Unidad 1

INTRODUCCIÓN A LAS BASES DE DATOS

1.- Sistema tradicional de ficheros: problemas

Los sistemas computacionales se utilizaron inicialmente en los negocios para funciones de contabilidad y, como eran funciones imprescindibles, el alto costo de los computadores era fácil de justificar.

A estos primeros sistemas se les llamó sistemas de procesamiento de datos y trataban de imitar los procedimientos manuales existentes.

Al principio, la mayoría de los archivos se almacenaban en cinta magnética, ya que el almacenamiento en disco era todavía caro, y se accedía a los datos de forma secuencial, lo que significa que cada registro puede leerse únicamente después de haber sido leídos todos los que le preceden. Estos archivos se procesaban por lotes, es decir, todos los registros de un archivo se procesaban al mismo tiempo.

Los archivos se empleaban en distintas aplicaciones. Un programa que realiza una tarea específica es un programa de aplicación y un conjunto de programas que trabajan en tareas relacionadas entre sí se llama sistema de aplicación.

Los archivos secuenciales servían para producir facturas e informes una o dos veces al mes pero para tareas rutinarias se necesitaba acceso directo a los datos (procesar directamente un registro dado). Los operadores debían introducir datos redundantes, lo que requería esfuerzo adicional y aumentaba la probabilidad de error.

Estos problemas se resolvieron parcialmente con la introducción de los archivos de acceso directo, particularmente los archivos secuenciales indexados (ISAM), que permitían la recuperación de registros aleatoriamente. Este tipo de archivos permiten utilizar uno o más campos para identificar un registro.

A finales de los sesenta se produjo la transición del procesamiento de los datos al procesamiento de la información. Se hace una distinción entre datos e información. Por datos se entienden hechos aislados, mientras que información son datos procesados.

Los archivos de acceso directo también tenían una serie de deficiencias:

Redundancia de datos. Como muchas aplicaciones utilizaban sus propios archivos, había algunos datos redundantes, lo que ocasionaba el aumento de introducción de datos y las probabilidades de inconsistencia entre diversas versiones de los mismos.

Pobre control de datos. El mismo elemento de los datos podía tener diversos nombres según el archivo en que se encontrara, lo cual creaba confusiones.

Capacidades inadecuadas de manipulación de datos. Los archivos secuenciales indexados permitían tener acceso a un registro particular pero no a un conjunto de registros interrelacionados.

Acoplamiento entre los programas y los ficheros. Existe una interdependencia muy fuerte entre los programas y los datos. Cualquier cambio en un archivo implica modificar todos los programas que acceden a él (si lo sabemos).

2.- Bases de datos

Una base de datos es una colección de elementos de datos interrelacionados que pueden procesarse por uno o más sistemas de aplicación. Un sistema de base de datos está formado por una base de datos, un sistema de gestión de bases de datos (SGBD), así como por el hardware y personal apropiado. Los sistemas de bases de datos superan estas limitaciones de los sistemas orientados a los archivos. Los datos se controlan por medio de un diccionario de datos/directorio, que está controlado por los administradores de la base de datos.

De este modo, conseguimos una serie de ventajas:

- Los datos se independizan de las aplicaciones La organización de los datos depende únicamente de las relaciones que haya entre ellos, no de los tratamientos que vayamos a realizar. Esto da mayor flexibilidad al sistema y la posibilidad de adaptarse a los cambios que provoque la evolución del mismo sin un coste excesivo.
- Coherencia de los resultados: Siempre que se realice la misma operación en las mismas condiciones de partida obtendremos los mismos resultados.
- Mejor disponibilidad. Las aplicaciones no son propietarias de los datos.
- Más información, al recogerse las relaciones entre los datos y restricciones a los mismos.
- Definiciones de datos almacenadas en la base de datos.
- Eficiencia
- Menor espacio de almacenamiento [1]

[1] Esto hay que matizarlo: Si bien al evitar las redundancias evitamos tener varias versiones del mismo dato, las bases de datos suelen almacenar informaciones adicionales por motivos de seguridad y eficiencia que acaban ocupando ingentes cantidades de espacio.

A pesar de sus evidentes ventajas (no hay más que observar su uso hegemónico) también presentan ciertos inconvenientes, tales como coste, necesidad de formación, tiempo de implantación, rentabilidad a corto, no estandarización y desfase entre la teoría y la práctica.

No obstante el uso de archivos tradicionales sólo está recomendado en situaciones muy concretas, tales como:

- Aplicaciones que sólo usan un archivo con tratamientos sencillos y sin relaciones (libreta de direcciones).
- Archivos de configuración o de parámetros de las aplicaciones.
- Intercambio de datos entre distintos sistemas (importación/exportación).
- Copias de seguridad.

OBJETIVOS

Los objetivos fundamentales de una base de datos son:

- Los datos deben estar compartidos. Hay diversas formas que se verán más adelante.
- El uso de los datos debe ser controlado. De esta tarea se encarga el sistema de gestión de base de datos (SGBD).
- Los datos se integran de una forma lógica, eliminando redundancias, resolviendo ambigüedades en la definición y manteniendo la consistencia interna entre los mismos.

MODELOS

Hay 3 modelos clásicos fundamentales:

- **Jerárquico.** Este modelo presume de que todas las interrelaciones entre los datos pueden estructurarse como jerarquías. Los archivos se conectan entre sí mediante punteros físicos (direcciones físicas que identifican dónde se puede encontrar un registro en disco) o campos de datos añadidos a los registros individuales. Tiene algunas limitaciones, ya que no todas las relaciones se pueden expresar de forma jerárquica.
- **En red.** Debido a la necesidad de manipular las interrelaciones, se desarrolló este modelo de base de datos que maneja relaciones en forma de red en lugar de jerárquicas. También utiliza punteros físicos.
- **Relacional.** La debilidad que tenían los punteros físicos era que había que definir las interrelaciones antes de que el sistema fuera puesto en

explotación. Codd argumentó que los datos deberían relacionarse mediante interrelaciones naturales, lógicas, inherentes a los datos. Propuso un modelo en el que los datos se representarían en tablas constituidas por filas y columnas, llamadas relaciones. También propuso dos lenguajes para manipular los datos en las tablas: el álgebra relacional y el cálculo relacional. En los sistemas de bases de datos relacionales, los archivos se pueden procesar con instrucciones sencillas, sin embargo, en los sistemas tradicionales se deben procesar de registro en registro

A estos se ha añadido el modelo orientado a objetos y algunos otros más experimentales que comerciales. Tenemos que matizar que muchas veces se llama "base de datos orientada a objetos" a una base de datos relacional con una capa añadida que permite usarla como OO.

COMPONENTES

- **Hardware.** Es el conjunto de dispositivos físicos sobre los que reside una base de datos. Pueden usarse mainframes o minicomputadoras para soportar acceso a varios usuarios, o computadoras personales que se utilizan con bases de datos autónomas controladas por un usuario único. Hay que señalar también que las unidades de disco son el mecanismo de almacenamiento principal para las bases de datos.
  Hoy en día el modelo más extendido es combinar en una red distintos servidores especializados que atienden las necesidades de los clientes.
  El avance y el abaratamiento de la alta tecnología (ley de Moore), ha contribuido a la difusión de los sistemas de bases de datos.
- **Software.** Hay dos tipos de software: el sistema de gestión de bases de datos (SGBD) y el software de aplicación (que usa las facilidades del SGBD para manipular las bases de datos). Este último suele ser desarrollado por la compañía para resolver un problema específico, mientras que el SGBD debe brindar varios servicios. La arquitectura de cada base de datos comercial es diferente, pero deben incluir programas para acceder a los datos, un motor de consultas, optimizadores, herramientas de administración, interfaces con el SO y los servicios de red, etc.
- **Datos.** Los datos tienen que ser cuidadosa y lógicamente estructurados y deben almacenarse de manera precisa en el diccionario de datos.
- **Personas.** Pueden ser usuarios que necesitan información de la base de datos para desarrollar su responsabilidad en el negocio o informáticos su

responsabilidad reside en el diseño y mantenimiento del sistema de la base de datos.

- Usuarios informáticos
  - Diseñadores
  - Administradores (DBA).
  - Analistas y programadores
- Usuarios finales
  - Internos a la organización
  - Externos (clientes, proveedores, etc)

OBJETIVOS DE LA BASE DE DATOS

- **Versatilidad para representar la información**. Los mismos datos podrán utilizarse de distintas formas
- **Rendimiento:** La base de datos debe soportar las condiciones de trabajo más exigentes sin degradar excesivamente los tiempos de respuesta.
- **Redundancia mínima:** Para que una base de datos sea efectiva hace falta eliminar en la medida de lo posible las redundancias, es decir, las repeticiones que puedan llevar a error, como el llamar a un mismo campo de distinta manera en varios archivos, ya que si no existe el riesgo de inconsistencia entre las distintas versiones de los mismos datos.
- **Capacidad de acceso:** Gracias al SGBD existe la posibilidad de que varios usuarios tengan acceso de forma rápida y eficiente a los datos de la base. Al centralizar los datos en una base de datos, aumentan las probabilidades de que se dé este caso. Si el SGBD permite esto, seguramente el trabajo realizado por los usuarios se vería dañado, por eso el SGBD debe proteger los datos de la actualización simultánea por otro usuario; para ello utiliza mecanismos sofisticados de bloqueo.
- **Integridad** de los datos: consiste en mantener la precisión y consistencia de los valores de los datos. Los mecanismos de seguridad protegen la integridad de los datos. También se pueden mantener en el diccionario de datos restricciones sobre los valores, aunque es una tarea que resulta complicada. Los mecanismos de copias de seguridad y restauración soportados por el SGBD deben preservar los datos de cualquier fallo del sistema.

- **Seguridad y privacidad:** Los ABDs (administradores de la base de datos) pueden restringir el acceso a los usuarios a cada elemento de la base de datos sólo para recuperación o permitir acceso y actualización. La información relativa a los derechos de acceso se almacena en el diccionario de datos.
  El acceso a la base de datos también es controlado por un mecanismo de contraseñas; El encargado de la asignación de contraseñas también es el ABD.
- **Afinamiento (Tuning):** La estructura interna de los datos debe poder ajustarse según las necesidades de rendimiento. Para ello se utilizan índices, cachés, etc. En configuraciones avanzadas de hardware cobra importancia la distribución de los datos entre distintos discos o servidores.
- **Interfaz con el pasado y el futuro:** El sistema ha de ser capaz de importar los datos de la organización desde el sistema precedente (si existe) y ha de adaptarse a los cambios que la evolución del negocio requiera con un coste mínimo.

3.- Arquitectura de las bases de datos

En una base de datos hay que lograr la independencia entre las estructuras lógica y física de los datos, lo que significa distinguir entre datos y aplicaciones.

El concepto de independencia de los datos implica la separación entre el almacenamiento y la organización lógica de los datos tal como éstos se contemplan por los distintos programas de aplicación que hacen uso de la base, con lo que se consigue que unos mismos datos se puedan presentar de distintas formas según las necesidades y, por otra parte, que el almacenamiento de los datos, su estructura lógica y los programas de aplicación sean independientes unos de otros.

El **modelo ANSI** establece 3 niveles de abstracción distintos en los que se podría dividir una base de datos:

- Nivel interno: está compuesto por la vista física de la base de datos (discos, direcciones, punteros...). Este nivel es el más cercano a la realidad física de los datos. Debe ser transparente para la mayoría de los usuarios.
- Nivel conceptual: Es la representación de los datos que intervienen en el problema. Como resultado se obtiene un esquema conceptual con todos los elementos de los datos y sus relaciones.
- Nivel externo: es la colección de las vistas de distintos grupos de usuarios sobre la base de datos, las cuales describen los elementos de los datos y sus

relaciones. Cada usuario puede tener una visión distinta de los datos, según sea más adecuado para sus necesidades.

La implementación de estos 3 niveles requiere que el SGBD haga corresponder cada nivel con el otro.

Ejemplos de aplicación

ESQUEMA EXTERNO: Visión parcial de las tablas de la base de datos según el usuario.

Vista para el programa de listado de notas de alumnos con los siguientes datos: Curso, Nombre, Nombre de asignatura y Nota.

Curso	Nombre	Nombre Asignatura	Nota
1	Ana	Programación en lenguajes estructurados	6
1	Ana	Sistemas informáticos multiusuario y en red	8
2	Rosa	Desa. de aplic. en entornos de 4.ª Generación y H. Case	5
2	Juan	Desa. de aplic. en entornos de 4.ª Generación y H. Case	7
1	Alicia	Programación en lenguajes estructurados	5
1	Alicia	Sistemas informáticos multiusuario y en red	4

ESQUEMA CONCEPTUAL: Definición de todas las Tablas, Columnas y Restricciones.

Tabla ALUMNOS. Columnas: N.º Matricula, Nombre, Curso, Direccion, Población. Clave: N.º de matricula.

N_Matric	Nombre	Curso	Direc	Poblac
11111	Ana	1	C/Pilón 10	Oropesa
11110	Rosa	2	C/Las Viñas 26	Lagartera
11122	Juan	2	C/ Amapolas 24, 3F	Berrocalejo
23445	Alicia	1	C/Lamina 34	Caleruela

Tabla ASIGNATURAS. Columnas: Codigo, Nombre de asignatura. Clave: Codigo.

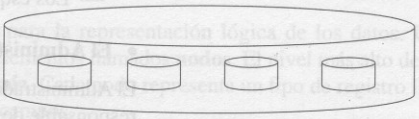
Codigo	Nombre Asignatura
1	Desa. de aplic. en entornos de 4.ª Generación y H. Case
2	Programación en lenguajes estructurados
3	Sistemas informáticos multiusuario y en red

Tabla NOTAS. Columnas: N.º Matricula, Codigo asignatura, Nota.

Matric	Codig	Nota
11111	2	6
11111	3	8
11110	1	5
11122	1	7
23445	2	5
23445	3	4

ESQUEMA INTERNO: Almacenamiento físico de los datos.

Archivo de Índices para ALUMNOS: Clave alumno, Dirección de la fila.
Archivo de Índices para ASIGNATURAS: Clave asignatura, Dirección de la fila.
Archivo de ALUMNOS: N.º Matrícula, Nombre, Curso, Dirección, Población.
Archivo de ASIGNATURAS: Código, Nombre de asignatura.



4.- SGBD

Un SGBD es un sistema computacional de propósito general que manipula la base de datos. A continuación se describen los diferentes servicios que ofrece.

El diccionario de datos/directorio (DD/D) almacena las definiciones de todos los elementos de los datos en la base de datos, así como las interrelaciones que existen entre las diversas estructuras de datos. A esto se le llaman **metadatos** o datos sobre los datos.

Mediante mecanismos de seguridad, el SGBD limita el acceso al personal autorizado y también lo restringe a ciertos datos. La **integridad** y la **consistencia** de la base de datos se protegen por medio de restricciones sobre los valores que pueden tomar los elementos de los datos y por las capacidades de recuperación y respaldo suministradas por el SGBD.

El SGBD proporciona los mecanismos físicos que permiten a varios usuarios tener acceso de forma rápida y eficiente a diferentes datos relacionados. También utiliza mecanismos de bloqueo para que la actualización de más de un usuario simultáneamente no afecte a los datos.

Se debe permitir a los usuarios formular sus consultas y pedir informes únicos directamente de la base de datos.

Por último, el SGBD ofrece al programador una serie de herramientas que facilitan la creación de software de aplicación.

5.- BASES DE DATOS Y REDES

CONCEPTO DE DISTRIBUCIÓN

Un sistema de base de datos distribuida consiste en varios sistemas de bases de datos operando en los sitios locales y conectados por líneas de comunicación.

PROCESAMIENTO DISTRIBUIDO

Una consulta o una actualización deja de ser un proceso simple controlado por un único módulo de software, se convierte en varios procesos cooperando entre sí controlado por varios módulos independientes. Pero para que funcione con efectividad, deben estar disponibles tecnologías adecuadas de comunicación y los SGBDs deben poder comunicarse entre sí.

VENTAJAS E INCONVENIENTES

Una clara ventaja es que es posible ubicar los datos en lugares donde se necesitan con más frecuencia, aunque también al mismo tiempo se permita a usuarios no locales acceder a los datos según sus necesidades. Esto mejora la relación costo-efectividad y la autonomía local.

PLATAFORMAS CLIENTE-SERVIDOR

Las plataformas cliente/servidor son sistemas abiertos, lo que significa que tratan de lograr la interoperabilidad entre dos o más sistemas, es decir que se comuniquen y contribuyan cada uno a alguna parte del trabajo común.

Los ordenadores clientes están interconectados a un servidor, así, un cliente que necesite hacer una consulta o actualización en la base de datos, envía una petición al servidor de la base de datos y este le devuelve los datos solicitados.

Gracias a esto, cada sistema se especializa en una parte del proceso: Los servidores de base de datos en cumplir los objetivos que nos hemos marcado (integridad,

seguridad, rendimiento, etc), mientras los clientes trabajan como front-end y son los que interactúan directamente con el Usuario.

6.- PROTECCIÓN DE DATOS: SEGURIDAD, INTEGRIDAD Y CONFIDENCIALIDAD

El uso concurrente de los datos comporta problemas de seguridad, integridad y confidencialidad, que el administrador debe paliar en la medida de lo posible. El SGBD facilita normalmente mecanismos para prevenir los fallos, detectarlos cuando se han producido y corregirlos después de haber sido detectados.

SEGURIDAD

El objetivo del concepto de seguridad (que suele conocerse también por recuperación, del inglés *recovery*) es el de proteger la BD contra fallos lógicos, físicos o humanos que destruyan los datos en todo o en parte. Estos fallos van desde catástrofes naturales, sabotajes, fallos del sistema operativo, fallos de discos u otras caídas del sistema intencionadas o no.

En lo que afecta al SGBD, existen dos tipos importantes de fallos:

- Los que provocan la pérdida de memoria volátil (interrupción del suministro eléctrico o anormal funcionamiento del equipo físico).
- Los que provocan la pérdida de memoria secundaria (por ejemplo, cuando patinan las cabezas de un disco).

Lo importante ante cualquier fallo es asegurar que la BD quede en un estado consistente, para lo cual se crean unidades de ejecución denominadas transacciones.

Transacción: Podemos definir transacción como una secuencia de operaciones que han de ejecutarse de forma atómica, es decir, o se realizan todas o no se realiza ninguna. Por ejemplo, una transferencia bancaria.

Las transacciones pueden acabar con éxito y ser grabadas o, por el contrario, pueden fracasar, en cuyo caso debe ser restaurado el estado en el que se encontraba la BD antes de que empezara a ejecutarse la transacción. Los SGBD suelen incorporar sentencias para la gestión de transacciones (por ejemplo, COMMIT, ROLLBACK, etc.) por si el usuario quisiera gestionarlas explícitamente. Además, las transacciones suelen gestionarse de forma implícita de manera que son abortadas si su finalización ha sido anormal y si su terminación es satisfactoria se grabarán automáticamente.

El método más extendido para anular y recuperar transacciones es la utilización de un fichero diario, llamado fichero log, en el que se guarda la información

necesaria para “deshacer”, en el caso de fracasar, o “rehacer”, en el caso de recuperar las transacciones. A veces existen dos ficheros diarios, uno con la imagen anterior a las modificaciones (llamado *before image log*) y otro con la imagen posterior a las modificaciones (*after image log*).

El fichero diario puede ser un fichero gestionado en forma de cola, que una vez llena va eliminando registros según van entrando registros nuevos. Los registros viejos se van grabando en un área intermedia y se pasan a la base de datos periódicamente.

Al ocurrir un fallo que dé lugar a pérdida de memoria volátil, es preciso realizar la operación denominada *recuperación en caliente*, en la que el sistema consulta el fichero diario para determinar las transacciones que hay que deshacer porque no han sido completadas y cuáles hay que rehacer porque si bien han sido completadas no han sido grabadas cuando se produjo el fallo.

Para evitar recorrer todo el fichero log se introduce el concepto de *punto de verificación* o *checkpoint* que se ejecuta periódicamente y que implica:

- Pasar el contenido de la memoria intermedia del diario al mismo diario.
- Escribir un registro de punto de recuperación en el diario.
- Pasar el contenido del área intermedia de la base a la base.
- Escribir la dirección del registro de recuperación en el fichero de rearranque.

Por ejemplo: supongamos que tenemos la siguiente secuencia de transacciones con el punto de Checkpoint y la caída del sistema de la figura:

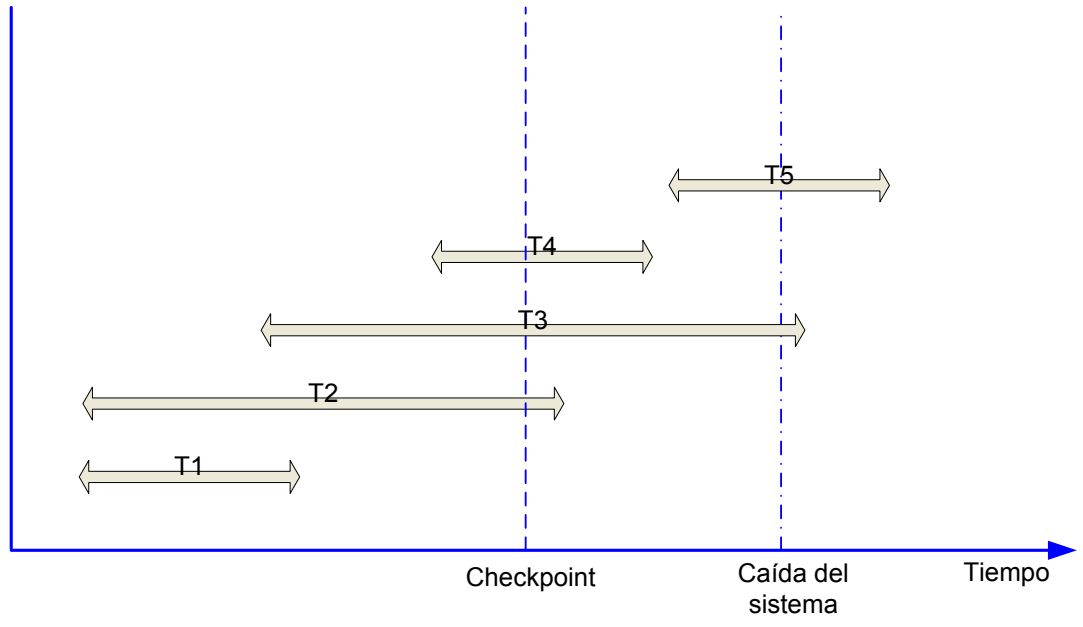


Fig. Estado de diversas transacciones ante un fallo del sistema.

- T1 no se ve afectada por la caída, puesto que se guardó en la base de datos en el momento de verificación T-c.
- T2 y T4 han terminado cuando se da la caída del sistema pero deben "rehacerse" porque no han sido grabadas en la base de datos, puesto que no se ha producido un Checkpoint posterior a su terminación.
- T3 y T5 deben "Deshacerse" puesto que no han finalizado a la caída del sistema.

En caso de fallo de memoria que afecte a la base de datos (*Recuperación en frío*) se utiliza un sistema de copia de seguridad o backup de la base, que permitirán junto con los ficheros diarios que se han ido produciendo desde que se realizó la copia, reconstruir la BD, llevándola de forma consistente a la situación anterior a que se produjera el fallo. Se sugiere copias de seguridad por errores fatales (ejemplo pérdida del diario, etc.).

Otra forma de garantizar la recuperación ante fallos es utilizar la técnica de páginas ocultas (recuperación en caliente) o *Shadow paging,* que consiste en mantener dos tablas de páginas durante la vida de la transacción. Al empezar la transacción ambas páginas son iguales, reflejándose todos los cambios en una de ellas (la *primaria*) y manteniendo la otra (la *secundaria*) sin cambios. Se graba la de los cambios si todo va bien desechando la secundaria, y si la transacción se aborta se desecha la primaria y se restablece la secundaria.

INTEGRIDAD

Hablamos de integridad en el sentido de corrección, validez o precisión de los datos de la base, con el objetivo de obtener un diseño íntegro para proteger la BD contra operaciones que introduzcan inconsistencia en los datos.

Existen dos tipos de operaciones que pueden atentar contra la integridad de los datos:

a) Operaciones semánticas inconsistentes. Son las que violan restricciones que ha definido el administrador. Podrían ser restricciones sobre los dominios, por ejemplo, en cierta base el dominio de la edad puede ser un entero entre 18 y 65, o sobre los atributos, por ejemplo si el estado civil de una persona es casado su edad debe ser mayor de 14 años.

b) Interferencia por accesos concurrentes. En sistemas multiusuarios es necesario un mecanismo de control de concurrencia. Para ello se utilizan técnicas que establecen controles previos para evitar que se produzcan situaciones de

inconsistencia, conocidas como técnicas pesimistas, o bien, técnicas que suponen que en principio no tiene por qué haber problemas y en caso de haberlos actúan para volver la base de datos a un estado consistente y que son conocidas como técnicas optimistas.

CONFIDENCIALIDAD

El objetivo de construir un diseño con esta característica es proteger a la BD de accesos no autorizados. La protección deberá incluir no sólo a los datos y sus interrelaciones, sino también a los esquemas, programas, etc. El tema de la confidencialidad atañe a:

- Aspectos legales, sociales y éticos.
- Cuestiones de política de empresa.
- Controles de acceso a las instalaciones.
- Controles basados sobre soporte físico que pueden llegar al extremo de usar la voz, las huellas dactilares e incluso la retina para identificación de los usuarios.

Los aspectos de confidencialidad han de tenerse en cuenta desde el diseño de la base de datos hasta las pruebas del último de los módulos. La falta de cuidado de los desarrolladores que tienden a soslayar las “incómodas” barreras de seguridad que les complican el trabajo han provocado terribles agujeros de seguridad en no pocas organizaciones.

También hemos de procurar que las medidas de seguridad entorpezcan lo menos posible el trabajo de los usuarios, de otro modo acabarán adoptando prácticas que las pongan en peligro. Por ejemplo, si establecemos una política de contraseñas muy complejas y las cambiamos a menudo, muchos usuarios las olvidarán o las anotarán.