TEMA 2
INTRODUCCIÓN A LAS BASES DE DATOS

1.- INTRODUCCIÓN

Como se ha dicho anteriormente en un sistema de gestión de datos basado en ficheros la información está dispersa en varios ficheros de datos y cierto número de programas que los recuperan y agrupan. A medida que crecen las necesidades de información en la empresa, se van creando nuevos programas para acceder a nuevos datos e integrar esa información con los ya existentes. Los diferentes departamentos de la empresa también tienen sus propios programas que manejan los mismos datos o diferentes datos de las mismas personas o cosas, de modo que la información disponible en la empresa para un mismo cliente, por ejemplo, es diferente según el departamento que solicite la información a sus programas.

En un momento dado hay que crear programas que integren datos de los distintos departamentos y mantengan esa información actualizada. Esto exige un gran esfuerzo por comprender los modos en que se han almacenado físicamente los datos en cada programa y cuál ha de ser el nuevo sistema de almacenamiento. Este nuevo sistema de datos se solapará al antiguo si no se crean además los programas equivalentes a los existentes para los datos integrados, por lo que la realización de cambios es una tarea dificultosa, comparada con los trabajos a desarrollar en un sistema de bases de datos, ya que los datos dependen del fichero y lenguaje con que fueron creados. Además, hay que tener en cuenta el empleo de distintos lenguajes de programación por diferentes programadores para resolver los problemas empresariales en un determinado momento, de modo que tanto los datos como los programas se almacenan con formatos diversos, lo que origina problemas de aislamiento de datos.

La no integración de datos, en sistemas basados en ficheros, procedentes de diferentes departamentos o por empleo de lenguajes distintos origina que se produzcan repeticiones innecesarias (redundancia) en los datos, que generarán problemas de integridad y consistencia de datos, ya que no estará actualizada toda la información a la vez para el mismo cliente, por ejemplo, de modo que se maneje información sobre dos cuentas corrientes o direcciones, en lugar de la dirección o cuenta corriente actualizada. Asociado a los problemas de redundancia están los derivados del almacenamiento (volumen ocupado inútilmente y sus costes) y recuperación (es más lenta que si no hubiese redundancia).

La eliminación de la redundancia pasa, como ya se ha dicho, por la integración de los datos y programas empresariales, en un único sistema basado en ficheros de datos.

Problemas en la gestión de datos con sistemas gestores de ficheros

Los sistemas de gestión de datos basados en ficheros tienen dos tipos de problemas, el primero respecto de los archivos y el segundo tipo de problemas relativo a los datos.

Los problemas respecto de los ficheros se deben a la necesidad de controlar la integridad semántica, el control de autorizaciones y la concurrencia de accesos o acceso simultáneo de varios usuarios al mismo fichero de datos.

La *integridad semántica* es un conjunto de restricciones, también llamadas reglas de validación o limitantes de consistencia, que permiten o no almacenar determinados valores en un objeto de la base de datos. Por ejemplo: el campo CIF permite almacenar 8 dígitos y un carácter. Esta letra o carácter se obtienen mediante un algoritmo en el que intervienen los dígitos anteriores. El algoritmo que calcula si una determinada letra cumple la condición de ser la correcta para que un DNI sea además un CIF es una regla de validación.

Los diferentes programas que permiten la entrada de datos pueden tener diferentes exigencias en cuanto a integridad semántica. Si cada fichero que permite la adición o modificación de datos posee sus propios limitantes de consistencia es difícil mantener o crear programas que tengan en cuenta a la vez todas las reglas de validación, por lo que se produce información inconsistente para todo el sistema aunque cumplan los datos introducidos las reglas de validación creadas para un determinado subsistema.

El *control de autorizaciones* trata de evitar que se produzcan accesos indebidos a los datos, para lo que a cada usuario se le otorga un identificador y una clave. El identificador permite discriminar a los usuarios reales del sistema de aquellos que intentan acceder de forma no autorizada. La clave autentifica al usuario, de modo que aunque se posea información acerca de los identificadores de acceso, todavía se necesita que el usuario sea autentificado por el sistema. Hasta aquí puede que no haya demasiados problemas en la implementación de un control de autorizaciones, pero como los elementos del sistema están distribuidos sin organizar a lo largo del sistema, no se puede crear un método eficaz que controle el acceso a cada elemento del sistema, de modo que un usuario puede acceder a determinados programas o datos sobre los que no debería tener permisos de acceso, ya que al modificar el programa para que tenga o no esos permisos, es difícil prever y crear los controles de seguridad adecuados, en cuyo caso puede llegar a permitirse, de forma no deseada, que usuarios no autorizados accedan a otros datos a los que no tiene necesidad de acceder. Otro problema añadido se da cuando un usuario ha otorgado permisos a otro sobre aquellos objetos sobre los que puede otorgarlos y más tarde se le retiran, de modo que no quede claro si se han eliminado los permisos a aquellas terceras personas a las que se les otorgaron.

El tercer problema respecto de los ficheros es la gestión del *control de concurrencia* o acceso simultáneo de varios usuarios a los mismos datos, ya que cuando varios usuarios acceden a la vez al mismo fichero de datos para modificar la información que contiene y no hay un programa que controle el orden de acceso al fichero de datos por los usuarios, no hay control de concurrencia, no habrá seguridad acerca de cuál de todas las informaciones modificadas será guardada y en qué orden se han producido esas modificaciones, de forma que la información obtenida será inconsistente respecto a las operaciones realizadas. La existencia de un control de

concurrencia permite el acceso simultáneo de lectura de los datos y accesos sucesivos individualizados, en cola, de modo que hasta que no termine un usuario de realizar modificaciones los demás no podrán realizar otras nuevas.

Los problemas respecto de los datos se deben a su estructura física, a su modo de estar almacenados en diferentes archivos. Destacan los siguientes problemas:

- *Redundancia* o repetición innecesaria de información en varios ficheros.
- *Inconsistencia* o información redundante en la que los datos de los distintos ficheros no concuerda entre sí.
- *Aislamiento* o fragmentación de la información. Se produce cuando los datos referentes a un objeto se almacenan en distintos ficheros, siendo difícil obtener a la vez toda la información relativa al mismo objeto.

2.- INTRODUCCIÓN HISTÓRICA

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

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

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

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

En 1970 Codd, de los laboratorios de investigación de IBM, escribió un artículo presentando el *modelo relacional*. En este artículo, presentaba también los inconvenientes de los sistemas previos, el jerárquico y el de red. Entonces, se comenzaron a desarrollar muchos sistemas relacionales, apareciendo los primeros a finales de los setenta y principios de los ochenta. Uno de los primeros es System R, de IBM, que se desarrolló para probar la funcionalidad del modelo relacional, proporcionando una implementación de sus estructuras de datos y sus operaciones. Esto condujo a dos grandes desarrollos:

- El desarrollo de un lenguaje de consultas estructurado denominado SQL, que se ha convertido en el lenguaje estándar de los sistemas relacionales.
- La producción de varios SGBD relacionales durante los años ochenta, como DB2 y SLQ/DS de IBM, y ORACLE de ORACLE Corporation.

Hoy en día, existen cientos de SGBD relacionales, tanto para microordenadores como para sistemas multiusuario, aunque muchos no son completamente fieles al modelo relacional. Otros sistemas relacionales multiusuario son INGRES de Computer Associates, Informix de Informix Software Inc. y Sybase de Sybase Inc. Ejemplos de sistemas relacionales de microordenadores son Paradox y dBase IV de Borland, Access de Microsoft, FoxPro y R:base de Microrim.

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

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

Un Sistema de Gestión de bases de datos (SGBD; en inglés, database management system: DBMS) es un conjunto de programas que permite a los usuarios crear y mantener una base de datos. Por tanto, el SGBD es un sistema de software de propósito general que facilita el proceso de definir, construir y manipular bases de datos para diversas aplicaciones. Para definir una base de datos hay que especificar los tipos de datos, las estructuras y las restricciones de los datos que se almacenarán en ella. Construir una base de datos es el proceso de guardar los datos mismos en algún medio de almacenamiento controlado por el SGBD. En la manipulación de una base de

datos intervienen funciones como consultar la base de datos para obtener datos específicos, actualizar la base de datos para reflejar cambios en el Universo del Discurso y generar informes a partir de los datos.

El diseño de bases de datos reporta fundamentalmente dos ventajas. La primera es la independencia lógica de datos, lo que significa que los cambios realizados en un objeto de la base de datos no obligan a modificar otros elementos de la base de datos. La segunda es la independencia física de los datos, es decir, la base de datos no depende del tipo de dispositivo de almacenamiento en que se guarde. Como consecuencia de las anteriores ventajas, aparecen dos nuevas ventajas sobreañadidas relativas a la disminución de costes de almacenamiento y mantenimiento de la base de datos.

3.- CARACTERÍSTICAS DESEABLES DE LAS BASES DE DATOS

El diseño de bases de datos comerciales hace necesaria la aparición de determinados elementos o características que ayuden a resolver las necesidades de los administradores y usuarios de las bases de datos:

1. Elementos que facilitan el uso de un determinado Sistema Gestor de Bases de Datos:

a) Independencia física de los datos: al cambiar los objetos de una base de datos de un equipo a otro (por ej. de un portátil a otro ordenador de una red) no será necesario realizar cambios en las rutinas o procedimientos de acceso a esos datos, independientemente del software o hardware del equipo en el que se vaya a trabajar con la B.D.

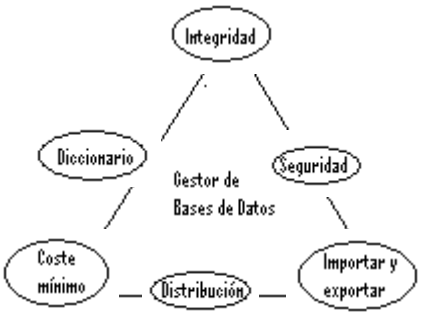
b) Independencia lógica de los datos: las modificaciones que se realizan en los objetos de la base de datos no exigen cambiar sus descriptores o la estructura general de la base de datos.

c) Mínima redundancia: consiste en evitar repeticiones innecesarias de datos en las bases de datos. Para ello se llevará a cabo un proceso de normalización que nos permitirá crear tablas con el mínimo número de campos posible, que estarán relacionadas entre sí y que nos permitirán recuperar toda la información posible evitando la redundancia.

d) Interfaz de alto nivel que permite a los programadores definir y crear sus rutinas, vistas o consultas, formularios o informes sin necesidad de conocer la organización interna y externa de la base de datos.

e) Lenguaje de usuario final: permite la generación automática de objetos de la base de datos y consulta de los mismos por los usuarios sin emplear lenguajes específicos de programación.

f) Asistentes que faciliten o simplifiquen tareas: instalación, configuración, creación de bases de datos, copias de seguridad, importación y exportación de datos…



Características de las bases de datos

2. Elementos de integridad de datos: es la propiedad de las bases de datos por la que:

a) Se puede recuperar la información perdida o dañada cuando se produzcan fallos en el sistema. Se trata de tener implementado un control de transacciones que se encargue de almacenar las operaciones sobre la base de datos en el mismo orden en el que se han ejecutado, de forma que las modificaciones realizadas puedan deshacerse en caso de que surja algún problema o error. Los sistemas gestores de bases de datos comerciales tienen implementado este nivel de integridad.

b) Los datos introducidos y mantenidos en la base de datos deben cumplir las reglas establecidas en la fase de análisis (integridad semántica)

Existe una herramienta denominada *gestor de consistencia de la base de datos* que se encarga de reparar los errores e inconsistencias que encuentra durante la verificación de la base de datos.

3. Elementos de seguridad:

a) Creación de copias de seguridad y restauración de datos de forma semiautomática o mediante asistentes.

b) Seguridad en cuanto a los accesos. Hay cuatro niveles:

- Impedir el acceso físico a los dispositivos de almacenamiento: evitar el robo de equipos o unidades de disco e impedir el acceso al sistema gestor de bases de datos vía sistema operativo.
- Impedir el acceso de usuarios no autorizados mediante el empleo de controles de acceso (identificador y contraseña). Y limitar el acceso de algunos usuarios a ciertos sectores del esquema del sistema (lectura, eliminación, etc…)
- Impedir las modificaciones y la recuperación de datos a quien no tenga permisos (datos codificados).
- Control de concurrencia para evitar errores de integridad debidos a accesos simultáneos de varios usuarios sobre los mismos datos (bases de datos distribuidas).

4. Utilidades para importar, exportar datos y empleo de técnicas de Data warehousing: importación, exportación y transformación de datos (entre diferentes sistemas de bases de datos).

a) Migración: posibilidad que tienen las bases de datos de recuperar datos almacenados en otras aplicaciones.

b) Compatibilidad de datos: los datos pueden ser copiados a y desde otras aplicaciones sin pérdida.

Algunos objetos, protocolos o servicios implicados en la importación, exportación y transformación de datos son:

- *Dataware:* almacén integrado de información procedente de distintas fuentes de datos que sirve de base para herramientas de soporte a la decisión y análisis de datos.
- *Servicios de transformación de datos (DTS):* permiten importar, exportar y transformar datos entre las bases de datos y los formatos OLE, BD, ODBC y archivos de texto. DTS permite generar data warehouses y data marts.
- *ODBC (Open DataBase Conectivity):* es un protocolo que permite compartir información entre bases de datos que están en distintos ordenadores.
- *OLE DB:* es un protocolo que además de permitir el acceso a fuentes de datos relacionales, posee un enlace a ODBC y permite el acceso a fuentes no relacionales de información (correo electrónico, sistemas de archivos, etc…)

- *Internet y correo electrónico:* las bases de datos actuales han de contar con asistentes que permitan la creación de páginas HTML a partir de sus datos. También han de soportar el empleo de tecnología ASP (Active Server Pages) que permita la actualización de la base de datos desde páginas HTML dinámicas (compras, ventas, facturación, stock→ comercio electrónico)

5. Unidades de Distribución de datos:

a) Replicación: es una propiedad de las bases de datos distribuidas por la que algunos datos están duplicados en distintos sitios de la red para así disminuir el tráfico de la red a la hora de realizar consultas. En algunas bases de datos distribuidas existe una base de datos principal y casa sitio posee una copia de los elementos que utiliza (aumenta la redundancia).

b) Sincronización: actualización periódica de las copias disponibles de la base de datos en cada sitio (si cambia algún dato que esté repetido en varios sitios hay que modificar su nuevo valor en todos estos sitios).

6. Metabase o Diccionario de datos: almacena en un esquema las definiciones y propiedades de los objetos que configuran la base de datos y los sinónimos de dichos elementos.

7. Coste mínimo: para conseguirlo hay que optimizar el coste de almacenamiento de los datos y también hay que contar con la posibilidad de usar varios procesadores en paralelo o con agrupaciones de disco (clustering) para un acceso más rápido a los datos.

4.- ELEMENTOS DE LAS BASES DE DATOS

De las características que se espera tenga una base de datos, citadas en el apartado anterior, parece lógico concluir que una base de datos está formada por cuatro elementos:

- Los datos, que han de estar integrados en el sistema de bases de datos correspondiente y con las características definidas anteriormente.
- El equipo físico sobre el que se instala el producto de gestión de bases de datos y almacena los datos y otros objetos de la base de datos y los equipos conectados en red al equipo servidor de datos.
- Software o producto de gestión de datos instalado, o lo que es lo mismo, el entorno de la base de datos.
- Usuarios de la base de datos:
  - Administrador, que es el encargado del diseño y modificación de la estructura de la base de datos.
  - Programadores, que diseñan los objetos de la base de datos para la correcta utilización de la base de datos, en lo que a almacenamiento y recuperación de información se refiere.
  - Usuarios finales, que son los encargados de utilizar los objetos de la base de datos puestos a su disposición, para facilitar la consecución de los objetivos empresariales propuestos.

5.- VENTAJAS E INCONVENIENTES DE LAS BASES DE DATOS

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

1. Ventajas por la integración de datos:

- *Control sobre la redundancia de datos.* Los sistemas de ficheros almacenan varias copias de los mismos datos en ficheros distintos. Esto hace que se desperdicie espacio de almacenamiento, además de provocar la falta de consistencia de datos. En los sistemas de bases de datos todos estos ficheros están integrados, por lo que no se almacenan varias copias de los mismos datos. Sin embargo, en una base de datos no se puede eliminar la redundancia completamente, ya que en ocasiones es necesaria para modelar las relaciones entre los datos, o bien es necesaria para mejorar las prestaciones.
- *Consistencia de datos.* Eliminando o controlando las redundancias de datos se reduce en gran medida el riesgo de que haya inconsistencias. Si un dato está almacenado una sola vez, cualquier actualización se debe realizar sólo una vez, y está disponible para todos los usuarios inmediatamente. Si un dato está duplicado y el sistema conoce esta redundancia, el propio sistema puede encargarse de garantizar que todas las copias se mantienen consistentes. Desgraciadamente, no todos los SGBD de hoy en día se encargan de mantener automáticamente la consistencia.
- *Más información sobre la misma cantidad de datos.* Al estar todos los datos integrados, se puede extraer información adicional sobre los mismos.
- *Compartición de datos.* En los sistemas de ficheros, los ficheros pertenecen a las personas o a los departamentos que los utilizan. Pero en los sistemas de bases de datos, la base de datos pertenece a la empresa y puede ser compartida por todos los usuarios que estén autorizados. Además, las nuevas aplicaciones que se vayan creando pueden utilizar los datos de la base de datos existente.
- *Mantenimiento de estándares.* Gracias a la integración es más fácil respetar los estándares necesarios, tanto los establecidos a nivel de la empresa como los nacionales e internacionales. Estos estándares pueden establecerse sobre el formato de los datos para facilitar su intercambio, pueden ser estándares de documentación, procedimientos de actualización y también reglas de acceso.

2. Ventajas por la existencia del SGBD:

- *Mejora en la integridad de datos.* La integridad de la base de datos se refiere a la validez y la consistencia de los datos almacenados. Normalmente, la integridad se expresa mediante restricciones o reglas que no se pueden violar. Estas restricciones se pueden aplicar tanto a los datos, como a sus relaciones, y es el SGBD quien se debe encargar de mantenerlas.
- *Mejora en la seguridad.* La seguridad de la base de datos es la protección de la base de datos frente a usuarios no autorizados. Sin unas buenas medidas de seguridad, la integración de datos en los sistemas de bases de datos hace que éstos sean más vulnerables que en los sistemas de ficheros. Sin embargo, los SGBD permiten mantener la seguridad mediante el establecimiento de claves para identificar al personal autorizado a utilizar la base de datos. Las autorizaciones se pueden realizar a nivel de operaciones, de modo que un usuario puede estar autorizado a consultar ciertos datos pero no a actualizarlos, por ejemplo.
- *Mejora en la accesibilidad a los datos.* Muchos SGBD proporcionan lenguajes de consultas o generadores de informes que permiten al usuario hacer cualquier tipo de consulta sobre los datos, sin que sea necesario que un programador escriba una aplicación que realice tal tarea.
- *Mejora en la productividad.* El SGBD proporciona muchas de las funciones estándar que el programador necesita escribir en un sistema de ficheros. A nivel básico, el SGBD proporciona todas las rutinas de manejo de ficheros típicas de los programas de aplicación. El hecho de disponer de estas funciones permite al programador centrarse mejor en la función

específica requerida por los usuarios, sin tener que preocuparse de los detalles de implementación de bajo nivel.

- *Mejora en el mantenimiento gracias a la independencia de datos.* En los sistemas de ficheros, las descripciones de los datos se encuentran inmersas en los programas de aplicación que los manejan. Esto hace que los programas sean dependientes de los datos, de modo que un cambio en su estructura, o un cambio en el modo en que se almacena en disco, requiere cambios importantes en los programas cuyos datos se ven afectados. Sin embargo, los SGBD separan las descripciones de los datos de las aplicaciones. Esto es lo que se conoce como independencia de datos, gracias a la cual se simplifica el mantenimiento de las aplicaciones que acceden a la base de datos.
- *Aumento de la concurrencia.* En algunos sistemas de ficheros, si hay varios usuarios que pueden acceder simultáneamente a un mismo fichero, es posible que el acceso interfiera entre ellos de modo que se pierda información o, incluso, que se pierda la integridad. La mayoría de los SGBD gestionan el acceso concurrente a la base de datos y garantizan que no ocurran problemas de este tipo.
- *Mejora en los servicios de copias de seguridad y de recuperación ante fallos.* Muchos sistemas de ficheros dejan que sea el usuario quien proporcione las medidas necesarias para proteger los datos ante fallos en el sistema o en las aplicaciones. Los usuarios tienen que hacer copias de seguridad cada día, y si se produce algún fallo, utilizar estas copias para restaurarlos. En este caso, todo el trabajo realizado sobre los datos desde que se hizo la última copia de seguridad se pierde y se tiene que volver a realizar. Sin embargo, los SGBD actuales funcionan de modo que se minimiza la cantidad de trabajo perdido cuando se produce un fallo.

Inconvenientes de los sistemas de bases de datos

- *Complejidad.* Los SGBD son conjuntos de programas muy complejos con una gran funcionalidad. Es preciso comprender muy bien esta funcionalidad para poder sacar un buen partido de ellos.
- *Tamaño.* Los SGBD son programas complejos y muy extensos que requieren una gran cantidad de espacio en disco y de memoria para trabajar de forma eficiente.
- *Coste económico del SGBD.* El coste de un SGBD varía dependiendo del entorno y de la funcionalidad que ofrece. Por ejemplo, un SGBD para un ordenador personal puede costar 500 euros, mientras que un SGBD para un sistema multiusuario que dé servicio a cientos de usuarios puede costar entre 10.000 y 100.000 euros. Además, hay que pagar una cuota anual de mantenimiento que suele ser un porcentaje del precio del SGBD.
- *Coste del equipamiento adicional.* Tanto el SGBD, como la propia base de datos, pueden hacer que sea necesario adquirir más espacio de almacenamiento. Además, para alcanzar las prestaciones deseadas, es posible que sea necesario adquirir una máquina más grande o una máquina que se dedique solamente al SGBD. Todo esto hará que la implantación de un sistema de bases de datos sea más cara.
- *Coste de la conversión.* En algunas ocasiones, el coste del SGBD y el coste del equipo informático que sea necesario adquirir para su buen funcionamiento, es insignificante comparado al coste de convertir la aplicación actual en un sistema de bases de datos. Este coste incluye el coste de enseñar a la plantilla a utilizar estos sistemas y, probablemente, el coste del personal especializado para ayudar a realizar la conversión y poner en marcha el sistema. Este coste es una de las razones principales por las que algunas empresas y organizaciones se resisten a cambiar su sistema actual de ficheros por un sistema de bases de datos.

- *Prestaciones.* Un sistema de ficheros está escrito para una aplicación específica, por lo que sus prestaciones suelen ser muy buenas. Sin embargo, los SGBD están escritos para ser más generales y ser útiles en muchas aplicaciones, lo que puede hacer que algunas de ellas no sean tan rápidas como antes.
- *Vulnerable a los fallos.* El hecho de que todo esté centralizado en el SGBD hace que el sistema sea más vulnerable ante los fallos que puedan producirse.
- No existe un único diseño de base de datos que permita dar solución a un problema empresarial, de modo que distintos analistas o administradores de bases de datos pueden obtener objetos diferentes para lograr el mismo resultado, por lo que puede haber problemas de redundancia, nombres y tipos de objetos a la hora de integrar bases de datos para obtener otras nuevas que funcionen de modo distribuido.

6.- NIVELES DE DESCRIPCIÓN DE DATOS

Una base de datos es un conjunto de datos que se refieren al mismo tipo de persona u objeto, y se muestran al usuario estructurados en filas y columnas. A cada estructura de filas y columnas se le denomina tabla. Los sistemas gestores de bases de datos que se pueden adquirir no sólo cumplen esta definición, sino que además presentan muy diversas utilidades, entre las que destacan las de creación de tablas, vistas, formularios e informes mediante interfaces gráficos.

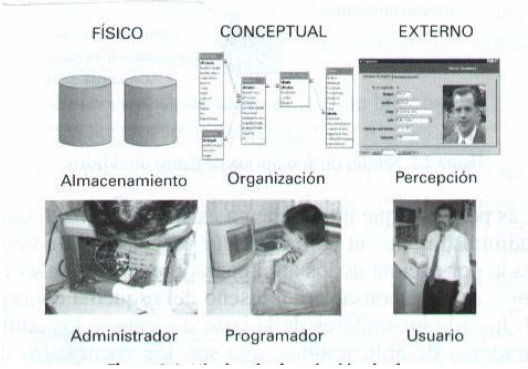
También contienen herramientas y lenguajes para la definición, descripción y manipulación de datos, para la creación de programas, funciones y procedimientos, así como elementos de gestión de usuarios: creación, definición y modificación de usuarios y permisos de acceso a datos o programas, que garanticen la seguridad de la base de datos.

Como la evolución o mejora de de las herramientas suministradas con los sistemas gestores de bases de datos es rápida (una nueva versión cada dos o tres años) hay que prever tamaños, capacidades, etc., de modo que las nuevas versiones funcionen sobre las máquinas actuales con un coste mínimo de actualización del hardware y/o software empleado, a no ser que se produzca un cambio tecnológico como el paso del DOS a WINDOWS.

Al intentar categorizar a las personas que acceden a la base de datos se pueden encontrar tres grupos diferenciados:

a) Aquellos que realizan tareas de administración.
b) Los programadores de aplicaciones y del sistema.
c) Los usuarios.

Para ocultar la complejidad de las bases de datos y lograr la independencia de los datos y programas, de acuerdo con el comité ANSI/X3/SPARC se definen en la arquitectura o diseño de una base de datos tres vistas o niveles de abstracción: físico, conceptual y externo, que se corresponden con los tres grupos de personas que acceden a las bases de datos.



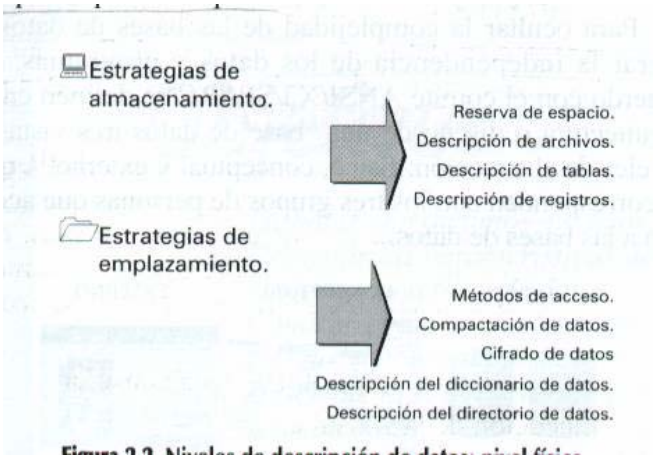
6.1. NIVEL FÍSICO

El nivel físico describe cómo se almacenan físicamente las estructuras de datos, no sólo se han de definir las tablas (almacenamiento lógico) sino los archivos físicos. La descripción del nivel físico se realiza mediante un esquema interno, que es un conjunto de definiciones y reglas que permite definir tablas y cómo se relacionan entre sí.

Desde el punto de vista de la organización de los datos en los dispositivos de almacenamiento se distinguen los siguientes aspectos:

a) Una estrategia de almacenamiento o asignación de espacios para el conjunto de datos.
b) Reserva del espacio necesario para almacenar los datos.
c) Descripción de los archivos de la base de datos: nombre, tipo de organización, unidad o volumen en el que se almacenan y método de acceso.
d) Descripción de las tablas de datos que contienen registros.
e) Definición de registros.
f) Longitud y tipo de dato de cada campo.
g) Una estrategia de emplazamiento para optimizar tiempos de respuesta y espacios de memoria secundaria.
h) Unos determinados métodos de acceso como las especificaciones de claves, índices o punteros y la descripción del diccionario y directorio de datos cuando el sistema no realice esa tarea automáticamente.

Además, se tienen en cuenta las técnicas de compresión de datos y de criptografiado o cifrado de datos. Esa a este nivel donde se describen los datos desde el punto de vista de la máquina que los soporta.



Las personas que intervienen a este nivel son:

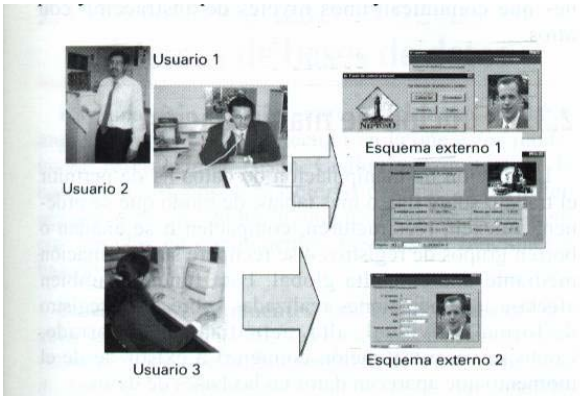
- El administrador empresarial, que está situado justo por encima de los administradores de bases de datos.
- Los administradores de la base de datos y los administradores de aplicaciones.

6.2. NIVEL CONCEPTUAL

El nivel conceptual o nivel lógico global describe la organización de los datos en la base de datos y las relaciones existentes entre ellos. La descripción de este nivel se realiza mediante un esquema conceptual, que permite definir las reglas de validación, las relaciones entre tablas y los campos y sus propiedades. Son los programadores de la base de datos los responsables de crear y guardar la información prevista, definida anteriormente por los analistas. Además, son responsables de crear adecuadamente las estructuras lógicas (tablas) de la base de datos. Es en este nivel donde se generan las órdenes de control de integridad y las restricciones o valores permitidos para campo.

6.3. NIVEL EXTERNO

El nivel externo, describe la base de datos tal y como es percibida por los usuarios. Para éstos, las tablas y sus registros existen físicamente. A este nivel, la seguridad de la base de datos hace que a cada usuario se le muestre la parte de la base de datos a la que tiene acceso. Cada usuario verá una base de datos (esquema externo) diferente según sea el nivel de acceso que se le haya concedido. Los objetos a los que puede acceder un usuario forman su nivel externo: tablas, vistas, formularios, informes, etc… Es decir, el nivel externo es la percepción de la base de datos por el usuario, de modo que hay tantos niveles externos distintos como usuarios haya en la base de datos.



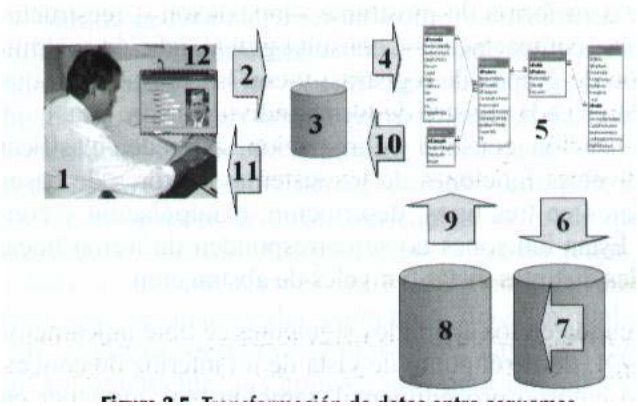
El nivel externo, desde el punto de vista funcional de los usuarios, consiste en un conjunto de herramientas de desarrollo visual de consultas, formularios e informes y un lenguaje de consulta que servirá para la creación de programas que permitan acceder a los datos desde la base de datos.

Las fases de transformación de datos de un esquema a otro son:

1) Un usuario solicita ver determinados datos de una tabla creando una consulta.
2) Verificación del esquema externo para ese usuario.
3) Aceptación del esquema externo.
4) Transformación de la solicitud al nivel conceptual.
5) Verificación del esquema conceptual.
6) Aceptación del esquema conceptual. Se manejan las tablas que contienen los datos solicitados por la consulta.

7) Transformación de la solicitud al nivel interno. Se busca la información sobre el almacenamiento físico de los datos, sobre los registros seleccionados en cada tabla.
8) Selección de la tabla objeto de la consulta.
9) Ejecución de la consulta.
10) Transformación del nivel interno al conceptual.
11) Transformación del nivel conceptual al externo.
12) Se muestran los registros correspondientes al usuario.

Aunque en la explicación anterior no se ha tenido en cuenta el diccionario/directorio de datos, hay que decir que será el encargado de las transformaciones.



7.- CLASIFICACIÓN DE LOS SISTEMAS GESTORES DE BASES DE DATOS

Una característica fundamental del enfoque de bases de datos es que proporciona cierto nivel de abstracción de los datos al ocultar detalles de almacenamiento que la mayoría de los usuarios no necesitan conocer. Los modelos de datos son el principal instrumento para ofrecer dicha abstracción.

Un **modelo de datos** es un conjunto de conceptos que pueden servir para describir la estructura de una base de datos. Con el concepto de estructura nos referimos a los tipos de datos, los vínculos y las restricciones que deben cumplirse para esos datos.

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

a) El *modelo relacional* se basa en el concepto matemático denominado ``relación", que gráficamente se puede representar como una tabla. En el modelo relacional, los datos y las relaciones existentes entre los datos se representan mediante estas relaciones matemáticas, cada una con un nombre que es único y con un conjunto de columnas. En el modelo relacional la base de datos es percibida por el usuario como un conjunto de tablas. Esta percepción es sólo a nivel lógico (en los niveles externo y conceptual de la arquitectura de

tres niveles), ya que a nivel físico puede estar implementada mediante distintas estructuras de almacenamiento.

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

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

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

Un segundo criterio para clasificar los SGBD es el **número de usuarios a los que da servicio el sistema**:

a) Los sistemas *monousuario* sólo atienden a un usuario a la vez, y su principal uso se da en los ordenadores personales.
b) Los sistemas *multiusuario*, entre los que se encuentran la mayor parte de los SGBD, atienden a varios usuarios al mismo tiempo.

Un tercer criterio es el **número de sitios en los que está distribuida la base de datos**:

a) Casi todos los SGBD son *centralizados*: sus datos se almacenan en un solo computador. Los SGBD centralizados pueden atender a varios usuarios, pero el SGBD y la base de datos en sí residen por completo en una sola máquina.
b) En los SGBD *distribuidos* la base de datos real y el propio software del SGBD pueden estar distribuidos en varios sitios conectados por una red.
c) Los SGBD *distribuidos homogéneos* utilizan el mismo SGBD en múltiples sitios. Una tendencia reciente consiste en crear software para tener acceso a varias bases de datos autónomas preexistentes almacenadas en SGBD *distribuidos heterogéneos*. Esto da lugar a los SGBD *federados* o *sistemas multibase de datos* en los que los SGBD participantes tienen

cierto grado de autonomía local. Muchos SGBD distribuidos emplean una arquitectura cliente-servidor.

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

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

8.- FUNCIONES DE LOS SISTEMAS GESTORES DE BASES DE DATOS

Las funciones esenciales de un SGBD son la de descripción, manipulación y utilización.

- **Función de Descripción o definición**. Esta función debe permitir al administrador de la base especificar los elementos de datos que la integran, su estructura y las relaciones que existen entre ellos, las reglas de integridad semántica, los controles a efectuar antes de autorizar el acceso a la base, etc., así como también las características de tipo físico y las vistas lógicas de los usuarios. Esta función, es llevada a cabo por el Lenguaje de Descripción o definición de Datos (LDD), propio de cada SGBD.
- **Función de Manipulación**. La función de manipulación permite a los usuarios de la base buscar, añadir, suprimir o modificar los datos de la misma, siempre de acuerdo con las especificaciones y normas de seguridad dictadas por el administrador. La función de manipulación se llevará a cabo por medio de un Lenguaje de Manipulación de Datos (LMD), que facilita los instrumentos necesarios para la realización de estas tareas.
- **Función de utilización**. Se trata de una función que reúne las interfaces que necesitan los diferentes usuarios para comunicarse con la base y proporciona un conjunto de procedimientos para el administrador, entre los que se encuentra el lenguaje de Control de Datos (LCD). En la mayoría de los SGBD existen funciones de servicio, como cambiar la capacidad de los ficheros, obtener estadísticas de utilización, cargar archivos, etc.; principalmente, las relacionadas con la seguridad física –copias de seguridad, rearranque en caso de caída del sistema, etc. – y de protección frente a accesos no autorizados, las cuales se encuentran comprendidas en la función de utilización.

9. COMPONENTES DE UN SISTEMA GESTOR DE BASES DE DATOS.

Los SGBD son paquetes de software muy complejos y sofisticados que deben proporcionar los servicios comentados en la sección anterior. No se puede generalizar sobre los elementos que componen un SGBD ya que varían mucho unos de otros. Sin embargo, es muy útil conocer sus componentes y cómo se relacionan cuando se trata de comprender lo que es un sistema de bases de datos.

Un SGBD tiene varios módulos, cada uno de los cuales realiza una función específica. El sistema operativo proporciona servicios básicos al SGBD, que es construido sobre él.

- El *procesador de consultas* es el componente principal de un SGBD. Transforma las consultas en un conjunto de instrucciones de bajo nivel que se dirigen al gestor de la base de datos.
- El *gestor de la base de datos* es el interface con los programas de aplicación y las consultas de los usuarios. El gestor de la base de datos acepta consultas y examina los esquemas externo y conceptual para determinar qué registros se requieren para satisfacer la petición. Entonces el gestor de la base de datos realiza una llamada al gestor de ficheros para ejecutar la petición.
- El *gestor de ficheros* maneja los ficheros en disco en donde se almacena la base de datos. Este gestor establece y mantiene la lista de estructuras e índices definidos en el esquema interno. Si se utilizan ficheros dispersos, llama a la función de dispersión para generar la dirección de los registros. Pero el gestor de ficheros no realiza directamente la entrada y salida de datos. Lo que hace es pasar la petición a los métodos de acceso del sistema operativo que se encargan de leer o escribir los datos en el buffer del sistema.
- El *preprocesador del LMD* convierte las sentencias del LMD embebidas en los programas de aplicación, en llamadas a funciones estándar escritas en el lenguaje anfitrión. El preprocesador del LMD debe trabajar con el procesador de consultas para generar el código apropiado.
- El *compilador del LDD* convierte las sentencias del LDD en un conjunto de tablas que contienen metadatos. Estas tablas se almacenan en el diccionario de datos.
- El *gestor del diccionario* controla los accesos al diccionario de datos y se encarga de mantenerlo. La mayoría de los componentes del SGBD acceden al diccionario de datos.

Los principales componentes del gestor de la base de datos son los siguientes:

- *Control de autorización*. Este módulo comprueba que el usuario tiene los permisos necesarios para llevar a cabo la operación que solicita.
- *Procesador de comandos.* Una vez que el sistema ha comprobado los permisos del usuario, se pasa el control al procesador de comandos.
- *Control de la integridad.* Cuando una operación cambia los datos de la base de datos, este módulo debe comprobar que la operación a realizar satisface todas las restricciones de integridad necesarias.
- *Optimizador de consultas.* Este módulo determina la estrategia óptima para la ejecución de las consultas.
- *Gestor de transacciones.* Este módulo realiza el procesamiento de las transacciones.
- *Planificador (scheduler).* Este módulo es el responsable de asegurar que las operaciones que se realizan concurrentemente sobre la base de datos tienen lugar sin conflictos.
- *Gestor de recuperación.* Este módulo garantiza que la base de datos permanece en un estado consistente en caso de que se produzca algún fallo.
- *Gestor de buffers.* Este módulo es el responsable de transferir los datos entre memoria principal y los dispositivos de almacenamiento secundario. A este módulo también se le denomina *gestor de datos*.

10.- LENGUAJES DE LOS SISTEMAS GESTORES DE BASES DE DATOS.

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

10.1. LENGUAJE DE DEFINICIÓN DE DATOS

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

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

10.2. LENGUAJE DE MANEJO DE DATOS

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

Hay dos tipos de LMD: los **procedurales** y los **no procedurales**.

a) Con un ***LMD procedural*** el usuario (normalmente será un programador) especifica qué datos se necesitan y cómo hay que obtenerlos. Esto quiere decir que el usuario debe especificar todas las operaciones de acceso a datos llamando a los procedimientos necesarios para obtener la información requerida. Estos lenguajes acceden a un registro, lo procesan y basándose en los resultados obtenidos, acceden a otro registro, que también deben procesar. Así se va accediendo a registros y se van procesando hasta que se obtienen los datos deseados. Las sentencias de un LMD procedural deben estar embebidas en un lenguaje de alto nivel, ya que se necesitan sus estructuras (bucles, condicionales, etc.) para obtener y procesar cada registro individual. A este lenguaje se le denomina *lenguaje anfitrión*. Las bases de datos jerárquicas y de red utilizan LMD procedurales.
b) Un ***LMD no procedural*** se puede utilizar de manera independiente para especificar operaciones complejas sobre la base de datos de forma concisa. En muchos SGBD se pueden introducir interactivamente instrucciones del LMD desde un terminal o bien embeberlas en un lenguaje de programación de alto nivel. Los LMD no procedurales permiten especificar los datos a obtener en una consulta o los datos que se deben actualizar, mediante una sola y sencilla sentencia. El usuario o programador especifica

qué datos quiere obtener sin decir cómo se debe acceder a ellos. El SGBD traduce las sentencias del LMD en uno o varios procedimientos que manipulan los conjuntos de registros necesarios. Esto libera al usuario de tener que conocer cuál es la estructura física de los datos y qué algoritmos se deben utilizar para acceder a ellos. A los LMD no procedurales también se les denomina *declarativos*. Las bases de datos relacionales utilizan LMD no procedurales, como **SQL** (Structured Query Language) o QBE (Query-By-Example). Los lenguajes no procedurales son más fáciles de aprender y de usar que los procedurales, y el usuario debe realizar menos trabajo, siendo el SGBD quien hace la mayor parte.

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

10.3. LENGUAJES DE CUARTA GENERACIÓN

No existe consenso sobre lo que es un ***lenguaje de cuarta generación* (4GL)**. Lo que en un lenguaje de tercera generación (3GL) como COBOL requiere cientos de líneas de código, tan solo necesita diez o veinte líneas en un 4GL. Comparado con un 3GL, que es procedural, un 4GL es un lenguaje no procedural: el usuario define qué se debe hacer, no cómo debe hacerse. Los 4GL se apoyan en unas herramientas de mucho más alto nivel denominadas herramientas de cuarta generación. El usuario no debe definir los pasos a seguir en un programa para realizar una determinada tarea, tan sólo debe definir una serie de parámetros que estas herramientas utilizarán para generar un programa de aplicación. Se dice que los 4GL pueden mejorar la productividad de los programadores en un factor de 10, aunque se limita el tipo de problemas que pueden resolver. Los 4GL abarcan:

- Lenguajes de presentación, como lenguajes de consultas y generadores de informes.
- Lenguajes especializados, como hojas de cálculo y lenguajes de bases de datos.
- Generadores de aplicaciones que definen, insertan, actualizan y obtienen datos de la base de datos.
- Lenguajes de muy alto nivel que se utilizan para generar el código de la aplicación.

Los lenguajes SQL y QBE son ejemplos de 4GL. Hay otros tipos de 4GL:

- Un *generador de formularios* es una herramienta interactiva que permite crear rápidamente formularios de pantalla para introducir o visualizar datos. Los generadores de formularios permiten que el usuario defina el aspecto de la pantalla, qué información se debe visualizar y en qué lugar de la pantalla debe visualizarse. Algunos generadores de formularios permiten la creación de atributos derivados utilizando operadores aritméticos y también permiten especificar controles para la validación de los datos de entrada.
- Un *generador de informes* es una herramienta para crear informes a partir de los datos almacenados en la base de datos. Se parece a un lenguaje de consultas en que permite al usuario hacer preguntas sobre la base de datos y obtener información de ella para un informe. Sin embargo, en el generador de informes se tiene un mayor control sobre el aspecto de la salida. Se puede dejar que el generador determine automáticamente el aspecto de la salida o se puede diseñar ésta para que tenga el aspecto que desee el usuario final.

- Un *generador de gráficos* es una herramienta para obtener datos de la base de datos y visualizarlos en un gráfico mostrando tendencias y relaciones entre datos. Normalmente se pueden diseñar distintos tipos de gráficos: barras, líneas, etc.
- Un *generador de aplicaciones* es una herramienta para crear programas que hagan de interface entre el usuario y la base de datos. El uso de un generador de aplicaciones puede reducir el tiempo que se necesita para diseñar un programa de aplicación. Los generadores de aplicaciones constan de procedimientos que realizan las funciones fundamentales que se utilizan en la mayoría de los programas. Estos procedimientos están escritos en un lenguaje de programación de alto nivel y forman una librería de funciones entre las que escoger. El usuario especifica qué debe hacer el programa y el generador de aplicaciones es quien determina cómo realizar la tarea.

11. ADMINISTRACIÓN DE DATOS Y DE LA BASE DE DATOS

El administrador de datos y el administrador de la base de datos son las personas o grupos de personas encargadas de gestionar y controlar todas las actividades que tienen que ver con los datos de la empresa y con la base de datos, respectivamente.

El **administrador de datos** es quien entiende los datos y las necesidades de la empresa con respecto a dichos datos. Su trabajo es decidir qué datos deben almacenarse en la base de datos y establecer políticas para mantener y gestionar los datos una vez hayan sido almacenados. Un ejemplo de tal política sería una que estableciera quién puede realizar qué operaciones sobre qué datos y en qué circunstancias.

La persona (o personas) que se encarga de implementar las decisiones del administrador de datos es el **administrador de la base de datos**. Su trabajo es crear la base de datos e implementar los controles necesarios para que se respeten las políticas establecidas por el administrador de datos. El administrador de la base de datos es el responsable de garantizar que el sistema obtenga las prestaciones deseadas, además de prestar otros servicios técnicos.

El administrador de datos juega un papel más importante que el administrador de la base de datos en las siguientes etapas del ciclo de vida: planificación de la base de datos, definición del sistema, recolección y análisis de los requisitos, diseño conceptual y diseño lógico de la base de datos. En el resto de las etapas es donde el administrador de la base de datos tiene el papel más importante: selección del SGBD, diseño de las aplicaciones, diseño físico, prototipado, implementación, conversión y carga de datos, prueba y mantenimiento.

12. USUARIOS DE LA BASE DE DATOS

Podemos definir a los usuarios como toda persona que tenga todo tipo de contacto con el sistema de base de datos desde que este se diseña, elabora, termina y se usa. Los usuarios que acceden una base de datos pueden clasificarse como:

1.-Programadores de aplicaciones: Los profesionales en computación que interactúan con el sistema por medio de llamadas en DML (Lenguaje de Manipulación de Datos), las cuales están incorporadas en un programa escrito en un lenguaje de programación (Por ejemplo, COBOL, PL/I, Pascal, C, etc.)

2.- Usuarios sofisticados: Los usuarios sofisticados interactúan con el sistema sin escribir programas. En cambio escriben sus preguntas en un lenguaje de consultas de base de datos.

3.- Usuarios especializados: Algunos usuarios sofisticados escriben aplicaciones de base de datos especializadas que no encajan en el marco tradicional de procesamiento de datos.

4.- Usuarios ingenuos: Los usuarios no sofisticados interactúan con el sistema invocando a uno de los programas de aplicación permanentes que se han escrito anteriormente en el sistema de base de datos, podemos mencionar al usuario ingenuo como el usuario final que utiliza el sistema de base de datos sin saber nada del diseño interno del mismo por ejemplo: un cajero.