

TEMA 1

BASES DE DATOS

Desarrollo de Aplicaciones Web

José Luis Comesaña



2011

TEMA 1

ÍNDICE

1.- Introducción.....	1
2.- Los ficheros de información.....	2
2.1.- ¿Qué es un fichero?.....	2
2.2.- Tipos de ficheros.....	3
2.3.- Los soportes de información.....	3
2.4.- Métodos de acceso.....	4
2.5.- Ficheros secuenciales.....	4
2.6.- Ficheros de acceso directo.....	5
2.7.- Ficheros indexados.....	6
2.8.- Otros (secuenciales indexados, hash.).....	7
a. Ficheros Secuenciales Indexados.....	7
b. Ficheros de Acceso Calculado o Hash.....	7
➔ Módulo.....	8
➔ Extracción.....	8
2.9.- Parámetros de utilización.....	8
3.- Bases de datos.....	9
3.1.- Conceptos.....	9
Base de datos.....	9
3.2.- Usos.....	10
¿Quién utiliza las bases de datos?.....	10
¿Para qué se utilizan las bases de datos?.....	11
3.3.- Ubicación de la información.....	11
4.- Modelos de bases de datos.....	13
4.1.- Modelo jerárquico.....	13
4.2.- Modelo en red.....	14
4.3.- Modelo relacional.....	14
4.4.- Modelo orientado a objetos.....	15
4.5.- Otros modelos.....	16
Modelo Objeto-Relacional.....	16
Modelo de bases de datos deductivas.....	16
Bases de datos multidimensionales.....	25
Bases de datos transaccionales.....	25
5.- Tipos de bases de datos.....	26
6.- Sistemas gestores de bases de datos.....	30
6.1.- Funciones.....	31
6.2.- Componentes.....	32
6.3.- Arquitectura.....	33
6.4.- Tipos.....	34
7.- SGBD comerciales.....	35
8.- SGBD libres.....	37
9.- Bases de datos centralizadas.....	38
10.- Bases de datos distribuidas.....	39
10.1.- Fragmentación.....	40
11.- Primeros pasos en Oracle Database 10g Express Edition.....	42
¿Qué es Oracle Database 10g Express Edition?.....	42
¿Por dónde empezamos?.....	42
¿Cómo se realiza la instalación?.....	42
Instalación de Oracle Database 10g Express Edition bajo Windows 7.....	42
Instalación de Oracle Database 10g Express Edition bajo Ubuntu Linux.....	43

Gestión básica de datos en Oracle Database 10g Express Edition	43
Administración simple de usuarios en Oracle Database 10g Express Edition	43

Almacenamiento de la información

Caso práctico

Ada sabe bien que BK Programación deberá hacer frente a retos importantes que requerirán del dominio adecuado de múltiples disciplinas. Tiene claro que el desarrollo de sus proyectos ha de estar apoyado sobre unas bases firmes, y una de ellas será la gestión adecuada de los datos.

*Considera que **Juan** y **María** deben conocer la evolución que han experimentado las técnicas de almacenamiento de información, destacando que el dominio de las bases de datos es fundamental para garantizar un funcionamiento óptimo de las aplicaciones que BK Programación va a tener que desarrollar.*

1.- Introducción.

¿Te has preguntado alguna vez dónde y de qué manera se almacenan y gestionan los datos que utilizamos diariamente? Si pensamos en cualquier acción de nuestra vida cotidiana, o si analizamos la mayoría de los ámbitos de actividad, nos encontramos que la utilización de las bases de datos está ampliamente extendida. Éstas, y los datos contenidos en ellas, serán imprescindibles para llevar a cabo multitud de acciones.

¿Crees que no es para tanto? Piensa en las siguientes situaciones:

- ✓ Cuando seleccionamos nuestro canal favorito en la TDT.
- ✓ Al utilizar la agenda del móvil para realizar una llamada telefónica.
- ✓ Cuando operamos en el cajero automático.
- ✓ Al solicitar un certificado en un organismo público.
- ✓ Cuando acudimos a la consulta del médico.
- ✓ Al inscribirnos en un curso, plataforma OnLine, etc.
- ✓ Si utilizas un GPS.
- ✓ Cuando reservamos unas localidades para un evento deportivo o espectáculo.
- ✓ Si consumimos ocio digital.
- ✓ Cuando consultamos cualquier información en Internet. (Bibliotecas, enciclopedias, museos, etc.)
- ✓ Al registrarte en una página de juegos OnLine, redes sociales o foros.
- ✓ Incluso, si tienes coche, puede ser que éste incorpore alguna base de datos.

Suponemos que no es necesario que continuemos más para darnos cuenta de que casi todo lo que nos rodea, en alguna medida, está relacionado con los datos, su almacenamiento y su gestión. El gran volumen de datos que actualmente manejamos y sus innumerables posibilidades requieren de la existencia de técnicos perfectamente formados y capaces de trabajar con ellos.

Este módulo profesional se centra en el estudio de las **Bases de Datos** y su uso en el desarrollo de aplicaciones. En esta primera unidad comenzaremos conociendo los primeros sistemas basados en ficheros para el almacenamiento y gestión de la información. Seguidamente, se desarrollarán los conceptos y definiciones básicas relacionadas con las bases de datos, posteriormente analizaremos sus modelos y tipos, un poco más adelante, podremos conocer las características y capacidades de los sistemas gestores de bases de datos y finalmente, identificaremos las herramientas reales con las que llevar a cabo la gestión dichas bases.

2.- Los ficheros de información.

Caso práctico

Juan le cuenta a María que hace poco visitó un museo en el que había una exposición sobre historia de la informática y que pudo ver soportes antiguos para almacenamiento de información: tarjetas perforadas, cintas magnéticas, tambores magnéticos, discos de diferentes tamaños y otros dispositivos de la época.

-Todo ha evolucionado muchísimo, la cantidad de datos y archivos que hoy podemos transportar en los modernos sistemas de almacenamiento y la velocidad a la que podemos acceder a ellos es sorprendente -comenta María.

Ada, mientras, prepara un DVD para realizar una copia de seguridad de los archivos de su portátil, destaca que gracias a las mejoras en el modo de organización de ficheros y soportes de información, se ha abierto un sin fin de posibilidades para la aplicación de las TIC en cualquier ámbito.

2.1.- ¿Qué es un fichero?

En la década de los setenta, los procesos básicos que se llevaban a cabo en una empresa se centraban en cuestiones relacionadas con contabilidad y facturación. Las necesidades de almacenamiento y gestión de información podían satisfacerse utilizando un número relativamente reducido de archivos en papel agrupados y ordenados, los típicos ficheros clásicos.

Al llevar a cabo una primera informatización, se pasó de tener los datos en formato papel a poder acceder a ellos de manera mucho más rápida a través del ordenador. En ese momento, la informática adaptó sus herramientas para que los elementos que el usuario maneja en el ordenador se parezcan a los que utilizaba manualmente. Así en informática se sigue hablado de ficheros, formularios, carpetas, directorios,...

La información debía ser trasladada desde el papel al formato digital y por lo general, era necesario almacenarla para su posterior recuperación, consulta y procesamiento. De este modo, para llevar a cabo un tratamiento eficiente de ésta era necesario establecer métodos adecuados para su almacenamiento. El elemento que permitió llevar a cabo el almacenamiento de datos de forma permanente en dispositivos de memoria masiva fue el fichero o archivo.

Fichero o archivo: conjunto de información relacionada, tratada como un todo y organizada de forma estructurada. Es una secuencia de dígitos binarios que organiza información relacionada con un mismo aspecto.

Los ficheros están formados por registros lógicos que contienen datos relativos a un mismo elemento u objeto (por ejemplo, los datos de usuarios de una plataforma educativa). A su vez, los registros están divididos en campos que contienen cada una de las informaciones elementales que forman un registro (por ejemplo, el nombre del usuario o su dirección de correo electrónico).

Hemos de resaltar que los datos están almacenados de tal forma que se puedan añadir, suprimir, actualizar o consultar individualmente en cualquier momento.

Como los ficheros suelen ser muy voluminosos, solo se pueden llevar a la memoria principal partes de ellos para poder procesarlos. La cantidad de información que es transferida entre el soporte en el que se almacena el fichero, y la memoria principal del ordenador, en una sola operación de lectura/grabación, recibe el nombre de registro físico o bloque.

Normalmente en cada operación de lectura/grabación se transfieren varios registros del fichero, es decir un bloque suele contener varios registros lógicos. Al número de registros que entran en un bloque se le conoce con el nombre de factor de bloqueaje, y a esta operación de agrupar varios registros en un bloque se le llama bloqueo de registros.

2.2.- Tipos de ficheros.

Según la función que vaya a desempeñar los ficheros, éstos pueden ser clasificados de varias maneras. En la siguiente imagen puedes observar una posible clasificación.



- a. **Ficheros permanentes:** contienen información relevante para una aplicación. Es decir, los datos necesarios para el funcionamiento de ésta. Tienen un periodo de permanencia en el sistema amplio. Estos se subdividen en:
- ✓ **Ficheros maestros:** contienen el estado actual de los datos que pueden modificarse desde la aplicación. Es la parte central de la aplicación, su núcleo. Podría ser un archivo con los datos de los usuarios de una plataforma educativa.
 - ✓ **Ficheros constantes:** son aquellos que incluyen datos fijos para la aplicación. No suelen ser modificados y se accede a ellos para realización de consultas. Podría ser un archivo con códigos postales.
 - ✓ **Ficheros históricos:** contienen datos que fueron considerados como actuales en un periodo o situación anterior. Se utilizan para la reconstrucción de situaciones. Podría ser un archivo con los usuarios que han sido dados de baja en la plataforma educativa.
- b. **Ficheros temporales:** Se utilizan para almacenar información útil para una parte de la aplicación, no para toda ella. Son generados a partir de datos de ficheros permanentes. Tienen un corto periodo de existencia. Estos se subdividen en:
- ✓ **Ficheros intermedios:** almacenan resultados de una aplicación que serán utilizados por otra.
 - ✓ **Ficheros de maniobras:** almacenan datos de una aplicación que no pueden ser mantenidos en memoria principal por falta de espacio.
 - ✓ **Ficheros de resultados:** almacenan datos que van a ser transferidos a un dispositivo de salida.

Supongamos una aplicación informática para gestionar una biblioteca, existirá un fichero con el catálogo de libros disponibles, otro con las editoriales, otro con información sobre libros que se han quedado obsoletos, etc. ¿A cuál de los siguientes tipos correspondería el fichero que almacena las editoriales?

- Fichero maestro.
- Fichero constante.**
- Fichero intermedio.

2.3.- Los soportes de información.

Los ficheros se almacenan en soportes de información manejados por dispositivos periféricos del ordenador, que permiten leer y grabar datos en el soporte. Los soportes más utilizados para almacenar los ficheros son las cintas magnéticas y los discos (magnéticos, ópticos, o magneto-ópticos). Dentro de estos dos tipos de soporte existen en el mercado una gran variedad de modelos. Inicialmente, los primeros sistemas de almacenamiento físico eran tambores de cinta magnética. Tenían unas dimensiones parecidas a los discos de vinilo. Estos tambores funcionaban de manera similar a los antiguos casetes, pero sus mayores dimensiones les permitían almacenar gran cantidad de datos en formato digital, es decir en ceros y unos, en orden secuencial.

Posteriormente, los sistemas de almacenamiento de información comenzaron a cambiar de la mano de los avances en el hardware, en concreto con la aparición del disquete y del disco duro. Eran

dispositivos de acceso aleatorio, no siendo necesario en ellos pasar por todos los datos desde el inicio hasta la zona donde se encuentra la información que nos interesa.

Por tanto, se distinguen dos tipos de soportes para el almacenamiento de datos:

- ✓ **Soportes de Acceso Directo a los datos** (Por ejemplo: discos). Son los más empleados y el acceso a los datos puede hacerse de forma directa, pudiendo colocarnos en la posición que nos interesa y leer a partir de ella.
- ✓ **Soportes de Acceso Secuencial** (Por ejemplo: cintas magnéticas). Se suelen usar en copias de seguridad y si deseamos leer un dato que está en la mitad de la cinta, tendremos que leer todo lo que hay hasta llegar a esa posición.

Conoce más sobre las características de cintas y discos a través de los enlaces que te proponemos:

http://es.wikipedia.org/wiki/Cinta_magn%C3%A9tica_de_almacenamiento_de_datos

<http://www.infodata.es/tour/disco-duro.html>

http://es.wikipedia.org/wiki/Disco_%C3%B3ptico

<http://www.consumer.es/web/es/tecnologia/hardware/2006/01/26/148862.php>

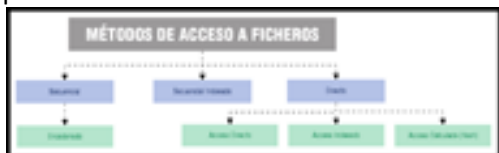
<http://www.infodata.es/tour/magneto-opticos.html>

2.4.- Métodos de acceso.

A medida que la tecnología ha ido evolucionando, atendiendo principalmente a los avances hardware, el acceso a la información contenida en los diferentes tipos de ficheros ha variado mucho. Los objetivos fundamentales de estas modificaciones pueden resumirse en los siguientes puntos:

- ✓ Proporcionar un acceso rápido a los registros.
- ✓ Conseguir economizar el almacenamiento.
- ✓ Facilitar la actualización de los registros.
- ✓ Permitir que la estructura refleje la organización real de la información.

Las distintas formas de organizar un fichero en un soporte de memoria o, lo que se conoce también por métodos de acceso a los ficheros se detallan en el siguiente gráfico.



Las organizaciones secuenciales, de acceso aleatorio o directo y de acceso indexado son las más comunes. En los siguientes epígrafes se detallarán las características de cada uno de los métodos de acceso a los ficheros.

Relaciona los diferentes métodos de acceso a los ficheros.

Método de acceso	Relación	Tipo de acceso
Encadenado.	2	1. Directo.
Indexado.	3	2. Secuencial.
Calculado o Hash.	1	3. Directo o secuencial.

Como ves el método de acceso Indexado existe en los dos tipos de acceso: directo y secuencial.

2.5.- Ficheros secuenciales.

Un fichero con organización secuencial se caracteriza porque sus registros están almacenados de forma contigua, de manera, que la única forma de acceder a él, es leyendo un registro tras otro desde el principio hasta el final. En los ficheros secuenciales suele haber una marca indicativa del fin

del fichero, que suele denominarse **EOF** (End of File). Para detectar el final del fichero sólo es necesario encontrar la marca EOF.

Este tipo de ficheros pueden utilizar dispositivos o soportes no direccionables o de acceso secuencial, como son las cintas magnéticas de almacenamiento de datos. También se utiliza en los CD de audio y los DVD de vídeo, en los que la música o las imágenes se almacenan a lo largo de una espiral continua.

Los registros almacenados se identifican por medio de una información ubicada en uno de sus campos, a este campo se le denomina **clave o llave**. Si se ordena un archivo secuencial por su clave, es más rápido realizar cualquier operación de lectura o escritura.

Otras características relevantes de los ficheros secuenciales son:

- ✓ La lectura siempre se realiza hacia delante.
- ✓ Son ficheros monousuario, no permiten el acceso simultáneo de varios usuarios.
- ✓ Tienen una estructura rígida de campos. Todos los registros deben aparecer en orden, es decir, la posición de los campos de cada registro siempre ha de ser la misma.
- ✓ El modo de apertura del fichero, condiciona la lectura o escritura.
- ✓ Aprovechan al máximo el soporte de almacenamiento, al no dejar huecos vacíos.
- ✓ Se pueden grabar en cualquier tipo de soporte, tanto en secuenciales como direccionables.
- ✓ Todos los lenguajes de programación disponen de instrucciones para trabajar con este tipo de ficheros.
- ✓ No se pueden insertar registros entre los que ya están grabados.



En el siguiente gráfico se observa la estructura de un fichero secuencial.

2.6.- Ficheros de acceso directo.

En este tipo de ficheros se puede acceder a un registro indicando la posición relativa del mismo dentro del archivo o, más comúnmente, a través de una clave que forma parte del registro como un campo más. Estos archivos deben almacenarse en dispositivos de memoria masiva de acceso directo, como son los discos magnéticos.

Campo clave: campo que permite identificar y localizar un registro de manera ágil y organizada.

Cada uno de los registros se guarda en una posición física, que dependerá del espacio disponible en memoria masiva, de ahí que la distribución de los registros sea aleatoria dentro del soporte de almacenamiento. Para acceder a la posición física de un registro se utiliza una dirección o índice, no siendo necesario recorrer todo el fichero para encontrar un determinado registro.

A través de una transformación específica aplicada a la clave, se obtendrá la dirección física en la que se encuentra el registro. Según la forma de realizar esta transformación, existen diferentes modos de acceso:



En el acceso directo la clave coincide con la dirección, debiendo ser numérica y comprendida dentro del rango de valores de las direcciones. Es el método más rápido.

La medida básica de posicionamiento del puntero en el fichero es el byte, dependiendo del tipo de codificación de caracteres que empleemos ([Unicode](#), [ANSI](#)) se utilizarán 1 o 2 bytes por carácter respectivamente. Teniendo esto en cuenta, el puntero avanzará de

uno en uno o de dos en dos bytes para poder leer o escribir cada carácter.

Otras características fundamentales de los ficheros de acceso directo o aleatorio son:

- ✓ Posicionamiento inmediato.
- ✓ Registros de longitud fija.
- ✓ Apertura del fichero en modo mixto, para lectura y escritura.
- ✓ Permiten múltiples usuarios utilizándolos.
- ✓ Los registros se borran colocando un cero en la posición que ocupan.
- ✓ Permiten la utilización de algoritmos de compactación de huecos.
- ✓ Los archivos se crean con un tamaño definido, es decir, con un máximo de registros establecido durante la creación.
- ✓ Esta organización sólo es posible en soportes direccionables.
- ✓ Se usan cuando el acceso a los datos de un registro se hace siempre empleando la misma clave y la velocidad de acceso a un registro es lo que más nos importa.
- ✓ Permiten la actualización de los registros en el mismo fichero, sin necesidad de copiar el fichero.
- ✓ Permiten realizar procesos de actualización en tiempo real.

En los ficheros de acceso directo los registros siempre se encuentran en posiciones contiguas dentro del soporte de almacenamiento.

Verdadero.

Falso

2.7.- Ficheros indexados.

Se basan en la utilización de **índices**, que permiten el acceso a un registro del fichero de forma directa, sin tener que leer los anteriores. Estos índices son similares a los de los libros. Si nos interesa leer un capítulo concreto podemos recurrir al índice que nos dice en que página comienza, y abrimos el libro por esa página, sin tener que mirar en todas las páginas anteriores para localizarlo.

Por tanto, existirá una **zona de registros** en la que se encuentran los datos del archivo y una **zona de índices**, que contiene una tabla con las claves de los registros y las posiciones donde se encuentran los mismos. La tabla de índices estará ordenada por el campo clave.

La tabla de índices será cargada en memoria principal para realizar en ella la búsqueda de la fila correspondiente a la clave del registro a encontrar, obteniéndose así la dirección donde se encuentra el registro. Una vez localizada la dirección, sólo hay que acceder a la zona de registros en el soporte de almacenamiento y posicionarnos en la dirección indicada. Puesto que la tabla debe prever la inclusión de todas las direcciones posibles del archivo, su principal inconveniente resulta determinar su tamaño y mantenerla ordenada por los valores de la clave.



Las características más relevantes de un fichero indexado, son las siguientes:

- ✓ El diseño del registro tiene que tener un campo, o combinación de campos, que permita identificar cada registro de forma única, es decir, que no pueda haber dos registros que tengan la misma información en él. A este campo se le llama **campo clave** y es el que va a servir de índice. Un mismo fichero puede tener mas de un campo clave, pero al menos uno de ellos no admitirá valores duplicados y se le llama clave primaria. A las restantes se les llama claves alternativas.
- ✓ Permiten utilizar el modo de **acceso secuencial** y el modo de **acceso directo** para leer la información guardada en sus registros.
- ✓ Para acceder a este tipo de ficheros utilizando el modo de acceso directo se hace conociendo el contenido del campo clave del registro que queremos localizar. Con esa información el sistema operativo puede consultar el índice y conocer la posición del registro dentro del fichero.

- ✓ Para acceder a este tipo de ficheros utilizando el modo de acceso secuencial los registros son leídos ordenados por el contenido del campo clave, independientemente del orden en que se fueron grabando (el orden lógico no es igual al orden físico), debido a que el acceso a los datos se hace a través del índice, que para hacer más fácil la búsqueda de los registros, permanece siempre ordenado por el campo clave.
- ✓ Solamente se puede grabar en un soporte direccionable. Por ejemplo, un disco magnético. Si esto no fuera así, no podría emplear el acceso directo.

2.8.- Otros (secuenciales indexados, hash.).

Existen otros tipos de organización de ficheros, ficheros secuenciales indexados y ficheros de acceso calculado, a continuación se detallan las características de cada uno de ellos.

a. Ficheros Secuenciales Indexados:

También llamados parcialmente indexados, al igual que en los ficheros indexados existe una **zona de índices** y otra **zona de registros de datos**, pero esta última se encuentra dividida en **segmentos** (bloques de registros) ordenados.

En la tabla de índices, cada fila hace referencia a cada uno de los segmentos. La clave corresponde al último registro y el índice apunta al registro inicial. Una vez que se accede al primer registro del segmento, dentro de él se localiza (de forma secuencial) el registro buscado.

Esta organización es muy utilizada, tanto para procesos en los que intervienen pocos registros como para aquellos en los que se maneja el fichero completo.

Las principales características son:

- Permite el acceso secuencial. Esto es muy interesante cuando la tasa de actividad es alta. En el acceso secuencial, además, los registros se leen ordenados por el campo clave.
- Permite el acceso directo a los registros. Realmente emula el acceso directo, empleando para ello las tablas de índices. Primero busca la clave en el área de índices y luego va a leer al área de datos en la dirección que le indica la tabla.
- Se pueden actualizar los registros en el mismo fichero, sin necesidad de crear un fichero nuevo de copia en el proceso de actualización.
- Ocupa más espacio en el disco que los ficheros secuenciales, debido al uso del área de índices.
- Solo se puede utilizar soportes direccionables.
- Obliga a una inversión económica mayor, por la necesidad de programas y, a veces, hardware más sofisticado.

b. Ficheros de Acceso Calculado o Hash:

Cuando utilizamos ficheros indexados es necesario siempre tener que consultar una tabla para obtener la dirección de almacenamiento a partir de la clave. La técnica del acceso calculado o **hash**, permite accesos más rápidos, ya que en lugar de consultar una tabla, se utiliza una transformación o función matemática (función de hashing) conocida, que a partir de la clave genera la dirección de cada registro del archivo. Si la clave es alfanumérica, deberá previamente ser transformada en un número.

El mayor problema que presenta este tipo de ficheros es que a partir de diferentes claves se obtenga la misma dirección al aplicar la función matemática o transformación. A este problema se le denomina **colisión**, y las claves que generan la misma dirección se conocen por **sinónimos**. Para resolver este problema se aplican diferentes métodos, como tener un bloque de excedentes o zona de sinónimos, o crear un archivo de sinónimos, etc.

Para llevar a cabo la transformación existen multitud de métodos, siendo algunos:

→ **Módulo:** La dirección será igual al resto de la división entera entre la clave y el número de registros.

→ **Extracción:** La dirección será igual a una parte de las cifras que se extraen de la clave.

Una buena transformación o función de hash, será aquella que produzca el menor número de colisiones. En este caso hay que buscar una función, a ser posible biunívoca, que relacione los posibles valores de la clave con el conjunto de números correlativos de dirección. Esta función consistirá en realizar una serie de cálculos matemáticos con el valor de la clave hasta obtener un número entre 1 y n, siendo n el número de direcciones que tiene el fichero.

En un fichero con acceso calculado:



Se utiliza la dirección como clave.



Hay una tabla en la que está cada clave con la dirección del registro correspondiente.



La dirección se obtiene a partir de la clave mediante un algoritmo.

2.9.- Parámetros de utilización.

En función del uso que se vaya a dar al fichero, serán adecuados unos tipos u otros de organización. Mediante la utilización de **parámetros de referencia**, podremos determinar el uso de un fichero. Estos parámetros son:

- Capacidad o volumen:** es el espacio, en caracteres, que ocupa el fichero. La capacidad podrá calcularse multiplicando el número previsto de registros por la longitud media de cada registro.
- Actividad:** permite conocer la cantidad de consultas y modificaciones que se realizan en el fichero. Para poder especificar la actividad se deben tener en cuenta:
 - **Tasa de consulta o modificación:** que es el porcentaje de registros consultados o modificados en cada tratamiento del fichero, respecto al número total de registros contenidos en él.
 - **Frecuencia de consulta o modificación:** número de veces que se accede al fichero para hacer una consulta o modificación en un periodo de tiempo fijo.
- Volatilidad:** mide la cantidad de inserciones y borrados que se efectúan en un fichero. Para determinar la volatilidad es necesario conocer:
 - **Tasa de renovación:** es el tanto por ciento de registros renovados en cada tratamiento del fichero, respecto al número total de registros contenidos en él.
 - **Frecuencia de renovación:** es el número de veces que se accede al fichero para renovarlo en un periodo de tiempo fijo.
- Crecimiento:** es la variación de la capacidad del fichero y se mide con la tasa de crecimiento, que es el porcentaje de registros en que aumenta el fichero en cada tratamiento.

La volatilidad de un fichero es un parámetro que indica:



La variación del volumen del fichero.



La cantidad de veces que se abre o cierra el fichero.



El peso de los procesos de inserción y borrado en dicho fichero (frecuencia de renovación).

3.- Bases de datos.

Caso práctico

Ada, Juan y María, se han reunido para aclarar ideas sobre qué sistema de gestión de información van a utilizar.

-Bases de datos, está claro. Pero, hay de varios tipos ¿no? -pregunta **Juan**.

Ada, asiente con la cabeza y confirma a sus dos compañeros que la práctica totalidad de los sistemas de información actuales utilizan acceso a bases de datos.

Continúa **Ada**: -Sé que todos conocemos lo que son las bases de datos, pero es necesario afianzar y aclarar muchos conceptos fundamentales que nos van hacer falta para plantear, diseñar y construir las bases de datos que nuestras aplicaciones utilizarán.

Si BK Programación va a desarrollar aplicaciones para diferentes ámbitos, deberá documentarse adecuadamente para poder seleccionar qué sistema de base de datos debe utilizar en cada situación. Para ello, todos sus miembros tendrán que recordar, actualizar o aprender gran cantidad de interesantes conocimientos relacionados con este campo de la informática.

Como hemos visto anteriormente, los ficheros permiten organizar y memorizar conjuntos de datos del mismo tipo o naturaleza con una determinada estructura, siendo un medio para el almacenamiento de los datos o resultados de una aplicación específica. Pero si las aplicaciones, al ser diseñadas, deben depender directamente de sus ficheros o archivos, se pierde independencia y surgen serios inconvenientes: como información duplicada, incoherencia de datos, fallos de seguridad, etc.

Estos problemas debían ser solucionados, es cuando aparece el concepto de base de datos. Una base de datos permitirá reunir toda la información relacionada en un único sistema de almacenamiento, pudiendo cualquier aplicación utilizarla de manera independiente y ofreciendo una mejora en el tratamiento de la información, así como una evolución para el desarrollo de aplicaciones.

La gestión de las bases de datos ha experimentado gran cantidad de cambios, partiendo de aplicaciones especializadas hasta llegar a convertirse en el núcleo de los entornos informáticos modernos. Con la llegada de Internet en los noventa, el número de usuarios de bases de datos creció exponencialmente, y aunque muchos de ellos no sean conscientes de ello, el acceso a dichas bases forma parte de la vida cotidiana de muchos de nosotros.

Conocer los sistemas que gestionan las bases de datos, sus conceptos fundamentales, el diseño, lenguajes y la implementación de éstas, podemos considerarlo imprescindible para alguien que se está formando en el campo de la informática.

3.1.- Conceptos.

A finales de los setenta, la aparición de nuevas tecnologías de manejo de datos a través de los sistemas de bases de datos supuso un considerable cambio. Los sistemas basados en ficheros separados dieron paso a la utilización de sistemas gestores de bases de datos, que son sistemas software centralizados o distribuidos que ofrecen facilidades para la definición de bases de datos, selección de estructuras de datos y búsqueda de forma interactiva o mediante lenguajes de programación.

Llegados a este punto, te preguntarás... ¿Qué es una base de datos?

Base de datos: Es una colección de datos relacionados lógicamente entre sí, con una definición y descripción comunes y que están estructurados de una determinada manera. Es un conjunto estructurado de datos que representa entidades y sus interrelaciones, almacenados con la mínima




redundancia y posibilitando el acceso a ellos eficientemente por parte de varias aplicaciones y usuarios.

La base de datos no sólo contiene los datos de la organización, también almacena una descripción de dichos datos. Esta descripción es lo que se denomina **metadatos**, se almacena en el **diccionario de datos o catálogo** y es lo que permite que exista **independencia de datos** lógica-física.

Una base de datos constará de los siguientes elementos:

- ✓ **Entidades:** objeto real o abstracto con características diferenciadoras de otros, del que se almacena información en la base de datos. En una base de datos de una clínica veterinaria, posibles entidades podrían ser: ejemplar, doctor, consulta, etc.
- ✓ **Atributos:** son los datos que se almacenan de la entidad. Cualquier propiedad o característica de una entidad puede ser atributo. Continuando con nuestro ejemplo, podrían ser atributos: raza, color, nombre, número de identificación, etc.
- ✓ **Registros:** donde se almacena la información de cada entidad. Es un conjunto de atributos que contienen los datos que pertenecen a una misma repetición de entidad. En nuestro ejemplo, un registro podría ser: 2123056, Sultán, Podenco, Gris, 23/03/2009.
- ✓ **Campos:** donde se almacenan los atributos de cada registro. Teniendo en cuenta el ejemplo anterior, un campo podría ser el valor Podenco.

Una base de datos es:

-  Un programa para gestionar archivos muy grandes.
-  El conjunto de datos de los usuarios almacenados en un único disco duro.
-  Conjunto de datos de distinto tipo relacionados entre sí, junto con un programa de gestión de dichos datos.

3.2.- Usos.

Ya sabemos lo que es una base de datos y sus características principales, pero es necesario conocer quien las usa y para qué.

¿Quién utiliza las bases de datos?

Existen cuatro tipos de personas que pueden hacer uso de una base de datos: el administrador, los diseñadores de la base de datos, los programadores de aplicaciones y los usuarios finales.

¿Quién utiliza las bases de datos?	
Tipo	Funciones y características
El administrador	Es la persona encargada de la creación o implementación física de la base de datos. Es quien escoge los tipos de ficheros, los índices que hay que crear, la ubicación de éstos, etc. En general, es quien toma las decisiones relacionadas con el funcionamiento físico del almacenamiento de información. Siempre teniendo en cuenta las posibilidades del sistema de información con el que trabaje. Junto a estas tareas, el administrador establecerá la política de seguridad y de acceso para garantizar el menor número de problemas.
Los diseñadores	Son las personas encargadas de diseñar cómo será la base de datos. Llevarán a cabo la identificación de los datos, las relaciones entre ellos, sus restricciones, etc. Para ello han de conocer a fondo los datos y procesos a representar en la base de datos. Si estamos hablando de una empresa, será necesario que conozcan las reglas de negocio en la que esta se mueve. Para obtener un buen resultado, el diseñador de la base de datos debe implicar en el proceso a todos los usuarios de la base de datos, tan pronto como sea

	posible.
Los programadores de aplicaciones	Una vez diseñada y construida la base de datos, los programadores se encargarán de implementar los programas de aplicación que servirán a los usuarios finales. Estos programas de aplicación ofrecerán la posibilidad de realizar consultas de datos, inserción, actualización o eliminación de los mismos. Para desarrollar estos programas se utilizan lenguajes de tercera o cuarta generación.
Los usuarios finales	Son los clientes finales de la base de datos. Al diseñar, implementar y mantener la base de datos se busca cumplir los requisitos establecidos por el cliente para la gestión de su información.

¿Para qué se utilizan las bases de datos?

Enumerar todos y cada uno de los campos donde se utilizan las bases de datos es complejo, aunque seguro que quedarán muchos en el tintero, a continuación se recopilan algunos de los ámbitos donde se aplican.

- ✓ Banca: información de clientes, cuentas, transacciones, préstamos, etc.
- ✓ Líneas aéreas: información de clientes, horarios, vuelos, destinos, etc.
- ✓ Universidades: información de estudiantes, carreras, horarios, materias, etc.
- ✓ Transacciones de tarjeta de crédito: para comprar con tarjetas de crédito y la generación de los extractos mensuales.
- ✓ Telecomunicaciones: para guardar registros de llamadas realizadas, generar facturas mensuales, mantener el saldo de las tarjetas telefónicas de prepago y almacenar información sobre las redes.
- ✓ Medicina: información hospitalaria, biomedicina, genética, etc.
- ✓ Justicia y Seguridad: delincuentes, casos, sentencias, investigaciones, etc.
- ✓ Legislación: normativa, registros, etc.
- ✓ Organismos públicos: datos ciudadanos, certificados, etc.
- ✓ Sistemas de posicionamiento geográfico.
- ✓ Hostelería y turismo: reservas de hotel, vuelos, excursiones, etc.
- ✓ Ocio digital: juegos online, apuestas, etc.
- ✓ Cultura: gestión de bibliotecas, museos virtuales, etc.
- ✓ Etc.

3.3.- Ubicación de la información.

Utilizamos a diario las bases de datos, pero ¿Dónde se encuentra realmente almacenada la información?. Las bases de datos pueden tener un tamaño muy reducido (1 MegaByte o menos) o bien, ser muy voluminosas y complejas (del orden de Terabytes). Sin embargo todas las bases de datos normalmente se almacenan y localizan en discos duros y otros dispositivos de almacenamiento, a los que se accede a través de un ordenador. Una gran base de datos puede necesitar servidores en lugares diferentes, y viceversa, pequeñas bases de datos pueden existir como archivos en el disco duro de un único equipo.

A continuación, se exponen los sistemas de almacenamiento de información más utilizados para el despliegue de bases de datos, comenzaremos por aquellos en los que pueden alojarse bases de datos de tamaño pequeño y mediano, para después analizar los sistemas de alta disponibilidad de grandes servidores.

- ✓ **Discos SATA:** Es una interfaz de transferencia de datos entre la placa base y algunos dispositivos de almacenamiento, como puede ser el disco duro, lectores y regrabadores de CD/DVD/BD, Unidades de Estado Sólido u otros dispositivos. SATA proporciona mayores velocidades, mejor aprovechamiento cuando hay varias unidades, mayor longitud del cable de transmisión de datos y capacidad para conectar unidades al instante, es decir, insertar el dispositivo sin tener que apagar el ordenador. La primera generación especifica en transferencias de 150 Megabytes por

segundo, también conocida por SATA 150 MB/s o Serial ATA-150. Actualmente se comercializan dispositivos SATA II, a 300 MB/s, también conocida como Serial ATA-300 y los SATA III con tasas de transferencias de hasta 600 MB/s.

- ✓ **Discos SCSI:** Son interfaces preparadas para discos duros de gran capacidad de almacenamiento y velocidad de rotación. Se presentan bajo tres especificaciones: SCSI Estándar (Standard SCSI), SCSI Rápido (Fast SCSI) y SCSI Ancho-Rápido (Fast-Wide SCSI). Su tiempo medio de acceso puede llegar a 7 milisegundos y su velocidad de transmisión secuencial de información puede alcanzar teóricamente los 5 MB/s en los discos SCSI Estándares, los 10 MBps en los discos SCSI Rápidos y los 20 MBps en los discos SCSI Anchos-Rápidos (SCSI-2). Un controlador SCSI puede manejar hasta 7 discos duros SCSI.
- ✓ **RAID:** acrónimo de Redundant Array of Independent Disks o matriz de discos independientes, es un contenedor de almacenamiento redundante. **Se basa en el montaje en conjunto de dos o más discos duros**, formando un bloque de trabajo, para obtener desde una ampliación de capacidad a mejoras en velocidad y seguridad de almacenamiento. Según las características que queramos primar, se establecen distintos sistemas de RAID.
- ✓ **Sistemas NAS:** Es el acrónimo de Network Attached Storage ó sistema de almacenamiento masivo en red. Estos sistemas de almacenamiento permiten compartir la capacidad de almacenamiento de un computador (Servidor) con ordenadores personales o servidores clientes a través de una red, haciendo uso de un sistema operativo optimizado para dar acceso a los datos a través de protocolos de comunicación específicos. Suelen ser dispositivos para almacenamiento masivo de datos con capacidades muy altas, de varios Terabytes, generalmente superiores a los discos duros externos y además se diferencian de estos al conectar por red.
- ✓ **Sistemas SAN:** Acrónimo de Storage Area Network o red de área de almacenamiento. Se trata de una red concebida para conectar servidores, matrices (arrays) de discos y librerías de soporte. La arquitectura de este tipo de sistemas permite que los recursos de almacenamiento estén disponibles para varios servidores en una red de área local o amplia. Debido a que la información almacenada no reside directamente en ninguno de los servidores de la red, se optimiza el poder de procesamiento para aplicaciones comerciales y la capacidad de almacenamiento se puede proporcionar en el servidor donde más se necesite.

Puedes ampliar más información sobre algunos de los sistemas de almacenamiento vistos, además de tendencias y curiosidades en almacenamiento, a través de los siguientes enlaces:

<http://es.wikipedia.org/wiki/RAID>
<http://www.ideasgeek.net/2010/05/12/nas-dispositivo-de-almacenamiento-externo-masivo-conectado-por-red/>
<https://tihuilo.wordpress.com/2010/05/27/sistemas-de-almacenamiento/>
<http://www.oracle.com/technetwork/server-storage/general/3d-demos-333955.html>
<http://databaseandtech.wordpress.com/2009/06/22/fusion-tables-google-trae-la-base-de-datos-a-la-nube-del-internet/>
<http://www.cosasquecontar.com/2011/04/bigtable-como-google-almacena-los-datos/>

Rellena los huecos con los conceptos adecuados.

Un tipo de red donde se optimiza el poder de procesamiento para aplicaciones comerciales, pudiendo proporcionarse la capacidad de almacenamiento en el servidor donde más se necesite, se denomina sistema SAN.

En efecto, se trata de una red de área de almacenamiento. Este tipo de tecnología permite conectividad de alta velocidad, de servidor a almacenamiento, almacenamiento a almacenamiento, o servidor a servidor. Este método usa una infraestructura de red por separado, evitando así cualquier problema asociado con la conectividad de las redes existentes.

4.- Modelos de bases de datos.

Caso práctico

Juan tiene ya experiencia con bases de datos: *-Registros, tablas, relaciones, claves,... tiene su teoría, pero dame un problema a resolver y casi puedo construir la base de datos en un abrir y cerrar de ojos.*

María ve como **Ada**, algo escéptica al respecto, aclara a Juan algunas ideas: *-Juan, la experiencia es un grado como siempre hemos destacado, pero es imprescindible conocer y dominar los conceptos más importantes sobre bases de datos. Al igual que comenzar a programar directamente codificando, implementar una base de datos directamente sin detenerse a realizar un análisis previo y emplear las herramientas adecuadas, puede provocar muchos quebraderos de cabeza.*

Ada indica a **María**: *-Las bases de datos no siempre han sido como las conocemos ahora, han habido diferentes modelos para su construcción y es bueno conocer la evolución de éstos para comprender por qué utilizaremos el modelo de bases de datos relacional.*

La clasificación tradicional de las bases de datos establece tres modelos de bases de datos: jerárquico, en red y relacional. En la actualidad el modelo de bases de datos más extendido es el relacional. Aunque, hay que tener en cuenta que dos de sus variantes (modelo de bases de datos distribuidas y orientadas a objetos) son las que se más se están utilizando en los últimos tiempos.

En los siguientes epígrafes analizaremos cada uno de ellos, así como otros modelos de bases de datos existentes.

Conoce las características generales y graba en tu memoria fotográfica los gráficos que representan a cada uno de los modelos expuestos en el siguiente artículo:

<http://es.kioskea.net/contents/bdd/bddtypes.php3>

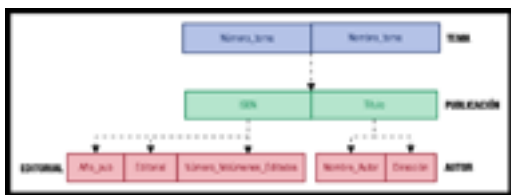
4.1.- Modelo jerárquico.

Cuando IBM creó su Sistema Administrador de Información o IMS, se establecieron las bases para que la gran mayoría de sistemas de gestión de información de los años setenta utilizaran el modelo jerárquico. También recibe el nombre de modelo en árbol, ya que utiliza una estructura en árbol invertido para la organización de los datos.

La información se organiza con una jerarquía en la que la relación entre las entidades de este modelo siempre es del tipo padre/hijo. De tal manera que existen nodos que contienen atributos o campos y que se relacionarán con sus nodos hijos, pudiendo tener cada nodo más de un hijo, pero un nodo siempre tendrá un sólo padre.

Los datos de este modelo se almacenan en estructuras lógicas llamadas segmentos. Los segmentos se relacionan entre sí utilizando arcos. La forma visual de este modelo es de árbol invertido, en la parte superior están los padres y en la inferior los hijos.

Hoy en día, debido a sus limitaciones, el modelo jerárquico está en desuso. En el siguiente gráfico puedes observar la estructura de almacenamiento del modelo jerárquico.



Si deseas completar tus conocimientos acerca de este modelo, te proponemos los siguientes enlaces:

http://es.wikipedia.org/wiki/Modelo_jer%C3%A1rquico

http://ddd.uab.cat/pub/elies/elies_a2000v9/4-2-1.htm

Rellena los huecos con los conceptos adecuados.

El modelo Jerárquico es un modelo muy rígido en el que las diferentes entidades se organizan en niveles múltiples, de acuerdo a una estricta relación **padre/hijo**, de manera que un **padre** puede tener más de un **hijo**, todos ellos localizados en el mismo nivel, y un **hijo** únicamente puede tener un **padre** situado en el nivel inmediatamente superior al suyo.

Como puedes ver en el gráfico anterior, la estructura representa las relaciones padre/hijo y las despliega en forma de árbol invertido. De esta manera un padre tiene un hijo o varios, y un hijo sólo podrá tener un padre.

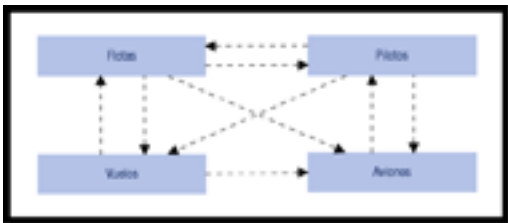
4.2.- Modelo en red.

El modelo de datos en red aparece a mediados de los sesenta como respuesta a limitaciones del modelo jerárquico en cuanto a representación de relaciones más complejas. Podemos considerar a IDS (Integrated Data Store) de Bachman como el primer sistema de base de datos en red. Tras él se intentó crear un estándar de modelo de red por parte de CODASYL, siendo un modelo que tubo gran aceptación a principios de los setenta.

El modelo en red organiza la información en registros (también llamados nodos) y enlaces. En los registros se almacenan los datos, mientras que los enlaces permiten relacionar estos datos. Las bases de datos en red son parecidas a las jerárquicas sólo que en ellas puede haber más de un padre.

En este modelo se pueden representar perfectamente cualquier tipo de relación entre los datos, pero hace muy complicado su manejo. Al no tener que duplicar la información se ahorra espacio de almacenamiento.

El sistema de gestión de información basado en el modelo en red más popular es el sistema IDMS.



Si deseas completar tus conocimientos acerca de este modelo, te proponemos los siguientes enlaces:

http://es.wikipedia.org/wiki/Modelo_de_red

http://ddd.uab.cat/pub/elies/elies_a2000v9/4-2-2.htm

4.3- Modelo relacional.

Este modelo es posterior a los dos anteriores y fue desarrollado por Codd en 1970. Hoy en día las bases de datos relacionales son las más utilizadas.

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

El modelo relacional utiliza **tablas bidimensionales** (relaciones) para la representación lógica de los datos y las relaciones entre ellos. Cada relación (tabla) posee un nombre que es único y contiene un conjunto de columnas.

Se llamará **registro, entidad o tupla** a cada fila de la tabla y **campo o atributo** a cada columna de la tabla.

A los conjuntos de valores que puede tomar un determinado atributo, se le denomina **dominio**.

Una **clave** será un atributo o conjunto de atributos que identifique de forma única a una tupla.

Las tablas deben cumplir una serie de requisitos:

- ✓ Todos los registros son del mismo tipo.
- ✓ La tabla sólo puede tener un tipo de registro.
- ✓ No existen campos o atributos repetidos.
- ✓ No existen registros duplicados.
- ✓ No existe orden en el almacenamiento de los registros.
- ✓ Cada registro o tupla es identificada por una clave que puede estar formada por uno o varios campos o atributos.

A continuación puedes observar cómo es una relación con sus tuplas y atributos en el modelo relacional.



El lenguaje habitual para construir las consultas a bases de datos relacionales es SQL, Structured Query Language o Lenguaje Estructurado de Consultas, un estándar implementado por los principales motores o sistemas de gestión de bases de datos relacionales.

Durante su diseño, una base de datos relacional pasa por un proceso al que se conoce como normalización de una base de datos.

Si deseas completar tus conocimientos acerca de este modelo, te proponemos el siguiente enlace:

http://es.wikipedia.org/wiki/Modelo_relacional

Rellena los huecos con los conceptos adecuados.

La **normalización** de bases de datos relacional consiste en definir las reglas que determinan las dependencias entre los datos de una base de datos relacional. Si definimos esta relación o dependencia entre los elementos de una determinada base de datos de la manera más sencilla posible, conseguiremos que la cantidad de espacio necesario para guardar los datos sea el menor posible y la facilidad para actualizar la relación sea la mayor posible. Es decir, optimizaremos su funcionamiento.

El proceso de normalización tiene una gran relevancia en el diseño de bases de datos. En futuras unidades de trabajo se abordarán las técnicas para llevar a cabo esta optimización.

4.4.- Modelo orientado a objetos.

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. Las operaciones de cada clase se especifican en términos de procedimientos predefinidos denominados **métodos**. Algunos sistemas existentes en el mercado, basados en el modelo relacional, han sufrido evoluciones incorporando conceptos orientados a objetos. A estos modelos se les conoce como sistemas **objeto-relacionales**.

El objetivo del modelo orientado a objetos es cubrir las limitaciones del modelo relacional. Gracias a este modelo se incorporan mejoras como la herencia entre tablas, los tipos definidos por el usuario, disparadores almacenables en la base de datos (triggers), soporte multimedia, etc.

Los conceptos más importantes del paradigma de objetos que el modelo orientado a objetos incorpora son:

- ✓ **Encapsulación** - Propiedad que permite ocultar la información al resto de los objetos, impidiendo así accesos incorrectos o conflictos.
- ✓ **Herencia** - Propiedad a través de la cual los objetos heredan comportamiento dentro de una jerarquía de clases.
- ✓ **Polimorfismo** - Propiedad de una operación mediante la cual puede ser aplicada a distintos tipos de objetos.

Desde la aparición de la programación orientada a objetos (POO u OOP) se empezó a pensar en bases de datos adaptadas a estos lenguajes. Este modelo es considerado como el fundamento de las bases de datos de tercera generación, siendo consideradas las bases de datos en red como la primera y las bases de datos relacionales como la segunda generación. Aunque no han reemplazado a las bases de datos relacionales, si son el tipo de base de datos que más está creciendo en los últimos años.

Si deseas completar tus conocimientos acerca de este modelo, te proponemos el siguiente enlace:

http://es.wikipedia.org/wiki/Base_de_datos_orientada_a_objetos

4.5.- Otros modelos.

Además de los modelos clásicos vistos hasta el momento, vamos a detallar a continuación las particularidades de otros modelos de bases de datos existentes y que, en algunos casos, son una evolución de los clásicos.

Modelo Objeto-Relacional

Las bases de datos pertenecientes a este modelo, son un híbrido entre las bases del modelo relacional y el orientado a objetos. El mayor inconveniente de las bases de datos orientadas a objetos radica en los costes de la conversión de las bases de datos relacionales a bases de datos orientadas a objetos.

En una base de datos objeto-relacional (BDOR) siempre se busca obtener lo mejor del modelo relacional, incorporando las mejoras ofrecidas por la orientación a objetos. En este modelo se siguen almacenando tuplas, aunque la estructura de las tuplas no está restringida sino que las relaciones pueden ser definidas en función de otras, que es lo que denominamos herencia directa.

El estándar en el que se basa este modelo es [SQL99](#). Este estándar ofrece la posibilidad de añadir a las bases de datos relacionales procedimientos almacenados de usuario, triggers, tipos definidos por el usuario, consultas recursivas, bases de datos [OLAP](#), tipos [LOB](#), ...

Otra característica a destacar es la capacidad para incorporar funciones que tengan un código en algún lenguaje de programación como por ejemplo: SQL, Java, C, etc.

La gran mayoría de las bases de datos relacionales clásicas de gran tamaño, como Oracle, SQL Server, etc., son objeto-relacionales.

Modelo de bases de datos deductivas

En este modelo las bases de datos almacenan la información y permiten realizar deducciones a través de [inferencias](#). Es decir, se derivan nuevas informaciones a partir de las que se han introducido explícitamente en la base de datos por parte del usuario.

Las bases de datos deductivas son también llamadas bases de datos lógicas, al basarse en lógica matemática. Surgieron para contrarrestar las limitaciones del modelo relacional para la respuesta a consultas [recursivas](#) y la deducción de relaciones indirectas entre los datos almacenados.

Si deseas completar tus conocimientos sobre las bases de datos deductivas, te proponemos el siguiente enlace:

La Lógica en el desarrollo de las Bases de Datos

Matilde Celma Giménez

Departamento de Sistemas Informáticos y Computación / Universidad Politécnica de Valencia

1. Lógica y Bases de Datos

La idea básica que subyace al uso de la lógica para el estudio de los sistemas de bases de datos es una idea común a todos los campos de la computación lógica: "la semántica por [teoría de modelos](#) de la lógica proporciona una base para la representación del conocimiento, y la semántica por [teoría de la demostración](#) proporciona una base para la computación" [J.W. Lloyd, en *Computational Logic*, 1990].

1. Lógica y Bases de Datos.

La lógica de primer orden ha sido utilizada en el desarrollo del [modelo relacional de datos](#) desde su aparición en 1970.

Problemas:

- formalización
- definición de lenguajes de consulta
- estudio del concepto de independencia del dominio
- actualización de vistas
- comprobación y restauración de la integridad.
- optimización de consultas
- diseño de bases de datos

2. Bases de datos deductivas.



Las Bases de Datos Deductivas extienden la capacidad expresiva de las bases de datos relacionales incluyendo un conjunto de reglas que permiten definir conocimiento implícito

2. Bases de datos deductivas.

Hechos = {tuplas de relaciones} (conocimiento explícito)

Reglas = {reglas deductivas} (conocimiento implícito)

ESQUEMA

Relaciones básicas:
 $R_i(A_{i1}, D_{i1}, A_{i2}, D_{i2}, \dots, A_{in}, D_{in})$
 $(1 \leq i \leq m)$ (m relaciones básicas)

Relaciones derivadas:
 $S_j(A_{j1}, D_{j1}, A_{j2}, D_{j2}, \dots, A_{jn}, D_{jn})$
 $(1 \leq j \leq s)$ (s relaciones derivadas)

Restricciones de Integridad
 $W_k; W_k$ es una expresión lógica
 $(1 \leq k \leq k)$ (k restricciones de integridad)

BASE DE DATOS

Relaciones básicas:

R_i	A_{i1}	A_{i2}	...	A_{in}

$R_i \subseteq (D_{i1} \times D_{i2} \times \dots \times D_{in})$
 $(1 \leq i \leq m)$ (m relaciones básicas)

Relaciones derivadas:
 $S_{ij}(x_1, x_2, \dots, x_{n_j}) \leftarrow W_{ij}$
 $(1 \leq i \leq s)$ (s relaciones derivadas)
 $(1 \leq j \leq K_i)$ (K_i reglas para la relación S_i)

2. Bases de datos deductivas

Relaciones básicas:

PIEZA (codpieza: D1, desc: D2, peso: D3)
 CP = (codpieza)

PROV (codprov: D4, nombre: D5, zona: D6)
 CP = (codprov)

PRECIOS (codprov: D4, codpieza: D1, precio: D7)
 CP = (codprov, codpieza)
 CAj = (codprov) → PROV
 CAj = (codpieza) → PIEZA

COMP (pieza1: D1, pieza2: D1)
 CP = (pieza1, pieza2)
 CAj = (pieza1) → PIEZA
 CAj = (pieza2) → PIEZA

Relaciones derivadas:

PRECIO3 (codprov: D4, codpieza: D1, precio: D7)
 CP = (codprov, codpieza)
 CAj = (codprov) → PROV
 CAj = (codpieza) → PIEZA

PRECIOS_EXT (codprov: D4, nombre: D5, codpieza: D1, desc: D2, precio: D7)
 CP = (codprov, codpieza)
 CAj = (codprov) → PROV
 CAj = (codpieza) → PIEZA

COMPONENTE (pieza1: D1, pieza2: D1)
 CP = (pieza1, pieza2)
 CAj = (pieza1) → PIEZA
 CAj = (pieza2) → PIEZA

Restricciones de integridad:
 $tx \neq y \mid \text{COMPONENTE}(x,y) \rightarrow \neg \text{COMPONENTE}(y,x)$

Esquema

2. Bases de datos deductivas

BDD

codprov	nombre	zona
pv1	Juan...	1
pv2	Carlos...	2
pv3	Luis...	3

codprov	nombre	zona
pv1	Juan...	1
pv2	Carlos...	2
pv3	Luis...	3

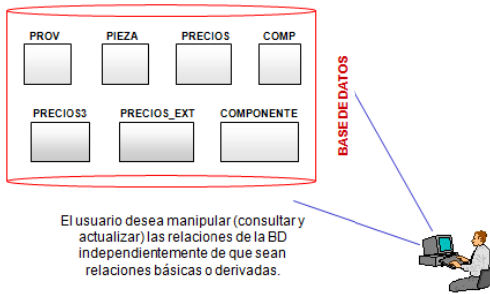
codprov	codpieza	precio
pv1	pz1	10
pv1	pz2	20
pv2	pz1	30
pv3	pz1	30

pieza1	pieza2
pv1	pv2
pv2	pv3

Reglas deductivas:

- $\text{precios3}(x, y, z) \leftarrow \exists z (\text{precios}(x, y, z) \wedge \text{prov}(x, w, 3))$
- $\text{componente}(x, y) \leftarrow \exists z (\text{comp}(x, z) \wedge \text{componente}(z, y))$
- $\text{componente}(x, y) \leftarrow \text{comp}(x, y)$
- $\text{precios_ext}(x, n, y, d, p) \leftarrow \exists z \exists w (\text{prov}(x, n, z) \wedge \text{pieza}(y, d, w) \wedge \text{precios}(x, y, p))$

2. Bases de datos deductivas



2. Bases de datos deductivas

- Mecanismo de **vistas** del modelo relacional → Definición de información implícita
- Relación derivada → VISTA
- Base de datos deductiva → Base de datos relacional con vistas

2. Bases de datos deductivas

Limitaciones del modelo relacional (SQL92):

- Definición de vistas → Limitaciones en la definición de vistas recursivas
- Actualización → Limitaciones en la actualización de las vistas
- SGBD relacionales → Ausencia de procedimientos para la evaluación de consultas recursivas

2. Bases de datos deductivas.

Los sistemas de gestión de bases de datos deductivas deben superar las limitaciones de los sistemas relacionales

PROBLEMAS:

- ✓ Formalización
- ✓ Actualización de la base de datos → **LÓGICA**
- ✓ Construcción de SGBD deductivos

2. Bases de datos deductivas.

Formalización: Si intentamos representar la información explícita y la información implícita en un mismo lenguaje (lenguaje de 1º orden) obtenemos un programa lógico:

- Base de datos deductiva Hechos
 - pieza (pz1, tornillo, 10)
 - ...
 - prov (pv1, Juan, 1)
 - ...
 - comp (pz1, pz3)
 - ...
- Reglas deductivas
 - $\text{precios3}(x, y, z) \leftarrow \exists w (\text{prov}(x, w, 3) \wedge \text{precios}(x, y, z))$
 - $\text{componente}(x, y) \leftarrow \exists z (\text{comp}(x, z) \wedge \text{componente}(z, y))$
 - $\text{componente}(x, y) \leftarrow \text{comp}(x, y)$
 - $\text{precios_ext}(x, n, y, d, p) \leftarrow \exists z \exists w (\text{prov}(x, n, z) \wedge \text{pieza}(y, d, w) \wedge \text{precios}(x, y, p))$

2. Bases de datos deductivas

MARCO FORMAL: Lógica de 1º orden (Programación Lógica)

Esquema de BDD:

- (L, RI): - L es un lenguaje de 1º orden
- RI es un conjunto de f.b.f de L (restricciones de integridad)

BDD: (programa lógico)

- {A: A es un átomo base} (hechos)
- {A ← L₁ ∧ L₂ ∧ ... ∧ L_n: A es un átomo y L_i es un literal} (reglas)

2. Bases de datos deductivas

Semántica de una BDD

Semántica de una BDD: definir el conocimiento existente en la base de datos.

¿qué es cierto en la BDD?

- Semántica declarativa: conocimiento en la BDD
- Semántica operacional: procedimiento para obtener el conocimiento

2. Bases de datos deductivas

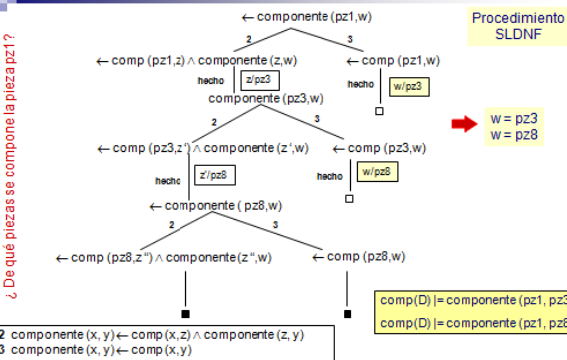
Semántica de una BDD

Programación lógica: semántica operacional: SLDNF
semántica declarativa: comp(D)

Semántica operacional: procedimiento SLDNF

- SLDNF: - procedimiento de refutación
- reglas de inferencia:
 - resolución
 - negación como fallo

Semántica declarativa asociada al SLDNF: completación de D

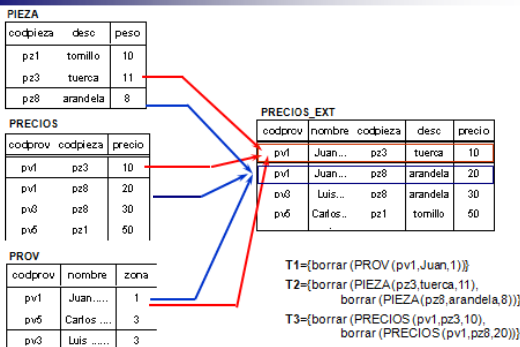


3. Actualización de base de datos deductivas



"Dada una base de datos D (D=BDI ∪ BDE) y un requisito de actualización insertar (A) (resp. borrar (A)) donde A es una tupla de una relación derivada, encontrar una transacción T sobre EDB tal que T(D) satisfaga el requisito de actualización"

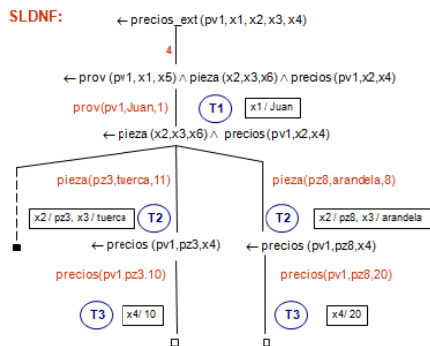
Ejemplo: DELETE FROM PRECIOS3 WHERE codprov=pv1



3. Actualización de bases de datos deductivas

Métodos para la actualización de bases de datos deductivas

Utilización de los procedimientos de evaluación de consultas para determinar los posibles caminos de derivación del conocimiento que se desea a actualizar



3.1 Actualización

Enunciado general del problema:

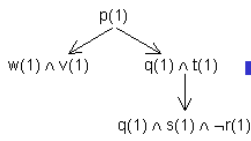
Dados el esquema (L,RI) de una base de datos deductiva, un estado de base de datos D de ese esquema tal que $\forall W \in RI$ se cumple que D *satisface* W, y dado un requisito de actualización U tal que U *no es cierto* en D entonces encontrar una transacción T tal que $\forall W \in RI, D' = T(D)$ *satisface* W y U *es cierto* en D'.

3.1 Actualización

Ejemplo 1

- 1. $p(x) \leftarrow q(x) \wedge t(x)$
- 2. $p(x) \leftarrow w(x) \wedge v(x)$
- 3. $t(x) \leftarrow s(x) \wedge \neg r(x)$

Actualización: $U = p(1)$



- 1) $\{w(1), v(1)\} \subseteq BDE$
- 2) $\{q(1), s(1)\} \subseteq BDE$ y $\{r(1)\} \notin BDE$
- 3) $\{p(1)\} \subseteq BDE$
- 4) $\{q(1), t(1)\} \subseteq BDE$

Obtener transacciones que aseguren una de estas cuatro situaciones

3.1 Actualización

Caracterización del problema:

- 1) Tiempo de generación de la solución.
- 2) Variables cuantificadas existencialmente
- 3) Recursividad
- 4) Información asumida
- 5) Tratamiento de restricciones de integridad

3.1 Actualización

1) Tiempo de generación de la solución.

- > **Tiempo de ejecución:** el árbol de derivación para el requisito de actualización se genera cuando la actualización es solicitada.
- > **Tiempo de definición:** el árbol de derivación para un requisito de actualización se estudia cuando se define el esquema de la base de datos, lo que supone una mejora ya que determinadas tareas sólo se realizan una vez.
- > **Mixto:** en este caso una parte de la solución se genera en tiempo de definición del esquema y se completa en tiempo de ejecución.

3.1 Actualización

En el **Ejemplo 1:**

- > un método que obtuviese la solución en tiempo de ejecución estudiaría el árbol de derivación de la actualización $p(1)$ para encontrar una solución.
- > un método que trabajase en tiempo de definición del esquema estudiaría el requisito genérico $p(x)$ para obtener soluciones que luego se instanciarían en tiempo de ejecución.

3.1 Actualización

2) Variables existencialmente cuantificadas.

Dada una regla deductiva de una base de datos normal, a las variables que aparecen en el cuerpo de la regla y no aparecen en la cabeza se les denomina **variables existencialmente cuantificadas**.

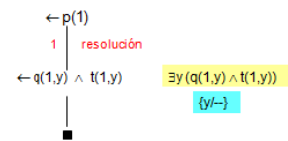
$$\forall X_1 \dots \forall X_i \dots \forall X_m (A \leftarrow L_1 \wedge \dots \wedge L_n) \equiv (x_i \text{ no aparece en } A) \forall X_1 \dots \forall X_{i-1} \forall X_{i+1} \dots \forall X_m (A \leftarrow \exists X_i (L_1 \wedge \dots \wedge L_n))$$

La presencia de variables existencialmente cuantificadas en las reglas deductivas puede provocar la aparición del problema llamado **falta de valores** durante la generación de las transacciones que resuelven un requisito de actualización.

3.1 Actualización

Ejemplo 2:

BDD: 1. $p(x) \leftarrow q(x,y) \wedge t(x,y)$
.....
t(1,2)



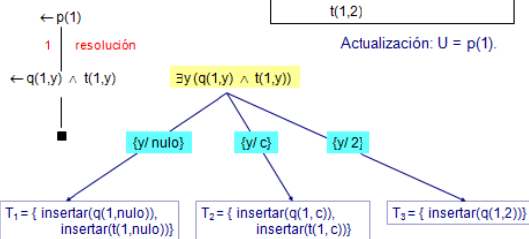
Actualización: $U = p(1)$.

Una solución sencilla a este problema consiste en utilizar el valor nulo para la variable de la que se desconoce el valor o bien un valor cualquiera proporcionado por el usuario o extraído sin criterio de la base de datos. Aunque en ocasiones esta solución es la única posible, en otras se puede elegir un valor tal que la transacción obtenida sea más sencilla.

3.1 Actualización

Ejemplo 2:

BDD: 1. $p(x) \leftarrow q(x,y) \wedge t(x,y)$
.....
t(1,2)



3.1 Actualización

3) Recursividad.

La presencia de reglas recursivas en la base de datos puede complicar la generación de la transacción, ya que el árbol de derivación puede ser infinito para un determinado requisito de actualización, lo que supone la existencia de infinitas transacciones posibles para satisfacerlo.

3.1 Actualización

Ejemplo 3:

BDD: 1. $p(x,y) \leftarrow q(x,y)$
2. $p(x,y) \leftarrow q(x,z) \wedge p(z,y)$

Actualización: $U = p(1,1)$.

Para satisfacer este requisito hay infinitas transacciones posibles:

- $T_1: \{ \text{insertar}(q(1,1)) \}$
- $T_2: \{ \text{insertar}(q(1,2)), \text{insertar}(q(2,1)) \}$
- $T_3: \{ \text{insertar}(q(1,2)), \text{insertar}(q(2,3)), \text{insertar}(q(3,1)) \}$
- ...

3.1 Actualización

4) Problema de la información asumida.

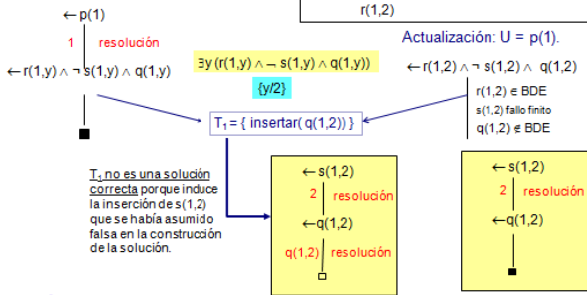
En presencia de negación en las reglas deductivas, es posible que algunas soluciones que podrían parecer correctas no lo sean, ya que alguna información que se ha supuesto cierta (resp. falsa), durante la construcción de la solución pase a ser falsa (resp. cierta) debido a las actualizaciones propuestas más adelante.

3.1 Actualización

Ejemplo 4:

BDD: 1. $p(x) \leftarrow r(x,y) \wedge \neg s(x,y) \wedge q(x,y)$
2. $s(1,2) \leftarrow q(1,2)$
 $r(1,2)$

Actualización: $U = p(1)$.



3.1 Actualización

5) Tratamiento de las restricciones de integridad.

La solución propuesta para un requisito de actualización puede suponer la violación de alguna restricción de integridad por lo que es interesante estudiar cómo integra cada método, si lo hace, su estrategia con la comprobación de las restricciones del esquema.

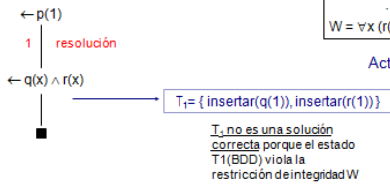
3.1 Actualización

Ejemplo 5:

BDD: 1. $p(x) \leftarrow q(x) \wedge r(x)$
...

$W = \forall x (r(x) \rightarrow t(x))$

Actualización: $U = p(1)$.



Un método que integre la generación de las soluciones con la restauración de la integridad podría generar la transacción {insertar(q(1)), insertar(r(1)), insertar(t(1))}.

3.1 Actualización

Estudio de un método de actualización:

1. Semántica asumida: la semántica declarativa determina el conjunto de posibles soluciones al requisito de actualización y la semántica operacional constituye la herramienta para computar esas soluciones.
2. Bases de datos: tipo de bases de datos y de restricciones de integridad para los que está definido el método.
3. Requisitos de actualización: forma sintáctica de los requisitos de actualización permitidos en el método.
4. Transacciones generadas: tipo de soluciones obtenidas y si sólo se obtiene una o todas las soluciones posibles.
5. Descripción del método: estrategia seguida para generar las transacciones.
6. Corrección y completitud del método.
7. Resumen: cómo el método resuelve los problemas antes mencionados.

La Lógica en el desarrollo de las Bases de Datos

1. Lógica y Bases de Datos.
2. Bases de datos deductivas.
3. Actualización de bases de datos deductivas.
 - 3.1 Actualización
 - 3.2 Comprobación de la integridad
 - 3.3 Restauración de la consistencia

3.2 Comprobación de la integridad

Restricción de integridad: propiedad del mundo real que una base de datos debe *satisfacer en cualquier instante* para ser consistente con cierto modelo del mundo real.

$$D_0 \xrightarrow{T_1} D_1 \xrightarrow{T_2} D_2 \xrightarrow{T_3} D_3 \xrightarrow{T_4} D_4 \xrightarrow{T_5} D_5 \xrightarrow{T_6} D_6 \xrightarrow{T_7} D_7 \xrightarrow{T_8} D_8 \xrightarrow{T_9} D_9 \xrightarrow{T_{10}} D_{10}$$

Evolución de una BD

Restricciones estáticas: hacen referencia a un único estado de la base de datos. Estas restricciones restringen los estados válidos con independencia de la secuencia de los mismos.

Restricciones dinámicas: hacen referencia a dos o más estados de la base de datos. Estas restricciones restringen las secuencias de estados válidas. Un caso particular de restricciones dinámicas son las restricciones de transición que restringen dos estados consecutivos válidos.

3.2 Comprobación de la integridad

Comprobación de la integridad: la comprobación de la integridad en bases de datos consiste en *comprobar* si el par de estados (D,D') implicados en una transacción T *satisface* las restricciones de transición y si el estado final D' *satisface* las restricciones estáticas.

Método de comprobación de la integridad: es un procedimiento de decisión tal que, dado un estado D y una restricción de integridad estática W, decide con una respuesta binaria si/no si el estado D *satisface/viola* la restricción estática W.

3.2 Comprobación de la integridad

La forma más sencilla de comprobar las restricciones estáticas es evaluar cada una de ellas después de la transacción; sin embargo esta aproximación puede ser muy costosa en bases de datos voluminosas.

La comprobación de la integridad podría simplificarse si consideráramos sólo los "cambios" que la transacción ha producido en la base de datos.

Todos los métodos propuestos para simplificar la comprobación de la integridad suponen que la base de datos era íntegra antes de la transacción. Apoyándose en esta hipótesis, los métodos comprueban sólo instancias de las restricciones generadas a partir de las actualizaciones (inserciones y borrados) de la transacción, evitando comprobar instancias que ya se satisfacían antes de la transacción y que además no se ven afectadas por ésta.

3.2 Comprobación de la integridad

Comprobación simplificada de la integridad en bases de datos relacionales:

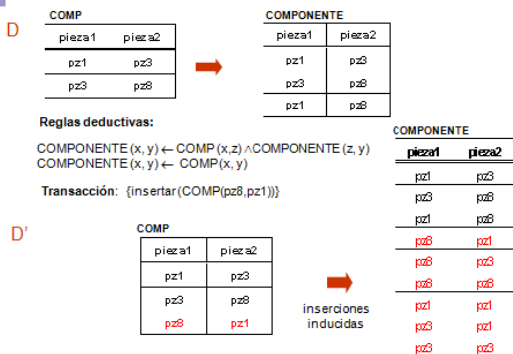
PRECIOS (codprov: D4, codpieza: D1, precio: D7)
 CP = {codprov, codpieza}
 CAj = {codprov} → PROV
 CAj = {codpieza} → PIEZA
 $W: \forall x \forall y \forall z (\text{precios}(x, y, z) \rightarrow \exists w \exists t (\text{prov}(x,w,t)))$
 T = { insertar (PRECIOS (pv11,pz3,100)) }
 D → T → D'

Si D es un estado íntegro entonces D' *satisface* W si y sólo si D' *satisface* $\exists w \exists t (\text{prov}(pv11,w,t))$

3.2 Comprobación de la integridad

Comprobación simplificada de la integridad en bases de datos deductivas:

En bases de datos deductivas las actualizaciones generadas por una transacción no son sólo las explícitamente requeridas por ésta (operaciones que la componen) sino también todas las actualizaciones que se pueden inducir por la presencia de reglas deductivas en la base de datos.



3.2 Comprobación de la integridad

Restricción de integridad:

$W: \forall x \forall y (\text{COMPONENTE} (x,y) \rightarrow \neg \text{COMPONENTE} (y,x))$
 insertar (COMPONENTE(pz8,pz1))
 ↓
 ¬COMPONENTE (pz1, pz8)

Si D es un estado íntegro entonces D' *satisface* W si y sólo si D' *satisface* ¬COMPONENTE (pz1, pz8) → D' *viola* W

3.2 Comprobación de la integridad

Enunciado general del problema:

Dados, el esquema (L,R) de una base de datos deductiva, un estado D de ese esquema tal que D *satisface* W (para toda W ∈ R), una transacción T formada por dos conjuntos de sentencias de base de datos, T = T_{ins} ∪ T_{del}, (T_{ins} ∩ T_{del} = ∅, T_{del} ⊆ D y T_{ins} ∩ D = ∅), el estado D' resultante de aplicar a D la transacción T, (D' = (D ∪ T_{ins}) \ T_{del}), comprobar que D' *satisface* W (para toda W ∈ R).

3.2 Comprobación de la integridad

Caracterización del problema:

- 1) Concepto de satisfacción
- 2) Corrección y completitud de un método
- 2) Fases en la comprobación simplificada de la integridad

3.2 Comprobación de la integridad

1) Concepto de satisfacción:

a) Punto de vista de la demostración:

D satisface W sii $Tr \models W$.

b) Punto de vista de la consistencia:

D satisface W sii $Tr \cup \{W\}$ es consistente.

El concepto de violación se define en términos del concepto de satisfacción:

D viola W sii no (D satisface W).

Diremos que un estado D es íntegro si, para toda restricción W perteneciente a RI, D satisface W.

3.2 Comprobación de la integridad

Ejemplo 1:

$$D = \{ p(a) \leftarrow q(x) \wedge \neg r(x), \\ q(a), \\ r(x) \leftarrow r(x) \}.$$

Si asumimos la semántica de la completión:

$$Tr = \text{comp}(D) = \{ \forall y (p(y) \leftrightarrow y = a \wedge \exists x (q(x) \wedge \neg r(x))), \\ \forall x (q(x) \leftrightarrow x = a), \\ \forall x (r(x) \leftrightarrow r(x)) \}.$$

Desde el punto de vista de la consistencia D satisface: $W_1 = q(a), W_2 = r(a), W_3 = p(a), W_4 = \neg r(a)$.

Desde el punto de vista de la demostración D satisface: $W_1 = q(a)$.

3.2 Comprobación de la integridad

Si asumimos la semántica del punto fijo iterado: $M_D = \{q(a), p(a)\}$

$$Tr(M_D) = \{ \forall x (p(x) \leftrightarrow x = a), \\ \forall x (q(x) \leftrightarrow x = a), \\ \forall x (\neg r(x)) \}.$$

D satisface $W_1 = q(a)$ y $W_2 = p(a)$ en los dos conceptos de satisfacción.

3.2 Comprobación de la integridad

2) Corrección y completitud de un método:

Sean:

$\checkmark M$ un método de comprobación de la integridad. D satisface_M (resp. viola_M) W significa que el método M decide que el estado D satisface (resp. viola) la restricción W ($W \in RI$).

$\checkmark CS$ el concepto de satisfacción asumido por el método M. D satisface_{CS} (resp. viola_{CS}) W significa que el estado D satisface (resp. viola) la restricción W ($W \in RI$) en el concepto de satisfacción CS.

Un método M es correcto cuando se cumple:

- si D satisface_M W entonces D satisface_{CS} W (correcto para satisfacción)
- si D viola_M W entonces D viola_{CS} W (correcto para violación).

Un método M es completo cuando se cumple:

- si D satisface_{CS} W entonces D satisface_M W (completo para satisfacción)
- si D viola_{CS} W entonces D viola_M W (completo para violación).

3.2 Comprobación de la integridad

3) Fases en la comprobación simplificada de la integridad

Hipótesis: D es íntegro.

Comprobación de la integridad:

FASE I: Fase de Generación

- Paso 1: Cálculo del conjuntos de literales que "representan" la diferencia entre los estados consecutivos D y D'.
- Paso 2: Identificación de las restricciones relevantes.
- Paso 3: Instanciación de las restricciones relevantes.
- Paso 4: Simplificación de las instancias de las restricciones relevantes.

FASE II: Fase de Evaluación

- Paso 5: Comprobación en D' de las instancias simplificadas de las restricciones relevantes.

3.2 Comprobación de la integridad

3) Fases en la comprobación simplificada de la integridad

Ejemplo 2:

$$D = \{ p(x) \leftarrow q(x,y) \wedge \neg t(y), \\ t(y) \leftarrow s(y), \\ q(a,b), q(b,a), s(c) \}$$

$$D' = \{ p(x) \leftarrow q(x,y) \wedge \neg t(y), \\ t(y) \leftarrow s(y), \\ t(y) \leftarrow q(y,z) \wedge t(z) \\ q(a,b), q(b,a), s(c), s(b) \}$$

$$T = \{ \text{ins}(s(b), \text{ins}(t(y) \leftarrow q(y,z) \wedge t(z))) \}$$

$$W_1 = \forall x \forall y (q(x,y) \rightarrow q(y,x))$$

$$W_2 = \forall x (s(x) \rightarrow p(x))$$

Fase de Generación

Paso 1: Cálculo del conjuntos de literales que "representan" la diferencia entre los estados consecutivos D y D'.

$$D = \{ p(x) \leftarrow q(x,y) \wedge \neg t(y), \\ t(y) \leftarrow s(y), \\ q(a,b), q(b,a), s(c) \} \quad \xrightarrow{T} \quad D' = \{ p(x) \leftarrow q(x,y) \wedge \neg t(y), \\ t(y) \leftarrow s(y), \\ t(y) \leftarrow q(y,z) \wedge t(z) \\ q(a,b), q(b,a), s(c), s(b) \}$$

$$T = \{ \text{ins}(s(b), \text{ins}(t(y) \leftarrow q(y,z) \wedge t(z))) \}$$

Actualizaciones inducidas por T:

Inserciones = $\{A: A \in B_D, \text{comp}(D') \models A \wedge \text{comp}(D) \not\models A\} = \{s(b), t(a), t(b)\}$

Borrados = $\{A: A \in B_D, \text{comp}(D) \models A \wedge \text{comp}(D') \not\models A\} = \{p(a), p(b)\}$

Se asume la semántica de la completión y el punto de vista de la demostración.

Fase de Generación

Paso 2: Identificación de restricciones relevantes

$$D = \{ p(x) \leftarrow q(x,y) \wedge \neg t(y), \\ t(y) \leftarrow s(y), \\ q(a,b), q(b,a), s(c) \} \quad \xrightarrow{T} \quad D' = \{ p(x) \leftarrow q(x,y) \wedge \neg t(y), \\ t(y) \leftarrow s(y), \\ t(y) \leftarrow q(y,z) \wedge t(z) \\ q(a,b), q(b,a), s(c), s(b) \}$$

$$T = \{ \text{ins}(s(b), \text{ins}(t(y) \leftarrow q(y,z) \wedge t(z))) \}$$

Inserciones = $\{s(b), t(a), t(b)\}$ \swarrow $W_1 = \forall x \forall y (q(x,y) \rightarrow q(y,x))$ no es relevante para T

Borrados = $\{p(a), p(b)\}$ \searrow $W_2 = \forall x (s(x) \rightarrow p(x))$ es relevante para T

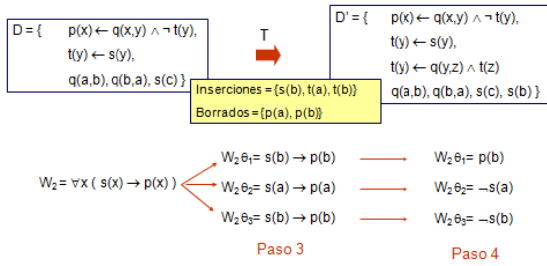
Paso 2

$W_2 = \forall x (s(x) \rightarrow p(x))$ es relevante para T

Fase de Generación

Paso 3: Instanciación de las restricciones relevantes

Paso 4: Simplificación de las instancias de las restricciones relevantes



3.2 Comprobación de la integridad

3) Fases en la comprobación simplificada de la integridad

Hipótesis: D es íntegro.

Comprobación de la integridad:

FASE I: Fase de Generación

- Paso 1: Cálculo del conjunto de literales que "capturan" la diferencia entre los estados consecutivos D y D'.
- Paso 2: Identificación de las restricciones relevantes.
- Paso 3: Instanciación de las restricciones relevantes.
- Paso 4: Simplificación de las instancias de las restricciones relevantes.

FASE II: Fase de Evaluación

- Paso 5: Comprobación en D' de las instancias simplificadas de las restricciones relevantes.

3.2 Comprobación de la integridad

3) Fases en la comprobación simplificada de la integridad

Hipótesis: D es íntegro.

Comprobación de la integridad:

Solución

FASE I: Fase de Generación Potencial

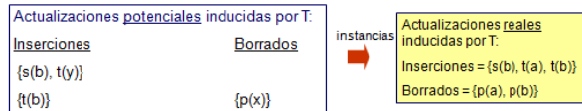
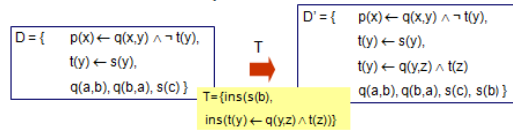
- Paso 1: Cálculo del conjunto de literales que "capturan" la diferencia entre los estados consecutivos D y D' sin acceder a la BDE.
- Paso 2: Identificación de las restricciones relevantes.
- Paso 3: Instanciación de las restricciones relevantes.
- Paso 4: Simplificación de las instancias de las restricciones relevantes.

FASE II: Fase de Evaluación

- Paso 5: Comprobación en D' de las instancias simplificadas de las restricciones relevantes.

Fase de Generación Potencial

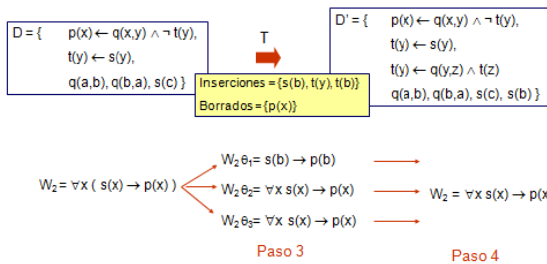
Paso 1: Cálculo del conjunto de literales que "capturan" la diferencia entre los estados consecutivos D y D'.



Fase de Generación Potencial

Paso 3: Instanciación de las restricciones relevantes

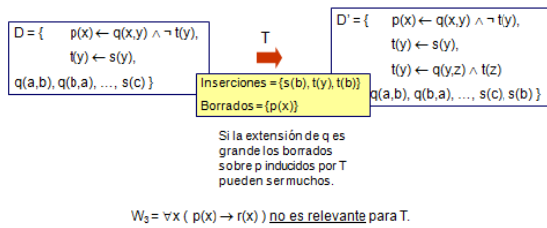
Paso 4: Simplificación de las instancias de las restricciones relevantes



Fase de Generación Potencial

Paso 3: Instanciación de las restricciones relevantes

Paso 4: Simplificación de las instancias de las restricciones relevantes



3.2 Comprobación de la integridad

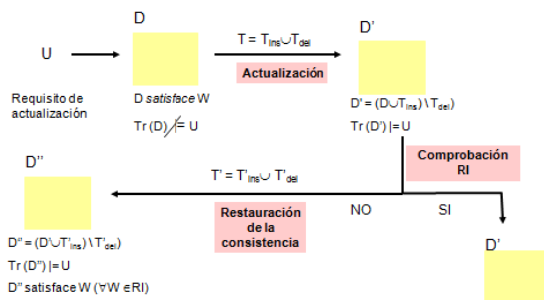
Estudio de un método de simplificación:

- Semántica asumida: referencia para interpretar el concepto de satisfacción.
- Concepto de satisfacción utilizado.
- Requisitos sintácticos: forma sintáctica de las reglas y de las restricciones de integridad.
- Corrección y completitud del método.
- Estrategia del método:
 - Fase de Generación: potencial (sin acceso a la BDE), real (con acceso a la BDE).
 - Intercalación de las fases de Generación y Evaluación.
 - Etapas de compilación independiente de la transacción.

La Lógica en el desarrollo de las Bases de Datos

- Lógica y Bases de Datos.
- Bases de datos deductivas.
- Actualización de bases de datos deductivas.
 - Actualización
 - Comprobación de la integridad
 - Restauración de la consistencia

3.2 Restauración de la consistencia



3.2 Restauración de la consistencia

Enunciado general del problema:

Dados, el esquema (L, RI) de una base de datos deduciva, un estado D de ese esquema tal que $D \text{ satisface } W$ (para toda $W \in RI$), una transacción $T = T_{ins} \cup T_{del}$, $(T_{ins} \cap T_{del} = \emptyset, T_{del} \subseteq D \text{ y } T_{ins} \cap D = \emptyset)$, el estado D' resultante de aplicar a D la transacción T , $(D' = (D \cup T_{ins}) \setminus T_{del})$, y una restricción $W \in RI$ tal que D' viola W , encontrar una transacción $T^* = T^*_{ins} \cup T^*_{del}$ ($T^*_{ins} \cap T^*_{del} = \emptyset, T^*_{del} \cap T_{ins} = \emptyset, T^*_{ins} \cap T_{del} = \emptyset$), tal que el estado D'' resultante de aplicar T^* a D' , $D'' = (D' \cup T^*_{ins}) \setminus T^*_{del}$, satisface W .

Bases de datos multidimensionales

Son bases de datos ideadas para desarrollar aplicaciones muy concretas. Básicamente almacena sus datos con varias dimensiones, es decir que en vez de un valor, encontramos varios dependiendo de los ejes definidos o una base de datos de estructura basada en dimensiones orientada a consultas complejas y alto rendimiento. En una base de datos multidimensional, la información se representa como matrices multidimensionales, cuadros de múltiples entradas o funciones de varias variables sobre conjuntos finitos. Cada una de estas matrices se denomina cubo. Eso facilita el manejo de grandes cantidades de datos dentro de empresas, dándole a esto una amplia aplicación dentro de varias áreas y diferentes campos del conocimiento humano.

Bases de datos transaccionales

Son bases de datos caracterizadas por su velocidad para gestionar el intercambio de información, se utilizan sobre todo en sistemas bancarios, análisis de calidad y datos de producción industrial. Son bases de datos muy fiables, ya que en ellas cada una de las operaciones de inserción, actualización o borrado se realizan completamente o se descartan.

5.- Tipos de bases de datos.

Caso práctico

María pregunta a Ada: —Si nuestras aplicaciones van a ser accesibles desde Internet ¿Qué tipo de base de datos utilizaremos?

Ada responde: —Pues la respuesta es amplia. Lo normal es que sean bases de datos de acceso múltiple, cuya información cambie en el tiempo, podrán ser centralizadas o distribuidas, además es probable que su acceso deba estar restringido sólo a los usuarios que se indiquen y su temática será diversa. Como ves, hay una gran variedad de tipos de bases de datos y dependiendo de las necesidades y presupuesto de nuestros clientes tendremos que adaptar nuestras aplicaciones a dichos tipos.

—Lo importante es que hagamos un buen diseño y planificación de nuestras bases de datos. De este modo, el software que desarrollemos irá sobre ruedas. — Añade **Juan**.

Como hemos visto, por cada modelo de datos se establecen sustanciales diferencias entre unas bases de datos y otras, pero, ¿Esta es la única clasificación de las bases de datos existente? No, vamos a ver a continuación una detallada descripción de los tipos de bases de datos teniendo en cuenta varios criterios.

Diferentes clasificaciones de las bases de datos

✓ Bases de datos según su contenido

- **Bases de datos con información actual:** Contienen información muy concreta y actualizada, normalmente, de tipo numérico: estadísticas, series históricas, resultados de encuestas, convocatorias de becas o subvenciones, convocatorias de eventos, ofertas de empleo, ...
- **Directorios:** recogen datos sobre personas o instituciones especializadas en una actividad o materia concreta. Hay directorios de profesionales, de investigadores, de centros de investigación, de bibliotecas, de revistas científicas, de empresas, de editoriales, ...
- **Bases de datos documentales:** En éste último grupo, cada registro se corresponde con un documento, sea éste de cualquier tipo: una publicación impresa, un documento audiovisual, gráfico. Dependiendo de si incluyen o no el contenido completo de los documentos que describen, podremos tener:
 - **Bases de datos de texto completo:** constituidas por los propios documentos en formato electrónico, con un volcado completo de su texto.
 - **Archivos electrónicos de imágenes:** Constituidos por referencias que permiten un enlace directo con la imagen del documento original, sea éste un documento iconográfico (fotografías, imágenes de televisión, ...) o un documento impreso digitalizado en formato de imagen.
 - **Bases de datos referenciales:** Sus registros no contienen el texto original sino tan sólo la información fundamental para describir y permitir la localización de documentos impresos, sonoros, iconográficos, audiovisuales o electrónicos. En estos sistemas de información sólo se puede obtener referencias sobre documentos que habrá que localizar posteriormente en otro servicio (archivo, biblioteca, fonoteca,...) o solicitar a un servicio de suministro de documentos.

✓ Bases de datos según su uso:

- **Base de datos individual:** Es una base de datos utilizada básicamente por una persona. El sistema administrador de la base de datos y los datos son controlados por el mismo usuario. Puede estar almacenada en la unidad de disco duro del usuario o en el servidor de archivos de una red de área local. Por ejemplo, un gerente de ventas podría contar con una base de datos para el control de sus vendedores y su desempeño.
- **Base de datos compartida:** Son bases de datos con múltiples usuarios y que muy probablemente pertenezcan a la misma organización, como la base de datos de una compañía. Se encuentra almacenada en una computadora potente y bajo el cuidado de un profesional en el área, el administrador de la base de datos. Los usuarios tienen acceso a la base de datos mediante una red de área local o una red de área extensa.
- **Bases de datos de acceso público:** Son bases de datos accesibles por cualquier persona. Puede no ser necesario pagar un canon para hacer uso de los datos contenidos en ellas.




- **Bases de datos propietarias o bancos de datos:** Se trata en general de bases de datos de gran tamaño, desarrolladas por una organización y que contienen temas especializados o de carácter particular. El público general puede tener acceso a estas bases a veces de forma gratuita y otras mediante el pago de una cuota. Pueden ofrecer información que va desde negocios, economía, inversión, técnica y científica hasta servicios de entretenimiento. Permiten encontrar en minutos lo que tardaría horas ojeando revistas.
- ✓ **Bases de datos según la variabilidad de la información**
 - **Bases de datos estáticas:** Son bases de datos de sólo lectura. Se utilizan para el almacenamiento de datos históricos que pueden ser analizados y utilizados para el estudio del comportamiento de un conjunto de datos a través del tiempo. Permiten realizar proyecciones y toma de decisiones.
 - **Bases de datos dinámicas:** Son bases de datos donde la información almacenada se modifica con el tiempo, permitiendo operaciones como actualización y adición de datos, además de las operaciones fundamentales de consulta.
- ✓ **Bases de datos según la localización de la información**
 - **Bases de datos centralizadas:** Se trata de bases de datos ubicadas en un único lugar, un único computador. Pueden ser bases de datos monousuario que se ejecutan en ordenadores personales o sistemas de bases de datos de alto rendimiento que se ejecutan en grandes sistemas. Este tipo de organización facilita las labores de mantenimiento, sin embargo, hace que la información contenida en dicha base, sea más vulnerable a posibles fallos y limita su acceso. Este tipo de bases de datos puede ofrecer dentro de la arquitectura Cliente/Servidor dos configuraciones:
 - **Basada en anfitrión:** ocurre cuando la máquina cliente y la máquina servidor son la misma. Los usuarios se conectarán directamente a la máquina donde se encuentra la base de datos.
 - **Basada en Cliente/Servidor:** ocurrirá cuando la base de datos reside en una máquina servidor y los resultados acceden a la base de datos desde su máquina cliente a través de una red
 - **Bases de datos distribuidas:** Según la naturaleza de la organización es probable que los datos no se almacenen en un único punto, sino que se sitúen en un lugar o lugares diferentes a donde se encuentran los usuarios. Una base de datos distribuida es la unión de las bases de datos mediante redes. Los usuarios se vinculan a los servicios de bases de datos distantes mediante una amplia variedad de redes de comunicación. Puede imaginarse una compañía con diferentes oficinas regionales, donde se encuentra distribuida la base de datos. Si embargo, los ejecutivos pueden tener acceso a la información de todas las oficinas regionales.
- ✓ **Bases de datos según el organismo productor**
 - **Bases de datos de organismos públicos y de la administración:** Las bibliotecas y centros de documentación de los ministerios, instituciones públicas, universidades y organismos públicos de investigación elaboran gran cantidad de recursos de información. Estos sistemas pueden ser:
 - Bases de datos de acceso público, sean gratuitas o no.
 - Bases de datos de uso interno, con información de acceso restringido.
 - **Bases de datos de instituciones sin ánimo de lucro:** Fundaciones, asociaciones, sindicatos y organizaciones no gubernamentales elaboran frecuentemente sus propios sistemas de información especializados.
 - **Bases de datos de entidades privadas o comerciales:** Los centros de documentación, bibliotecas y archivos de las empresas pueden elaborar distintos tipos de sistemas de información:
 - Bases de datos de uso interno para facilitar la circulación de información dentro de la empresa.
 - Bases de datos de uso interno que ocasionalmente ofrecen servicio hacia el exterior (usuarios particulares u otras instituciones)

- Bases de datos comerciales, diseñadas específicamente para ser utilizadas por usuarios externos.
- **Bases de datos realizadas por cooperación en red:** Se trata de sistemas de información cuya elaboración es compartida por diversas instituciones. Bases de datos internacionales se elaboran a través de este sistema de trabajo, con diversos centros nacionales responsables de la información perteneciente a cada país.
- ✓ **Bases de datos según el modelo de acceso:**
 - **Bases de datos de acceso local:** Para consultarlas es necesario acudir al organismo productor, a su biblioteca o centro de documentación. Pueden ser consultables en monopuesto o en varios puntos de una red local.
 - **Bases de datos en CD-ROM:** Pueden adquirirse por compra o suscripción bien directamente por un particular o por una biblioteca o centro de documentación que permita su consulta a sus usuarios. En algunas instituciones se instalan diferentes CD-ROM en una red local para permitir su consulta desde cualquier ordenador conectado a la misma.
 - **Bases de datos en línea:** Pueden consultarse desde cualquier ordenador conectado a Internet. La consulta puede ser libre (gratuita) o exigir la solicitud previa de una clave personal entrada (denominada comúnmente con el término inglés *password*). Para obtener un password puede exigir la firma de un contrato. Hay diferentes tipos de acceso en línea:
 - **Acceso via telnet o mediante línea de Internet:** el usuario realiza una conexión estable al host (gran ordenador) en donde se halla la base de datos, a través de Internet. La interfaz de usuario instalada en dicho ordenador remoto determinará si la interrogación debe realizarse por menú o por comandos o expresiones de un lenguaje determinado. Cuando un usuario entra en una base de datos vía telnet establece una sesión de trabajo interactiva con el programa que gestiona la base de datos, que le permite aplicar todas las posibilidades de interrogación que tenga el sistema selección, combinación y visualización o impresión de resultados. En cualquier momento podrá visualizar todas las búsquedas realizadas hasta ese instante y establecer combinaciones entre ellas.
 - **Acceso vía web:** conexión a través de un formulario existente en una página web de Internet, diseñado para lanzar preguntas a una base de datos.

Una misma base de datos puede tener acceso local y además una edición en CD-ROM y un sistema de acceso en línea. Sin embargo, puede haber diferencias en el contenido presente en cada uno de estos formatos o en el grado de actualización de la información. Por ejemplo, el productor de una base de datos puede ofrecer la conexión en línea a la base de datos completa con actualización diaria y, en cambio, editar un CD_ROM que tan sólo contenga los últimos cinco años de información y se actualice semestralmente.
- ✓ **Bases de datos según cobertura temática:**
 - **Bases de datos científico-tecnológicas:** Contienen información destinada a los investigadores de cualquier ámbito científico o técnico. A su vez, este grupo puede dividirse en:
 - Bases de datos multidisciplinarias: abarcan varias disciplinas científicas o técnicas.
 - Bases de datos especializadas: recopilan y analizan documentos pertinentes para una disciplina o subdisciplina concreta: investigación biomédica, farmacéutica, química, agroalimentaria, social, humanística, etc
 - **Bases de datos económico-empresariales:** Contienen información de interés para empresas, entidades financieras, ...
 - **Bases de datos de medios de comunicación:** contienen información de interés para los profesionales de medios de comunicación de masas: prensa, radio, televisión,...
 - **Bases de datos de ámbito político-administrativo y jurídico:** Contienen información de interés para los organismos de la administración y los profesionales del Derecho: legislación, jurisprudencia, ...

- **Bases de datos de ámbito sanitario:** Además de las propias del primer grupo especializadas en ciencias de la salud, existen otros sistemas con información de interés sanitario: historiales médicos, archivos hospitalarios, ...
- **Bases de datos para el gran público:** Contiene información destinada a cubrir necesidades de información general, de interés para un gran número de usuarios.

Las bases de datos en las que sus registros no contienen el texto original sino tan sólo la información fundamental para describir y permitir la localización de documentos impresos, sonoros, iconográficos, audiovisuales o electrónicos, reciben el nombre de:

-  Bases de datos documentales.
-  Bases de datos distribuidas.
-  **Bases de datos referenciales.**

6 - Sistemas gestores de bases de datos

Caso práctico

Ada explica a **Juan** y **María** que la elección de un buen Sistema Gestor de Base de Datos es fundamental. A través de esta herramienta podrán definir, construir y manejar las bases de datos con las que sus aplicaciones informáticas han de trabajar. Conocer sus funciones, componentes y tipos será la base fundamental para llevar a cabo una elección adecuada.

Juan dice: -Yo he utilizado varios sistemas gestores diferentes, cada uno tiene sus ventajas e inconvenientes, pero en general todos se parecen un poco. Eso sí, facilitan mucho el trabajo, son fiables y ahorran tiempo.

Para poder tratar la información contenida en las bases de datos se utilizan los sistemas gestores de bases de datos o SGBD, también llamados DBMS (DataBase Management System), que ofrecen un conjunto de programas que permiten acceder y gestionar dichos datos.

El objetivo fundamental de los SGBD es proporcionar eficiencia y seguridad a la hora de recuperar o insertar información en las bases de datos. Estos sistemas están diseñados para la manipulación de grandes bloques de información.

Sistema Gestor de Base de Datos: Conjunto coordinado de programas, procedimientos, lenguajes, etc., que suministra, tanto a los usuarios no informáticos, como a los analistas programadores, o al administrador, los medios necesarios para describir y manipular los datos contenidos en la base de datos, manteniendo su integridad, confidencialidad y seguridad.

El SGBD permite a los usuarios la creación y el mantenimiento de una base de datos, facilitando la definición, construcción y manipulación de la información contenida en éstas. Definir una base de datos consistirá en especificar los tipos de datos, las estructuras y las restricciones que los datos han de cumplir a la hora de almacenarse en dicha base. Por otro lado, la construcción de la base será el proceso de almacenamiento de datos concretos en algún medio o soporte de almacenamiento que esté supervisado por el SGBD. Finalmente, la manipulación de la base de datos incluirá la posibilidad de realización de consultas para recuperar información específica, la actualización de los datos y la generación de informes a partir de su contenido.

Las ventajas del uso de SGBD son:

- ✓ Proporcionan al usuario una visión abstracta de los datos, ocultando parte de la complejidad relacionada con cómo se almacenan y mantienen los datos.
- ✓ Ofrecen **Independencia física**, es decir, la visión que tiene de la información el usuario, y la manipulación de los datos almacenados en la Base de Datos, es independiente de cómo estén almacenados físicamente.
- ✓ Disminuyen la redundancia y la inconsistencia de datos.
- ✓ Aseguran la integridad de los datos.
- ✓ Facilitan el acceso a los datos, aportando rapidez y evitando la pérdida de datos.
- ✓ Aumentan la seguridad y privacidad de los datos.
- ✓ Mejoran la eficiencia.
- ✓ Permiten compartir datos y accesos concurrentes.
- ✓ Facilitan el intercambio de datos entre distintos sistemas.
- ✓ Incorporan mecanismos de copias de seguridad y recuperación para restablecer la información en caso de fallos en el sistema.



El SGBD interacciona con otros elementos software existentes en el sistema, concretamente con el sistema operativo (SO). Los datos almacenados de forma estructurada en la base de datos son utilizados indistintamente por otras aplicaciones, será el SGBD quien ofrecerá una serie de facilidades

a éstas para el acceso y manipulación de la información, basándose en las funciones y métodos propios del sistema operativo.

6.1.- Funciones.

Un SGBD desarrolla tres funciones fundamentales como son las de descripción, manipulación y utilización de los datos. A continuación se detallan cada una de ellas.

1. **Función de descripción o definición:** Permite al diseñador de la base de datos crear las estructuras apropiadas para integrar adecuadamente los datos. Esta función es la que permite definir las tres estructuras de la base de datos: Estructura interna, Estructura conceptual y Estructura externa. (Estos conceptos se verán más adelante en el epígrafe sobre arquitectura del SGBD).

Esta función se realiza mediante el **lenguaje de descripción de datos o DDL**. Mediante ese lenguaje: se definen las estructuras de datos, se definen las relaciones entre los datos y se definen las reglas (restricciones) que han de cumplir los datos.

Se especificarán las características de los datos a cada uno de los tres niveles.

- **A nivel interno** (estructura interna), se ha de indicar el espacio de disco reservado para la base de datos, la longitud de los campos, su modo de representación (lenguaje para la definición de la estructura externa).
- **A nivel conceptual** (estructura conceptual), se proporcionan herramientas para la definición de las entidades y su identificación, atributos de las mismas, interrelaciones entre ellas, restricciones de integridad, etc.; es decir, el esquema de la base de datos (lenguaje para la definición de estructura lógico global).
- **A nivel externo** (estructura externa), se deben definir las vistas de los distintos usuarios a través del lenguaje para la definición de estructuras externas. Además, el SGBD se ocupará de la transformación de las estructuras externas orientadas a los usuarios a las estructuras conceptuales y de la relación de ésta y la estructura física.

2. **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 las normas de seguridad dictadas por el administrador. Se llevará a cabo por medio de un **lenguaje de manipulación de datos (DML)** que facilita los instrumentos necesarios para la realización de estas tareas.

También se encarga de definir la **vista externa** de todos los usuarios de la base de datos o vistas parciales que cada usuario tiene de los datos definidos con el DDL.

Por manipulación de datos entenderemos:

- La recuperación de información almacenada en la base de datos, lo que se conoce como **consultas**.
- La inserción de información nueva en la base de datos.
- El borrado de información de la base de datos.
- La modificación de información almacenada en la base de datos.

3. **Función de control:** permite al administrador de la base de datos establecer mecanismos de protección de las diferentes visiones de los datos asociadas a cada usuario, proporcionando elementos de creación y modificación de dichos usuarios. Adicionalmente, incorpora sistemas para la creación de copias de seguridad, carga de ficheros, auditoría, protección de ataques, configuración del sistema, etc. El lenguaje que implementa esta función es el **lenguaje de control de datos o DCL**.

¿Y a través de qué lenguaje podremos desarrollar estas funciones sobre la base de datos? Lo haremos utilizando el **Lenguaje Estructurado de Consultas (SQL: Structured Query Language)**. Este lenguaje proporciona sentencias para realizar operaciones de DDL, DML y DCL. SQL fue publicado por el ANSI en 1986 (American National Standard Institute) y ha ido evolucionando a lo largo del tiempo. Además, los SGBD suelen proporcionar otras herramientas que complementan a estos lenguajes como generadores de formularios, informes, interfaces gráficas, generadores de aplicaciones, etc.

El DDL de una base de datos sirve para:

- ☐ La introducción de los datos en una base de datos.
- ☐ **Definir la estructura lógica de la base de datos.**
- ☐ Interrogar a la base de datos (consultar la información de dicha base).

6.2.- Componentes.

Una vez descritas las funciones que un SGBD debe llevar a cabo, imaginarás que un SGBD es un paquete de software complejo que ha de proporcionar servicios relacionados con el almacenamiento y la explotación de los datos de forma eficiente. Para ello, cuenta con una serie de componentes que se detallan a continuación:

1. **Lenguajes de la base de datos.** Cualquier sistema gestor de base de datos ofrece la posibilidad de utilizar lenguajes e interfaces adecuadas para sus diferentes tipos de usuarios. A través de los lenguajes se pueden especificar los datos que componen la BD, su estructura, relaciones, reglas de integridad, control de acceso, características físicas y vistas externas de los usuarios. Los lenguajes del SGBD son: Lenguaje de Definición de los Datos (**DDL**), Lenguaje de Manejo de Datos (**DML**) y Lenguaje de Control de Datos (**DCL**).
2. **El diccionario de datos.** Descripción de los datos almacenados. Se trata de información útil para los programadores de aplicaciones. Es el lugar donde se deposita la información sobre la totalidad de los datos que forman la base de datos. Contiene las características lógicas de las estructuras que almacenan los datos, su nombre, descripción, contenido y organización. En una base de datos relacional, el diccionario de datos aportará información sobre:
 - ✓ Estructura lógica y física de la BD.
 - ✓ Definición de tablas, vistas, índices, disparadores, procedimientos, funciones, etc.
 - ✓ Cantidad de espacio asignado y utilizado por los elementos de la BD.
 - ✓ Descripción de las restricciones de integridad.
 - ✓ Información sobre los permisos asociados a cada perfil de usuario.
 - ✓ Auditoría de acceso a los datos, utilización, etc.
3. **El gestor de la base de datos.** Es la parte de software encargada de garantizar el correcto, seguro, íntegro y eficiente acceso y almacenamiento de los datos. Este componente es el encargado de proporcionar una interfaz entre los datos almacenados y los programas de aplicación que los manejan. Es un intermediario entre el usuario y los datos. Es el encargado de garantizar la privacidad, seguridad e integridad de los datos, controlando los accesos concurrentes e interactuando con el sistema operativo.
4. **Usuarios de la base de datos.** En los SGBD existen diferentes perfiles de usuario, cada uno de ellos con una serie de permisos sobre los objetos de la BD. Generalmente existirán:
 - ✓ El **administrador de la base de datos** o Database Administrator (DBA), que será la persona o conjunto de ellas encargadas de la función de administración de la base de datos. Tiene el control centralizado de la base de datos y es el responsable de su buen funcionamiento. Es el encargado de autorizar el acceso a la base de datos, de coordinar y vigilar su utilización y de adquirir los recursos software y hardware que sean necesarios.
 - ✓ Los **usuarios de la base de datos**, que serán diferentes usuarios de la BD con diferentes necesidades sobre los datos, así como diferentes accesos y privilegios. Podemos establecer la siguiente clasificación:
 - ➔ Diseñadores.
 - ➔ Operadores y personal de mantenimiento.
 - ➔ Analistas y programadores de aplicaciones.
 - ➔ Usuarios finales: ocasionales, simples, avanzados y autónomos.
5. **Herramientas de la base de datos.** Son un conjunto de aplicaciones que permiten a los administradores la gestión de la base de datos, de los usuarios y permisos, generadores de formularios, informes, interfaces gráficas, generadores de aplicaciones, etc.

6.3.- Arquitectura.

Un SGBD cuenta con una arquitectura a través de la que se simplifica a los diferentes usuarios de la base de datos su labor. El objetivo fundamental es separar los programas de aplicación de la base de datos física.

Encontrar un estándar para esta arquitectura no es una tarea sencilla, aunque los tres estándares que más importancia han cobrado en el campo de las bases de datos son ANSI/SPARC/X3, CODASYL y ODMG (éste sólo para las bases de datos orientadas a objetos). Tanto ANSI (EEUU), como ISO (Resto del mundo), son el referente en cuanto a estandarización de bases de datos, conformando un único modelo de bases de datos.

La arquitectura propuesta proporciona tres niveles de abstracción: **nivel interno o físico, nivel lógico o conceptual y nivel externo o de visión del usuario**. A continuación se detallan las características de cada uno de ellos:

- ✓ **Nivel interno o físico:** En este nivel se describe la estructura física de la base de datos a través de un esquema interno encargado de detallar el sistema de almacenamiento de la base de datos y sus métodos de acceso. Es el nivel más cercano al almacenamiento físico. A través del esquema físico se indican, entre otros, los archivos que contienen la información, su organización, los métodos de acceso a los registros, los tipos de registros, la longitud, los campos que los componen, las unidades de almacenamiento, etc.
- ✓ **Nivel lógico o conceptual:** En este nivel se describe la estructura completa de la base de datos a través de un esquema que detalla las entidades, atributos, relaciones, operaciones de los usuarios y restricciones. Los detalles relacionados con las estructuras de almacenamiento se ocultan, permitiendo realizar una abstracción a más alto nivel.
- ✓ **Nivel externo o de visión del usuario:** En este nivel se describen las diferentes vistas que los usuarios percibirán de la base de datos. Cada tipo de usuario o grupo de ellos verá sólo la parte de la base de datos que le interesa, ocultando el resto.

Para una base de datos, sólo existirá un único esquema interno, un único esquema conceptual y podrían existir varios esquemas externos definidos para uno o varios usuarios.

Gracias a esta arquitectura se consigue la **independencia de datos** a dos niveles:

- ✓ **Independencia lógica:** Podemos modificar el esquema conceptual sin alterar los esquemas externos ni los programas de aplicación.
- ✓ **Independencia física:** Podemos modificar el esquema interno sin necesidad de modificar el conceptual o el externo. Es decir, se puede cambiar el sistema de almacenamiento, reorganizar los ficheros, añadir nuevos, etc., sin que esto afecte al resto de esquemas.

En el siguiente gráfico se puede apreciar la estructura de la que estamos hablando:



El esquema conceptual de la totalidad de la base de datos puede obtenerse de la unión de todos los esquemas externos definidos para cada usuario de la base de datos.

Verdadero

Falso

6.4.- Tipos.

¿Qué tipos de SGBD existen? Para responder a esta pregunta podemos realizar la siguiente clasificación, atendiendo a diferentes criterios:

- a. El primer criterio que se suele utilizar es **por el modelo lógico en que se basan**. Actualmente, el modelo lógico que más se utiliza es el **relacional**. Los modelos en red y jerárquico han quedado obsoletos. Otro de los modelos que más extensión está teniendo es el modelo **orientado a objetos**. Por tanto, en esta primera clasificación tendremos:
 - ✓ Modelo Jerárquico.
 - ✓ Modelo de Red.
 - ✓ Modelo Relacional.
 - ✓ Modelo Orientado a Objetos.(Para recordar los modelos de bases de datos vistos, sitúate en el epígrafe 4 de esta Unidad de Trabajo y analiza su contenido.)
- b. El segundo criterio de clasificación se centra en el **número de usuarios** a los que da servicio el sistema:
 - ✓ **Monousuario**: sólo atienden a un usuario a la vez, y su principal uso se da en los ordenadores personales.
 - ✓ **Multiusuario**: entre los que se encuentran la mayor parte de los SGBD, atienden a varios usuarios al mismo tiempo.
- c. El tercer criterio se basa en el **número de sitios en los que está distribuida la base de datos**:
 - ✓ **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.
 - ✓ **Distribuidos (Homogéneos, Heterogéneos)**: la base de datos real y el propio software del SGBD pueden estar distribuidos en varios sitios conectados por una red. Los sistemas 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 sistemas 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.
- d. **El cuarto criterio toma como referencia el coste**. La mayor parte de los paquetes cuestan entre 10.000 y 100.000 euros. Los sistemas monousuario más económicos para microcomputadores cuestan entre 0 y 3.000 euros. En el otro extremo, los paquetes más completos cuestan más de 100.000 euros.
- e. El quinto, y último, criterio establece su clasificación **según el propósito**:
 - ✓ **Propósito General**: pueden ser utilizados para el tratamiento de cualquier tipo de base de datos y aplicación.
 - ✓ **Propósito Específico**: Cuando el rendimiento es fundamental, se puede diseñar y construir un software de propósito especial para una aplicación específica, y este sistema no sirve para otras aplicaciones. Muchos sistemas de reservas de líneas aéreas son de propósito especial y pertenecen a la categoría de **sistemas de procesamiento de transacciones en línea**, que deben atender un gran número de transacciones concurrentes sin imponer excesivos retrasos.

7.- SGBD comerciales.

Caso práctico

—¿Conocéis la multinacional Oracle? ¿Y su sistema de gestión de bases de datos Oracle 10g?
—Pregunta **Ada**.

Juan, que está terminando de instalar un nuevo disco duro en su equipo le responde: —Por supuesto **Ada**, es el número uno en el mundo de las bases de datos y sus productos tienen una gran aceptación en el mercado. Según conozco, sus posibilidades y fiabilidad son impresionantes, aunque hay que pagar una licencia.

BK Programación ha de tener en cuenta que sus aplicaciones deben estar sustentadas en un sistema que ofrezca garantías, pero que se ajuste a sus necesidades, dimensiones y presupuesto. Es el momento de pensar su próxima jugada.

Actualmente, en el mercado de software existen multitud de sistemas gestores de bases de datos comerciales. En este epígrafe se desglosan las características fundamentales de los más importantes y extendidos hasta la fecha. Pero, como podrás observar, la elección de un SGBD es una decisión muy importante a la hora de desarrollar proyectos. A veces, el sistema más avanzado, "el mejor" según los entendidos, puede no serlo para el tipo de proyecto que estemos desarrollando. Hemos de tener en cuenta qué volumen de carga debe soportar la base de datos, qué sistema operativo utilizaremos como soporte, cuál es nuestro presupuesto, plazos de entrega, etc.

A través de la siguiente tabla se exponen los SGBD comerciales más utilizados y sus características más relevantes:

Sistemas Gestores de Bases de Datos Comerciales.		
SGBD	Descripción	URL
ORACLE	Reconocido como uno de los mejores a nivel mundial. Es multiplataforma, confiable y seguro. Es Cliente/Servidor. Basado en el modelo de datos Relacional. De gran potencia, aunque con un precio elevado hace que sólo se vea en empresas muy grandes y multinacionales. Ofrece una versión gratuita Oracle Database 10g Express Edition .	http://www.oracle.com/us/products/database/product-editions-066501.html?ssSourceSiteId=ocomes
MYSQL	Sistema muy extendido que se ofrece bajo dos tipos de licencia, comercial o libre. Para aquellas empresas que deseen incorporarlo en productos privativos, deben comprar una licencia específica. Es Relacional, Multihilo, Multiusuario y Multiplataforma. Su gran velocidad lo hace ideal para consulta de bases de datos y plataformas web.	http://www.mysql.com/
DB2	Multiplataforma, el motor de base de datos relacional integra XML de manera nativa, lo que IBM ha llamado pureXML, que permite almacenar documentos completos para realizar operaciones y búsquedas de manera jerárquica dentro de éste, e integrarlo con búsquedas relacionales.	http://www.ibm.com/developerworks/ssa/download/im/udbexp/
INFORMIX	Otra opción de IBM para el mundo empresarial que necesita un DBMS sencillo y confiable. Es un gestor de base de datos relacional basado en SQL. Multiplataforma. Consume menos recursos que Oracle, con utilidades muy avanzadas respecto a conectividad y funciones relacionadas con tecnologías de Internet/Intranet, XML, etc.	http://www-01.ibm.com/software/es/data/informix/discover-informix/index.html
Microsoft SQL SERVER	Sistema Gestor de Base de Datos producido por Microsoft. Es relacional, sólo funciona bajo Microsoft Windows, utiliza arquitectura Cliente/Servidor. Constituye la alternativa a	http://www.microsoft.com/spain/sql/2008/overview.aspx

	otros potentes SGBD como son Oracle, PostgreSQL o MySQL.
SYBASE	Un DBMS con bastantes años en el mercado, tiene 3 versiones para ajustarse a las necesidades reales de cada empresa. Es un sistema relacional, altamente escalable, de alto rendimiento, con soporte a grandes volúmenes de datos, transacciones y usuarios, y de bajo costo. http://www.sybase.es/products/databasemanagement/adaptiveserverenterprise

Otros SGBD comerciales importantes son: DBASE, ACCESS, INTERBASE y FOXPRO.

Puedes completar más información sobre estos y otros sistemas a través de los enlaces que te proponemos a continuación:

[http://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_bases_de_datos#SGBD no lib
es](http://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_bases_de_datos#SGBD_no_libres)

[http://4.bp.blogspot.com/BF5cCSiGL2M/TUjX04Zuzgl/AAAAAAAAAlo/2_0gKVhrpO0/s1600/
gartnerMQ_2011.bmp](http://4.bp.blogspot.com/BF5cCSiGL2M/TUjX04Zuzgl/AAAAAAAAAlo/2_0gKVhrpO0/s1600/gartnerMQ_2011.bmp)

8.- SGBD libres.

Caso práctico

Juan, que tiene especial debilidad por el software libre, comenta que existen alternativas muy potentes a coste cero. **Ada**, agradece la información que **Juan** aporta e indica que también tendrán en cuenta los sistemas gestores de bases de datos libres en sus desarrollos, ya que algunos de ellos están ampliamente extendidos y ofrecen importantes ventajas. **María**, que ha trabajado alguna vez con MySQL, está deseosa de aprender nuevos sistemas gestores ya sean comerciales o libres.

La alternativa a los sistemas gestores de bases de datos comerciales la encontramos en los SGBD de código abierto o libres, también llamados Open Source. Son sistemas distribuidos y desarrollados libremente. En la siguiente tabla se relacionan los cinco más utilizados actualmente, así como sus principales características y enlaces a sus páginas web:

Sistemas Gestores de Bases de Datos Libres.		
SGBD	Descripción	URL
MySQL	Es un sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. Distribuido bajo dos tipos de licencias, comercial y libre. Multiplataforma, posee varios motores de almacenamiento, accesible a través de múltiples lenguajes de programación y muy ligado a aplicaciones web.	http://www.mysql.com/
PostgreSQL	Sistema Relacional Orientado a Objetos. Considerado como la base de datos de código abierto más avanzada del mundo. Desarrollado por una comunidad de desarrolladores que trabajan de forma desinteresada, altruista, libre y/o apoyados por organizaciones comerciales. Es multiplataforma y accesible desde múltiples lenguajes de programación.	http://www.postgresql.org/
Firebird	Sistema Gestor de Base de Datos relacional, multiplataforma, con bajo consumo de recursos, excelente gestión de la concurrencia, alto rendimiento y potente soporte para diferentes lenguajes.	http://www.firebirdsql.org/
Apache Derby	Sistema Gestor escrito en Java, de reducido tamaño, con soporte multilinguaje, multiplataforma, altamente portable, puede funcionar embebido o en modo cliente/servidor.	http://db.apache.org/derby/
SQLite	Sistema relacional, basado en una biblioteca escrita en C que interactúa directamente con los programas, reduce los tiempos de acceso siendo más rápido que MySQL o PostgreSQL, es multiplataforma y con soporte para varios lenguajes de programación.	http://www.sqlite.org/

El tamaño máximo de una tabla en PostgreSQL es de 1,6 Terabytes.

Verdadero

Falso

9.- Bases de datos centralizadas.

Caso práctico

Ada, Juan y María están visitando un centro de cómputo cercano a BK Programación. La estructura del sistema informático está centralizada y limita las posibilidades de uso de la información contenida en dicho sistema. **Ada** indica que con la ayuda de la tecnología de redes de computadoras la información se puede mantener localizada en diversos lugares, permitiendo accesos más rápidos y múltiples ventajas adicionales en comparación con los sistemas centralizados. Los tres continúan su visita, analizando las ventajas e inconvenientes del sistema centralizado que están viendo.

Si nos preguntamos cómo es la arquitectura de un sistema de base de datos, hemos de saber que todo depende del sistema informático que la sustenta. Tradicionalmente, la arquitectura centralizada fue la que se utilizó inicialmente, aunque hoy en día es de las menos utilizadas.

Sistema de base de datos centralizado: Es aquella estructura en la que el SGBD está implantado en una sola plataforma u ordenador desde donde se gestiona directamente, de modo centralizado, la totalidad de los recursos. Es la arquitectura de los centros de proceso de datos tradicionales. Se basa en tecnologías sencillas, muy experimentadas y de gran robustez.

Los sistemas de los años sesenta y setenta eran totalmente centralizados, como corresponde a los sistemas operativos de aquellos años, y al hardware para el que estaban hechos: un gran ordenador para toda la empresa y una red de terminales sin inteligencia ni memoria.

Las principales características de las bases de datos centralizadas son:

- ✓ Se almacena completamente en una ubicación central, es decir, todos los componentes del sistema residen en un solo computador o sitio.
- ✓ No posee múltiples elementos de procesamiento ni mecanismos de intercomunicación como las bases de datos distribuidas.
- ✓ Los componentes de las bases de datos centralizadas son: los datos, el software de gestión de bases de datos y los dispositivos de almacenamiento secundario asociados.
- ✓ Son sistemas en los que su seguridad puede verse comprometida más fácilmente.

En la siguiente tabla se representan las ventajas e inconvenientes destacables de esta arquitectura de bases de datos.

Ventajas e inconvenientes de las bases de datos centralizadas.	
Ventajas	Inconvenientes
Se evita la redundancia debido a la posibilidad de inconsistencias y al desperdicio de espacio.	Un mainframe en comparación de un sistema distribuido no tiene mayor poder de cómputo.
Se evita la inconsistencia. Ya que si un hecho específico se representa por una sola entrada, la no-concordancia de datos no puede ocurrir.	Cuando un sistema de bases de datos centralizado falla, se pierde toda disponibilidad de procesamiento y sobre todo de información confiada al sistema.
La seguridad se centraliza.	En caso de un desastre o catástrofe, la recuperación es difícil de sincronizar.
Puede conservarse la integridad.	Las cargas de trabajo no se pueden difundir entre varias computadoras, ya que los trabajos siempre se ejecutarán en la misma máquina.
El procesamiento de los datos ofrece un mejor rendimiento.	Los departamentos de sistemas retienen el control de toda la organización.
Mantenimiento más barato. Mejor uso de los recursos y menores recursos humanos.	Los sistemas centralizados requieren un mantenimiento central de datos.

10.- Bases de datos distribuidas.

Caso práctico

Para poder apreciar la diferencia, **Ada** ha organizado una videoconferencia en la que intervienen dos técnicos de bases de datos y un gerente de una gran cadena hotelera, amigos suyos. Cada uno de ellos se encuentra en sedes diferentes dispersas geográficamente. **Juan** y **María**, permanecen atentos a las intervenciones que se realizan y toman buena nota de las valoraciones de los sistemas de bases de datos distribuidos hechas por los conferenciantes.

La necesidad de integrar información de varias fuentes y la evolución de las tecnologías de comunicaciones, han producido cambios muy importantes en los sistemas de bases de datos. La respuesta a estas nuevas necesidades y evoluciones se materializa en los sistemas de bases de datos distribuidas.

Base de datos distribuida (BDD): es un conjunto de múltiples bases de datos lógicamente relacionadas las cuales se encuentran distribuidas entre diferentes nodos interconectados por una red de comunicaciones.

Sistema de bases de datos distribuida (SBDD): es un sistema en el cual múltiples sitios de bases de datos están ligados por un sistema de comunicaciones, de tal forma que, un usuario en cualquier sitio puede acceder los datos en cualquier parte de la red exactamente como si los datos estuvieran almacenados en su sitio propio.

Sistema gestor de bases de datos distribuida (SGBDD): es aquel que se encarga del manejo de la BDD y proporciona un mecanismo de acceso que hace que la distribución sea transparente a los usuarios. El término transparente significa que la aplicación trabajaría, desde un punto de vista lógico, como si un solo SGBD ejecutado en una sola máquina, administrara esos datos.

Un SGBDD desarrollará su trabajo a través de un conjunto de sitios o nodos, que poseen un sistema de procesamiento de datos completo con una base de datos local, un sistema de gestor de bases de datos e interconectados entre sí. Si estos nodos están dispersos geográficamente se internocerarán a través de una red de área amplia o WAN, pero si se encuentran en edificios relativamente cercanos, pueden estar interconectados por una red de área local o LAN. Este tipo de sistemas es utilizado en: organizaciones con estructura descentralizada, industrias de manufactura con múltiples sedes (automoción), aplicaciones militares, líneas aéreas, cadenas hoteleras, servicios bancarios, etc.



En la siguiente tabla se representan las ventajas e inconvenientes destacables de las BDD:

Ventajas e inconvenientes de las bases de datos distribuidas.	
Ventajas	Inconvenientes
El acceso y procesamiento de los datos es más rápido ya que varios nodos comparten carga de trabajo.	La probabilidad de violaciones de seguridad es creciente si no se toman las precauciones debidas.
Desde una ubicación puede accederse a información alojada en diferentes lugares.	Existe una complejidad añadida que es necesaria para garantizar la coordinación apropiada entre los nodos.
Los costes son inferiores a los de las bases	La inversión inicial es menor, pero el

centralizadas.	mantenimiento y control puede resultar costoso.
Existe cierta tolerancia a fallos. Mediante la replicación, si un nodo deja de funcionar el sistema completo no deja de funcionar.	Dado que los datos pueden estar replicados, el control de concurrencia y los mecanismos de recuperación son mucho más complejos que en un sistema centralizado.
El enfoque distribuido de las bases de datos se adapta más naturalmente a la estructura de las organizaciones. Permiten la incorporación de nodos de forma flexible y fácil.	El intercambio de mensajes y el cómputo adicional necesario para conseguir la coordinación entre los distintos nodos constituyen una forma de sobrecarga que no surge en los sistemas centralizados.
Aunque los nodos están interconectados, tienen independencia local.	Dada la complejidad del procesamiento entre nodos es difícil asegurar la corrección de los algoritmos, el funcionamiento correcto durante un fallo o la recuperación.

Si deseas completar más información sobre las bases de datos distribuidas, puedes hacerlo a través del siguiente documento:

<http://sinbad.dit.upm.es/docencia/grado/curso0910/Tema%20VII%20Arquitecturas%20SGBD%20Distribuidos/2009-10%20Docu%20Todo%20el%20Tema%20VII%20BSDT.pdf>

10.1.- Fragmentación.

Sabemos que en los sistemas de bases de datos distribuidas la información se encuentra repartida en varios lugares. La forma de extraer los datos consultados puede realizarse mediante la fragmentación de distintas tablas pertenecientes a distintas bases de datos que se encuentran en diferentes servidores. El problema de fragmentación se refiere al particionamiento de la información para distribuir cada parte a los diferentes sitios de la red.

Pero hay que tener en cuenta el **grado de fragmentación** que se aplicará, ya que éste es un factor determinante a la hora de la ejecución de consultas. Si no existe fragmentación, se tomarán las relaciones o tablas como la unidad de fragmentación. Pero también puede fragmentarse a nivel de tupla (fila o registro) o a nivel de atributo (columna o campo) de una tabla. No será adecuado un grado de fragmentación nulo, ni tampoco un grado de fragmentación demasiado alto. El grado de fragmentación deberá estar equilibrado y dependerá de las particularidades de las aplicaciones que utilicen dicha base de datos. Concretando, el objetivo de la fragmentación es encontrar un nivel de particionamiento adecuado en el rango que va desde tuplas o atributos hasta relaciones completas.

Cuando se lleva a cabo una fragmentación, existen tres reglas fundamentales a cumplir:

- ✓ **Complejidad.** Si una relación R se descompone en fragmentos R1, R2, ..., Rn, cada elemento de datos que pueda encontrarse en R deberá poder encontrarse en uno o varios fragmentos Ri.
- ✓ **Reconstrucción.** Si una relación R se descompone en una serie de fragmentos R1, R2, ..., Rn, la reconstrucción de la relación a partir de sus fragmentos asegura que se preservan las restricciones definidas sobre los datos.
- ✓ **Disyunción.** Si una relación R se descompone verticalmente, sus atributos primarios clave normalmente se repiten en todos sus fragmentos.

Existen tres tipos de fragmentación:

- ✓ **Fragmentación horizontal:** La fragmentación horizontal se realiza sobre las tuplas de la relación, dividiendo la relación en subrelaciones que contienen un subconjunto de las tuplas que alberga la primera. Existen dos variantes de la fragmentación horizontal: la primaria y la derivada.
- ✓ **Fragmentación vertical:** La fragmentación vertical, en cambio, se basa en los atributos de la relación para efectuar la división. Una relación R produce fragmentos R1, R2, ..., Rr, cada uno de los cuales contiene un subconjunto de los atributos de R así como la llave primaria de R. El

objetivo de la fragmentación vertical es particionar una relación en un conjunto de relaciones más pequeñas de manera que varias de las aplicaciones de usuario se ejecutarán sobre un fragmento. En este contexto, una fragmentación óptima es aquella que produce un esquema de fragmentación que minimiza el tiempo de ejecución de las consultas de usuario. La fragmentación vertical es más complicada que la horizontal, ya que existe un gran número de alternativas para realizarla.

- ✓ **Fragmentación Híbrida o mixta:** Podemos combinar ambas, utilizando por ello la denominada fragmentación mixta. Si tras una fragmentación vertical se lleva a cabo otra horizontal, se habla de la fragmentación mixta (HV). Para el caso contrario, estaremos ante una fragmentación (VH). Para representar los dos tipos de fragmentación, se utilizan los árboles.

Una base de datos almacenada entre distintos computadores conectados en red, de forma que unos tienen acceso a los datos de otros, se dice que:

- Utiliza un modelo jerárquico
- Es de tipo distribuido con fragmentación**
- Utiliza un modelo en red

11.- Primeros pasos en Oracle Database 10g Express Edition.

Caso práctico

Después de valorar todas las opciones (comerciales y libres) existentes en el mercado, BK Programación se decantará por un consagrado sistema de base de datos comercial, pero en su versión gratuita. Será **Oracle Database 10g Express Edition**, que ofrece ser completamente gratuito para desarrollar y distribuir los desarrollos de la empresa, está disponible para Microsoft Windows y Linux, puede ser actualizado a versiones superiores de Oracle 10g y permite trabajar con diferentes lenguajes de programación.

Juan y María están muy interesados en aprender a manejar este sistema, saben que Oracle es una de las herramientas más potentes en el mundo de las bases de datos y están dispuestos a afrontar el reto.

¿Qué es Oracle Database 10g Express Edition?

Es un sistema de bases de datos libre para el desarrollo, implementación y distribución. Es un sistema para la iniciación, con un consumo reducido de recursos, basado en el producto Oracle Database 10g revisión 2. Su descarga es rápida y brinda un sistema de administración sencillo. Es un buen sistema de iniciación para desarrolladores en PHP, Java, XML y aplicaciones de código abierto, para administradores de bases de datos que necesitan una base de datos para su adiestramiento e implementación, para proveedores independientes de software o hardware que desean una base de datos inicial para distribuir libre de costes sus productos o para instituciones educativas o estudiantes que necesitan una base de datos libre con la que completar su curriculum.

Si quieres conocer más características destacables de este sistema de bases de datos, aquí puedes acceder a la hoja de especificación de Oracle 10g Express Edition (en Inglés).

<http://www.oracle.com/technetwork/database/express-edition/overview/dbxe-datasheet-130365.pdf>

¿Por dónde empezamos?

El primer paso que debemos dar es descargar el software necesario desde la página oficial de Oracle. A través del siguiente enlace podrás acceder a la zona de descarga de Oracle Database 10g Express Edition, regístrate, escoge el que se ajuste a tus necesidades y descárgalo en tu ordenador.

<http://www.oracle.com/technetwork/database/express-edition/downloads/index.html>

¿Cómo se realiza la instalación?

Para llevar a cabo la instalación del software descargado, dependiendo de tu sistema operativo, puedes visualizar alguno de los vídeos que te proponemos a continuación:

Instalación de Oracle Database 10g Express Edition bajo Windows 7

El vídeo comienza con los datos de la autora y la universidad a la que pertenece. A continuación, se accede al escritorio de un equipo con sistema operativo Windows 7, en el que ya se encuentra descargado el instalador de Oracle Database 10g Express Edition. Haciendo doble clic sobre dicho instalador, se inicia el asistente de instalación con una barra de progreso de color verde que se va completando. Una vez ha terminado de prepararse el instalador, aparece en pantalla una ventana de bienvenida del producto, se pulsa en el botón de siguiente y aparece el texto de aceptación de licencia de uso. Hay que seleccionar que se aceptan las condiciones y se pulsa en siguiente. Seguidamente, se solicita si se desea cambiar la ubicación de la instalación, en el vídeo se ha dejado la ubicación por defecto. En la siguiente ventana se solicita que introduzcamos una contraseña para la cuenta SYS y SYSTEM para la base de datos. Introducimos la contraseña elegida y se pulsa en siguiente. Se presenta ahora un resumen de lo que se va a instalar y dónde, para que lo confirmemos. Pulsamos en siguiente. El proceso de instalación se inicia y aparece una barra de progreso que se va completando. Una vez completada la instalación, el asistente nos pregunta si queremos iniciar la página inicial de gestión de la base de datos. Se abre el navegador web y aparece un formulario de login, en el que se introduce el nombre de usuario SYSTEM y la contraseña la que se

definió anteriormente. Una vez introducidos se pulsa en conectar y tras unos instantes, aparece un interfaz web para la gestión de la base de datos.

Instalación de Oracle Database 10g Express Edition bajo Ubuntu Linux.

Inicialmente, se muestran las librerías y requisitos necesarios para poder llevar a cabo la instalación en un equipo con sistema operativo Ubuntu. Se indica, a continuación, que se realice la descarga del paquete de instalación desde la página de Oracle, que se dejen los puertos por defecto de la instalación y que se realice un login con el usuario y contraseña establecidos durante este proceso. Una vez hechas estas indicaciones, se muestra una ventana del explorador Nautilus en el que aparece el paquete de instalación con extensión **.deb**. Al hacer doble clic sobre él, se inicia el instalador de paquetes de Ubuntu. Durante el proceso de instalación se solicita la contraseña de administrador del sistema. Una vez completada la instalación, se muestra en pantalla el comando para realizar la configuración de esta herramienta. Al teclear en una terminal dicho comando, se abre un asistente de configuración en modo texto en el que se van indicando diferentes parámetros de configuración: puerto http, puerto de escucha de la base de datos, contraseña del usuario SYSTEM y SYS y establecimiento de Oracle como aplicación que se inicia por defecto en el arranque. Tras unos instantes, la configuración se lleva a cabo y el interfaz de consola de comandos indica la dirección web que ha de insertarse en el navegador web para acceder al interfaz web de Oracle 10g Express Edition. Se carga en el navegador web dicha dirección, se abre sesión con el usuario SYSTEM y finalmente, el vídeo termina recordando que los requisitos iniciales es importante cumplirlos.

Gestión básica de datos en Oracle Database 10g Express Edition

A partir del inicio de sesión con el usuario SYSTEM, se accede al interfaz web de la base de datos. Se inicia el recorrido por el interfaz accediendo al explorador de objetos. Al pulsar sobre él aparece un menú de acciones, se selecciona crear y dentro de este submenú, se selecciona tabla. Se crea una tabla: carne, nombre, dirección y teléfono. A continuación, se pulsa en siguiente y se solicita cuál es el campo clave primaria de la tabla. En este caso, no se establece clave. Tampoco claves foráneas, ni restricciones. Se pulsa sobre el botón de crear y se visualiza la tabla y sus características en pantalla. El siguiente elemento del interfaz web que se analiza es el titulado SQL, al pulsar sobre su icono se muestra una consola de comandos SQL. A través de esta consola se realiza un ejemplo de inserción en la tabla mediante comandos. Una vez preparado el comando, se pulsa sobre ejecutar y se muestra el resultado de la ejecución de dicho comando de inserción. Para comprobar la inserción, a través de la misma consola, se lanza una consulta de datos y se obtiene el resultado que confirma la inserción, a través de la misma consola, se lanza una consulta de datos y se obtiene el resultado que confirma la inserción. Posteriormente, se ejecuta un comando de modificación de datos y una consulta asociada para confirmar la modificación. Por último, se lleva a cabo un borrado de datos mediante línea de comandos y consulta asociada para ver los resultados.

Administración simple de usuarios en Oracle Database 10g Express Edition

Utilizando el navegador web, en el interfaz web de la aplicación, se posiciona sobre el menú de administración. Se despliega un submenú en el que se selecciona usuarios de la base de datos y en su interior, gestión de usuarios. Seguidamente, se visualiza un único usuario con nombre HR que está bloqueado y cuya cuenta ha expirado. Se pulsa sobre el usuario y se muestra un formulario en el que pueden modificarse los datos básicos de dicho usuario, su clave, desbloqueo y privilegios. A continuación, una vez desbloqueado el usuario HR y asignada una nueva clave, se cierra sesión con el usuario SYSTEM y se entra con el usuario HR. En el mismo interfaz web inicial, se accede al explorador de objetos y puede verse que el usuario puede crear tablas u otros objetos, existiendo diferentes modelos para utilizar como base. Se cierra sesión con el usuario HR y se cierra el navegador. Ahora, en el escritorio del equipo se selecciona el icono de Equipo y se pulsa con botón derecho. Se selecciona la opción administrar y en servicios y aplicaciones, selecciona servicios y busca por orden alfabético los servicios asociados a Oracle. Encuentra los servicios OracleServiceXE y OracleXETSListener, los selecciona con doble clic y hace que no se inicien de forma automática al

iniciarse Windows 7. De este modo consigue que el arranque de Windows sea mucho más rápido al no iniciarse por defecto estos servicios.