**UNIDAD 1. ALMACENAMIENTO DE LA INFORMACIÓN**

Introducción

En nuestro día a día, tanto personal como profesional, interactuamos constantemente con datos de diversas formas y tamaños.

¿Alguna vez te has detenido a pensar cómo y dónde se gestionan y almacenan estos datos que nos son tan vitales? Tal vez no lo hayas notado, pero las bases de datos juegan un papel crucial en casi todas nuestras actividades cotidianas, y el manejo de estos datos se ha convertido en un elemento imprescindible en nuestra sociedad.

¿Aún no te has convencido de su importancia? Veamos algunos ejemplos:

• Al buscar un libro en una biblioteca o tienda en línea.   
• Al realizar una transacción con tu tarjeta de crédito.   
• Al utilizar una aplicación de mapas para llegar a un destino.   
• Al reservar un vuelo o alojamiento para tus próximas vacaciones.   
• Al recibir recomendaciones personalizadas en una plataforma de streaming.   
• Al guardar tus contactos en tu teléfono móvil.

La lista podría continuar, pero el punto parece claro: vivimos en un mundo impulsado por datos. Y en medio de este mar de información, se necesita de profesionales capacitados y competentes que sepan cómo manejar y trabajar con estas bases de datos.

Este módulo está diseñado para darte una comprensión sólida de las bases de datos y su relevancia en el desarrollo de aplicaciones. En esta primera unidad, comenzaremos nuestra travesía desde los fundamentos, explorando los sistemas de almacenamiento basados en ficheros.

A medida que progresamos, desglosaremos los conceptos y definiciones esenciales asociados con las bases de datos, exploraremos sus diferentes modelos y tipos, y examinaremos las funciones y características de los sistemas gestores de bases de datos.

Para culminar, nos familiarizaremos con las herramientas prácticas que nos permitirán manejar efectivamente estas bases de datos. Así que, ¡prepárate para un emocionante viaje a través del mundo de las bases de datos!

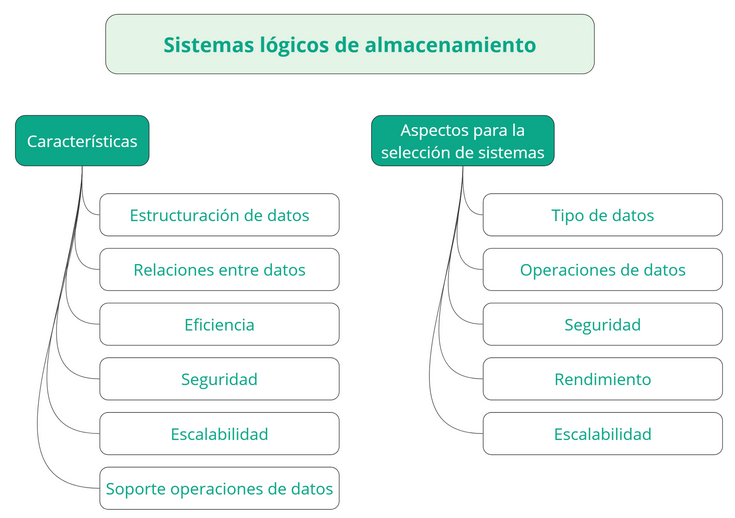
**1. Sistemas lógicos de almacenamiento**

Un sistema lógico de almacenamiento es un marco que dicta cómo se organizan, almacenan y acceden los datos en un sistema de información. No se trata solo de guardar datos en una ubicación física, sino de cómo se estructuran y se relacionan entre sí los datos en esa ubicación.

En su nivel más básico, un sistema lógico de almacenamiento se compone de registros, que son las unidades más pequeñas de datos identificables. Estos registros se organizan en archivos, que luego se agrupan en bases de datos.

El concepto de sistema lógico de almacenamiento puede ser mejor entendido con un ejemplo simple: piensa en una biblioteca. No almacena libros de manera aleatoria, sino que los organiza por autor, título, género, etc. Este es un tipo de sistema de almacenamiento lógico.

Eva y Andrés se encuentran con un reto similar en InnovaTech. Necesitan idear un sistema lógico para almacenar los datos de la aplicación de la librería, desde la información de cada libro hasta los datos de los clientes y las transacciones de ventas. Necesitan un sistema que no solo almacene estos datos, sino que permita una recuperación eficiente de la información cuando sea necesario. Entender el concepto de sistemas de almacenamiento lógico es su primer paso hacia la resolución de este desafío.



**1.1. Características**

Los sistemas lógicos de almacenamiento poseen ciertas características que los hacen indispensables en el mundo de la informática y los sistemas de información. Veamos algunas de las más relevantes:

* Estructuración de datos: Los sistemas de almacenamiento lógico organizan los datos de manera coherente y estructurada, lo que facilita el acceso y la recuperación de información cuando es necesario.
* Relaciones entre datos: Estos sistemas no solo almacenan datos, sino que también establecen relaciones entre ellos. Esto permite a los usuarios realizar consultas complejas y obtener información significativa.
* Eficiencia: Gracias a su estructura organizada, los sistemas de almacenamiento lógico permiten una rápida recuperación de los datos, lo que se traduce en un mejor rendimiento de las aplicaciones que los utilizan.
* Seguridad: Algunos sistemas de almacenamiento lógico ofrecen medidas de seguridad para proteger los datos almacenados, como control de acceso y cifrado.
* Escalabilidad: Son capaces de adaptarse a la creciente cantidad de datos, lo que los hace aptos para su uso en una amplia gama de aplicaciones, desde las más pequeñas hasta las más grandes y complejas.
* Soporte para operaciones de datos: Los sistemas lógicos de almacenamiento proporcionan mecanismos para operaciones de datos como inserción, actualización, eliminación y búsqueda de registros.

**1.1.1. Aspectos a considerar para la selección de sistemas**

Para seleccionar el sistema de almacenamiento lógico más adecuado para nuestras necesidades, es crucial identificar y analizar una serie de aspectos.

Cada uno de ellos tiene un impacto directo en la funcionalidad y la eficiencia del sistema.

* Tipo de datos: Es importante tener en cuenta qué tipo de datos se van a almacenar en el sistema. Los datos pueden ser numéricos, alfanuméricos, fechas, etc., y cada tipo puede requerir un sistema de almacenamiento lógico diferente. Dependiendo de los datos que se vayan a manejar, se pueden requerir sistemas de almacenamiento diferentes.

Por ejemplo, los datos textuales largos pueden requerir un almacenamiento orientado a documentos, mientras que los datos numéricos pueden ser manejados de manera más eficiente por sistemas de almacenamiento basados en columnas.

* Operaciones de datos: Debemos considerar qué operaciones se realizarán en los datos, que pueden incluir inserciones, actualizaciones, eliminaciones y búsquedas. El sistema de almacenamiento lógico debería ser capaz de manejar todas estas operaciones de manera eficiente.

El análisis de estas operaciones de datos permitirá determinar cómo se debe organizar y acceder a los datos para lograr una eficiencia óptima. Por ejemplo, si se realizan frecuentemente consultas complejas, puede ser útil un sistema con capacidades de indexación avanzadas.

* Seguridad: Este es un aspecto esencial en cualquier sistema que maneje datos. Necesitamos saber qué medidas de seguridad ofrece el sistema, como el control de acceso y el cifrado. El análisis de las necesidades de seguridad ayudará a determinar qué medidas se requieren. Esto puede implicar el control de acceso a nivel de usuario, cifrado de datos, o incluso funciones de auditoría para rastrear el acceso y las modificaciones a los datos.
* Rendimiento: Un factor importante a tener en cuenta es el rendimiento del sistema, que puede mejorar la experiencia del usuario. El rendimiento puede verse afectado por cómo se almacenan los datos, cómo se accede a ellos y cómo se manejan las operaciones de datos.

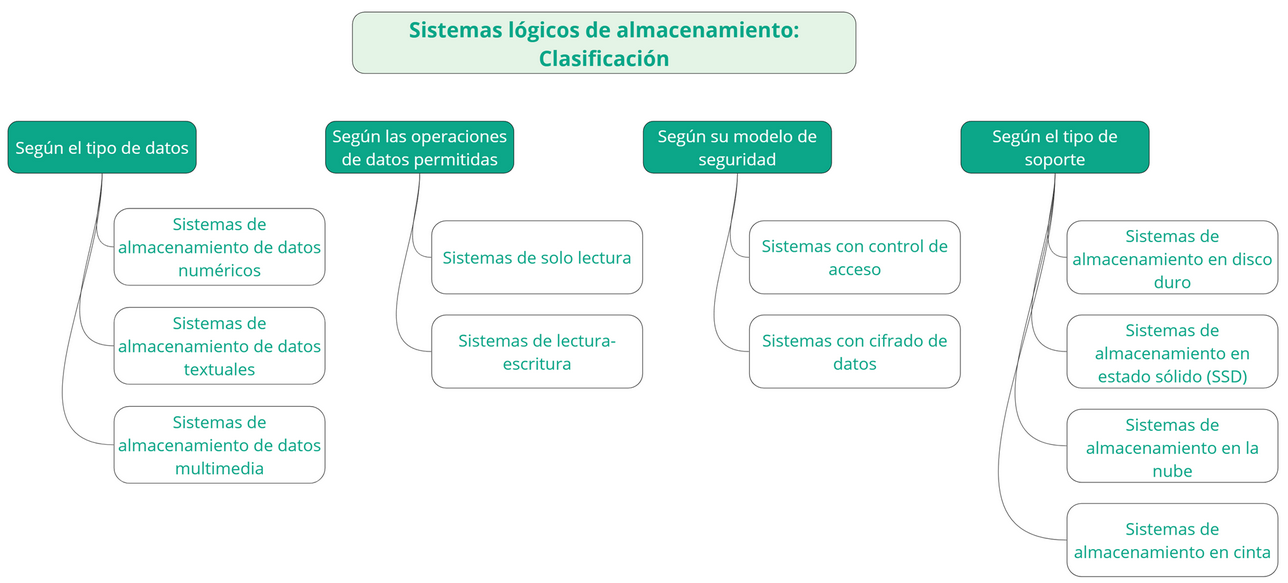
Analizar el rendimiento implica considerar la velocidad de las operaciones de datos, el tiempo de respuesta a las consultas, y la capacidad del sistema para manejar cargas de trabajo pesadas sin ralentizarse.

* Escalabilidad: La capacidad de escalar es esencial, especialmente si esperamos que la cantidad de datos que manejamos crezca con el tiempo. El sistema de almacenamiento lógico debería ser capaz de manejar un aumento en la cantidad de datos sin degradar su rendimiento.   
  Si se espera un crecimiento significativo en la cantidad de datos, es crucial un sistema que pueda escalar para manejar esta creciente carga de datos sin degradar el rendimiento.

**1.2. Clasificación**

Los sistemas lógicos de almacenamiento se pueden clasificar de varias maneras, dependiendo de los aspectos que estemos considerando.

Aquí se presentan algunas de las formas más comunes de clasificación:



* Según el tipo de datos: El tipo de datos que el sistema es capaz de almacenar es un criterio importante para su clasificación. Algunos sistemas están optimizados para tipos de datos específicos, mientras que otros pueden manejar una variedad de tipos de datos. Por ejemplo:

• Sistemas de almacenamiento de datos numéricos: Estos sistemas están optimizados para almacenar y procesar datos numéricos, y son comúnmente utilizados en aplicaciones financieras y científicas.

• Sistemas de almacenamiento de datos textuales: Estos sistemas están diseñados para manejar grandes cantidades de texto, como documentos o registros de transacciones.

• Sistemas de almacenamiento de datos multimedia: Estos sistemas son capaces de manejar datos de audio, vídeo e imágenes.

* Según las operaciones de datos permitidas: Los sistemas lógicos de almacenamiento también se pueden clasificar según las operaciones de datos que permiten. Algunos sistemas permiten una amplia gama de operaciones, mientras que otros se centran en operaciones específicas para optimizar el rendimiento:

• Sistemas de solo lectura: Estos sistemas solo permiten leer datos, y no permiten modificaciones. Son útiles para el almacenamiento de datos que no cambian con frecuencia, como los datos de configuración o los archivos de sistema.

• Sistemas de lectura-escritura: Estos sistemas permiten tanto la lectura como la modificación de datos. Son los más comunes y se utilizan en la mayoría de las aplicaciones.

* Según su modelo de seguridad: Los sistemas lógicos de almacenamiento se pueden clasificar también según su modelo de seguridad. Algunos sistemas incorporan funciones avanzadas de seguridad, mientras que otros son más básicos:

• Sistemas con control de acceso: Estos sistemas implementan mecanismos para controlar quién puede acceder a los datos y qué operaciones pueden realizar.

• Sistemas con cifrado de datos: Estos sistemas utilizan técnicas de cifrado para proteger los datos almacenados, de manera que solo puedan ser leídos por aquellos que tienen la clave correcta.

* Según el tipo de soporte: El soporte físico donde se almacenan los datos también puede ser un criterio para clasificar los sistemas lógicos de almacenamiento. Por ejemplo:

• Sistemas de almacenamiento en disco duro: Los discos duros han sido durante mucho tiempo el soporte de almacenamiento principal para la mayoría de los sistemas informáticos. Pueden almacenar grandes cantidades de datos y proporcionar un acceso relativamente rápido a esos datos.

• Sistemas de almacenamiento en estado sólido (SSD): Los SSD son más rápidos que los discos duros y son cada vez más populares, especialmente en dispositivos móviles y portátiles.

• Sistemas de almacenamiento en la nube: La nube permite almacenar y acceder a datos a través de Internet, sin la necesidad de tener una unidad de almacenamiento física en el dispositivo del usuario. Estos sistemas pueden proporcionar una gran cantidad de almacenamiento y son fácilmente escalables.

• Sistemas de almacenamiento en cinta: Aunque son más lentos y menos accesibles que los discos duros o los SSD, las cintas siguen siendo una opción popular para el almacenamiento a largo plazo y la archivación de datos debido a su bajo costo y su alta capacidad.

**2. Ficheros**

Los ficheros son la forma más básica y tradicional de almacenar y organizar los datos. Un fichero no es más que un conjunto de datos almacenados en un dispositivo de almacenamiento como un disco duro o un SSD. Los ficheros pueden contener cualquier tipo de datos, desde texto y números hasta imágenes y sonidos. En los sistemas informáticos modernos, los ficheros suelen ser la forma en que los datos se almacenan y se organizan a nivel físico.

Pero los ficheros no son solo una forma de almacenamiento de datos. También son una forma de organizar los datos de manera lógica. Por ejemplo, un fichero de texto puede contener los registros de una base de datos, con cada registro almacenado en una línea separada. O un fichero de imagen puede contener los píxeles de una imagen, con cada píxel almacenado como una serie de números.

Los ficheros son una parte fundamental de cualquier sistema informático y proporcionan la base sobre la cual se construyen los sistemas de bases de datos más avanzados. En esta sección, examinaremos más de cerca los ficheros, cómo se organizan, cómo se accede a ellos y cómo se manipulan. Aprenderemos también sobre las limitaciones de los sistemas basados en ficheros y cómo estas limitaciones llevaron al desarrollo de los sistemas de bases de datos.

**2.1. Tipos de ficheros**

Dependiendo de diversos factores como la naturaleza de los datos, cómo se accede a ellos, su propósito y su permanencia, los ficheros pueden clasificarse en distintas categorías. En este apartado, abordaremos dos clasificaciones principales para los tipos de ficheros.

Por un lado, tenemos una clasificación basada en la organización y el acceso a los datos. En este grupo, encontramos:

• Ficheros secuenciales

• Ficheros indexados

• Ficheros de acceso directo.

Cada uno de estos tipos tiene características particulares que los hacen más adecuados para ciertos escenarios y aplicaciones.

Por otro lado, encontramos una clasificación basada en la permanencia y el propósito de los datos almacenados. Aquí se distinguen:

• Ficheros permanentes, que contienen datos que deben conservarse durante largos periodos de tiempo

• Ficheros temporales, que almacenan datos de uso efímero, generalmente ligados a la ejecución de un proceso específico.

A lo largo de este apartado, profundizaremos en cada uno de estos tipos de ficheros, explorando sus características, usos y ejemplos que ayuden a comprender mejor sus aplicaciones prácticas. Conocer estas categorías y sus particularidades permitirá tomar decisiones más informadas y efectivas a la hora de diseñar e implementar sistemas de almacenamiento de datos.

**2.2. Clasificación Basada en la Organización y Acceso a los Datos**

En la base de esta clasificación se encuentran principios de organización de datos que se han desarrollado a lo largo del tiempo en respuesta a la necesidad de manejar cantidades cada vez mayores de datos y a la demanda de un acceso más rápido y eficiente a estos datos.

Los ficheros en esta clasificación se dividen en tres tipos principales: secuenciales, indexados y de acceso directo. Cada uno de estos tipos representa una estrategia diferente para almacenar y recuperar datos, y cada uno tiene sus propias ventajas y desventajas.

Los ficheros secuenciales representan la forma más básica de almacenamiento, con datos dispuestos y accedidos en el mismo orden, proporcionando una eficiencia considerable para tareas que procesan todos los datos en orden.

Los ficheros indexados emplean una estructura de índice, que actúa como un "mapa" para los datos, facilitando el acceso más rápido a los datos específicos, ideal para tareas que requieren un acceso frecuente a datos particulares.

Los ficheros de acceso directo permiten la recuperación y almacenamiento directo de datos usando una clave única. Cada registro tiene una dirección única, proporcionando un acceso muy rápido a los datos, sin importar el tamaño del fichero.

**2.2.1. Ficheros Secuenciales**

Los ficheros secuenciales, como su nombre lo indica, son aquellos donde los registros de datos están organizados y accedidos en un orden secuencial, siguiendo una línea lógica uno tras otro.

Este orden secuencial puede estar basado en algún criterio particular como el tiempo de creación o alguna clave específica inherente a los datos. Este formato es la forma más antigua y simple de organizar los datos en almacenamiento informático y es la base de los métodos de almacenamiento más complejos.

Un aspecto importante de los ficheros secuenciales es que, aunque se necesita recorrer todos los registros para acceder a uno en particular, lo que puede resultar en un tiempo de acceso lento para ficheros grandes, son excepcionalmente eficientes para operaciones que requieren la lectura o procesamiento de todos los registros en un orden particular.

Ejemplos:

• Un ejemplo común de uso de ficheros secuenciales sería en los procesos de respaldo de datos (backup), donde se realiza una copia completa de todos los datos en un sistema. En este caso, los ficheros se leen secuencialmente y se copian a otro medio de almacenamiento.

• Otros ejemplos de uso pueden incluir la impresión de informes, en los que se necesita acceder a todos los registros uno tras otro, o en tareas de procesamiento por lotes, donde se requiere procesar una gran cantidad de datos en un orden específico.

• Además, los ficheros secuenciales son muy utilizados en la gestión de registros de logs, donde los eventos se registran en orden cronológico.

Recuerda   
Si se requiere procesar todos los registros de principio a fin, los ficheros secuenciales son una opción altamente eficiente. Esto se debe a que el tiempo necesario para leer los datos del disco es mucho mayor que el tiempo necesario para procesarlos, por lo que si los datos se leen de forma secuencial (es decir, sin saltos), la velocidad de lectura del disco se maximiza.

**2.2.2. Ficheros Indexados**

Los ficheros indexados, tal como sugiere su denominación, emplean una estructura de índice para ofrecer una vía de acceso rápida y eficiente a los datos almacenados. En lugar de recorrer secuencialmente todo el fichero para encontrar un registro específico, el índice, que se puede visualizar como una especie de tabla de contenido o un mapa, permite un acceso directo al lugar donde está almacenado el dato en el fichero.

El índice en los ficheros indexados es una estructura de datos independiente que mantiene una correspondencia entre las claves de los registros y sus ubicaciones en el fichero. Al igual que un índice en un libro proporciona números de página para temas específicos, el índice en un fichero indexado proporciona la ubicación de registros específicos dentro del fichero. Esta metodología permite minimizar el tiempo de búsqueda cuando se necesitan recuperar registros con claves específicas.

Ejemplos

• El uso de ficheros indexados es ideal en situaciones en las que se necesitan realizar consultas frecuentes a registros específicos en una base de datos. Por ejemplo, una base de datos de clientes que se accede con frecuencia para buscar información sobre clientes individuales, se beneficiaría enormemente de un sistema de ficheros indexados. Los ficheros indexados permitirían a los usuarios localizar la información de un cliente específico rápidamente, sin tener que buscar en todo el fichero.

• Otros ejemplos donde los ficheros indexados serían útiles incluyen bibliotecas digitales, donde los usuarios pueden buscar libros o artículos por título, autor, o temas específicos; sistemas de gestión de documentos, donde los usuarios pueden buscar documentos por palabras clave o temas; y bases de datos de productos en tiendas de comercio electrónico, donde los usuarios pueden buscar productos por nombre, categoría, o características específicas.

Recuerda   
Cualquier situación donde se necesite un acceso frecuente y rápido a registros específicos se beneficiará del uso de ficheros indexados.

**2.2.3. Ficheros de Acceso Directo**

Los ficheros de acceso directo, también conocidos como ficheros de acceso aleatorio, proporcionan un medio de acceder instantáneamente a cualquier registro dentro del fichero sin tener que recorrer los registros anteriores, gracias a la asignación de una dirección única para cada registro. Esto se logra a través de una clave única, que se utiliza para calcular la dirección del registro en el fichero.

La característica distintiva de los ficheros de acceso directo es su capacidad para leer, escribir y actualizar registros individuales directamente. A diferencia de los ficheros secuenciales o indexados, donde la velocidad de acceso puede depender del tamaño del fichero, los ficheros de acceso directo pueden acceder a cualquier registro con la misma rapidez, independientemente del tamaño del fichero.

Estos ficheros son ideales para situaciones en las que se necesita acceso instantáneo a registros específicos, independientemente de su ubicación en el fichero. Son útiles en contextos donde la velocidad y la eficiencia de acceso a los datos son críticas.

Ejemplos:

• Serían útiles en una base de datos bancaria donde se necesita acceder a los registros de las cuentas de los clientes de forma inmediata.

• Otro caso de uso de los ficheros de acceso directo es en los sistemas de reservas de aerolíneas. Cada reserva es un registro único y puede necesitarse en cualquier momento para cambios, cancelaciones o consultas de los clientes. En estos casos, un fichero de acceso directo permite un acceso inmediato a la información.

Recuerda   
Los ficheros de acceso directo son una solución eficiente para situaciones que requieren un acceso rápido y directo a registros individuales en grandes conjuntos de datos. Su flexibilidad y eficiencia los hacen ideales para una amplia gama de aplicaciones que manejan grandes volúmenes de datos.

**2.3. Clasificación Basada en la Permanencia y Propósito de los Datos**

En la gestión de ficheros, la durabilidad y el propósito de los datos que contienen son factores cruciales a considerar. Por lo tanto, una forma común de clasificar los ficheros es por su permanencia y el propósito de los datos almacenados. En esta categoría, diferenciamos entre ficheros permanentes y ficheros temporales, cada uno con roles y usos específicos.

Los ficheros permanentes contienen datos fundamentales para un sistema o aplicación, persistiendo durante largos periodos de tiempo. Incluyen ficheros de configuración, bases de datos y registros de auditoría.

Los ficheros temporales, en cambio, almacenan datos transitorios o de uso breve. Facilitan el intercambio de datos entre procesos, guardan estados de proceso y almacenan datos intermedios en procesamientos de grandes conjuntos de datos.

**2.3.1. Ficheros Permanentes**

Los ficheros permanentes son aquellos que se almacenan en un sistema de almacenamiento de datos y permanecen allí durante un período de tiempo prolongado. Generalmente, contienen información que es esencial para las operaciones de una organización y que necesita ser preservada por razones operativas, legales o históricas. Su contenido no suele cambiar con frecuencia y se utilizan principalmente para el almacenamiento a largo plazo.

Un ejemplo de un fichero permanente puede ser un archivo que almacena la información de los clientes de una empresa. Este archivo puede contener detalles como el nombre del cliente, dirección, número de teléfono, y demás datos relevantes. Este tipo de fichero se considera permanente porque la información que contiene es esencial para las operaciones de la empresa y se mantendrá durante todo el tiempo que el cliente esté vinculado a la empresa.

Otro ejemplo de un fichero permanente puede ser el registro de transacciones financieras de una empresa. Este fichero puede contener detalles de todas las transacciones financieras realizadas por la empresa durante un período de tiempo específico. Estos datos son esenciales para la contabilidad y la auditoría, y deben mantenerse durante un largo período de tiempo para cumplir con las regulaciones financieras y fiscales.

**2.3.2. Ficheros Temporales**

Los ficheros temporales son aquellos que existen únicamente durante un cierto período de tiempo, generalmente mientras se está ejecutando un programa específico o durante una sesión de trabajo específica. Una vez que se completa la tarea para la que fueron creados, estos ficheros se eliminan para liberar espacio de almacenamiento. Los ficheros temporales se utilizan principalmente para almacenar información intermedia que puede ser necesaria durante la ejecución de un programa, pero que no necesita ser guardada de forma permanente.

Un ejemplo común de uso de ficheros temporales se da en los procesadores de texto. Cuando estás escribiendo un documento, el procesador de texto puede crear un fichero temporal que almacena lo que estás escribiendo. Este fichero permite al procesador de texto recuperar tu trabajo en caso de un fallo inesperado, como un bloqueo del software o un apagado inesperado del sistema. Una vez que cierras el documento y lo guardas, el fichero temporal se elimina.

Otro ejemplo de uso de ficheros temporales se da en los navegadores web. Cuando visitas una página web, tu navegador puede almacenar imágenes, scripts y otros datos de la página en ficheros temporales. Esto permite al navegador cargar la página más rápidamente si vuelves a visitarla durante la misma sesión de navegación. Una vez que cierras el navegador, estos ficheros temporales se eliminan.

**2.3.3. Ficheros Volátiles**

Los ficheros volátiles son un tipo especial de ficheros temporales que residen en la memoria volátil del sistema, como la RAM, en lugar de en un almacenamiento no volátil, como un disco duro. Estos ficheros existen únicamente mientras el sistema está en funcionamiento y se pierden cuando el sistema se apaga o se reinicia.

Los ficheros volátiles se utilizan para almacenar datos que necesitan ser accesibles muy rápidamente y que no necesitan persistir entre diferentes sesiones de ejecución. Un ejemplo de esto podría ser la información de sesión de un usuario en una aplicación web. Esta información podría almacenarse en un fichero volátil en el servidor mientras el usuario está interactuando con la aplicación. Cuando el usuario cierra la sesión o después de un cierto período de inactividad, la información de la sesión se descarta.

Otro uso común de los ficheros volátiles se da en la caché de las aplicaciones o los sistemas operativos. Estos ficheros pueden almacenar datos que se acceden con frecuencia para reducir el tiempo de carga y mejorar el rendimiento. Sin embargo, como estos datos son sólo una copia de los datos originales que están almacenados de forma segura en otro lugar, no es necesario que persistan una vez que el sistema se apaga.

**2.3.4. Ficheros de Copia de Seguridad**

Los ficheros de copia de seguridad, también conocidos como ficheros de respaldo, son copias exactas de los datos originales que se crean para ser utilizadas en caso de pérdida, corrupción o borrado accidental de los datos originales. Estos ficheros se almacenan en un medio de almacenamiento seguro y a menudo en una ubicación física separada para protegerlos de desastres como incendios o inundaciones que podrían destruir tanto los datos originales como las copias de seguridad si se almacenaran en la misma ubicación.

Por ejemplo, una empresa podría realizar copias de seguridad de sus datos de cliente en un disco duro externo cada noche. En caso de que los datos de cliente en el sistema principal se corrompan de alguna manera, la empresa podría restaurar los datos a partir de la copia de seguridad más reciente, perdiendo solo un día de datos como máximo. Además, en muchos sistemas modernos, se realizan copias de seguridad automáticas en la nube. Esto permite una mayor seguridad y facilidad de acceso a las copias de seguridad. Sin embargo, también plantea problemas de seguridad y privacidad, ya que los datos deben transmitirse y almacenarse de forma segura para evitar el acceso no autorizado.

Las políticas de copia de seguridad varían según las necesidades de la organización y los datos que se están respaldando. Algunas organizaciones pueden realizar copias de seguridad de sus datos una vez al día, mientras que otras pueden hacerlo una vez a la semana o incluso una vez al mes. Las copias de seguridad también pueden ser incrementales (donde solo se copian los datos que han cambiado desde la última copia de seguridad) o completas (donde se copian todos los datos cada vez).

**2.4. Clasificación Basada en el Propósito de los Datos**

Los ficheros también pueden clasificarse en base a su propósito dentro del sistema. Esto puede ayudar a entender cómo se utilizan los datos y cómo se deben gestionar.

El propósito de los datos puede variar enormemente, desde la contabilidad y las ventas hasta la gestión de proyectos y la planificación de recursos. En general, esta clasificación se divide en cinco categorías principales: ficheros de trabajo, ficheros históricos, ficheros maestros, ficheros de transacciones y ficheros de respaldo.

Cada uno de estos tipos de ficheros tiene un propósito específico y se maneja de una manera única para maximizar su eficacia. Comencemos examinando cada uno de estos tipos de ficheros en detalle.

**2.4.1. Ficheros de Trabajo**

Los ficheros de trabajo son aquellos que contienen datos temporales que se utilizan durante el procesamiento de transacciones o la realización de cálculos y operaciones específicas. Este tipo de ficheros puede contener información variada y no necesariamente necesita ser conservada a largo plazo.

Los ficheros de trabajo son útiles en varias situaciones, por ejemplo:

• En un sistema de gestión de proyectos, los ficheros de trabajo podrían contener el estado actual de las tareas asignadas y su progreso. Esta información podría ser actualizada frecuentemente y no ser necesaria una vez que el proyecto ha concluido.

• En una empresa de comercio electrónico, los ficheros de trabajo podrían almacenar los detalles del carrito de compras de un cliente mientras está realizando compras. Esta información es crucial mientras el cliente está activo en el sitio, pero una vez que la transacción ha concluido, los datos pueden ser eliminados o archivados.

Los ficheros de trabajo, por su naturaleza efímera, necesitan ser gestionados cuidadosamente para evitar la pérdida de datos críticos durante su uso, pero a la vez mantener un uso eficiente del espacio de

almacenamiento al eliminarlos cuando ya no son necesarios.

**2.4.2. Ficheros de Respaldo**

Los ficheros de respaldo, también conocidos como archivos de backup, son copias de seguridad de los datos que se almacenan para recuperar información en caso de pérdida de datos o fallos en el sistema.

Los ficheros de respaldo pueden ser creados para cualquier tipo de archivo y son esenciales para mantener la integridad y la disponibilidad de los datos.

Aquí algunos ejemplos de cómo se utilizan los ficheros de respaldo:

Un sistema de gestión de documentos puede crear ficheros de respaldo de todos los documentos importantes de la empresa al final de cada día laboral. Si se pierden los archivos originales debido a un fallo del hardware, se pueden restaurar desde los ficheros de respaldo.

En un sistema de bases de datos, los ficheros de respaldo pueden ser creados regularmente para garantizar que la información más reciente esté disponible en caso de fallos del sistema.

La creación de ficheros de respaldo debe ser una parte integral de cualquier estrategia de gestión de datos. Esto es especialmente importante para los sistemas que manejan datos críticos o sensibles, donde la pérdida de datos puede tener graves consecuencias. Además, los ficheros de respaldo deben almacenarse en un lugar seguro y posiblemente separado del sistema original para protegerlos de los mismos riesgos que los datos originales.

**2.4.2. Ficheros Históricos**

Los ficheros históricos son aquellos que almacenan información que ya no se modifica, pero se mantiene por razones de registro histórico, auditoría, cumplimiento legal, o potencialmente para análisis retrospectivos. Estos archivos representan una versión inmutable de los datos que fueron válidos o relevantes en un punto específico en el tiempo.

Aquí algunos ejemplos de cómo se utilizan los ficheros históricos:

• En una empresa de ventas, los ficheros históricos pueden incluir registros de todas las transacciones de ventas que se han realizado en el pasado. Estos datos pueden ser útiles para realizar análisis de tendencias de ventas, pronósticos, o para auditorías financieras.

• En el ámbito de la medicina, los ficheros históricos pueden incluir registros médicos de pacientes que ya no están en tratamiento pero que deben mantenerse por razones legales o para referencia futura.

La gestión de ficheros históricos puede requerir soluciones de almacenamiento a largo plazo y estrategias de archivado. También es importante considerar las políticas de retención de datos y las obligaciones legales relacionadas con el almacenamiento de datos históricos.

**3.1. Conceptos y usos de las bases de datos**

Las bases de datos son una herramienta esencial en la era digital. Estas representan una manera sistemática y organizada de recopilar, gestionar y recuperar una gran cantidad de información. En esencia, una base de datos es un conjunto de datos relacionados entre sí de alguna manera, usualmente estructurados en tablas de datos que permiten el procesamiento y análisis eficiente de la información.

Desde el mantenimiento de registros financieros y la administración de inventarios, hasta el análisis de tendencias de mercado y la toma de decisiones basada en datos, las bases de datos desempeñan un papel crucial en diversas aplicaciones en todos los sectores de la industria.

En este apartado, exploraremos la definición y los componentes de una base de datos, su relevancia y aplicabilidad en diversos contextos, y algunos ejemplos de cómo las bases de datos se utilizan en diferentes industrias. Este conocimiento será fundamental para comprender cómo se gestionan los datos en un mundo cada vez más impulsado por la información.

**3.2. Principales características de las bases de datos**

Las bases de datos son sistemas organizados de almacenamiento que permiten la gestión de grandes cantidades de información de manera eficiente.

Son esenciales en cualquier sistema informático que necesite tratar con datos de forma estructurada y ofrecen una serie de características que les dan ventaja sobre otros sistemas de almacenamiento. Aquí se destacan algunas de las características más relevantes:

• Persistencia: Los datos almacenados en una base de datos se mantienen de manera permanente, independientemente de si la base de datos está en uso activo o no. Esto asegura la disponibilidad de los datos cada vez que se requieran.

• Consistencia: Las bases de datos mantienen un alto nivel de consistencia, lo que significa que los datos almacenados son coherentes y exactos. Esto se logra a través del uso de transacciones, que permiten a los cambios de datos realizarse de manera atómica (o sea, todas las actualizaciones de datos se realizan o ninguna se realiza).

• Concurrencia: Las bases de datos permiten el acceso simultáneo a los datos por parte de múltiples usuarios. Esto es esencial en entornos de trabajo colaborativos y aplicaciones multiusuario.

• Recuperabilidad: Las bases de datos ofrecen la posibilidad de recuperar datos en caso de fallos o errores. Los sistemas de gestión de bases de datos (DBMS) realizan copias de seguridad y registros de transacciones para permitir la recuperación de datos.

• Seguridad: Las bases de datos proporcionan mecanismos de seguridad para proteger los datos almacenados. Esto incluye la autenticación de usuarios, el control de acceso y el cifrado de datos.

Un ejemplo de uso de una base de datos puede ser un sistema de gestión de alumnos de una universidad. En este sistema, los datos de los alumnos (como su nombre, dirección, cursos matriculados, calificaciones, etc.) se almacenarían en una base de datos. El sistema permitiría a los administradores y profesores acceder y modificar estos datos de manera consistente y segura, al mismo tiempo que proporcionaría la capacidad de recuperar los datos en caso de un fallo del sistema.

**3.3. Estructura y componentes de una base de datos**

La estructura de una base de datos define cómo se organizan los datos.

En una base de datos, la estructura es determinada por el modelo de datos que se utiliza. Los modelos de datos más comunes son el modelo relacional, el modelo jerárquico, el modelo de red y el modelo de objetos.

Los componentes clave de una base de datos incluyen:

• Entidades: Las entidades son los objetos que se representan en la base de datos. Por ejemplo, en una base de datos para una librería, las entidades podrían ser los libros, los autores, los clientes y las transacciones de compra.

• Atributos: Los atributos son las propiedades que describen las entidades. Por ejemplo, los atributos de la entidad "Libro" podrían ser el título, el autor, el género y el precio.

• Registros: Un registro es una instancia individual de una entidad. Por ejemplo, cada libro en la librería sería un registro en la base de datos. Cada registro consta de una serie de atributos.

• Campos: Un campo es un único elemento de datos dentro de un registro. Por ejemplo, el título de un libro sería un campo dentro del registro de ese libro.

• Tablas: En una base de datos relacional, las entidades y sus atributos se organizan en tablas. Cada tabla representa una entidad, y cada fila en la tabla es un registro.

• Relaciones: Definiciones de cómo se relacionan las diferentes entidades en la base de datos.

• Restricciones: Reglas aplicadas para mantener la integridad de los datos en la base de datos.

**3.4. Usos, Aplicaciones y Ejemplos de Bases de Datos**

Las bases de datos son una parte crucial de casi todos los aspectos de nuestra vida moderna. Proporcionan la capacidad de almacenar, organizar y recuperar grandes cantidades de información de manera eficiente.

A continuación, presentamos una serie de usos y aplicaciones de las bases de datos en diversas industrias, acompañados de ejemplos concretos:

• Negocios y Comercio Electrónico: Las bases de datos desempeñan un papel esencial en la gestión de la información de clientes, proveedores y productos en los negocios. Las empresas de comercio electrónico como Amazon y eBay utilizan bases de datos para almacenar y gestionar millones de productos y transacciones de clientes. También permiten personalizar la experiencia del usuario en función de su historial de compras y preferencias.

• Banca y Finanzas: Las instituciones financieras utilizan bases de datos para llevar un registro de las transacciones de sus clientes, proporcionar servicios de banca en línea, calcular intereses y gestionar cuentas de inversión. Por ejemplo, los bancos utilizan bases de datos para llevar un registro de los saldos de las cuentas de sus clientes y para automatizar el proceso de préstamos.

• Salud y Medicina: Las bases de datos se utilizan para almacenar y gestionar la información del paciente, incluyendo su historial médico, medicamentos recetados, resultados de pruebas y citas médicas. Un ejemplo de esto es la base de datos de pacientes de un hospital, que se utiliza para rastrear el tratamiento y el progreso de cada paciente.

• Educación: Las instituciones educativas utilizan bases de datos para manejar información relacionada con estudiantes, profesores, cursos, calificaciones y más. Por ejemplo, una universidad puede usar una base de datos para almacenar la información de inscripción de los estudiantes, las calificaciones de los cursos y los horarios de las clases.

• Redes Sociales y Entretenimiento: Las redes sociales como Facebook y Twitter, y las plataformas de streaming como Netflix y Spotify, también utilizan bases de datos para gestionar la información de los usuarios, sus interacciones, preferencias y recomendaciones. Por ejemplo, Netflix utiliza algoritmos de aprendizaje automático que se alimentan de bases de datos para recomendar contenido a los usuarios basado en su historial de visualización.

Como estos ejemplos ilustran, las bases de datos se utilizan en una amplia gama de aplicaciones y sectores, resaltando su importancia en nuestra sociedad actual.

**3.5. Tipos según el modelo de datos**

La estructura de una base de datos define cómo se organizan los datos. En una base de datos, la estructura es determinada por el modelo de datos que se utiliza. Los modelos de datos más comunes son el modelo relacional, el modelo jerárquico, el modelo de red y el modelo de objetos.

Los componentes clave de una base de datos incluyen:

Bases de Datos Jerárquicas: Organizan los datos en una estructura de árbol, con una jerarquía de padres e hijos. Son muy eficientes para ciertos tipos de operaciones y fueron algunos de los primeros sistemas de bases de datos.

Bases de Datos de Red: Mejoran las bases de datos jerárquicas al permitir múltiples padres, lo que ofrece una red más compleja de relaciones.

Bases de Datos Relacionales: Son el tipo más común y utilizan tablas para almacenar datos. Las relaciones entre los datos se establecen a través de claves primarias y extranjeras.

Bases de Datos de Objetos: Almacenan datos como objetos, similares a cómo se manejan los objetos en la programación orientada a objetos.

Bases de Datos NoSQL: Este es un término amplio que abarca una variedad de modelos de bases de datos que no se ajustan a la estructura relacional. Estos incluyen bases de datos de clave-valor, bases de datos de documentos, bases de datos de columnas anchas y bases de datos de grafos.

Cada uno de estos modelos de bases de datos tiene sus propias ventajas, desventajas y casos de uso ideales. En las siguientes secciones, vamos a explorar cada uno de estos modelos en detalle, discutiendo sus características, ventajas, desventajas y ejemplos de uso en el mundo real.

**3.5.1. Bases de Datos Jerárquicas**

Las bases de datos jerárquicas son uno de los modelos de bases de datos más antiguos, que data de la época en que los sistemas informáticos comenzaron a almacenar grandes cantidades de datos. En este modelo, los datos se organizan en una estructura de árbol, similar a un organigrama, con un nodo padre y varios nodos hijos.

Características

• Estructura de árbol: En una base de datos jerárquica, los datos se organizan en una estructura de árbol. Un nodo tiene solo un nodo padre, pero puede tener varios nodos hijos.

• Eficiencia: Las bases de datos jerárquicas pueden ser muy eficientes en términos de velocidad y rendimiento cuando se utilizan para determinados tipos de operaciones de consulta, especialmente cuando la mayoría de las consultas son conocidas y predecibles.

• Integridad de los datos: Este modelo garantiza que no habrá datos duplicados ya que cada nodo hijo tiene un solo nodo padre.

Desventajas

• Falta de flexibilidad: Las bases de datos jerárquicas son rígidas en su estructura, lo que puede hacer que sea difícil adaptarlas a cambios en los requisitos de los datos.

• Dificultad para manejar relaciones muchos a muchos: En un modelo jerárquico, cada hijo tiene un solo padre. Esto hace que sea difícil representar relaciones muchos a muchos.

Ejemplo de uso

Un ejemplo clásico de una base de datos jerárquica es el sistema de gestión de bases de datos de IBM, IMS (Information Management System). Este sistema se utiliza en aplicaciones empresariales para gestionar grandes cantidades de datos y transacciones. Un uso común de IMS es en los sistemas de reservas de aerolíneas, donde los datos se organizan jerárquicamente (por ejemplo, aerolínea > vuelo > pasajero).

**3.5.2. Bases de Datos de Red**

Las bases de datos de red son una extensión del modelo jerárquico, diseñadas para abordar algunas de sus limitaciones. En lugar de limitarse a una estructura de árbol, las bases de datos de red adoptan una estructura de gráfico que permite a múltiples registros tener relaciones con otros.

Características

• Estructura de gráfico: En una base de datos de red, los datos se organizan en una estructura de gráfico, lo que permite una mayor flexibilidad en las relaciones entre los datos. Un nodo puede tener múltiples nodos padres y múltiples nodos hijos.

• Soporte de relaciones muchos a muchos: Este modelo puede manejar fácilmente relaciones muchos a muchos entre los datos, algo que las bases de datos jerárquicas no pueden hacer de manera eficiente.

• Recuperación eficiente de datos: Al igual que las bases de datos jerárquicas, las bases de datos de red pueden ser muy eficientes para ciertos tipos de consultas.

Desventajas

• Complejidad: La estructura de gráfico de una base de datos de red puede ser bastante compleja, especialmente a medida que el volumen de datos crece.

• Mantenimiento: Debido a su complejidad, estas bases de datos pueden ser difíciles de mantener y requieren un alto nivel de habilidad técnica para su gestión.

Ejemplo de uso

El modelo de base de datos de red se utilizó ampliamente en la década de 1980. Un ejemplo de esto es el Integrated Data Store (IDS) de Charles Bachman. El IDS fue ampliamente utilizado en aplicaciones que requerían un alto rendimiento y la capacidad de manejar relaciones complejas, como en sistemas de control de inventario, donde un producto puede estar relacionado con múltiples proveedores y categorías.

**3.5.3. Bases de Datos Relacionales**

Las bases de datos relacionales son las más utilizadas y conocidas en la actualidad. Este tipo de bases de datos se basan en el modelo relacional, introducido por Edgar F. Codd en 1970. La idea central es que todos los datos se representan en términos de tuplas, agrupadas en relaciones.

Características

• Estructura de tabla: En una base de datos relacional, los datos se organizan en tablas o "relaciones". Cada tabla tiene un conjunto definido de columnas (atributos) y un número indefinido de filas (tuplas).

• Integridad referencial: Este es un sistema de reglas que las bases de datos relacionales utilizan para garantizar que las relaciones entre las tablas permanezcan consistentes. Esto se logra a través de claves primarias y claves foráneas.

• Soporte de consultas SQL: SQL (Structured Query Language) es el lenguaje estándar para interactuar con las bases de datos relacionales. Permite a los usuarios realizar una variedad de tareas, incluyendo consultas, actualizaciones, inserciones y eliminaciones de datos.

Desventajas

• Rendimiento: Las bases de datos relacionales pueden tener dificultades para manejar grandes volúmenes de datos o consultas altamente complejas. Esto se debe en parte a que la estructura de tabla no siempre es la forma más eficiente de almacenar o acceder a los datos.

• Flexibilidad limitada: Aunque el modelo relacional es bastante flexible en términos de los tipos de datos que puede representar, no es ideal para todos los casos de uso. Por ejemplo, no es particularmente bueno para manejar datos jerárquicos o gráficos.

Ejemplos de uso

Las bases de datos relacionales son la columna vertebral de muchas aplicaciones empresariales. Por ejemplo, un sistema de gestión de relaciones con los clientes (CRM) puede usar una base de datos relacional para almacenar información sobre clientes, interacciones de ventas, productos, etc. Ejemplos de sistemas de gestión de bases de datos relacionales incluyen MySQL, Oracle, SQL Server y PostgreSQL.

**3.5.4. Bases de Datos de Objetos**

Las bases de datos orientadas a objetos (OODB) representan un cambio de paradigma respecto a las bases de datos relacionales. En lugar de basarse en tablas y relaciones, las OODB organizan los datos en torno a "objetos", que son instancias de clases, similares a cómo se hace en la programación orientada a objetos (OOP).

Estos objetos pueden contener tanto datos (atributos) como comportamientos (métodos). Esto proporciona una flexibilidad adicional ya que los objetos pueden heredar características de otros objetos, lo que permite crear estructuras de datos más complejas y dinámicas. Por ejemplo, un objeto "estudiante" podría heredar características de un objeto "persona", y a su vez tener atributos específicos como "materia" o "nota".

Las OODB también son capaces de almacenar y manipular objetos como una entidad única, en lugar de descomponerlos en elementos individuales, lo que es útil para manejar datos complejos como los gráficos 3D, las interfaces de usuario y los componentes de software.

Son ampliamente utilizadas en el desarrollo de videojuegos, sistemas CAD/CAM, y cualquier lugar donde los datos complejos deben ser manipulados de manera eficiente.

Una de las desventajas de las bases de datos orientadas a objetos es que requieren un cambio de mentalidad y de prácticas de programación en comparación con las bases de datos relacionales. También pueden ser menos eficientes en el manejo de operaciones de búsqueda y consulta a gran escala en comparación con las bases de datos relacionales.

Recuerda   
Las bases de datos orientadas a objetos ofrecen una gran flexibilidad y son excelentes para manejar datos complejos y relacionados, pero pueden ser más desafiantes de usar y menos eficientes en ciertos contextos en comparación con las bases de datos relacionales.

**3.5.5. Bases de Datos NoSQL**

Las bases de datos NoSQL (Not only SQL) representan un cambio en la forma en que pensamos sobre el almacenamiento de datos. A diferencia de las bases de datos relacionales, que estructuran los datos en tablas y dependen de un esquema predefinido, las bases de datos NoSQL son más flexibles y pueden manejar una amplia variedad de tipos de datos, lo que las hace especialmente útiles para trabajar con grandes volúmenes de datos y datos semi-estructurados o no estructurados. También ofrecen un alto rendimiento y escalabilidad.

Existen varios tipos de bases de datos NoSQL, cada una optimizada para diferentes tipos de tareas:

• Bases de datos de clave-valor: Estas son las bases de datos NoSQL más simples. Almacenan datos como pares de claves y valores. Son altamente escalables y rápidas para recuperar datos, pero la información tiende a estar menos estructurada y puede ser más difícil de buscar en comparación con otros tipos. Ejemplos de este tipo de bases de datos son Redis y DynamoDB.

• Bases de datos de documentos: Almacenan datos en documentos, que son registros semi-estructurados que pueden contener varios campos diferentes. Estos campos pueden variar entre documentos, lo que las hace flexibles para manejar datos no estructurados. MongoDB es el ejemplo más conocido de este tipo de base de datos.

• Bases de datos de columnas: Organizan los datos por columnas en lugar de por filas, lo que puede hacer que las operaciones de lectura sean muy rápidas. Este tipo de base de datos es especialmente útil para análisis de datos y para aplicaciones que requieren la agregación de grandes cantidades de datos, como contar el número de visitas a un sitio web. Un ejemplo de este tipo de base de datos es Cassandra.

• Bases de datos de grafos: Estas bases de datos son únicas en su capacidad para almacenar datos y también las relaciones entre esos datos. Son especialmente útiles para entender y analizar conexiones y relaciones entre entidades, como la red de amigos en una plataforma de medios sociales. Ne04j es un ejemplo de este tipo de base de datos.

**3.6. Tipos según la ubicación de la información**

En función de donde se almacenan y gestionan los datos, podemos clasificar las bases de datos en tres tipos principales: centralizadas, distribuidas y en la nube.

Más adelante, en el temario, les dedicaremos un capítulo exclusivamente para tratarlas en profundidad. Ahora, simplemente las presentamos resumidamente.

• Bases de datos centralizadas: En una base de datos centralizada, todos los datos están almacenados y gestionados en un solo lugar, como un servidor o un centro de datos. Este tipo de base de datos es accesible desde cualquier lugar a través de la red, pero todos los cambios, consultas y gestiones de los datos se realizan en la ubicación centralizada.

Las bases de datos centralizadas son simples de administrar debido a su única ubicación. Sin embargo, pueden presentar problemas de escalabilidad y rendimiento a medida que aumenta el volumen de datos y la cantidad de usuarios que acceden a ellos. Un ejemplo de base de datos centralizada podría ser la utilizada por una pequeña empresa para gestionar sus operaciones diarias. Por ejemplo, una tienda de ropa podría tener una base de datos centralizada en su oficina principal, donde guarda la información de sus productos, ventas, proveedores, etc. Esta base de datos puede ser accesible para los empleados en las tiendas a través de una red interna.

• Bases de datos distribuidas: En una base de datos distribuida, los datos se almacenan en múltiples ubicaciones físicas. Pueden estar divididos (particionados) entre diferentes sitios, o pueden replicarse, donde cada ubicación tiene una copia de todos los datos. Las bases de datos distribuidas pueden ser más complejas de administrar debido a su naturaleza distribuida, pero ofrecen ventajas en términos de rendimiento, escalabilidad y resistencia a fallos.

Un ejemplo de base de datos distribuida podría ser la base de datos de una empresa de telecomunicaciones que tiene presencia en varios países. Cada país podría tener su propia base de datos que almacena la información de los clientes locales, pero todas estas bases de datos podrían estar conectadas para proporcionar una visión global a la empresa. Esto permitiría a la empresa manejar grandes volúmenes de datos y proporcionar un servicio más rápido a sus clientes, ya que las consultas y transacciones pueden ser manejadas localmente en lugar de ser enviadas a una ubicación central.

• Bases de datos en la nube: Las bases de datos en la nube aprovechan la infraestructura de los proveedores de servicios en la nube para almacenar y gestionar datos. Ofrecen escalabilidad, flexibilidad y costos operativos bajos, ya que no requieren la inversión y el mantenimiento de hardware físico. Los datos en una base de datos en la nube pueden ser accesibles desde cualquier lugar con una conexión a Internet, lo que las hace ideales para aplicaciones que requieren alta disponibilidad.

Un ejemplo de base de datos en la nube podría ser la base de datos utilizada por una aplicación de redes sociales como Instagram. Instagram utiliza Amazon Web Services (AWS), un proveedor de servicios en la nube, para almacenar y gestionar los datos de sus usuarios, como las imágenes que suben, los comentarios que hacen y los perfiles que siguen. Esta base de datos en la nube permite a Instagram manejar la enorme cantidad de datos que genera y proporciona a los usuarios un acceso rápido y fiable a sus datos desde cualquier lugar del mundo.

**4. Sistemas gestores de base de datos (SGBD**)

Las bases de datos son una herramienta esencial en la era digital. Estas representan una manera sistemática y organizada de recopilar, gestionar y recuperar una gran cantidad de información. En esencia, una base de datos es un conjunto de datos relacionados entre sí de alguna manera, usualmente estructurados en tablas de datos que permiten el procesamiento y análisis eficiente de la información.

Desde el mantenimiento de registros financieros y la administración de inventarios, hasta el análisis de tendencias de mercado y la toma de decisiones basada en datos, las bases de datos desempeñan un papel crucial en diversas aplicaciones en todos los sectores de la industria.

En este apartado, exploraremos la definición y los componentes de una base de datos, su relevancia y aplicabilidad en diversos contextos, y algunos ejemplos de cómo las bases de datos se utilizan en diferentes industrias. Este conocimiento será fundamental para comprender cómo se gestionan los datos en un mundo cada vez más impulsado por la información.

**4.1. Funciones de un SGBD**

Las funciones clave de un SGBD incluyen:

• Creación y Mantenimiento de la Base de Datos: Un SGBD permite a los usuarios definir la estructura de una base de datos, creando y modificando tablas, campos y registros. También permite la creación de restricciones y reglas para garantizar la consistencia y la integridad de los datos.

• Manipulación de Datos: Un SGBD facilita operaciones como la inserción, actualización, eliminación y consulta de datos en la base de datos. Esto se hace mediante un lenguaje de manipulación de datos, que puede ser estructurado, como SQL, o no estructurado.

• Control de Acceso: Un SGBD tiene mecanismos para controlar el acceso a los datos. Puede proporcionar diferentes niveles de acceso a diferentes usuarios, garantizando así la seguridad y privacidad de los datos.

• Gestión de Transacciones: Un SGBD asegura que todas las transacciones se realicen de forma segura y consistente, incluso en casos de fallos del sistema. Esto se logra mediante mecanismos como el bloqueo y la simultaneidad de transacciones.

• Recuperación de Datos: En caso de un fallo del sistema, un SGBD tiene la capacidad de recuperar los datos perdidos a su estado anterior. Esta característica es esencial para asegurar la durabilidad y la integridad de los datos.

• Optimización de Consultas: Un SGBD tiene la capacidad de optimizar las consultas realizadas a la base de datos para garantizar un rendimiento eficiente. Para ello, utiliza diversas técnicas y algoritmos para determinar el plan de consulta más eficiente.

Ejemplos de SGBD incluyen MySQL, PostgreSQL, Oracle Database, Microsoft SQL Server, y MongoDB, cada uno con sus propias características y ventajas.

**4.1.1. Evaluación de la utilidad de un SGBD**

La utilidad de un Sistema Gestor de Bases de Datos (SGBD) puede ser evaluada desde múltiples perspectivas, y su valor se refleja tanto en la eficiencia de las operaciones diarias como en la capacidad de una organización para tomar decisiones basadas en datos. A continuación, se describen algunas de las formas en las que un SGBD puede ser útil:

* Manejo Eficiente de Grandes Cantidad de Datos: Los SGBD son capaces de gestionar grandes cantidades de datos de manera eficiente. Su diseño permite la inserción, actualización, eliminación y consulta de datos a gran escala, facilitando la manipulación y análisis de los mismos.
* Integridad de Datos: Los SGBD tienen mecanismos para garantizar la integridad de los datos. Estos mecanismos evitan que los datos se vuelvan inconsistentes o incorrectos, proporcionando confianza en la exactitud de los datos.
* Seguridad de Datos: Los SGBD ofrecen una variedad de características de seguridad, incluyendo el control de acceso, el registro de auditoría y la encriptación de datos. Estas características protegen los datos de accesos no autorizados y brechas de seguridad.
* Facilita el Acceso y Compartición de Datos: Los SGBD permiten a múltiples usuarios acceder y manipular datos simultáneamente. Esto facilita la colaboración y el intercambio de información entre diferentes departamentos o equipos de trabajo.
* Resiliencia ante Errores: Los SGBD están diseñados para ser resistentes a los errores. En caso de una interrupción inesperada, los SGBD tienen mecanismos para recuperar los datos a su estado anterior, asegurando la durabilidad de los datos.
* Reducción de Redundancia de Datos: Los SGBD permiten a las organizaciones reducir la redundancia de datos al almacenar la información en un solo lugar. Esto no solo ahorra espacio de almacenamiento, sino que también mejora la eficiencia y la coherencia de los datos.

**4.2. Estructura y componentes de un SGBD**

Un Sistema Gestor de Bases de Datos (SGBD) está formado por varios componentes interrelacionados que trabajan juntos para proporcionar funciones de gestión de datos. Estos son los principales componentes de un SGBD:

• Motor de Base de Datos: El motor de la base de datos es el núcleo de un SGBD. Es responsable de la gestión de los datos almacenados, lo que incluye operaciones de consulta, inserción, actualización y eliminación. El motor de base de datos también maneja la optimización de consultas para garantizar que las operaciones de la base de datos se realicen de la manera más eficiente posible.

• Diccionario de Datos: El diccionario de datos, también conocido como catálogo del sistema, almacena metadatos sobre la base de datos. Esto incluye información sobre la estructura de la base de datos, tales como las tablas existentes, los campos en cada tabla, las relaciones entre las tablas, las restricciones y los índices.

• Administrador de Transacciones: Este componente maneja las transacciones en la base de datos, asegurándose de que sean ejecutadas de manera correcta y segura. El administrador de transacciones también es responsable de mantener la integridad de los datos en caso de fallos del sistema, mediante el uso de técnicas como el registro de transacciones y la recuperación después de fallos.

• Administrador de Concurrencia: El administrador de concurrencia controla el acceso simultáneo a los datos para garantizar la consistencia y evitar conflictos entre transacciones simultáneas. Utiliza mecanismos como el bloqueo y el protocolo de commit de dos fases para gestionar el acceso concurrente.

• Administrador de Seguridad: Este componente es responsable de la seguridad de la base de datos. Controla el acceso a los datos y protege contra amenazas a la seguridad, como el acceso no autorizado. El administrador de seguridad implementa políticas de seguridad a través de características como el control de acceso basado en roles, la autenticación y la encriptación de datos.

• Interfaz de Usuario: La interfaz de usuario proporciona un medio para que los usuarios interactúen con la base de datos. Puede variar desde una simple línea de comandos hasta una interfaz gráfica de usuario completa, dependiendo del SGBD específico.

Estos componentes trabajan juntos para proporcionar un entorno robusto y seguro para la gestión de datos. La estructura y las funcionalidades específicas pueden variar entre los diferentes SGBD, pero estos componentes representan los elementos fundamentales que comúnmente se encuentran en un SGBD.

**4.3. Tipos de SGBD**

Los Sistemas Gestores de Bases de Datos se pueden clasificar de varias formas, según otros criterios. Algunos de los criterios de clasificación más comunes son los siguientes:

1. Según el modelo de datos:

Relacionales

Orientados a objetos

NoSQL

Distribuidos

En la nube

2. Según la arquitectura:

Centralizados

Cliente-Servidor

Peer-to-Peer

Federados

Sistemas distribuidos de bases de datos

3. Según el número de usuarios:

Monousuario: Solo un usuario puede acceder a la base de datos a la vez.

Multiusuario: Muchos usuarios pueden acceder a la base de datos

simultáneamente.

4. Según el número de sitios:

Locales: Cuando la base de datos se ubica en un solo lugar.

Distribuidos: Cuando los datos se reparten en diferentes ubicaciones

5. Según la forma de uso:

Comerciales: Son aquellos SGBD que son propiedad de empresas y se venden con licencia. Ejemplos de estos son Oracle Database, Microsoft SQL Server, IBM DB2, etc.

De código abierto: Son SGBD cuyo código fuente está disponible para el público y se puede modificar y distribuir libremente. Ejemplos de estos son MySQL, PostgreSQL, MongoDB, etc.

Es importante tener en cuenta que estos criterios de clasificación no son mutuamente excluyentes, y un SGBD puede pertenecer a varias categorías al mismo tiempo. Por ejemplo, un SGBD puede ser a la vez relacional y multiusuario, o puede ser de código abierto y distribuido. La elección de un tipo de SGBD sobre otro dependerá de las necesidades específicas del proyecto o la organización.

**4.3.1. Clasificación según el modelo de datos**

Vamos a detallar un poco más el principal criterio de clasificación de los SGBD, que es según el modelo de datos. Nos encontramos con:

• SGBD Relacionales (RDBMS): Son los sistemas de bases de datos más comunes. Estos almacenan los datos en tablas y utilizan el lenguaje SQL para gestionar los datos. Ejemplos de RDBMS incluyen Oracle, MySQL, MS SQL Server, PostgreSQL, entre otros.

• SGBD Orientados a Objetos (OODBMS): Estos sistemas gestores de bases de datos almacenan los datos en forma de objetos, tal como se usan en la programación orientada a objetos. Aunque no son tan comunes como los RDBMS, se utilizan en aplicaciones específicas que requieren un manejo complejo de los datos, como sistemas CAD (Diseño Asistido por Computadora), sistemas de simulación, y más. Un ejemplo de OODBMS es MongoDB.

• SGBD NoSQL: Estos sistemas de bases de datos son una alternativa a los SGBD relacionales y se utilizan en aplicaciones que requieren la gestión de grandes volúmenes de datos que no se ajustan bien a los modelos tabulares. Existen varios tipos de bases de datos NoSQL, incluyendo bases de datos de documentos (por ejemplo, MongoDB, CouchDB), bases de datos de columnas (por ejemplo, Cassandra, HBase), bases de datos de clave-valor (por ejemplo, Redis, DynamoDB) y bases de datos de grafos (por ejemplo, Ne04j, GraphDB).

• SGBD Distribuidos: Un SGBD distribuido gestiona una base de datos que se almacena en varias máquinas. Estos sistemas se utilizan para mejorar el rendimiento y la disponibilidad de los datos.

• SGBD en la nube: Un SGBD en la nube es un tipo de SGBD distribuido que almacena los datos en la nube. Proporcionan escalabilidad y flexibilidad, y reducen la necesidad de mantenimiento de hardware local. Ejemplos de este tipo de sistemas son Amazon RDS, Google Cloud SQL, y Microsoft Azure SQL Database.

**4.4. Comparativa entre SGBD y sistemas de ficheros clásicos**

Los sistemas gestores de bases de datos (SGBD) y los sistemas de ficheros clásicos son dos enfoques diferentes para gestionar la información. Aunque ambos son útiles y tienen sus ventajas, hay diferencias fundamentales entre ellos que los hacen más apropiados para diferentes aplicaciones. Aquí presentamos una comparación de estas dos formas de gestionar los datos.

Sistemas de Ficheros Clásicos:

• Estructura de Datos: En un sistema de ficheros, los datos se almacenan en archivos independientes. Cada archivo puede tener una estructura propia, y los datos se acceden mediante aplicaciones de software específicas que conocen esta estructura.

• Redundancia de Datos: Los sistemas de ficheros pueden dar lugar a una alta redundancia de datos, ya que los mismos datos pueden almacenarse en varios archivos.

• Integridad de los Datos: En los sistemas de archivos, la responsabilidad de mantener la integridad de los datos recae en la aplicación que accede a los datos.

• Acceso a los Datos: En un sistema de ficheros, los datos se acceden a través del sistema operativo, lo que puede hacer que el acceso a los datos sea más lento.

Sistemas Gestores de Bases de Datos (SGBD):

• Estructura de Datos: En un SGBD, los datos se almacenan en tablas relacionadas entre sí. La estructura de las tablas es consistente y los datos se acceden mediante un lenguaje estándar de consulta de bases de datos (SQL).

• Redundancia de Datos: Los SGBD minimizan la redundancia de datos, ya que cada dato se almacena solo una vez en la base de datos. Cuando se necesita la información en varias tablas, se utilizan las relaciones entre las tablas en lugar de duplicar los datos.

• Integridad de los Datos: Los SGBD mantienen la integridad de los datos automáticamente, aplicando restricciones de integridad en la base de datos.

• Acceso a los Datos: En un SGBD, los datos se acceden directamente a través del gestor de la base de datos, lo que puede hacer que el acceso a los datos sea más rápido.

Recuerda   
Los SGBD proporcionan una gestión de datos más eficiente, consistente y segura que los sistemas de ficheros clásicos. Sin embargo, los sistemas de ficheros pueden seguir siendo útiles para aplicaciones más sencillas que no requieren la complejidad de una base de datos.

**4.5. Sistemas gestores de bases de datos comerciales y libres**

Los Sistemas Gestores de Bases de Datos (SGBD) son herramientas esenciales para la gestión de la información en la era digital. Estos sistemas varían en complejidad y funcionalidad, y se pueden dividir en dos categorías principales: los SGBD comerciales y los SGBD libres o de código abierto.

SGBD comerciales:

Los SGBD comerciales son aquellos que requieren la compra de una licencia para su uso. Estos sistemas suelen estar respaldados por grandes empresas de software y cuentan con soporte técnico profesional. Ofrecen una amplia gama de características avanzadas y suelen ser muy fiables y eficientes. Ejemplos de SGBD comerciales incluyen:

• Oracle Database: Es uno de los SGBD más populares y potentes del mundo. Oracle es conocido por su escalabilidad, su soporte para transacciones complejas y su alto rendimiento.

• Microsoft SQL Server: Es una solución completa para la gestión de bases de datos, análisis y reportes. SQL Server es especialmente popular en el entorno empresarial.

• IBM DB2: DB2 es una familia de productos de gestión de datos de IBM que ofrecen soluciones avanzadas para la gestión de datos, incluyendo la inteligencia artificial.

SGBD Libres o de Código Abierto:

Los SGBD de código abierto son aquellos que están disponibles para el público de forma gratuita. El código fuente de estos sistemas está abierto a la comunidad de desarrolladores, lo que permite la colaboración y la mejora continua. Ejemplos de SGBD de código abierto incluyen:

• MySQL: Es uno de los SGBD de código abierto más populares. MySQL es utilizado por muchas empresas de internet de alto perfil, incluyendo Facebook y Twitter.

• PostgreSQL: Este SGBD es conocido por su conformidad con los estándares y su soporte para características avanzadas como las transacciones y subconsultas.

• MongoDB: MongoDB es una base de datos NoSQL de código abierto que es especialmente útil para trabajar con grandes volúmenes de datos y para aplicaciones que requieren una alta disponibilidad.

Cada tipo de SGBD tiene sus propias ventajas. Los SGBD comerciales suelen ofrecer un rendimiento superior, características avanzadas y soporte profesional. Por otro lado, los SGBD de código abierto ofrecen la flexibilidad de la personalización, la transparencia del código y la reducción de costos.

La elección entre un SGBD comercial y uno libre depende en última instancia de las necesidades específicas de cada proyecto o empresa.

**5. Fragmentación de la información**

Vamos a detallar un poco más el principal criterio de clasificación de los SGBD, que es según el modelo de datos. Nos encontramos con:

Fragmentación horizontal: La fragmentación horizontal divide la base de datos en filas. Cada fragmento en este caso contiene un subconjunto de las filas de la tabla original. Esta fragmentación es útil cuando diferentes usuarios necesitan acceder a diferentes filas de una tabla. Por ejemplo, un banco puede tener una tabla de clientes y decidir dividirla horizontalmente basándose en la ubicación geográfica de los clientes. De este modo, los empleados en una región específica solo tendrán acceso a los datos de los clientes de su propia región.

Fragmentación vertical: La fragmentación vertical divide la base de datos en columnas. En este caso, cada fragmento contiene un subconjunto de las columnas de la tabla original. Esta fragmentación es útil cuando diferentes usuarios necesitan acceder a diferentes columnas de una tabla. Por ejemplo, en una compañía de seguros, el departamento de siniestros puede necesitar solo la información de contacto del cliente y los detalles del siniestro, mientras que el departamento de facturación puede necesitar la información de contacto y los detalles de facturación. Mediante la fragmentación vertical, cada departamento tiene acceso solo a los datos que necesita.

Fragmentación mixta: La fragmentación mixta combina la fragmentación horizontal y vertical para dividir la base de datos tanto en filas como en columnas. Esta técnica se utiliza cuando se necesita un enfoque más personalizado para acceder a los datos. Por ejemplo, una empresa de comercio electrónico puede dividir su base de datos de clientes tanto horizontal como verticalmente, permitiendo que cada departamento acceda solo a la información relevante para su área geográfica y a las columnas de datos específicas que necesita.