UNIDAD 1: ALMACENAMIENTO DE LA INFORMACIÓN

Módulo Profesional: Bases de Datos

Índice

RESUMEN INTRODUCTORIO

En la presente unidad se estudiarán conceptos relacionados con el almacenamiento de la información. En este sentido, cobran cierto interés los sistemas de la información. Como tal, la información debe ser almacenada en archivos, para ser gestionada y estar accesible siempre que sea necesario. Esta accesibilidad también será estudiada en la unidad, así como los diferentes soportes que permiten almacenar archivos. Grosso modo, los sistemas capaces de almacenar y gestionar la información son los Sistemas Gestores de Bases de Datos (S.G.B.D.), para los cuales esta unidad analizará su función, sus objetivos y los componentes que incorpora. Asimismo, se hace necesario conocer la Arquitectura ANSI-SPARC, como base de muchos sistemas gestores, cuya estructura está dividida en 3 niveles: nivel interno, nivel conceptual y nivel externo. Existen diferentes modelos de bases de datos, en función de si están basados en registros u orientados a objetos. Independientemente de ellos, se analizará la importancia de las bases de datos distribuidas y su diferencia con las centralizadas, aportando ventajas y soluciones a problemáticas que pueden ocurrir con los sistemas que las implantan. Por último, se introduce el concepto de base de dato no relacional, para estudiar las diferencias con las relacionales.

INTRODUCCIÓN

Las bases de datos permiten que la información sea almacenada y recuperada de forma rápida y sencilla. En este sentido, son utilizadas en muchos ámbitos de la vida cotidiana. Muchas de las acciones que se realizan diariamente están íntimamente relacionadas con las bases de datos. Ejemplos de ello podría ser: la consulta de la disponibilidad de un libro de la biblioteca, la búsqueda de un contacto en una agenda telefónica, la investigación de información de cualquier tipo a través de un motor de búsqueda tipo Google, pasar un artículo por el lector en el supermercado, almacenar la información de un hospital, como puede ser el historial de los pacientes, la ficha de los trabajadores, etc.

CASO INTRODUCTORIO

La empresa de Paula ha sufrido un daño colateral. De un día para otro los servidores de bases de datos han dejado de funcionar por una serie de apagones y subidas de tensión, problema del cual no se conoce su causa. Tras este varapalo, Paula ha decidido migrar toda la información que ha podido recuperar a servidores de bases de datos distribuidos, con la intención de evitar el mismo problema en el futuro. Esta es una de las muchas razones por las que los sistemas distribuidos son más seguros que los centralizados, como en el caso de la empresa de Paula. Además, las copias de seguridad no se habían almacenado en diferentes soportes y mucha información no se ha podido recuperar. Es importante conocer este tipo de problemas sin llegar a sufrirlos como le ha ocurrido a Paula, para poder adelantarse a ellos y ofrecer soluciones desde el primer momento en el que se trabaja con información de suma importancia.

Al finalizar la unidad el alumnado:

- ✓ Aprenderá las diferentes técnicas existentes para el almacenamiento de la información.
- ✓ Será capaz de reconocer los elementos de las bases de datos analizando sus funciones.
- ✓ Conocerá las ventajas que aportan los Sistemas Gestores de Bases de Datos (SGBD).

1. CONCEPTOS BÁSICOS

En este tema se analizan los sistemas de almacenamiento que utilizan las bases de datos más comunes. Para ello es necesario conocer algunos conceptos, como los que se detallan a continuación.

1.1 Sistemas de información

Un sistema de información es un conjunto de actividades que administran la información relevante en una entidad, generalmente, una empresa. Se encarga de la distribución de los datos según unos determinados requerimientos, de la correcta compartición de la información entre sus usuarios y de su almacenamiento en soportes adecuados basados en ordenadores y en los avances de las telecomunicaciones. Los grandes volúmenes de información manejados por un sistema se agrupan en conjuntos más pequeños para poder ser nombrados y representados en las aplicaciones que los utilizan.

La unidad más pequeña que permite representar información es el **bit**, que admite dos valores: 0 y 1. Un grupo de 8 **bits** forman el **byte**. Cada dato al que puede hacerse referencia en un sistema de información se denomina **campo** y está formado por un grupo de **bytes**. Los datos que forman parte de una entidad común se agrupan en un **registro**, que está formado por un conjunto de **campos**. El campo o grupo de campos que identifican unívocamente a cada **registro** de un archivo se denomina campo **clave**. Todos los registros del mismo tipo forman un **archivo**. El conjunto de archivos que representa la información de un sistema forma una **base de datos**.

Imagen: Almacenamiento de la información

1.2 Estructura básica de almacenamiento: El archivo

Para almacenar la información se ofrece una estructura de datos de alto nivel llamada archivo que permite distribuir los datos en dispositivos externos de almacenamiento. Normalmente los archivos de datos serán almacenados en

registros de tipo lógico. Estos registros lógicos no son más que estructuras de datos formados por uno o más elementos llamados campos que constituyen una unidad para un determinado proceso. Cuando se necesita utilizar un archivo en una aplicación, se define el tipo de registro que lo formará y, posteriormente, lo utilizará según las operaciones que actúen sobre él el lenguaje de programación de la aplicación:

Nombre del archivo: CLIENTES
Nombre del registro: R_CLIENTES
Campo clave: NIF
Formato del registro:

Campo	Nombre	Tipo de datos	Longitud
1	NIF	Alfanumérico	10
2	APELLIDOS	Alfanumérico	20
3	NOMBRE	Alfanumérico	15
4	NACIMIENTO	Fecha	8

Ejemplo del contenido del archivo:

05695822F	MARTÍN PÉREZ	GONZALO	05121981
12548993U	LÓPEZ MARTÍNEZ	TERESA	29031989
05988346Y	GONZÁLEZ ALONSO	JUAN PEDRO	13072001
13449487X	FERNÁNDEZ GÓMEZ	YOLANDA	07101977

Las características más importantes de los archivos son:

- Residen en soportes externos (DD), por lo que su existencia no está limitada al tiempo de ejecución del programa que lo crea, sino que permanece cuando éste termina.
- Los datos pueden transportarse de un equipo a otro.
- Tienen capacidad de almacenamiento ilimitada ya que, aunque un soporte dispone de capacidad limitada, un archivo puede distribuirse en varios soportes.

La clasificación de los archivos puede hacerse atendiendo a la función que realizan y se pueden dividir de la siguiente forma:

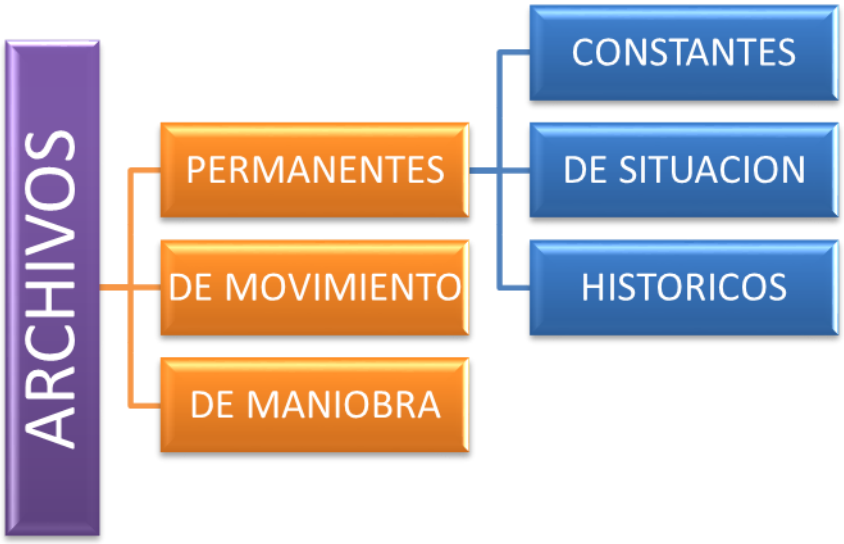


Imagen: Clasificación de los archivos según su función

- **Permanentes**: Sus registros varían poco en el tiempo. También se conocen con el nombre de archivos **maestros**. Se distinguen 3 tipos:
    - **Constantes**: Su información permanece prácticamente invariada, utilizándose como archivos de consulta.
    - **De situación**: Reflejan el estado o situación actual de una empresa o entidad. Se actualizan periódicamente para adaptarlos a una nueva situación. Son los "menos permanentes".
    - **Históricos**: Se obtienen de los anteriores cuando estos dejan de utilizarse. Se utilizan para estudios estadísticos o de consulta.

- **De movimientos**: En ellos se almacena temporalmente la información que se utiliza para actualizar los archivos de situación.
- **De maniobra**: Son archivos temporales creados durante la ejecución de un programa y borrados habitualmente al terminar el mismo. Por ejemplo, archivos intermedios utilizados en procesos de ordenación.

Las operaciones que pueden realizarse sobre un archivo son:

CREATE. Crear la estructura del archivo, establece la estructura y posición del archivo en el dispositivo de almacenamiento.

OPEN. Abrir un archivo creado para poder utilizarlo. Cuando el archivo ya existe, debe abrirse para trabajar con él, bien sea para consultarlo o para actualizarlo.

READ. Leer un registro de un archivo. Transfiere la información del registro actual al área de datos del programa que solicita la lectura.

WRITE. Escribir un registro en un archivo. Graba en el soporte de almacenamiento el contenido de un registro con los datos especificados en el área de datos del programa.

CLOSE. Cerrar un archivo cuando ya no va a utilizarse. Esta operación es obligatoria antes de concluir un programa. Si un archivo queda abierto, se corre el riesgo de perder información. Actualiza la situación real del archivo y elimina de memoria la tabla mantenida por el sistema para agilizar las operaciones de acceso al archivo.

RENAME. Renombrar un archivo. Permite cambiar el nombre a un archivo.

DELETE. Eliminar un archivo. Cuando deja de tener validez, es conveniente borrarlo del dispositivo de almacenamiento para no desperdiciar espacio.

COPY. Copiar un archivo. Consiste en duplicar la información de un archivo en otro.

EDIT. Editar un archivo. Permite modificar el contenido de un archivo. Se utiliza, sobre todo, en archivos de texto.

Indexar un archivo. Es una operación de alto nivel que permite dar a un archivo organización indexada. El acceso a los registros del archivo se hará a través de un archivo índice que estará ordenado por el campo que se utilizó para indexarlo (clave).

Imagen: Operaciones sobre los archivos

2. ACCESO A LA INFORMACIÓN DE LOS REGISTROS

Independientemente de cómo estén organizados los datos en un archivo, para las aplicaciones es importante la forma en que puede accederse a los registros. Existen básicamente 3 modos de acceso:

- Acceso secuencial
- Acceso directo
- Acceso por índice

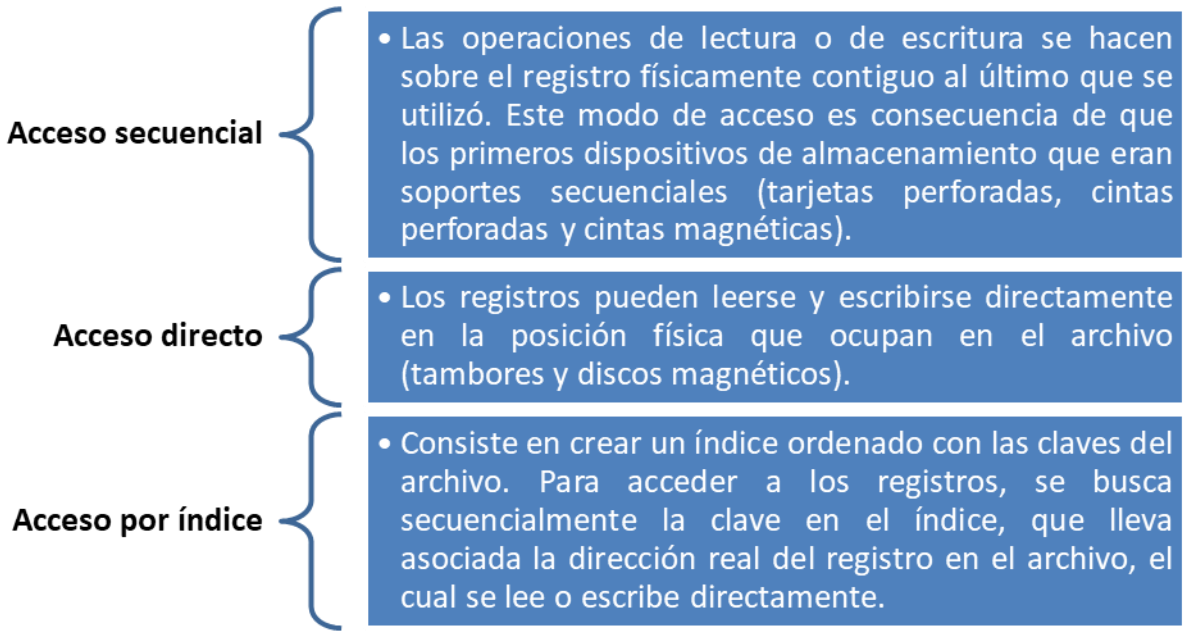


Imagen: Tipos de acceso a la información de los registros de un archivo

3. GESTIÓN DE ARCHIVOS EN SOPORTES

La gestión de archivos es una de las tareas principales que deben atender los sistemas operativos. El trabajo es distinto según el tipo de soporte que se utilice. Existen 2 tipos de soporte:

- **Secuenciales**: Los datos se graban unos a continuación de otros, de tal forma que el acceso a un dato se realiza accediendo sobre todos los datos que le preceden en el soporte. El ejemplo clásico es la cinta magnética.

- **Direccionables**: El espacio de almacenamiento se divide en espacios parciales direccionables individualmente, pudiendo acceder a un dato por la dirección en que está almacenado, sin que sea necesario pasar por los

datos almacenados en direcciones físicas anteriores. Por ejemplo, los CD-ROM, DVD, etc.

Imagen: Comparativa entre soporte secuencial y direccional

PARA SABER MÁS

Para ampliar información sobre soportes tanto direccionables como secuenciales visita la web:
http://www.elgrupoinformatico.com/repaso-las-tecnologias-almacenamiento-actuales-t19307.html

Actualmente, la información se almacena en soportes direccionables, dejando los secuenciales para realizar grandes copias de seguridad. Por ello, se explica cómo se distribuyen los archivos en discos.

El disco se divide en unidades mínimas de E/S llamadas clúster o bloques. A cada archivo se le asignan bloques según su organización y el método de asignación de bloques que utilice el sistema operativo. Existen 3 formas teóricas de asignar bloques:

- La **asignación contigua** de bloques requiere que todos los bloques de un archivo ocupen posiciones contiguas de disco para una eficiente utilización. El acceso secuencial es relativamente sencillo, porque basta con ir leyendo bloques en la secuencia en la que están en el disco. El acceso directo tampoco es complicado, conociendo la dirección del primer bloque del archivo. No obstante, este método de asignación tiene el problema de la predeterminación de los bloques que ocupará un archivo. También es posible que en el disco haya suficiente número de bloques para almacenar el archivo, pero no ocupen posiciones contiguas. Este problema se conoce con el nombre de fragmentación externa.

- La **asignación enlazada** fragmenta el archivo en bloques que pueden estar distribuidos aleatoriamente por el disco. Para conocer la secuencia, el sistema operativo guarda la dirección del primer bloque y cada bloque almacena en sus últimos bytes la dirección del siguiente bloque en secuencia. La dirección del último bloque es un valor prefijado de antemano. El acceso secuencial es muy sencillo, el directo es imposible.

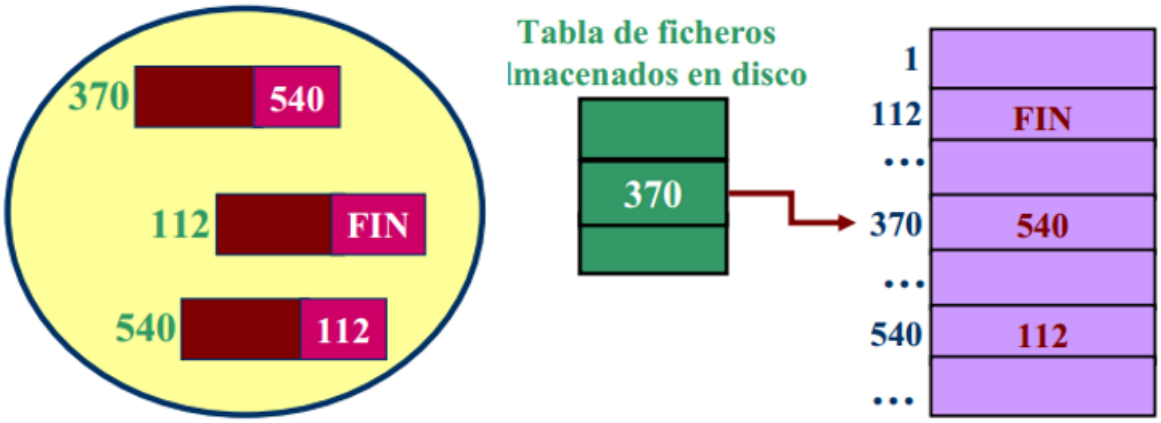


Imagen: Ejemplo de asignación enlazada

- La **asignación indexada** consiste en reunir todos los punteros en un bloque llamado índice. El bloque índice es una tabla de entradas, donde la entrada i-ésima tiene la dirección del bloque i-ésimo del archivo. El acceso secuencial se realiza recorriendo las entradas del bloque índice y el acceso directo a un bloque b se hace yendo directamente a la dirección que indica la entrada b del bloque índice. También facilita el acceso por índices.

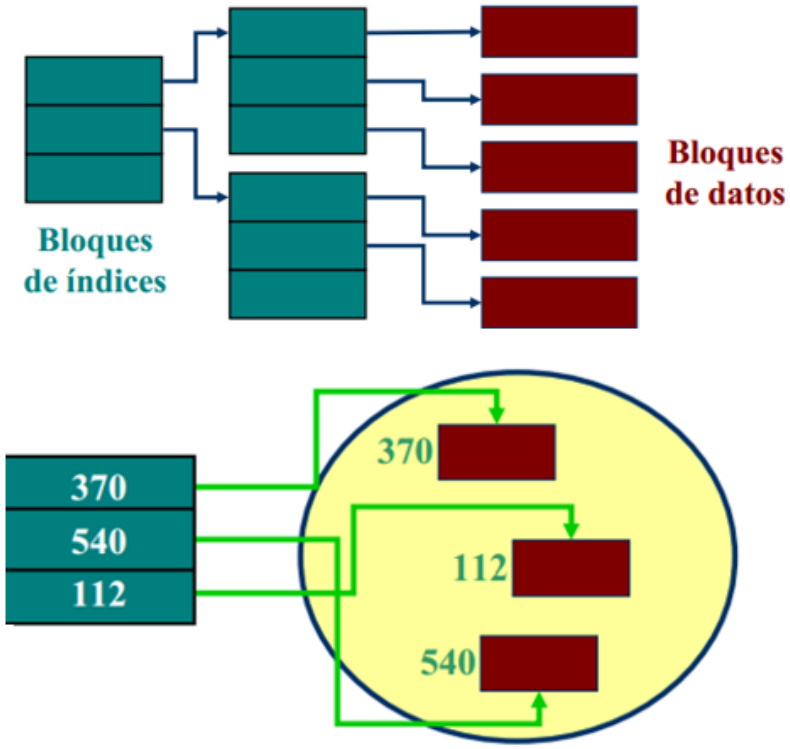
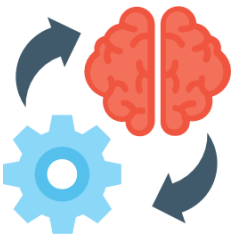


Imagen: Ejemplo de asignación enlazada



RECUERDA

1. Asignación contigua todos los bloques de un archivo están contiguos en el disco.
2. La asignación enlazada fragmenta el archivo en bloques que pueden estar distribuidos aleatoriamente por el disco.
3. La asignación indexada consiste en reunir todos los punteros en un bloque llamado índice.

4. SISTEMAS GESTORES DE BASES DE DATOS (S.G.B.D.)

Una base de datos es un conjunto de datos almacenados de forma organizada y estructurada en un soporte de información que es manejado por un ordenador. La información se guarda en archivos independientes integrados en la base y puede ser compartida por distintos usuarios que la utilicen para fines diferentes en instantes de tiempo que pueden coincidir. Sus componentes más importantes son el hardware, los datos, los usuarios y los procedimientos encargados del uso correcto de la información. Un sistema gestor de base de datos (SGBD) es un conjunto de programas que permiten la administración y gestión de la información de una base de datos.

Imagen: Componentes de un sistema gestor de Bases de Datos

PARA SABER MÁS

Para ampliar la información sobre las Bases de datos y los SGBD se recomienda:

http://www.um.es/docencia/barzana/IAGP/Iagp6.html

En estos sistemas, los programadores y usuarios no tienen que saber cómo está distribuida y organizada la información. De ello se encarga un conjunto de procedimientos que forma parte del sistema gestor de la base de datos. Una de sus funciones es proporcionar diferentes niveles de abstracción de la información, dependiendo del tipo de usuario que la maneja. Para la mayoría, se ocultan los detalles de la forma y el lugar en que están almacenados los datos, así como los procedimientos de recuperación y actualización de la información. También se encarga de conectar e implementar los distintos niveles de la arquitectura de la base de datos.

Imagen: Ejemplo de algunos SGBD existentes en el mercado

PARA SABER MÁS

Para ampliar información acerca de los SGBD disponibles en el mercado y una comparativa entre los SGBD comerciales y libres, visita a la web: http://www.slideshare.net/oruizz/comparacion-software-comercial-vs-libre-gestores-de-base-de-datos

El sistema de control de base de datos es el encargado de transformar los datos requeridos por los programas de aplicación en registros físicos a leer, hacer la petición al sistema operativo y, cuando la información está disponible, transferirla al área de trabajo del programa que la solicitó.

El administrador del sistema organiza y controla los recursos del sistema. Sus principales funciones son:

- Definir el esquema conceptual con el lenguaje de definición del SGBD.
- Controlar el acceso a la base de datos, concediendo permisos a los usuarios.
- Definir estrategias de recuperación frente a posibles fallos.

4.1 Objetivos

Los objetivos que debe cumplir una base de datos para ser lo más rápida, eficaz y polivalente son los siguientes:

- Debe independizar los datos de las aplicaciones. Es lo que se conoce como independencia física (se puede modificar el esquema físico sin que afecte a los superiores) e independencia lógica (si se modifica el esquema conceptual no es necesario modificar los programas de aplicación).

- Conseguir que los datos repetidos innecesariamente en la base sean los mínimos posibles (redundancia mínima de información).

- Suministrar mecanismos de seguimiento de las operaciones realizadas en la base de datos. Esto se consigue mediante procesos espías que mantienen archivos en los que se almacena la fecha y hora de conexión de usuarios, operaciones realizadas en cada sesión, datos modificados, etc.

- Proporcionar versatilidad en las posibilidades de búsqueda de información facilitando al usuario varios criterios.

- Asegurar la protección de los datos contra accesos no autorizados.

- Controlar la integridad de la información en la base de datos. Para ello, debe dar respuesta a posibles fallos de hardware, defectos en el código de los programas de aplicación, actualizaciones incompletas, inserción de datos incorrectos o no válidos, etc.

- Dotar de mecanismos para realizar copias de seguridad de la información.

- Conseguir un tiempo de respuesta suficientemente pequeño para evitar que el usuario se desespere. El tiempo de respuesta se define como el tiempo que transcurre desde que el usuario termina de realizar una petición al sistema hasta que empieza a recibir repuesta.

- Solucionar los problemas planteados por la concurrencia. Actualmente es posible que una base de datos esté compartida por varios usuarios situados en distintos terminales. Por ello, puede darse el caso de que dos o más usuarios distintos intenten actualizar de forma concurrente (en el mismo instante de tiempo) un registro determinado. Fundamentalmente, estos problemas son la actualización incorrecta y el bloqueo mutuo.

El primer problema se soluciona mediante una técnica llamada cierre, que consiste en poner un semáforo que se encuentre cerrado cuando el registro está

siendo actualizado y abierto en otro caso. Las transacciones que se encuentran con el semáforo cerrado se guardan en una cola de espera y se procesan cuando el semáforo cambia de estado.

El segundo problema es consecuencia de la solución al primero. Se produce bloqueo mutuo cuando dos o más transacciones se encuentran en una espera circular indefinida. Para resolver el problema se puede impedir que suceda, algo difícil que afectaría al rendimiento del SGBD. También puede detectarse y corregirlo.

PARA SABER MÁS

Para conocer el SGBD MySQL se propone visitar la web oficial de MySQL y en concreto este apartado:

https://dev.mysql.com/doc/

4.2 Componentes

El SGBD está dividido en 4 módulos que llevan a cabo sus funciones asociadas. Se compone del núcleo, lenguaje, utilidades y diccionario de datos.

Núcleo: Este módulo incorpora un conjunto de programas cuya labor es la de coordinar y controlar todo el funcionamiento del SGBD. Presenta una serie de características que se listan en la siguiente imagen.

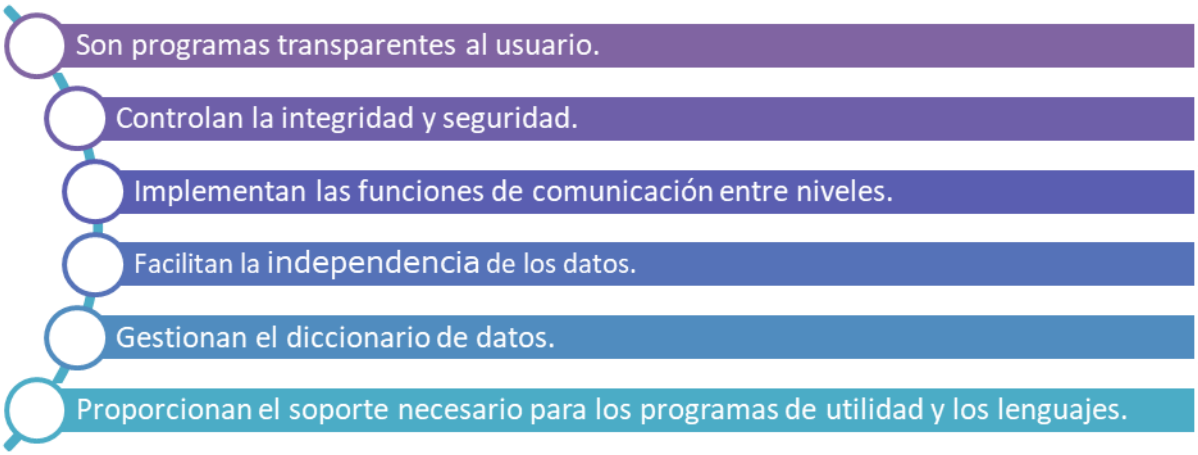


Imagen: Características del núcleo de un SGBD

Lenguajes: El SGBD proporciona lenguajes que permiten la definición y el manejo de los datos de la base. Cada SGBD tiene una estructura y organización

particular, pero todos tienen la característica común de ofrecer al administrador y a los usuarios 2 lenguajes:

- El lenguaje de descripción de datos (DDL) se usa para definir el esquema conceptual y los distintos subesquemas externos de la base de datos. Se ofrece establecer parte de la seguridad de la base de datos, permitiendo asignar derechos sobre las operaciones que pueden realizar usuarios.

- El lenguaje de manipulación de datos (DML) permite gestionar la información de la base de datos (añadir, eliminar, modificar registros y recuperar información de forma estructurada de la base de datos).

Imagen: Lenguajes de un SGBD

Los SGBD son capaces de procesar peticiones de manipulación de datos que se han formulado desde programas escritos en otros lenguajes de programación.

Utilidades: Aplicaciones que facilitan el trabajo a los usuarios y programadores. Tienen la característica común de incorporar un interfaz fácil de entender. Se basan en menús que guían al usuario para conseguir el objetivo final.

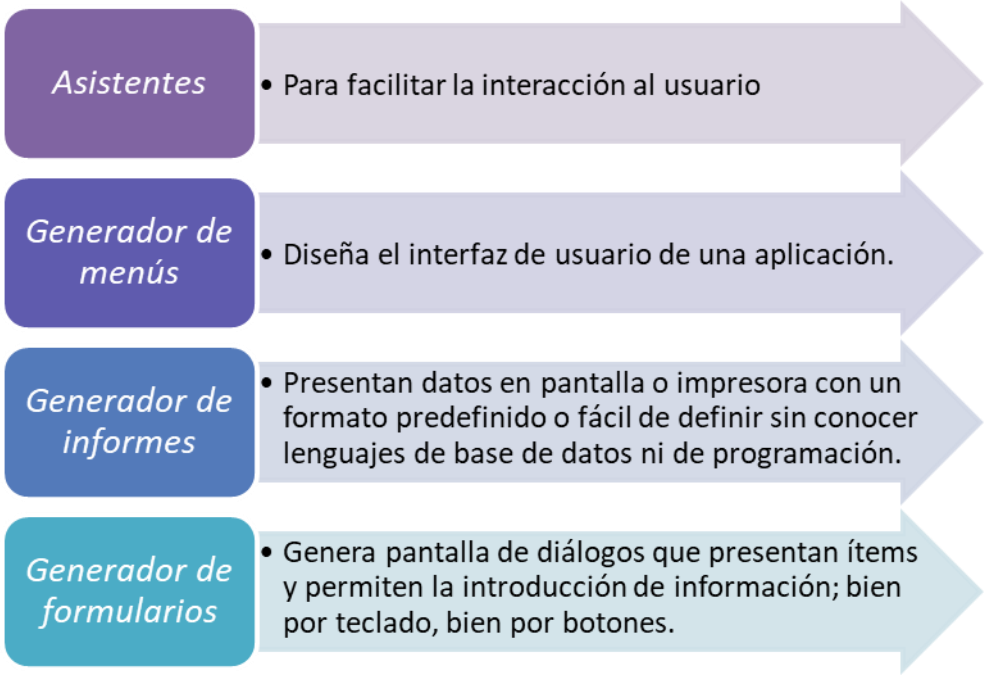


Imagen: Utilidades de un SGBD

Diccionario de datos: Es un almacén integrado en el que se guarda toda la información referente a la descripción, gestión e implantación de la base de datos. También se le conoce como catálogo del sistema. Contiene la descripción de los esquemas interno, conceptual y externo, tablas de usuarios con sus permisos respectivos, los programas de aplicación que utilizan, las operaciones que realizan dichos usuarios en el sistema, otros recursos implicados en el mismo, etc. Los diccionarios de datos están estructurados entres capas o niveles. Si la complejidad del sistema de base de datos lo requiere, las capas pueden subdividirse recursivamente. Las tres capas son:

- La capa de mayor nivel de abstracción se llama global, donde se reproduce la información común a todos los usuarios, incluido el administrador de la base de datos.

- A continuación, existe una capa intermedia que organiza la relación entre las capas globales y local. Pueden existir varias capas y, normalmente, se representan mediante vistas, que son percepciones individuales que tienen los usuarios de la base de datos.

- Finalmente, al más bajo nivel de abstracción, se encuentra la capa local, en la que se representan los datos como grupos de información específica.

ARTÍCULO DE INTERÉS

Artículo interesante sobre las 10 bases de datos más grandes del mundo:

http://www.20minutos.es/noticia/203609/0/bases/datos/grandes

5. ARQUITECTURA ANSI-SPARC

Uno de los objetivos de un sistema de base de datos es proporcionar una visión lo más abstracta posible de la información, es decir, ocultar detalles referentes a la forma en que los datos están organizados y almacenados. Existen 3 características importantes inherentes a los sistemas de bases de datos: la separación entre los programas de aplicación y los datos, el manejo de múltiples vistas por parte de los usuarios y el uso de un catálogo para almacenar el esquema de la base de datos.
En 1975, el comité ANSI-SPARC (American National Standard Institute – Standards Planning and Requirements Committee) propuso una arquitectura de 3 niveles para los sistemas de bases de datos. El objetivo de la arquitectura de tres niveles es el de separar los programas de aplicación de la base de datos física.

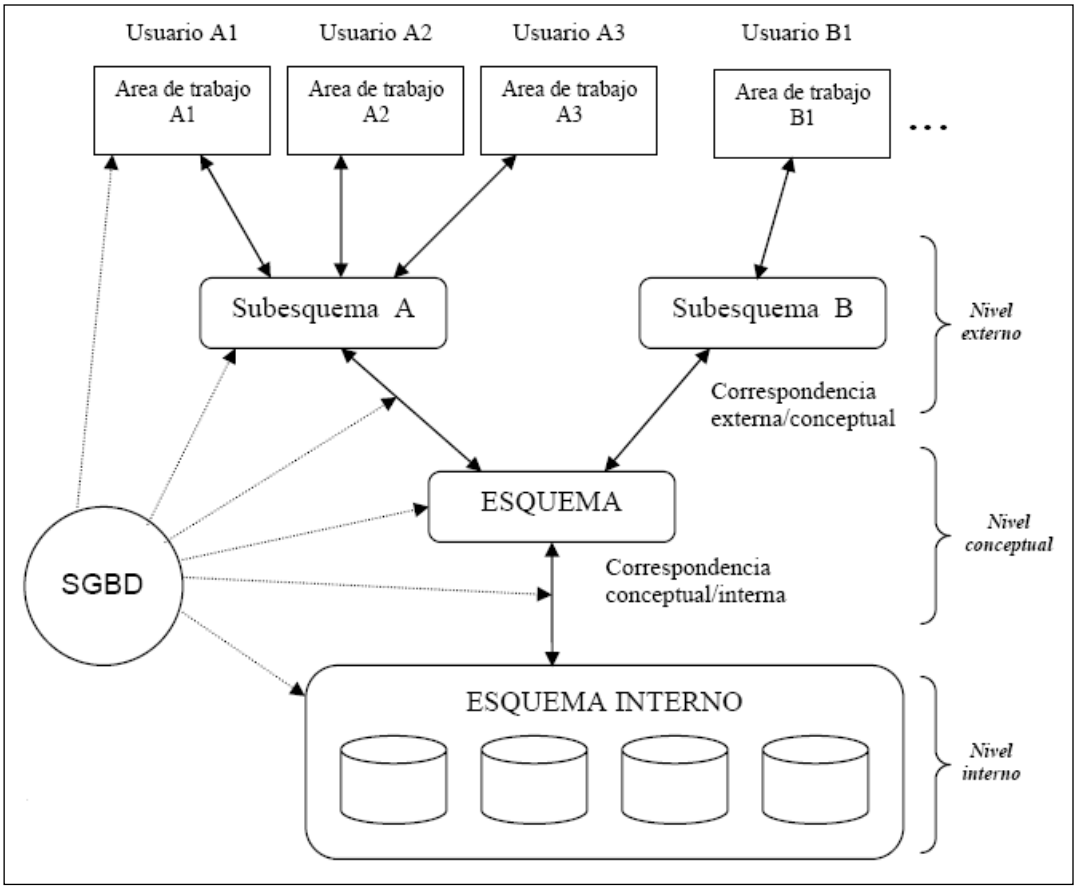


Imagen: Esquema de la arquitectura ANSI/SPARC (3 niveles)

La mayoría de los SGBD no distinguen del todo estos tres niveles. Algunos incluyen detalles del nivel físico en el esquema conceptual. En casi todos los SGBD que se manejan vistas de usuario, los esquemas externos se especifican con el mismo modelo de datos que describe la información a nivel conceptual,

aunque en algunos se pueden utilizar diferentes modelos de datos en el nivel conceptual y externo.

Hay que destacar que los 3 esquemas no son más que descripciones de los mismos datos, pero con distintos niveles de abstracción. Los únicos datos que existen realmente están a nivel físico, almacenados en un dispositivo como puede ser un disco. En un SGBD basado en la arquitectura de tres niveles, cada grupo de usuarios hace referencia exclusivamente a su propio esquema externo. Por lo tanto, el SGBD debe transformar cualquier petición expresada en términos de un esquema externo a una petición expresada en términos del esquema conceptual, y luego, a una petición en el esquema interno, que se procesará sobre la base de datos almacenada. Si la petición es de una obtención (consulta) de datos, será preciso modificar el formato de la información extraída de la base de datos almacenada, para que coincida con la vista externa del usuario. El proceso de transformar peticiones y resultados de un nivel a otro se denomina correspondencia o transformación. Estas correspondencias pueden requerir bastante tiempo, por lo que algunos SGBD no cuentan con vistas externas.

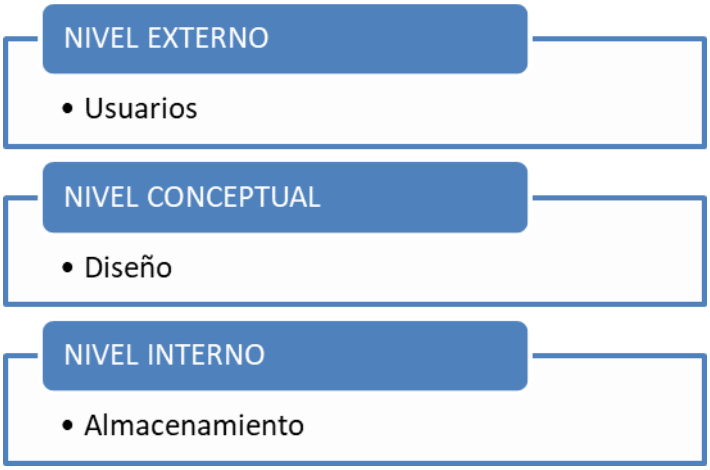


Imagen: Correspondencias de la arquitectura ANSI/SPARC (3 niveles)

La arquitectura de 3 niveles es útil para explicar el concepto de independencia de datos que se puede definir como la capacidad para modificar el esquema en un nivel del sistema sin tener que modificar el esquema del nivel inmediato superior. Se pueden definir 2 tipos de independencia de datos:

- La **independencia lógica** es la capacidad de modificar el esquema conceptual sin tener que alterar los esquemas externos ni los programas de aplicación. Se puede modificar el esquema conceptual para ampliar la base de datos o para reducirla. Si, por ejemplo, se reduce la base de datos eliminando una entidad, los esquemas externos que no se refieran a ella no deberán verse afectados.

- La **independencia física** es la capacidad de modificar el esquema interno sin tener que alterar el esquema conceptual (o los externos). Por ejemplo, puede ser necesario reorganizar ciertos ficheros físicos con el fin de mejorar el rendimiento de las operaciones de consulta o de actualización de datos. Dado que la independencia física se refiere sólo a la separación entre las aplicaciones y las estructuras físicas de almacenamiento, es más fácil de conseguir que la independencia lógica.

En los SGBD que tienen la arquitectura de varios niveles es necesario ampliar el catálogo o diccionario, de modo que incluya información sobre cómo establecer la correspondencia entre las peticiones de los usuarios y los datos, entre los diversos niveles. El SGBD utiliza una serie de procedimientos adicionales para realizar estas correspondencias haciendo referencia a la información de correspondencia que se encuentra en el catálogo. La independencia de datos se consigue porque al modificarse el esquema en algún nivel, el esquema del nivel inmediato superior permanece sin cambios, sólo se modifica la correspondencia entre los dos niveles. No es preciso modificar los programas de aplicación que hacen referencia al esquema del nivel superior. Por lo tanto, la arquitectura de 3 niveles puede facilitar la obtención de la verdadera independencia de datos, tanto física como lógica. Sin embargo, los dos niveles de correspondencia implican un gasto extra durante la ejecución de una consulta o de un programa, lo cual reduce la eficiencia del SGBD. Es por esto que muy pocos SGBD han implementado esta arquitectura completa.

La arquitectura más estándar y, por tanto, la más utilizada, es la que hace una división en niveles de la base de datos. Se consideran tres niveles según la perspectiva desde la que sea vista la información.

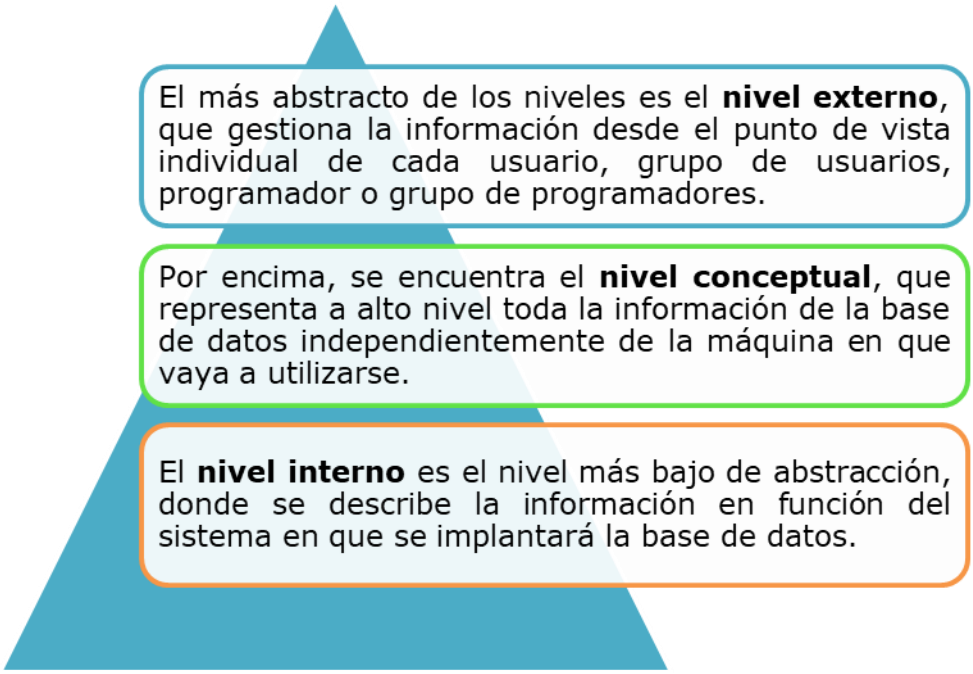


Imagen: Nivel externo, conceptual e interno

5.1 Nivel interno

En el nivel interno se describe la **estructura física** de la base de datos mediante la definición del esquema interno. Este esquema especifica todos los detalles para el almacenamiento de la base de datos, así como los métodos de acceso. En el esquema interno se especifican:

- Los archivos que contienen la información y su organización.
- La forma de acceder a los registros.
- El tipo y longitud del registro.

5.2 Nivel conceptual

Este nivel está asociado al **esquema conceptual**, donde se definen todos los datos que intervendrán en el sistema. Se obtiene a partir de los requerimientos de los usuarios potenciales del sistema de base de datos a implantar, sin importar la forma ni el lugar en el que se almacenarán y recuperarán los datos. Contiene los datos elementales (campos), los datos compuestos (registros), las relaciones existentes entre campos elementales y compuestos, las reglas que rigen el funcionamiento de la empresa, etc.

En el nivel conceptual se describe la estructura de toda la base de datos para una comunidad de usuarios (todos los de una empresa u organización). Este

esquema oculta los detalles de las estructuras de almacenamiento y se concentra en describir entidades, atributos, relaciones, operaciones de los usuarios y restricciones. El esquema conceptual contiene:

- Los registros, que representan las entidades del mundo real (por ejemplo: empleados, departamentos, etc.).
- Los campos, que son los atributos de estas entidades (por ejemplo, para la entidad empleado: nombre, dirección, DNI, sueldo, etc.)
- Relaciones entre las distintas entidades (un empleado pertenece a un departamento, un departamento contiene varios empleados, etc.)
- Podrían incluirse verificaciones de integridad (por ejemplo: un empleado no podría pertenecer a un departamento que no existe).

5.3 Nivel externo

Es el conjunto de **percepciones individuales** de la base de datos. Cada visión individual se denomina subesquema o vista. Un subesquema podrá ser compartido por varios usuarios y cada usuario tendrá la posibilidad de acceder a distintos subesquemas. Al crear un subesquema, es posible mezclar campos de distintos registros, omitir campos, cambiar el orden de los campos, añadir campos que puedan ser calculados a partir de los descritos en el esquema conceptual, etc.

En el nivel externo se describen varios esquemas externos o vistas de usuario. Cada esquema externo describe la parte de la base de datos que interesa a un determinado grupo de usuarios y oculta a ese grupo el resto de la base de datos. En este nivel se puede utilizar un modelo conceptual o un modelo lógico para especificar los esquemas. Para una base de datos específica, hay un único esquema interno y conceptual, pero puede haber varios esquemas externos, cada uno definido para uno o varios usuarios.

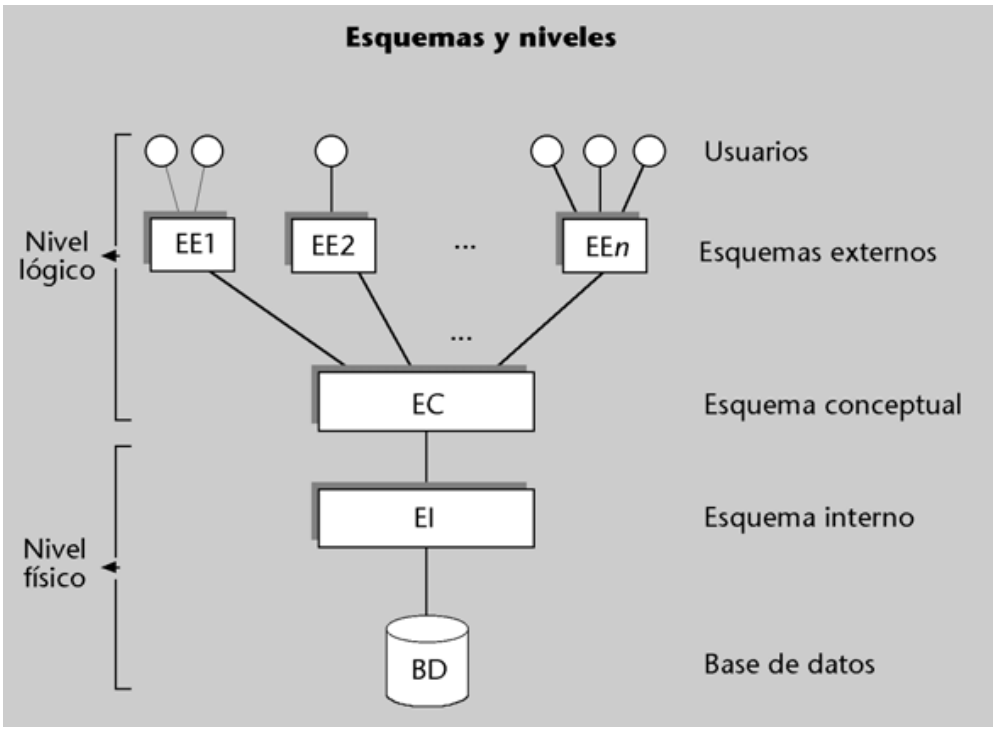


Imagen: Esquemas y niveles ANSI-SPARC

6. MODELOS DE BASES DE DATOS

Una base de datos almacena la información de forma estructurada. La manera en que ésta se almacene dará origen a un tipo u otro de base de datos. Cada conjunto de reglas y mecanismos de abstracción utilizados para la definición de los datos se corresponde con un modelo lógico de datos; estos modelos se dividen en dos grupos: los basados en objetos y los basados en registros.

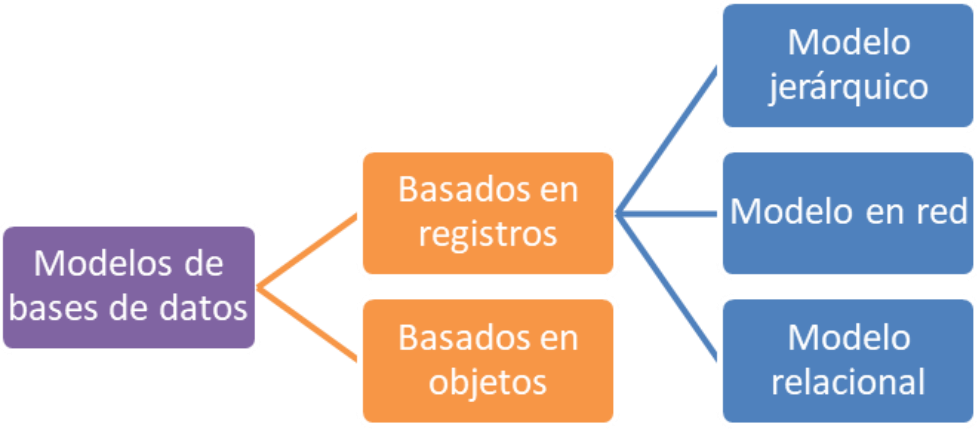


Imagen: Modelos de bases de datos basados en registros y en objetos

6.1 Modelos basados en registros

Describen los datos a nivel conceptual y físico. Se caracterizan porque la base de datos está estructurada en registros de varios tipos, cada uno con un número fijo de campos. Estos modelos han ido evolucionando con el tiempo, pasando del modelo jerárquico y en red, al modelo relacional, más extendido en la actualidad.

Modelo Jerárquico: Es el primer modelo de SGBD que se implantó (años 70), y muchos de los actuales modelos se basan en él. Utiliza estructuras con forma de árbol para la representación lógica de los datos. Dicho árbol está compuesto por una jerarquía de elementos llamados nodos, que se representa con la raíz en la parte superior. Cada nodo simboliza un tipo de registro (llamado segmento), que contiene un determinado número de campos. Un nodo tiene un único padre, y puede tener varios registros hijos. Cada nodo contiene los campos que son comunes a todos sus hijos. En el siguiente ejemplo, para un registro de tipo Club tiene relacionados varios registros de tipo "Jugador".

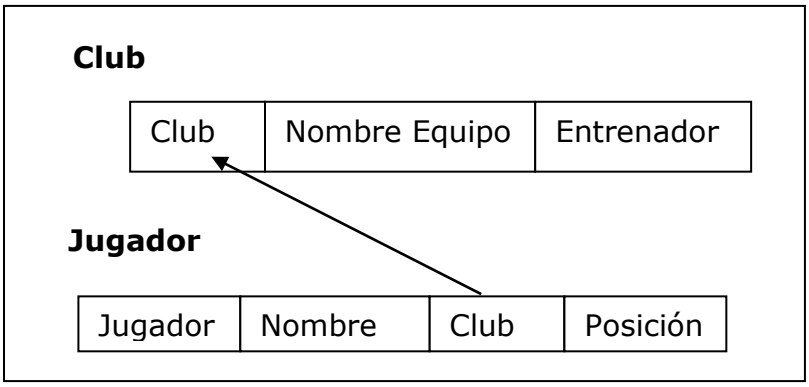


Imagen: Ejemplo de modelo jerárquico

Modelo en Red: Es un modelo de datos propio de los años 70. Consiste en una estructura de nodos conectados entre sí, como en el modelo jerárquico, con la diferencia de que un nodo puede tener varios padres. Este modelo permite representar cualquier tipo de relación, incluyendo las relaciones de varios a varios y las reflexivas, que no incluía el modelo jerárquico. Además, permite más de una relación entre nodos.

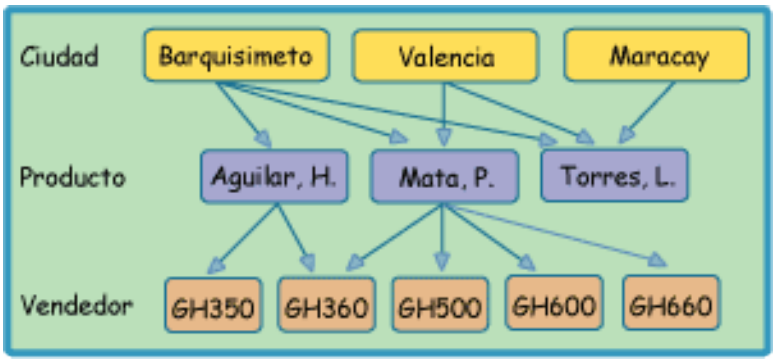


Imagen: Ejemplo de modelo en red

Modelo Relacional: Codd planteó a mediados de los años 70 una alternativa las bases de datos jerárquicas y en red, con la que pretendía obtener más flexibilidad y rigor en el tratamiento de los datos. Una base de datos relacional está formada por tablas. Una tabla es una estructura bidimensional formada por una sucesión de registros del mismo tipo, cada uno de ellos compuesto por un número fijo de campos. Las tablas se caracterizan porque:

- No pueden contener campos repetidos.
- No pueden existir registros duplicados, por lo que tienen un campo clave que identifica al registro y lo hace único.

Imagen: Ejemplo de modelo relacional (tablas clientes y pedidos)

NIF	Nombre	Departamento	Sueldo
25324117J	José García	Informática	12.000
23451899K	María González	Ventas	11.000

Ejemplo de tabla del modelo relacional

PARA SABER MÁS

Para ampliar información y ver ejemplos sobre el modelado de datos:

http://www.desarrolloweb.com/articulos/modelos-base-datos.html

6.2 Modelos orientados a objetos

Los sistemas de gestión de bases de datos orientados a objetos (SGBDOO) son polivalentes, ya que facilitan las funciones de los SGBD, algunos de los lenguajes de programación y otras propias del desarrollo de sistemas orientados a objetos. Los SGBDOO están siendo objeto de estudios, y mediante su evolución se espera que desplacen a los sistemas relacionales. El principal obstáculo con el que se han encontrado es el aumento de complejidad del modelo, así como el hecho de proporcionar un mayor número de funciones.

- **Objetos e identidad**: Cada entidad del mundo real es modelada como un objeto. Cada objeto tiene un identificador único y está asociado a un estado que se representa por los valores de sus atributos y un comportamiento que se define por procedimientos llamados métodos que actúan sobre él.

- **Objetos complejos**. Los valores de los atributos de un objeto pueden ser objetos.

- **Encapsulamiento**: Cada objeto tiene definidos en su interior los métodos y la interfaz para tener acceso y capacidad de manipulación de dicho objeto.

- **Clases**: Todos los objetos definidos por los mismos atributos y métodos forman una clase. Cada objeto que pertenece a una clase se dice instancia de la clase.

- **Herencia**: Una clase (subclase) se puede definir como una especialización de otras clases (superclases) existentes, por lo que heredará sus atributos y métodos.

- **Sobrecarga**: Una operación puede tener asociados distintos métodos. El sistema decide qué método realiza la operación. Por ejemplo, un objeto

puede incluir la operación resta (que toma 2 parámetros, hace su diferencia y retorna el resultado) y tener varios métodos para implementar la operación (uno para parámetros enteros, otro para restar fechas, otro para restar cadenas, etc.).

La arquitectura de los SGBDOO está basada en un enfoque cliente-servidor donde el servidor soporta las funciones del SGBD. El modelo de datos se apoya en los conceptos de clase, objeto y función. No hay distinción entre atributos y métodos, ya que ambos se tratan como funciones. Las funciones heredadas pueden ser redefinidas, de modo que dos clases pueden tener una función con el mismo nombre, pero con definiciones diferentes. Los 2 elementos principales de la arquitectura son, por un lado, el administrador de objetos que implementa el modelo orientado a objetos y facilita el soporte para definir esquemas y realizar consultas; y, por otro, el administrador de almacenamiento externo que actualmente se implementa sobre un modelo relacional.

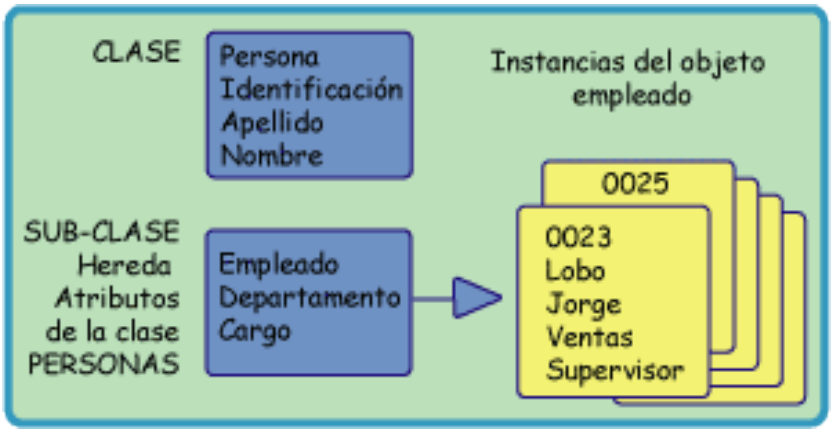


Imagen: Ejemplo de modelo orientado a objetos

7. BASES DE DATOS DISTRIBUIDAS

Una **Base de Datos Distribuida (BDD)** es una colección de datos distribuidos en diferentes nodos de una red de computadoras.

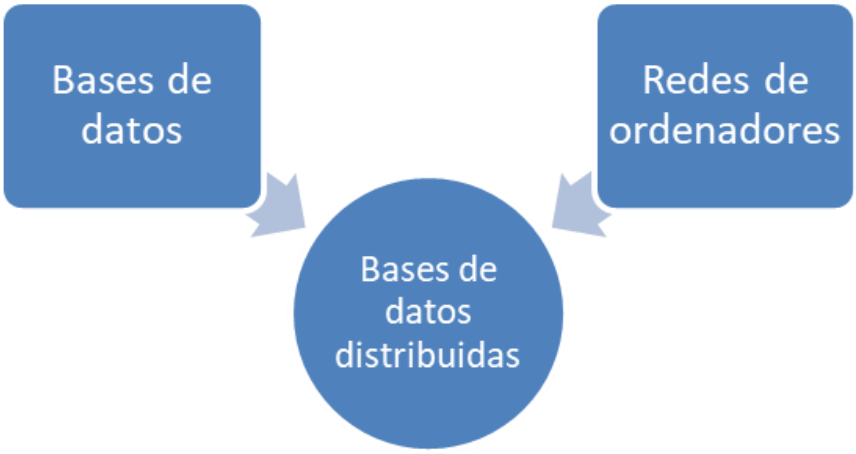


Imagen: Bases de datos distribuidas

Cada sitio de la red es autónomo, puede ejecutar aplicaciones locales y al menos una aplicación global, lo cual requiere el acceso a datos, ubicados en varios sitios, usando un subsistema de comunicación. Para el diseño de BDD se han definido dos grandes estrategias: el enfoque Top-Down y el Bottom-Up. En el enfoque Top-Down se comienza diseñando el esquema global, luego se concibe la fragmentación de la base de datos y la localización de los fragmentos en los sitios. Se completa ejecutando, en cada sitio, el diseño físico de los datos. Por otro lado, el enfoque Bottom-Up se basa en la integración de esquemas ya creados en un esquema global a partir de las bases de datos existentes.

Los Sistemas de Bases de Datos Distribuidas representan más naturalmente la estructura geográficamente descentralizada de una organización, aumentan la disponibilidad de los datos, reducen el tráfico de comunicación y es justificable, además, por el abaratamiento de los costos en el equipamiento y la infraestructura de comunicaciones de las redes de computadoras. El diseño de las Bases de Datos Distribuidas posee las fases del diseño centralizado y cuenta, además, con dos nuevos problemas que caracterizan el proceso de distribución de datos, e incluyen la determinación de: cómo dividir la base de datos en componentes para localizarlos en diferentes sitios, qué cantidad de datos debe ser replicados y cómo deben los fragmentos replicados ser localizados.

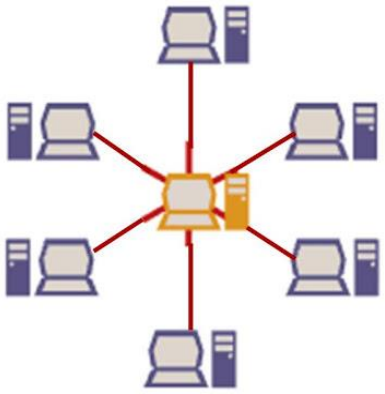


Imagen: Base de datos distribuidas

7.1 Ventajas y Desventajas de los sistemas distribuidos

Ventajas:

- El acceso a los datos es más rápido debido a que los datos se localizan más cercanos al lugar donde se utilizan.
- El procesamiento es rápido debido a que varios nodos intervienen en el procesamiento de una carga de trabajo, nuevos nodos se pueden agregar fácil y rápidamente.
- La probabilidad de que una falla en un solo nodo afecte al sistema es baja y existe una autonomía e independencia entre los nodos. Control local de los datos con que se interactúa.
- Mayor tolerancia a los fallos.

Desventajas:

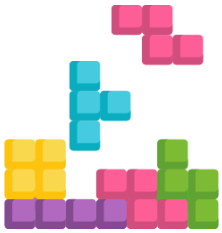
- Es más complicado el control y la manipulación de los datos.
- Es compleja el aseguramiento de la integridad de la información en presencia de fallas no predecibles tanto de componentes de hardware como de software. La integridad se refiere a la consistencia, validez y exactitud de la información.
- El control de concurrencia y los mecanismos de recuperación son mucho más complejos que en un sistema centralizado dado que los datos pueden estar replicados.

La distribución de la base de datos requiere determinar la fragmentación y la localización. Existen dos alternativas para fragmentar datos: fragmentación horizontal (FH) y fragmentación vertical (FV). La combinación de las anteriores resulta en una fragmentación híbrida. Es importante seguir tres reglas, las cuales aseguran que la base de datos no tenga cambios semánticos durante la fragmentación: completitud, reconstrucción y disyunción.

7.2 Sistema de Gestión de Base de Datos Distribuida

Un **sistema de gestión de bases de datos distribuidas** (SGBDD) es un Sistema de Gestión de bases de datos que gestiona la base de datos distribuida.
Funcionalidades adicionales de un SGBDD:

- Accede a sitios remotos y transmite consultas y datos a través de varios sitios mediante una red de comunicación.
- Almacena el esquema de distribución y replicación de los datos en el catálogo del sistema.
- Establece las estrategias de ejecución de las consultas y las transacciones que acceden a los datos en más de un sitio.
- Decide sobre cual copia de los datos replicados acceder.
- Mantiene la consistencia de las copias de los datos replicados.
- Realiza la recuperación ante los fallos.



EJEMPLO PRÁCTICO

Se dispone de una entidad bancaria la cual maneja datos de millones de usuarios. Es necesario que dicha entidad disponga de varios sitios donde albergar esa información de manera que no se pierda en ningún momento.

Solución:
Cualquier modificación de los datos implica que éstos se varíen en todos los ordenadores de la red de la base de datos distribuida.

Las BDD pueden ser:

- **Homogéneas**: Todos los sitios tienen el mismo SGBD, son conscientes de la existencia de los demás sitios y cooperan en el procesamiento de las solicitudes. Los sitios locales mantienen un mismo esquema y SGBD.

- **Heterogéneas**: Cada sitio puede tener un SGBD distinto, así como esquemas diferentes. Puede que algunos sitios no conozcan a otros. Puede que solo ofrezcan facilidades limitadas para la cooperación en el procesamiento de transacciones.

7.2.1 Problemas a resolver en las bases datos distribuidas

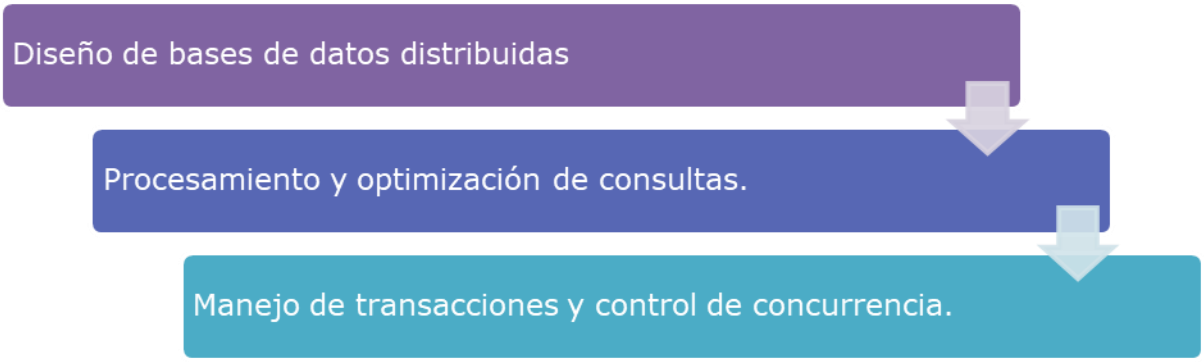


Imagen: Problemas de las bases de datos distribuidas

Etapas del diseño: La etapa diferenciadora entre el diseño de una base de datos centralizada y una base de datos distribuidos es el diseño de la distribución, que consta de dos actividades:

- **Fragmentación**: Decidir "como" se divide la base de datos y en "que" partes.
- **Asignación**: Decidir "donde" se ubica cada parte, así como si se tiene replicación de datos.

Aspectos a considerar en el diseño de una base de datos distribuida:

- **Fragmentación**: Una relación puede ser dividida en un número de sub-relaciones, denominadas fragmentos, los cuales son entonces distribuidas.
- **Asignación**: cada fragmento debe ser almacenado en un sitio en base a una distribución óptima.
- **Replicación**: El SGBDD puede mantener una copia de un fragmento en diferentes sitios.

Fragmentación: El problema de fragmentación se refiere al particionamiento de la información para distribuir cada parte a los diferentes sitios de la red.

Objetivos de la fragmentación: El objetivo de la fragmentación consiste en dividir la relación en un conjunto de relaciones más pequeñas tal que algunas de las aplicaciones de usuario sólo hagan uso de un fragmento. Sobre este marco, una fragmentación óptima es aquella que produce un esquema de división que minimiza el tiempo de ejecución de las aplicaciones que emplean esos fragmentos. La unidad de fragmentación ideal no es la tabla sino una subdivisión de ésta. Esto es debido a:

- Las aplicaciones usan vistas definidas sobre varias relaciones, es decir, se forman a partir de "trozos" de varias tablas. Si se consigue que cada una de las vistas esté definida sobre subtablas locales (o en su defecto lo más

"cerca" posible) a cada aplicación, es de esperar un incremento en el rendimiento.

- Si múltiples vistas de diferentes aplicaciones están definidas sobre una tabla no fragmentada, se tiene:
    - Si la tabla no está replicada entonces se produce generación de tráfico por accesos remotos.
    - Si la tabla está replicada en todos o algunos de los sitios donde residen cada una de las aplicaciones entonces la generación de tráfico innecesario es producida por la necesidad de la actualización de las copias.

Ventajas y desventajas:

Ventajas:

- Se permite el procesamiento concurrente de transacciones ya que no se bloquean tablas enteras sino subtablas, por lo que dos consultas pueden acceder a la misma tabla a fragmentos distintos.
- Se permite la paralelización de consultas al poder descomponerlas en subconsultas, cada una de la cuales trabajará con un fragmento diferente incrementándose así el rendimiento.

Desventajas:

- Degradación del rendimiento en vistas definidas sobre varios fragmentos ubicados en sitios distintos (es necesario realizar operaciones con esos trozos lo cual es costoso).
- El control semántico se dificulta y el rendimiento se degrada debido que la verificación de restricciones de integridad (claves ajenas, uniques, etc.) implican buscar fragmentos en múltiples localizaciones.

7.2.2 Las 12 reglas de un SGBDD

El principio fundamental de las Bases de Datos Distribuidas o regla cero plantea que: "Desde el punto de vista del usuario, un sistema distribuido deberá ser idéntico a un sistema no distribuido".

La regla cero conduce a las 12 reglas restantes. Todas las reglas no son independientes entre sí, ni tienen igual importancia, pero son útiles para entender la tecnología distribuida.

- **1. Autonomía Local**: Los sitios de un sistema distribuido deben ser autónomos, La autonomía Local Implica:
    - Propietario local.

- Administración local.
- Responsabilidad local.
- Integración local.
- Representación local.

- **2. No dependencia de un sitio central**, no debe existir un único sitio, ya que implicaría:
  - Cuello de botella.
  - Vulnerabilidad.

- **3. Operación continua.**
  - Adición de elementos.
  - Actualización de versiones.

- **4. Independencia de Localización**, el usuario desconoce dónde están físicamente los datos.

- **5. Independencia de fragmentación**. Deseable porque simplifica los programas de los usuarios y sus actividades en la terminal.

- **6. Independencia de réplica**. La creación y destrucción de réplicas debe hacerse transparente al usuario, la réplica proporciona:
  - Ventajas: Mayor Prestación, ya que los datos son locales y Mayor disponibilidad, ya que los datos son accesibles siempre.
  - Desventajas: Hay que propagar las actualizaciones.

- **7. Procesamiento distribuido de consultas**. Los Sistemas relacionales brindan herramientas de consulta muy eficientes.
  - Varias maneras de trasladar los datos.

- **8. Manejo distribuido de transacciones**:
  - Transacción distribuida: varios agentes de la transacción en varios lugares.
  - Control de recuperación: una transacción atómica. Todos los agentes avanzan o retroceden juntos.
  - Control de concurrencia: Bloqueos mediante paso de mensajes.

- **9. Independencia con respecto al equipo**. El SGBD se ejecutará igual sea cual sea el equipo.

- **10. Independencia con respecto al Sistema Operativo**. El SGBD debe ser multioperativo sin afectar al usuario.

- **11. Independencia con respecto a la Red**. El SGBD debe soportar múltiples redes sin afectar al usuario.

- **12. Independencia con respecto al SGBD**. Se pueden manejar distintas copias de SGBD si manejan la misma norma estándar de SQL: Oracle, Informix, Multibase, etc.

8. BASES DE DATOS NO RELACIONALES

En los últimos años, el avance de la tecnología ha llegado a impulsar un tipo de base de datos que, en cierto modo, ha roto con el esquema tradicional de las bases de datos relacionales. Las **bases de datos no relacionales** o NoSQL han supuesto una revolución en el sector del almacenamiento de la información y son muchas las empresas que están migrando sus datos a ellas. Este tipo de bases de datos son verdaderamente útiles en las siguientes situaciones:

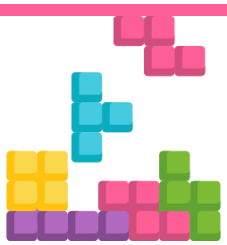
- Cuando una aplicación necesita almacenar o acceder a grandes cantidades de información en poco tiempo se necesita una base de datos que con alta velocidad. Este tipo de bases de datos son bastante más rápidas que las relacionales, con capacidad de gestionar una gran cantidad de peticiones que requieren muchas operaciones por segundo.
- Las bases de datos NoSQL permiten almacenar grandes cantidades de información, a diferencia de las relacionales, con capacidad más limitada.
- En las bases de datos relacionales, el esquema de la información que se almacena queda predefinida de antemano. Las bases de datos NoSQL, en cambio, permiten almacenar diferentes tipos de información, aportando gran flexibilidad en este sentido.
- Este tipo de bases de datos, como su propio nombre indica, no hace uso del lenguaje SQL tradicional, a diferencia de las relacionales.

Los tipos de bases de datos NoSQL son clave-valor, documentales, en grafo y orientadas a objetos. Como ejemplos de este tipo de bases de datos se encuentran: Cassandra, Redis, MongoDB y CouchDB.

PARA SABER MÁS

Para ampliar información sobre este tipo de bases de datos, acceder al siguiente enlace:

https://www.acens.com/wp-content/images/2014/02/bbdd-nosql-wp-acens.pdf



EJEMPLO PRÁCTICO

Imagine que se dispone de una colección de alumnos, cuya información se recoge en base a los siguientes criterios: nombre, apellidos, fecha de nacimiento y, a su vez, en un conjunto de módulos o asignaturas, los cuales presentan un nombre y un número de créditos. Es posible que de un alumno no se disponga de la misma información que de otro, es decir, haya campos no conocidos. ¿Qué tipo de base de datos utilizaría?

Solución:
Para estos casos una buena solución sería MongoDB, una base de datos no relacional ya que lo que se almacena es información en base a estructuras.

RESUMEN FINAL

En esta unidad se han repasado tanto las principales características de los sistemas de almacenamiento, como las organizaciones de archivos. Se han analizado los distintos modelos de datos, así como los sistemas de gestión de bases de datos que permiten implementar dichos modelos, atendiendo a los 2 tipos de sistemas soportados: distribuidos y centralizados.

Con respecto a las bases de datos distribuidas, son cada vez más usadas por las empresas y suponen una ventaja competitiva frente a las centralizadas, siempre y cuando la empresa en cuestión tenga necesidad de usar una base de datos con las características propias de los sistemas distribuidos. Lo más habitual es disponer de varias sedes y tener que manejar información común, para lo cual las bases de datos distribuidas son especialmente útiles.