos contiene no sólo a la base de datos misma sino también una definición completa de su estructura. La definición se almacena en el **catálogo** del sistema. Esta información también se conoce como **meta-datos** o **diccionario de datos** o **directorío de datos**. El catálogo es usado por el DBMS para poder trabajar con *cualquier base de datos*. En un solo catálogo se mantienen las estructuras de las diversas bases de datos requeridas por los usuarios; éste se define una sola vez, salvo cambios posteriores, y después es usado por diversas aplicaciones. Normalmente la redundancia es poca y los cambios son más fáciles de hacer.

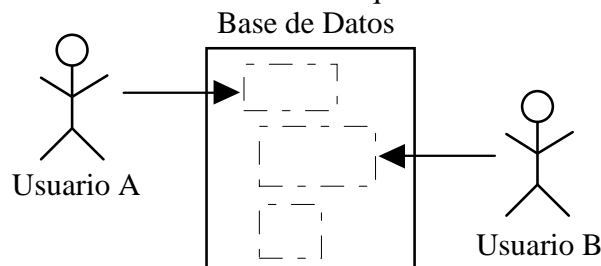
- **Aislamiento entre datos y programas**

Los módulos del DBMS son independientes de cualesquier archivos específicos, ya que la estructura de éstos está almacenada en el catálogo del DBMS. Esta propiedad se conoce como **independencia entre programas y datos**.

El DBMS proporciona a los usuarios una **representación lógica** de los datos sin incluir muchos de los detalles de cómo éstos son almacenados.

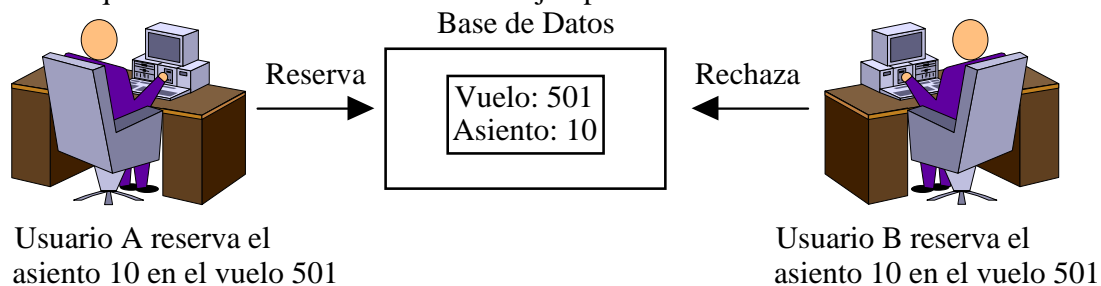
- **Soporte de múltiples vistas de los datos**

Una base de datos normalmente tiene muchos usuarios, cada uno de los cuales puede requerir diferentes perspectivas o **vistas** de la base de datos. Una vista puede ser un subconjunto de la base de datos o puede contener datos **virtuales** que son derivados de otros datos de la base.



- **Compartimiento de datos y procesamiento de transacciones multiusuario**

El DBMS debe incluir software para **control de la concurrencia** a fin de asegurar que varios usuarios que tratan de actualizar los mismos datos, lo hagan en una forma controlada de tal manera que los resultados sean correctos. Ejemplo:



A este tipo de procesamiento se le conoce como de **transacciones multiusuario**.

1.1.3 Ventajas de usar un DBMS

- **Control de la redundancia**

Cuando se hace el diseño de una base de datos se debe almacenar *una sola vez* cada dato lógico, tal como el nombre o fecha de nacimiento de una persona. Esto evita inconsistencias en la información y a la vez ahorra espacio de almacenamiento. En algunos casos es deseable, o necesario, tener una **redundancia controlada** para optimizar tiempos de respuesta en consultas o porque así lo exige el modelo de datos empleado. El DBMS debe proporcionar este recurso.

- **Restricciones de integridad**

Esta característica se encuentra relacionada con la del punto anterior. Estas restricciones van desde permitir que a la base sólo ingresen **datos correctos**, por ejemplo valores en un rango, hasta forzar que un registro **esté relacionado** con otros registros de una cierta manera. Normalmente estas restricciones se derivan del significado o **semántica** de los datos del minimundo. La mayoría de las restricciones pueden ser controladas por el DBMS; otras deben controlarse por medio de módulos especializados programados.

- **Restricción a accesos no autorizados (Seguridad)**

Normalmente el DBMS debe proporcionar un subsistema de **seguridad y autorización** que el DBA utiliza para asignar cuentas y privilegios (restricciones) a los usuarios de una base de datos. Con este subsistema se puede definir el tipo de operaciones que un usuario puede efectuar y a qué partes de la base puede tener acceso. El uso de vistas es otra forma de controlar el acceso.

- **Múltiples interfaces de usuario**

Debido a los diversos tipos de usuarios que usan una base de datos, el DBMS debe proporcionar una variedad de interfaces para los mismos. Éstas pueden incluir: lenguajes de consulta; lenguajes de programación, con capacidades gráficas inclusive; interfaces basadas en menús e interfaces de lenguaje natural.

- **Representación de vínculos complejos entre datos**

El DBMS debe tener la capacidad de representar una gran variedad de vínculos complejos entre los datos, así como poder recuperarlos y actualizarlos fácil y eficientemente.

- **Respaldo (Backup) y Recuperación (Recovery)**

El DBMS debe tener los recursos para recuperarse de fallas de hardware y de software. El subsistema de **respaldo y recuperación** es el responsable por esta tarea. Por ejemplo, si el sistema falla a mitad de una transacción, el DBMS debe poder restaurar la base de datos hasta el estado previo que tenía antes de la falla, y así asegurar que su contenido es válido.

1.1.4 Implicaciones adicionales del enfoque de base de datos

La utilización de bases de datos en las organizaciones presenta varias implicaciones las cuales pueden beneficiar a las funciones computacionales de las mismas.

- **Potencial para establecer estándares**

En una organización grande el enfoque de bases de datos permite al DBA establecer estándares entre los usuarios de las mismas. Esto facilita la comunicación y la cooperación entre los departamentos, proyectos y usuarios dentro de la organización. Los estándares pueden ser definidos para los nombres y formatos de los datos, formatos de despliegue, estructuras de reportes, terminología, etc.

- **Reducción del tiempo de desarrollo de aplicaciones**

Una característica principal de las bases de datos es que el desarrollo de una nueva aplicación toma poco tiempo. El diseñar e implementar una nueva base de datos a partir de cero puede llevar mucho más tiempo que escribir una sola aplicación con archivos especializados. Sin embargo, ya que la base de datos está en funcionamiento, se requiere mucho menos tiempo para crear nuevas aplicaciones usando las facilidades del DBMS. Se estima un ahorro de un sexto a un cuarto de tiempo con respecto a sistemas de archivos especializados.

- **Disponibilidad de información "al día"**

Cuando una actualización a la base de datos es hecha por un usuario, todos los demás usuarios pueden ver inmediatamente dicha actualización. Esta disponibilidad de información "al día" es esencial para muchas transacciones, tales como sistemas de reservaciones y bases de datos de bancos.

1.1.5 Ventajas y desventajas en el uso de un DBMS

Ventajas

- Al minimizar la redundancia, disminuye el margen de error.
- Independencia entre hardware y datos.
- Independencia entre datos y programas de aplicación.
- Los datos pueden compartirse (conurrencia).
- Disminución del tiempo de programación.
- Nuevas aplicaciones se pueden agregar más fácilmente.
- Mecanismos de seguridad superiores a los convencionales.
- Es más fácil recuperar información cuando ocurre una falla.

Desventajas

- Un sistema de bases de datos normalmente es menos eficiente (en tiempo y en espacio) que el mismo sistema con procesamiento de archivos.
- El DBMS consume muchos recursos de hardware (memoria central y secundaria), si no es para PC.
- Mayor costo (de adquisición y de mantenimiento), si no es para PC.
- En bases de datos centralizadas el sistema es más vulnerable a catástrofes físicas.

1.1.6 Usuarios de bases de datos

Para bases de datos **personales**, normalmente la misma persona define, construye y manipula la base de datos. Para bases de datos **multiusuario grandes**, con varias decenas o cientos de usuarios, diversos tipos de personal están involucrados:

- **Administrador de la base de datos (DBA)**

Es el responsable por la administración de los recursos tanto del DBMS como de las bases de datos que tenga la organización. El DBA se encarga de autorizar el acceso a las bases de datos, coordinar y supervisar su uso, adecuar el sistema para un uso eficiente, etc. El DBA puede ser una persona o un equipo de personas de la organización.

- **Diseñadores de la base de datos**

Son los responsables de identificar los datos a ser almacenados en una base de datos y de escoger las estructuras apropiadas para representar y almacenar estos datos. Normalmente definen las vistas de los diversos grupos de usuarios para al final integrar una base de datos con todas ellas.

- **Programadores de aplicaciones (ingenieros de software)**

Son los encargados de determinar los requerimientos de los usuarios, de desarrollar las especificaciones para los programas y de implementarlos. También se encargan de hacer las pruebas, depuraciones, documentación y mantenimiento de las aplicaciones.

- **Usuarios finales**

Son las personas que tienen acceso a la base de datos para hacer consultas y actualizaciones y generar reportes. El rango es amplio y puede ir desde usuarios simples (capturistas) hasta usuarios experimentados (ingenieros, científicos, analistas de negocios, etc.).

1.2 Ejemplos de bases de datos

En esta sección se describen varios ejemplos de bases de datos que muestran el espectro de sistemas en las que éstas se pueden usar, así como las características importantes de cada tipo de base de datos.

- **Micro-empresa para pintar casas**

Es una micro-empresa, compuesta por dos personas de tiempo completo y algunos trabajadores eventuales, que se dedica a pintar casas. Es una micro-empresa que ya tiene varios años en el mercado, con una buena reputación, que consigue la mayor parte de sus trabajos por recomendaciones de clientes y, ocasionalmente, por contratistas.

Cuentan con una cartera de varias docenas de clientes y emplean una pequeña base de datos personal para guardar información importante acerca de trabajos realizados anteriormente. Esto con el fin de poder obtener datos relevantes cuando se va a efectuar un nuevo trabajo, ya sea para un cliente nuevo (recomendado o no) o para un cliente anterior.

Enseguida se muestra un ejemplo de esta base de datos:

Table: Clientes		Table: Trabajos	
Field Name	Data Type	Field Name	Data Type
Id_cliente	Number	Id_trabajo	Number
Nombre	Text	Fecha	Date/Time
Teléfono	Text	Descripción	Text
Domicilio	Text	Presup_original	Currency
Cod_postal	Number	Cant_cobrada	Currency
Id_fuente	Number	Id_cliente	Number

Table: Contactos	
Field Name	Data Type
Id_fuente	Number
Nombre	Text
Teléfono	Text

Las bases de datos personales, en general, no son grandes, contienen pocas tablas y almacenan poca información.

Los DBMS típicos para estas bases de datos son: Microsoft Access, Microsoft FoxPro, MySQL y las versiones para PC de los manejadores grandes (normalmente gratuitas).

- **Agencia de venta de automóviles**

Las bases de datos pueden ser más complicadas que la del ejemplo anterior. Considérese ahora una agencia de venta de automóviles en la cual laboran dos supervisores, cuatro agentes de ventas y un administrador.

La base de datos contiene información sobre modelos en venta, clientes, ventas realizadas, etc. Esta base de datos es compartida por el personal de la agencia y está localizada en una red de área local formada por:

- un servidor de base de datos,
- una PC para el administrador,
- dos PC's para los supervisores, y
- cuatro PC's para los agentes de ventas.

La base de datos requerida para gestionar la información de esta aplicación es más complicada que la de la micro-empresa que pinta casas. Un ejemplo de las tablas que podría contener es el siguiente (la notación es del modelo relacional, que se verá en el capítulo 3):

Modelos(IdModelo, Descripción, PrecioActual, CantUnidades)

Unidades(NoSerie, FechaAdquisición, IdModelo)

Agentes(IdAgente, Nombre, Teléfono, FechaIngreso)

Clientes(IdCliente, Nombre, Domicilio, Teléfono, Notas)

VentasHechas(IdVenta, Fecha, IdAgente, IdCliente)

UnidadesVendidas(IdVenta, NoSerie, PrecioVenta)

Equipos(IdEquipo, Nombre, Descripción, Costo)

EquipoExtra(IdVenta, NoSerie, IdEquipo)

Obviamente, la información almacenada en esta base de datos es mucho mayor que en el primer caso. Además, esta base de datos es multiusuario.

Los DBMS típicos para estas bases de datos son: SQL Server, MySQL (versión avanzada) y las versiones menores o intermedias de los manejadores grandes.

- **Sistema de transacciones bancarias (cuentas de cheques)**

Este es un ejemplo de un sistema de base de datos de mucho mayor complejidad que los dos anteriores. El sistema debe poder:

- almacenar los datos generales de los clientes y de las cuentas que poseen,
- registrar las transacciones que se realizan y el lugar en que se efectúan,
- generar estados de cuenta,
- llevar a cabo consolidaciones, etc.

Además, el sistema debe permitir acceso multiusuario ya que las diversas transacciones se pueden efectuar:

- en ventanilla (en varias decenas de sucursales),
- a través de cajero automático (varias decenas),
- por Internet,
- por teléfono, etc.

Esta base de datos atiende simultáneamente a cientos de usuarios. Así, la base de datos es grande y compleja, pudiendo constar de varias decenas de tablas, conteniendo cientos de miles de datos.

- **Agencia de viajes turísticos**

Este ejemplo es el de un sistema de base de datos asociado a un sitio de Internet que se utiliza para: promover lugares turísticos y las características de los mismos, obtener datos y direcciones electrónicas de los clientes que visitan el sitio, y almacenar solicitudes de reservación para estos lugares, incluyendo hoteles y servicios.

La información que se maneja para la base de datos incluye datos multimedia: fotos, videos, sonido, etc. Además, la aplicación permite a los usuarios navegar por la misma, mostrando ventanas que contienen más detalles sobre determinadas características de los lugares. La cantidad de información manejada puede ser similar a la del segundo ejemplo.

Las características principales que distinguen a este ejemplo de los tres anteriores son las siguientes:

- Gran parte de la información manejada es estructurada (como pueden ser los datos de los clientes), pero otra parte es no estructurada (la información multimedia).
- El acceso a la información por parte de los clientes se hace a través de un navegador estándar, por lo que no se necesita software adicional en su PC.
- Se usa la tecnología estándar orientada a Internet para transferir los datos entre el navegador del cliente, la aplicación del sitio y la base de datos.

Comparación de las bases de datos

Estos cuatro ejemplos representan una muestra de los usos de la tecnología de las bases de datos. Cientos de miles de ellas son como la usada por la micro-empresa que pinta casas: bases de datos monousuario, con una cantidad relativamente pequeña de datos. Las formas y reportes que generan normalmente son simples y directos.

Otras son como la usada por la agencia de autos; tienen más de un usuario pero generalmente menos de veinte o treinta usuarios en total. Contienen una cantidad moderada de datos y las formas y reportes deben ser de cierta complejidad para satisfacer la diferentes funciones de la organización.

Las bases de datos más grandes son como la del sistema bancario que tiene cientos de usuarios y varios gigabytes de datos. En general, muchas aplicaciones las usan, teniendo cada una sus propias formas y reportes. En la siguiente tabla se resumen las características importantes de cada una de estos tipos de bases de datos:

Tipo de base de datos	Ejemplo	No. típico de usuarios simultáneos	Tamaño típico	Cantidad típica de tablas
Personal	Micro-empresa pintora	1	Varias decenas de megabytes	<10
Grupo	Agencia de autos	<25	Varias centenas de megabytes	<30
Organización	Sistema bancario	Cientos o miles	Varias decenas/centenas de gigabytes	>30
Internet	Agencia de viajes	Cientos o miles	Varias decenas/centenas de megabytes o gigabytes	<30

1.3 Conceptos y arquitectura de un sistema de base de datos

1.3.1 Modelos de datos, esquemas e instancias

Una característica fundamental del enfoque de base de datos es que proporciona un cierto nivel de abstracción de los datos, ocultando detalles de su almacenamiento que no son necesarios para la mayor parte de los usuarios de una base de datos. Un modelo de datos es la herramienta principal usada para proporcionar esta abstracción.

Un modelo de datos:

- es un conjunto de conceptos usado para describir la **estructura** de una base de datos (el término estructura se refiere a la conjunción de tipos de datos, vínculos y restricciones que deben observarse para los datos);
- en la mayoría de los casos, incluye un conjunto de **operaciones básicas** para especificar recuperaciones y actualizaciones sobre la base de datos;
- puede incluir conceptos para especificar **comportamiento**, lo cual se refiere a especificar un conjunto de **operaciones definidas por el usuario** permitidas sobre la base de datos.

Muchos modelos de datos han sido propuestos. Se pueden definir tres categorías según el tipo de conceptos que proporcionan:

- **conceptuales (o de alto nivel):** proporcionan conceptos que están más cercanos a la forma en que los usuarios perciben los datos. Frecuentemente se auxilian en diagramas para plasmar un modelo para un caso específico;
- **de representación (o de implementación):** proporcionan conceptos que pueden ser entendidos por usuarios finales pero que no están demasiado lejos del modo en que los datos están organizados dentro de la computadora. Estos modelos son los usados en la mayoría de los DBMS comerciales: relacional, de objetos y objeto/relacional.
- **físicos (o de bajo nivel):** brindan conceptos que describen los detalles de la manera en que los datos están almacenados en la computadora. En general, estos conceptos están dirigidos a especialistas en Computación más que a usuarios finales típicos;

Esquemas e instancias

En cualquier modelo de datos es importante distinguir entre la *descripción* de la base de datos y la *base de datos en sí misma*.

La descripción de la base de datos se conoce como **esquema de la base de datos (o metadatos)**. Este esquema se especifica durante el diseño de la base de datos y se espera que no cambie con frecuencia. Un esquema dibujado se conoce como **diagrama del esquema**. Cada unidad en el esquema es un **elemento del esquema**.

Ejemplo:

PROFESOR

Id_prof	Nom_prof	Categoría
---------	----------	-----------

CURSO

Clave	Nom_cu	Creds	Id_prof
-------	--------	-------	---------

ALUMNO

Matri	Nom_al	Carr	Prom
-------	--------	------	------

CURSA

Matri	Clave	Calif
-------	-------	-------

Un diagrama de esquema muestra sólo *algunos aspectos* del esquema, tal como el nombre de los elementos y de los campos de datos; pero no muestra otros, como el tipo de los datos, los vínculos entre ellos o restricciones sobre los mismos.

Los datos actuales en una base de datos pueden cambiar frecuentemente. Al conjunto de datos que está en la base de datos en un momento particular del tiempo se le conoce como **estado de la base de datos** (o conjunto de **ocurrencias** o **instancias**). En un estado dado, cada elemento del esquema tiene su propio *conjunto actual* de instancias.

Ejemplo:

CURSO

Clave	Nom_cu	Creds	Id_prof
300	Computacion I	9	1
310	Algoritmica	9	1
605	Economia V	8	5

Cada que se actualiza la base de datos (inserción, modificación o eliminación de datos), ésta cambia de un estado a otro.

Cuando se **define** una nueva base de datos, sólo se especifica su esquema al DBMS. Su estado inicial es el "estado vacío". Cuando se **cargan** los primeros datos, éstos definen el "estado inicial" de la base de datos. Después, cada actualización hace que la base de datos pase de un estado a otro.

En cualquier momento, un estado de la base de datos debe ser **válido**, esto es, el estado debe satisfacer tanto la estructura como las restricciones especificadas para la base de datos.

1.3.2 Lenguajes e interfaces de un DBMS

Lenguajes

El lenguaje **SQL** (Structured Query Language) es el **estándar** para manejadores relacionales. Contiene elementos para:

- definir el esquema lógico de la base de datos. Esta definición se compila y se guarda en el catálogo del DBMS,
- especificar el esquema interno de la base de datos. No es común encontrar esta parte en los DBMS, salvo, posiblemente, en algunos de gran complejidad,
- especificar vistas de usuarios,
- manipular los datos de la base, lo cual incluye recuperar, insertar, eliminar y modificar los datos, y
- administrar los recursos del DBMS y de las bases de datos.

En ocasiones, los DBMS, sobre todo los de mayor complejidad, brindan un lenguaje adicional **no estándar** que permite efectuar ciertas labores de programación con las bases de datos. Este lenguaje puede tener algunas pocas instrucciones de programación, o puede ser de cierta complejidad. Normalmente este lenguaje se usa para programar rutinas, llamadas **procedimientos almacenados**, con el objetivo de tener accesos muy eficientes a la base de datos (por el hecho de que siempre son almacenados junto con ella).

Por otro lado, el DBMS puede permitir trabajar con SQL en forma interactiva, a través de terminal o de PC comunicada con la base de datos, o puede ejecutar instrucciones de SQL embebidas en un lenguaje de programación de propósito general. En este último caso, el lenguaje de programación se conoce como **lenguaje anfitrión (host)**.

Interfaces

Son las diferentes interfaces que se pueden encontrar en un ambiente de bases de datos. Algunas pueden ser proporcionadas por el DBMS; otras, programadas por medio de un lenguaje de 3a. o 4a. generación.

- **Basadas en menús**, presentan al usuario listas de opciones que lo van guiando a través del sistema. Los menús pueden presentarse por medio de caracteres, exclusivamente, o por medio de un ambiente de ventanas.
- **Gráficas**, típicamente despliegan al usuario un diagrama el cual éste va llenando con información. Normalmente están combinadas con menús basados en ventanas.
- **Basadas en formas**, presenta un **forma** al usuario para que éste edite datos; esto es, para que inserte, elimine o modifique datos.
- **De lenguaje natural**, las cuales aceptan solicitudes escritas en lenguaje natural (normalmente, inglés). No son comunes y se puede decir que aún están en desarrollo.
- **Para usuarios paramétricos**, que se emplean para presentar al usuario un conjunto reducido de operaciones, muchas de ellas asignadas a teclas del teclado. Están destinadas a usuarios que efectúan operaciones repetitivas, como los cajeros de bancos.

- **Para el DBA**, que normalmente contienen instrucciones privilegiadas para ser usadas exclusivamente por el DBA.

1.3.3 Componentes de un DBMS

Un DBMS es un sistema de software grande y complejo. A continuación se describen brevemente las partes importantes de un DBMS.

Catálogo del Sistema
Compilador DDL
Compilador DML
Administrador de datos
Procesador Run-time
Subsistema de control de concurrencia
Subsistema de seguridad
Subsistema de respaldo/ recuperación

- **Catálogo del sistema**, es una mini-base de datos almacenada en disco que puede estar separada físicamente o no de las bases de datos controladas por el DBMS. Contiene información como nombres de archivos, campos de datos, tipos de datos, detalles de almacenamiento, información de transformación entre esquemas y restricciones.
- **Compilador DDL (Data Definition Language)**, procesa las definiciones de los esquemas y almacena sus descripciones (meta-datos) en el catálogo.
- **Compilador DML (Data Manipulation Language)**, analiza y traduce las instrucciones de SQL que actualizan (insertan, borran y cambian) y consultan a la base de datos. Normalmente genera llamados al procesador run-time para que éste las ejecute.
- **Administrador de datos**, controla el acceso a la información almacenada en disco, ya sea que forme parte de una base de datos o del catálogo. Usa servicios del sistema operativo para intercambiar datos entre disco y memoria central.
- **Procesador run-time**, recibe operaciones de recuperación y de actualización y las ejecuta sobre la base de datos; para esto se apoya en el administrador de datos.
- **Subsistema de control de concurrencia**, controla el acceso simultáneo a una base de datos realizado por varios usuarios.
- **Subsistema de seguridad**, restringe el acceso a la base de datos, tanto a usuarios no autorizados como a partes de ésta que sólo determinado grupo de usuarios puede usar.

- **Subsistema de respaldo/recuperación**, permite crear respaldos de las bases de datos en otro dispositivo, ya sea disco o cinta magnética, para poder recuperar la información, posteriormente, en caso de una falla catastrófica.

1.3.4 Programas de aplicación

Una aplicación para una base de datos consiste de **formas, consultas, reportes, menús y módulos especializados**. El conjunto de programas que constituyen a la aplicación se elaboran utilizando un lenguaje de programación y a través de interfaces se comunican con el DBMS para tener acceso a la base de datos. El lenguaje de programación puede ser un:

- **Lenguaje de 3a. generación**: son lenguajes de tipo algorítmico, basados en procedimientos, en los cuales se insertan llamados a instrucciones del DBMS para realizar procesos con la base de datos. Ejemplo de estos lenguajes son: C, Pascal, Cobol, Basic, etc. Actualmente no es común usarlos, ya que es engorroso crear las aplicaciones y darles mantenimiento, salvo en aquellos casos en los que se requiere una gran eficiencia, en tiempos de respuesta sobre todo.
- **Lenguaje de 4a. generación asociado al DBMS**: normalmente son lenguajes de 4a. generación, de caracteres o ventanas, que son vendidos por la misma empresa propietaria del DBMS. Son bastante versátiles y la comunicación con el DBMS es casi directa. Tienen como desventaja el que son especializados para un DBMS particular. Ejemplos: Power Builder (Sybase), WebSphere (DB2).
- **Lenguaje de 4a. generación de terceros**: son lenguajes basados en ventanas, en su gran mayoría, producidos por empresas terceras y que no están ligados a DBMS's particulares. Son bastante poderosos y requieren siempre de una interfaz (ODBC) para comunicarse con algún DBMS particular. Como ejemplos están Visual Basic y Java, principalmente. La interfaz y las ventanas pueden ocasionar tiempos de respuesta un poco lentos.

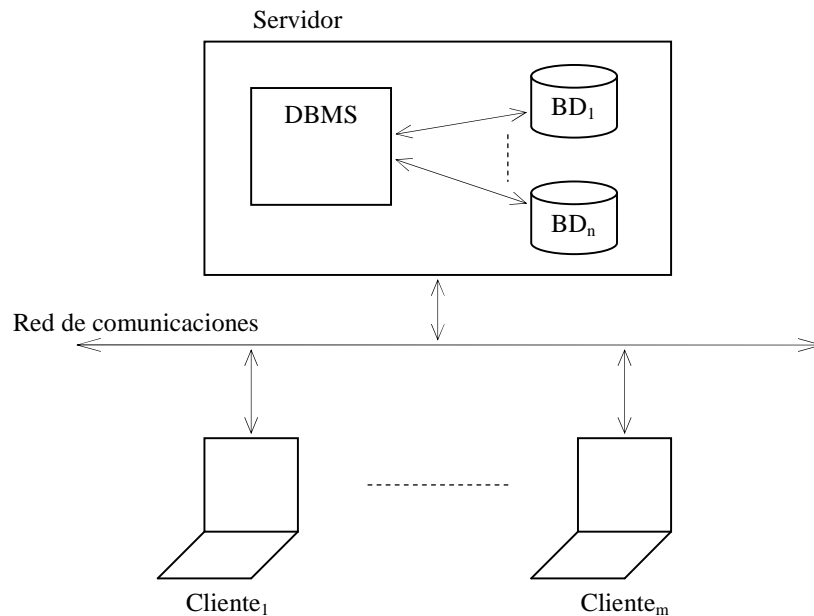
También se pueden usar lenguajes orientados a objetos (C++, SmallTalk, etc.) para crear programas de aplicación, aunque en general éstos se asocian más a bases de datos orientadas a objetos que a bases en otro paradigma.

1.4 Arquitecturas entre bases de datos y aplicaciones

Existen básicamente tres tipos de arquitectura para el procesamiento de bases de datos desde aplicaciones, relacionadas todas ellas con el sitio en el cual se encuentra cada parte del sistema. La más simple es la **arquitectura local** en la cual todos los componentes del sistema (base de datos, DBMS y aplicaciones) se encuentran en la misma computadora. Esta arquitectura normalmente es la que siguen los sistemas personales. Los otros dos tipos se describen a continuación.

1.4.1 Arquitectura cliente-servidor

Es una arquitectura en la cual el sistema de base de datos se divide en dos partes: el **servidor** (llamado también parte *dorsal* o *servicios de fondo*) y los **clientes** (llamados también partes *frontales* o *aplicaciones*).

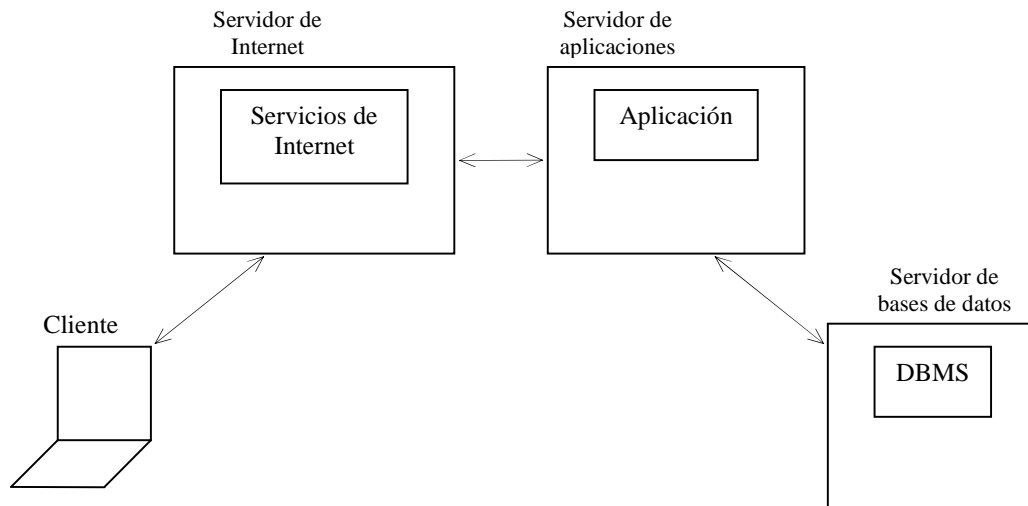


El servidor está formado precisamente por el DBMS llevando a cabo la administración y la manipulación de las bases de datos que controla. Los clientes son las diversas aplicaciones que trabajan con la información que está en las bases de datos, tanto aplicaciones escritas por usuarios como aplicaciones integradas (proporcionadas por el fabricante del DBMS o por terceros). Los clientes hacen peticiones al servidor (el DBMS), éste las recibe y las procesa, y envía las respuestas de vuelta a los clientes.

Es común usar computadoras personales o estaciones de trabajo sencillas, del lado de los clientes, y estaciones de trabajo poderosas o **mainframes**, del lado del servidor. Obviamente debe existir una red y software de comunicación para que clientes y servidor puedan intercambiar información y así poder implementar este tipo de arquitectura. Este tipo es el que se podría emplear para el segundo y tercer ejemplos, sobre todo el segundo, mencionados en la sección 1.2.

1.4.2 Arquitectura multicapa

También es conocida como arquitectura de **procesamiento distribuido**. En este caso el sistema se decompone en varias **capas**, cada una llevando a cabo un tipo de procesamiento específico.

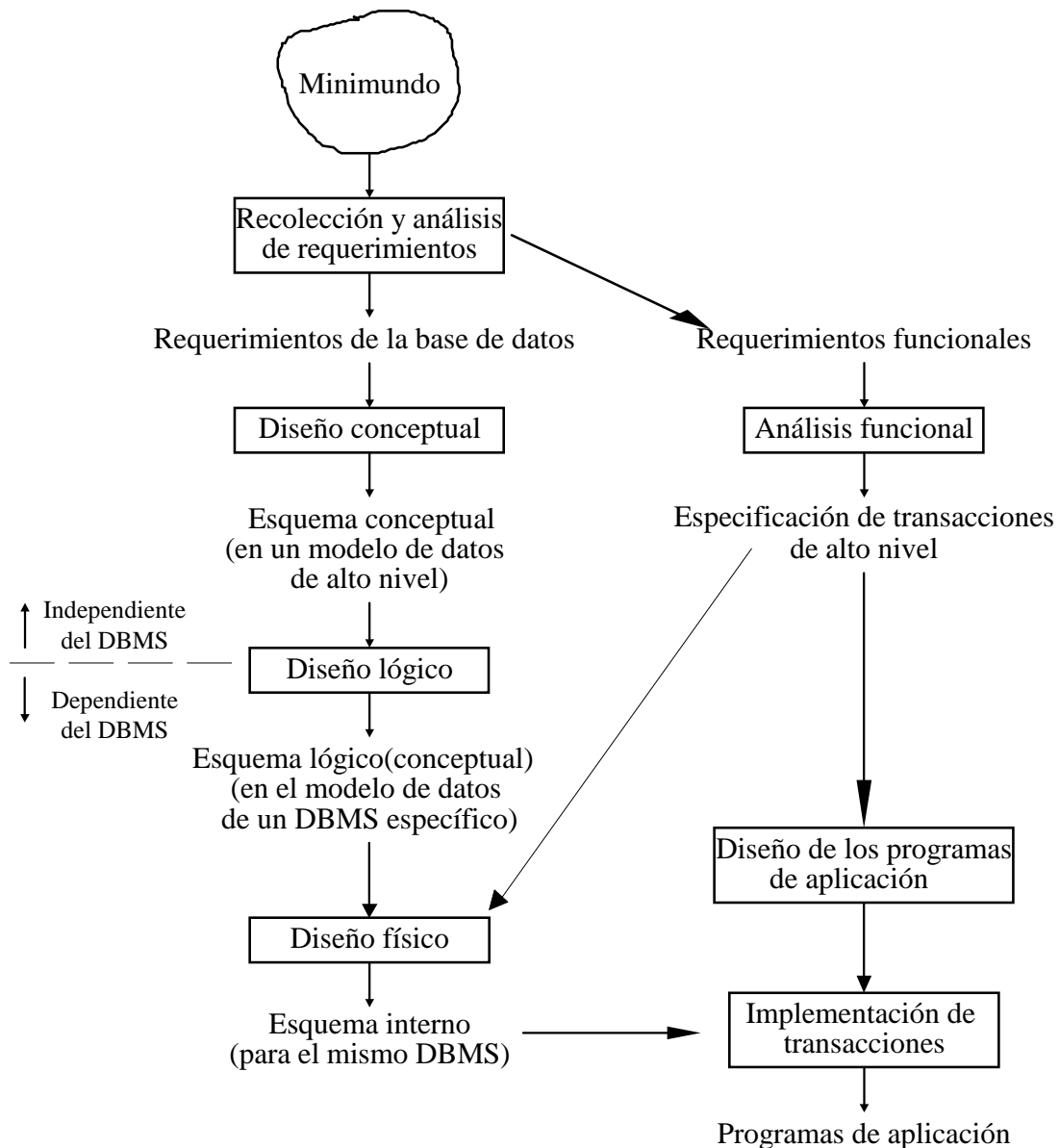


Por ejemplo, en la capa más cercana al usuario se podría tener un programa con interfaces gráficas poderosas para facilitar la actualización/consulta de la información de la base de datos a través de ventanas. En la siguiente capa se podría tener un **servidor de Internet** que llevara el control de todas las **páginas** que se mostrarían al usuario como interfaces de la aplicación. La tercera capa podría ser un **servidor de aplicaciones** el cual contendría las aplicaciones que implementan la lógica (reglas) del negocio. Y, finalmente, la última capa contendría al **servidor de bases de datos**.

Este tipo de arquitectura está muy asociado al uso de Internet, aunque no se emplea exclusivamente con él, y es el que se podría utilizar en el cuarto y, también, tercer ejemplos descritos en la sección 1.2.

1.5 Fases en el desarrollo de una base de datos

En esta sección se describen a grandes rasgos las fases usuales seguidas durante el desarrollo de una base de datos. En conjunción se comentan las actividades que normalmente se realizan para el desarrollo de programas de aplicación asociados.



El diagrama anterior muestra las etapas del desarrollo, enmarcadas con rectángulo, y los productos que se obtienen en cada etapa, sin enmarcar.

- **Recolección y análisis de requerimientos**, es la etapa en la que se hacen entrevistas con los usuarios potenciales de la base de datos con el fin de entender y documentar sus *requerimientos de datos*. El resultado es un conjunto de requerimientos de usuario. Estos deberán ser especificados de una manera tan detallada, completa y precisa como sea posible.

En paralelo se deben especificar los *requerimientos funcionales* de la aplicación. Estos consisten de operaciones definidas por el usuario (o **transacciones**) que serán aplicadas a la base de datos, e incluyen tanto recuperaciones como actualizaciones. Es común usar técnicas como los *diagramas de flujo de datos* para especificar estos requerimientos.

- **Diseño conceptual de la base de datos**, aquí se crea un **modelo conceptual** de la base de datos usando para ello un modelo de datos de alto nivel. El modelo conceptual es una descripción concisa de los requerimientos de datos de los usuarios y comprende descripciones de los *tipos de datos, vínculos y restricciones para los datos*; expresados por medio de los elementos que proporciona el modelo de datos de alto nivel. Dado que estos elementos no incluyen detalles de implementación, pueden ser usados para comunicarse con usuarios no especializados a fin de asegurar que todos sus requerimientos son cubiertos y que no hay contradicciones.

Después que el modelo conceptual ha sido elaborado, por medio de las operaciones básicas del modelo de datos se deben especificar las *transacciones de alto nivel* correspondientes a las operaciones del usuario identificadas durante el análisis funcional. Esto también sirve para confirmar que el modelo conceptual satisface todos los requerimientos funcionales.

- **Diseño lógico de la base de datos**, es la etapa en la cual se *implementa* la base de datos usando un DBMS comercial. Aquí se transforma el modelo conceptual de la base de datos, obtenido en la fase anterior, a una implementación usando el modelo de datos empleado por el DBMS: relacional, objetual, de red o jerárquico (los dos últimos ya de poco uso). El resultado es un **esquema lógico** (conceptual, según la arquitectura de los tres esquemas) de la base de datos.
- **Diseño físico de la base de datos**, durante esta fase son especificadas las *estructuras de almacenamiento internas y las organizaciones de archivos* para la base de datos. Normalmente el DBMS controla estos aspectos y el diseñador tiene poca injerencia en los mismos.

En paralelo con estas actividades, los programas de aplicación son diseñados e implementados como *transacciones de la base de datos* correspondientes a las especificaciones de las transacciones de alto nivel.