Unidad 1

Almacenamiento de la información

Gestión de bases de datos

Índice

Introducción

La RAE (Real Academia de la Lengua Española), dice sobre la informática lo siguiente:

> "Conjunto de conocimientos científicos y técnicos que hacen posible el tratamiento automático de la información por medio de ordenadores".

Por otra parte, la universidad de Cambridge comenta los siguiente:

> "Ciencia y actividad encargada de usar ordenadores y otras herramientas electrónicas para almacenar y enviar información".

Después de estas dos definiciones, podemos decir que un sistema de información es un conjunto de procesos y funciones que tienen como finalidad procesar la información de distintas formas dentro de un mismo ámbito u organización.

Cuando aún no existían las bases de datos, toda esta recogida de información se hacía a través de ficheros que eran específicos para cada sistema de información, lo que lo hacía muy trabajoso.

Englobados en este método, no se contemplaba que la información se gestionará a medio o largo plazo debido a los siguientes problemas:

- Redundancia de datos
- Mal aprovechamiento del almacenamiento.
- Tiempos de proceso mucho más largos.
- La redundancia provocaba una inconsistencia en los datos almacenados.
- La información no se podía trasmitir de unos a otros sistemas.

A lo largo de esta unidad vamos a ver cómo han ido evolucionando los sistemas de información y algunas nociones sobre ellos.

Al finalizar esta unidad

- Sabremos que inconvenientes posee el almacenamiento óptimo de la información.
- Aprenderemos historia sobre cómo se han desarrollado los sistemas de información.
- Conoceremos las características de los tipos de fichero.
- Sabremos a que nos referimos con índice.
- Describiremos los diferentes modelos de bases de datos.
- Comprenderemos las funciones de los sistemas gestores de bases de datos.

1.1. Ficheros

En la década de los setenta, los procesos básicos en una empresa se centraban en el tratamiento de la información de su actividad, principalmente en la facturación. Estos procesos requerían relativamente pocos archivos de papel para el almacenamiento y gestión de información. Con el paso del tiempo, el volumen de operaciones que tenían las empresas fue incrementándose, dando lugar a trabajar con un gran cantidad de documentación en papel.

El origen de los ficheros surgió en la primera informatización permitiendo un acceso más rápido a los datos en formato digital. La información se transfería del papel al formato digital y generalmente necesitaba ser almacenada para su posterior recuperación, consulta y procesamiento.

Podemos definir un **fichero** o **archivo** como una herramienta para el almacenamiento permanente de datos en dispositivos de memoria masiva. Su información es tratada como un todo y está organizada de manera estructurada.

Los ficheros están formados por registros lógicos que contienen datos sobre un solo elemento y están divididos en campos que contienen información individual. Los datos se almacenan de manera que pueden ser agregados, eliminados, actualizados o consultados individualmente en cualquier momento.

Dependiendo de la cantidad de datos que contienen los ficheros, a menudo son demasiado grandes para caber en la memoria principal del ordenador, por lo que solo se transfiere parte de ellos para su procesamiento. La cantidad de información transferida entre el soporte de almacenamiento y la memoria principal del ordenador en una sola operación de lectura / escritura se conoce como **registro físico** o **bloque**. Por lo general, se transfieren varios registros en cada operación de lectura / escritura y el número de registros en un bloque se conoce como **factor de bloqueo** o **bloqueo de registros**.

1.1.1. Tipos de fichero según su estructura de almacenamiento

En lo que a su contenido se refiere, tenemos dos tipos básicos de ficheros:

- **Texto plano.** Se trata de un fichero en el que en su interior nos encontramos cadenas de caracteres legibles por el ser humano y codificados en algún modo determinado (lo normal es UTF-8 o UTF-16). Se pueden leer con programas como un bloc de notas o un procesador de texto. Un ejemplo sería un archivo .txt.

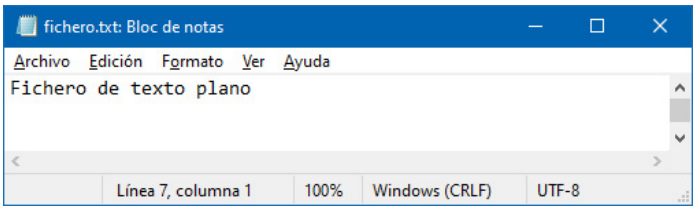


Imagen 1. Ejemplo de fichero de texto plano

- **Binarios.** Los procesan las aplicaciones ya que en su interior se encuentra una información codificada en binario y no se puede leer con un editor de texto. Se trata por ejemplo de un archivo .exe. o una imagen .jpg.

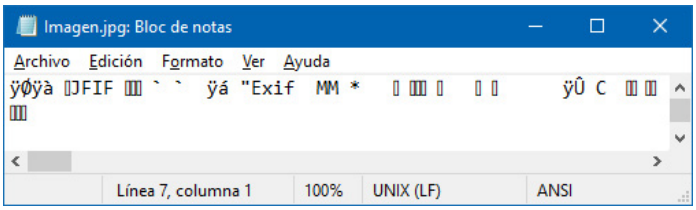


Imagen 2. Ejemplo de fichero binario

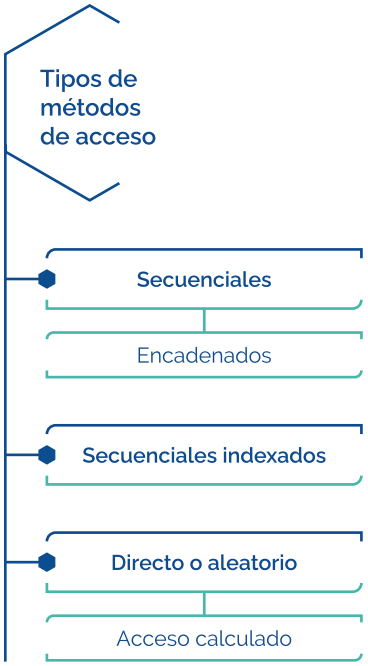
NOTA

Hay que tener en cuenta que, aunque se encuentre en binario, y no sea texto plano, nosotros podemos leer un archivo en binario abriendo por ejemplo un PDF, pero no se puede procesar ese texto.

Si hablamos de ficheros de texto plano, estos podemos distribuirlos según su contenido.

Los ficheros según el método de acceso se pueden clasificar en:

- Ficheros secuenciales.
- Ficheros de acceso directo o aleatorio.
- Ficheros secuenciales encadenados.
- Ficheros secuenciales indexados.
- Ficheros secuenciales indexado-encadenados.
- Ficheros de acceso calculado o Hash.



Ficheros secuenciales

En estos ficheros, los datos se organizan secuencialmente en el orden en el que fueron grabados. Para leer los últimos datos hay que leer todos los anteriores.

Ventajas que tienen estos ficheros:

- Rápidos para obtener registros contiguos.
- No hay huecos en el fichero porque se ha grabado toda la información de manera secuencial.

Las **desventajas** son las siguientes:

- Cuando se realizan consultas, resultan muy lentas ya que tienen que leer todos los datos anteriores al dato que se desea.
- Si queremos reordenar los datos es necesario volver a crearlo en muchas ocasiones.

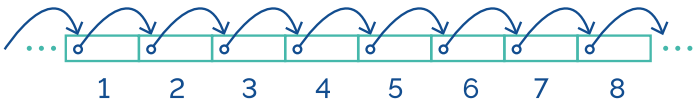


Imagen 3. Representación de un fichero secuencial

Ficheros de acceso directo o aleatorio

Podemos leer una posición concreta del fichero siempre que sepamos exactamente donde se encuentra. Normalmente este dato viene en bytes.

Ventajas:

- Como no tenemos que leer los datos anteriores, el acceso es rápido.
- En ocasiones en las que en una organización sigue una estructuración numérica que se usa también en el fichero es muy útil.

Desventajas:

- Si se hace una consulta a multitud de registros al mismo tiempo, resultan más lentas que las consultas a ficheros secuenciales.

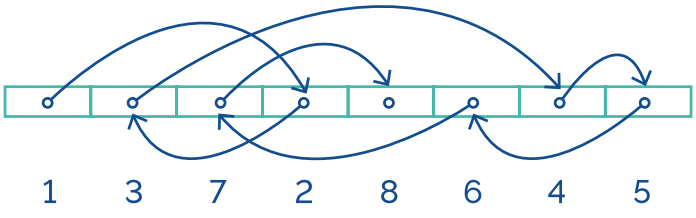


Imagen 4. Representación de un fichero de acceso directo o aleatorio

Ficheros secuenciales encadenados

Estos ficheros se gestionan mediante punteros. Los punteros son datos que nos indican la dirección de cada registro específico de un fichero.

En cada registro tenemos un puntero que nos lleva a la dirección del siguiente y que podemos cambiarlo, esto hacer que se recorra el fichero en un orden previamente dictaminado y no de manera continua.

Si introducimos un registro nuevo, es al final, sin alterar los registros, pero se deben de reordenar los punteros.

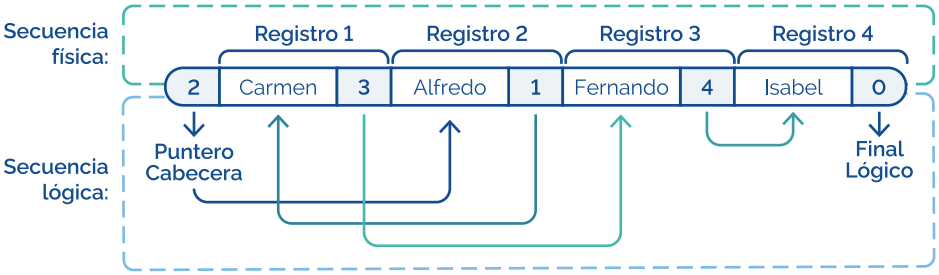


Imagen 5. Ejemplo de fichero secuencial encadenado

Ventajas:

- Se mantiene el orden en que se añaden los registros, pero se sigue un segundo orden en base al puntero.
- Si queremos reorganizar el fichero, no es necesario cambiarlo todo, sino simplemente modificar los punteros.

Desventajas:

- Si se añaden registros o se modifican las claves, hay que reorganizar los punteros.

Ficheros secuenciales indexados

En estos ficheros, se necesitan dos ficheros para almacenar los datos, en uno almacenaremos los registros secuencialmente y en el otro tendemos una tabla que relaciona los punteros con las claves de los registros.

Este segundo fichero recibe el nombre de *índice*.

Lo más importante en estos ficheros es que el de datos tenga todos sus registros ordenados, o si no, no podremos llegar nunca a este a través del índice.

Existe un tercer archivo donde se van añadiendo los nuevos registros antes de pasar a formar parte del fichero de datos, esta es la llamada *área de desbordamiento.* En este fichero los registros están desordenados y solo se accederá a él en caso de que no encontremos el registro en el fichero principal.

Cada cierto tiempo se reorganiza el fichero principal y es ahí donde se añaden los que se encuentren en *overflow* o desbordamiento.

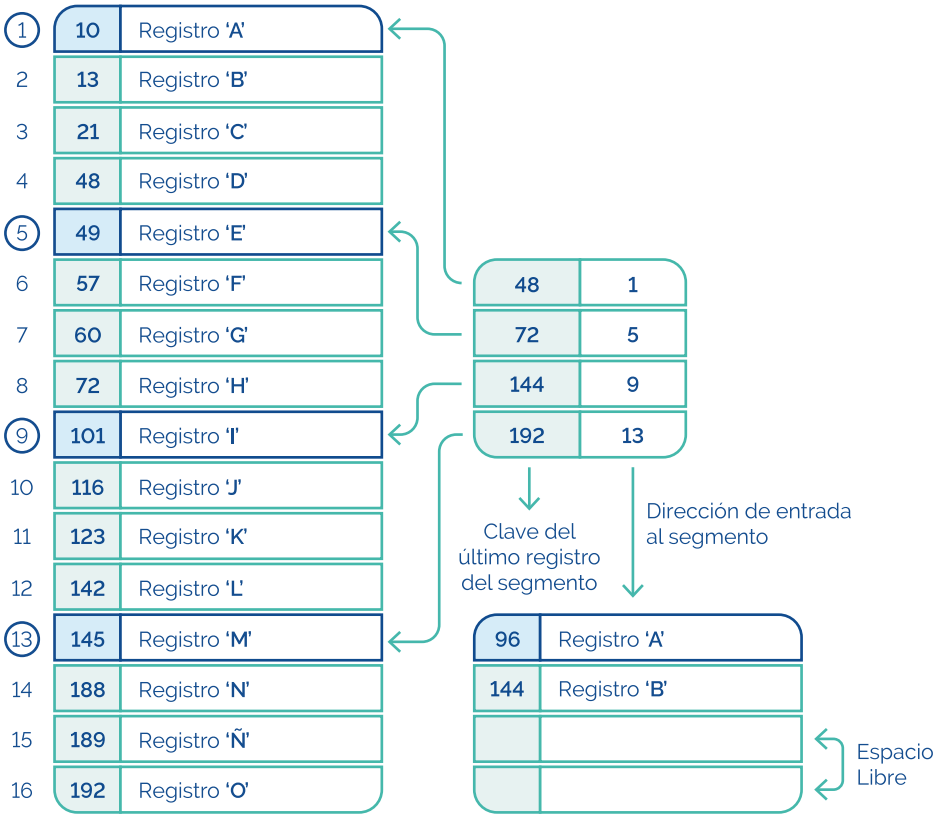


Imagen 6. Esquema de un fichero secuencial indexado

Ventajas:

- El archivo está siempre ordenado conforme estipulen sus claves.
- Se pude hacer una búsqueda de datos de manera casi inmediata.

Desventajas:

- Su uso óptimo pasa por reorganizar el fichero cada vez que se añada uno nuevo, operación que resulta costosa.
- Además, al añadir un registro, también habría que reorganizar los índices, lo que lo hace aún más costoso.

Ficheros secuenciales indexado-encadenados

Es exactamente igual que el anterior con la diferencia de que en este caso, los registros añadidos en el área de desbordamiento se hacen de manera secuencial, lo que lo hace aún más fácil de trabajar.

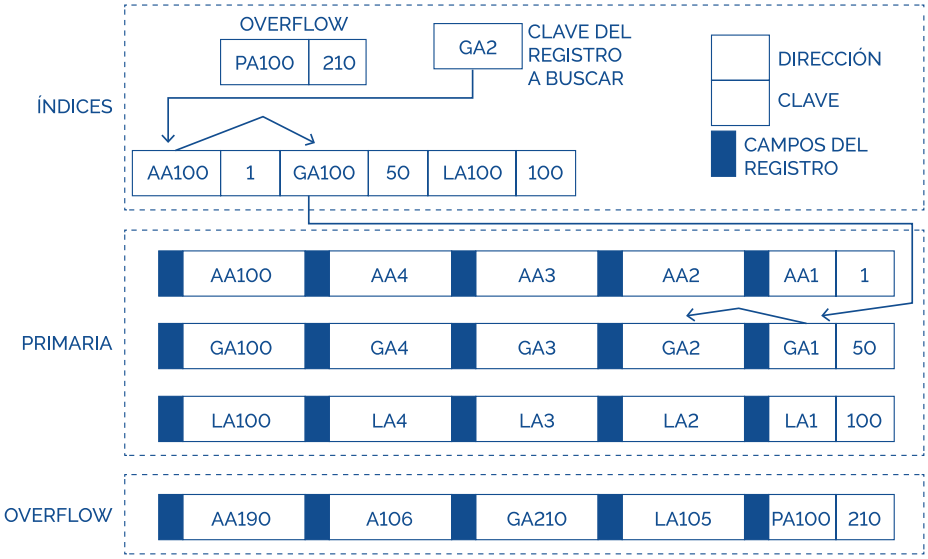


Imagen 7. Esquema de un fichero secuencial indexado-encadenado

Ventajas:

- Las mismas que el anterior tipo de ficheros con la adición de que son aún más rápidos porque al reorganizar el fichero solo se cambian los punteros.

Desventajas:

- Los datos deben de compactarse cada cierto tiempo para así poder reorganizar los índices y vaciar el fichero de overflow.

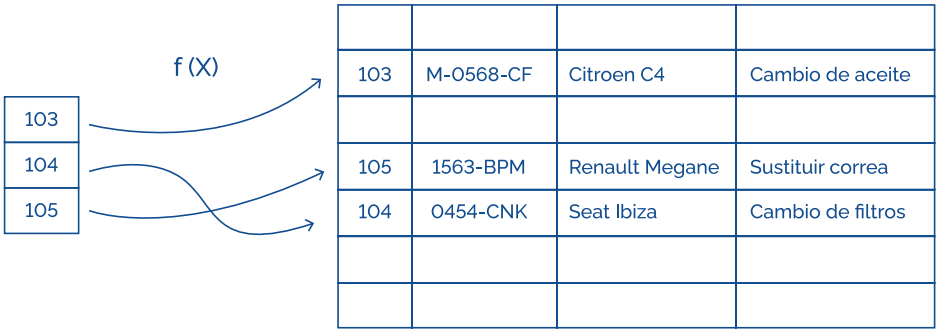
Ficheros de acceso calculado o Hash

También denominado transformación de claves, dispersión o hashing. Los registros se almacenan en una posición específica del fichero que es obtenida al aplicar funciones matemáticas o transformaciones aritméticas sobre la clave.

El resultado de aplicar la función matemática o hash debe dar como resultado un valor que identifique de forma inequívocamente un registro, cuando dos registros diferentes tienen el mismo valor de la función matemática se denomina colisión. Y cuando dos valores diferentes de la función matemática apunta a un mismo registro se denomina sinónimo.

Una buena función matemática o hash, será aquella que produzca el menor número de colisiones. En este caso hay que buscar una función, a ser posible biunívoca, que relacione los posibles valores de la clave con el conjunto de números correlativos de dirección.

Esta función consistirá en realizar una serie de cálculos matemáticos con el valor de la clave hasta obtener un número entre 1 y n, siendo n el número de direcciones que tiene el fichero.



103	M-0568-CF	Citroen C4	Cambio de aceite
105	1563-BPM	Renault Megane	Sustituir correa
104	0454-CNK	Seat Ibiza	Cambio de filtros

Imagen 8. Esquema de un fichero con acceso calculado o hash

Ventajas:

- Aumenta la velocidad de búsqueda sin necesidad de tener los elementos ordenados.
- El tiempo de búsqueda es independiente de la cantidad de registros.
- Dentro del espacio de direccionamiento puede haber posiciones vacías.

Desventajas:

- Es muy dependiente de una buena fórmula matemática o hashing.
- Necesita más uso de procesamiento por aplicar cálculos.
- El acceso ordenado a los registros consume tiempos elevados.

1.1.2. Tipos de soporte de almacenamiento

Si nos centramos en cómo se organizan los datos físicamente, tenemos dos tipos de soportes:

- **Secuenciales**. Si queremos llegar a un dato en concreto, tendremos que recorrer todos y cada uno de los datos anteriores contenidos en el medio físico. Un ejemplo es una cinta magnética.
- **Direccionables**. En este soporte se accede al dato directamente, sin la necesidad de pasar los datos anteriormente, Un ejemplo es un disco duro.

1.2. Bases de datos

El uso de los ficheros traía consigo muchas pegas e inconvenientes que hizo que se tuviera que plantear un nuevo método de acceso a la información estandarizado. Esto se hizo para que independientemente del tipo de información, un diseño físico en particular fuera aplicable a todas las aplicaciones de un mismo sitio u organización. Es aquí cuando nace el concepto de base de datos, que se centra en los datos y no en el proceso; esto conseguía eliminar la redundancia al mismo tiempo que favorecía el traspaso de información entre unas y otras aplicaciones.

1.2.1. Definición

La RAE define el término de Base de Datos como:

> "Memoria informática en la que pueden integrarse datos dispuestos de modo que sean accesibles individualmente por medios electrónicos o de otra forma."

1.2.2. Tipos de bases de datos

El modelo de *base de datos* hace referencia a la arquitectura mediante la cual almacenamos la información gestionándola e interrelacionándola entre sí misma. Se pueden clasificar las bases de datos en los siguientes tipos:

- **Modelo Jerárquico**. El primero y más antiguo, reestructura la idea que se usaba en los ficheros indexados. Se crea una jerarquía entre varios datos de distintos ficheros, propiciando la limitación semántica. Su uso fue casi siempre en grandes máquinas y un uso muy conocido fue en **IMS de IBM**.

- **Modelo en red**. Este modelo surge para mejorar el anterior introduciendo multitud de novedades como la flexibilidad de los datos, pero su nivel de complejidad aumento también de manera considerable. Su uso se puede encontrar en **IDMS (Integrated Database Management System)** o **DMS 1100 de Univac**

- **Modelo Relacional**. El más usado en las bases de datos actuales, el más eficiente hasta el momento y el primero en enseñarnos las bases de datos que se usan en la actualidad. En ese modelo que trataremos con detalle más adelante, la información se almacena en entidades que se relacionan entre ellas formando cadenas de relaciones que hacen que toda la estructura en conjunto tenga una lógica y un sentido. Un ejemplo es **MySQL**, **MariaDB** y **Microsoft SQL Server**.

- **Modelo Orientado a objetos**. Este modelo incorpora la orientación a objetos en las bases de datos, lo que permite almacenar datos complejos, como imágenes y videos, junto con las relaciones lógicas entre ellos. Un ejemplo de su uso es **ObjectDB**. Este modelo es útil cuando se necesita manejar objetos completos y sus interrelaciones dentro de una base de datos.

- **Objeto-relacional.** Aunque hemos dicho antes que las bases de datos relacionales son las más usadas, no es del todo cierto, porque es la fusión entre estas y las orientadas a objetos la que se usa hoy en día. Estas surgen de implementar la programación orientada a objetos en las bases de datos relacionales, surgiendo así el modelo de base de datos más eficiente actualmente.

  Un ejemplo es la base de datos de **Oracle** o **PostGreSQL**, consideradas las más potentes del mundo en muchos aspectos.

- **Documental**. Diseñada para la gestión de documentos completos, que a menudo son grandes y complejos. En este modelo, los documentos pueden contener diversos tipos de información no estructurada, como texto, imágenes, y otros tipos de contenido multimedia. Ejemplos de este tipo pueden ser **Alfresco** o **OpenText**

- **NoSQL.** Agrupa bases de datos que no siguen el modelo relacional tradicional y están diseñadas para manejar grandes volúmenes de datos distribuidos, no estructurados o semi-estructurados.

  Estas bases de datos son altamente escalables y flexibles, lo que las hace ideales para aplicaciones modernas que requieren procesamiento en tiempo real y grandes cargas de trabajo distribuidas.

  Sobre este modelo podemos encontrar otros subtipos, como:

  - **Orientado al documento**. Este modelo está diseñado para la gestión de documentos no estructurados o semi-estructurados, como JSON, XML o BSON. Es particularmente útil en entornos donde los datos no tienen un esquema fijo. Un ejemplo es **MongoDB**.

  - **Clave-Valor.** Almacenan datos como pares clave-valor, donde cada clave es única. Por ejemplo, **Redis** y **Amazon DynamoDB**.

  - **Basado en grafos.** Se utilizan para modelar redes complejas de relaciones, como las redes sociales o recomendaciones. Ejemplos son **Neo4j** y **Amazon Neptune**.

  - **Columnares.** Los datos se almacenan por columnas en lugar de filas, lo que mejora el rendimiento de consultas analíticas. Un ejemplo es **HBase**.

- **Multidimensional.** Se utiliza principalmente en entornos de inteligencia de negocios (BI) y almacenes de datos para análisis OLAP (procesamiento analítico en línea). Este modelo organiza los datos en cubos multidimensionales, lo que permite realizar consultas analíticas rápidas sobre grandes volúmenes de datos. Un ejemplo de uso de este modelo es en bases de datos como **SAP HANA**.

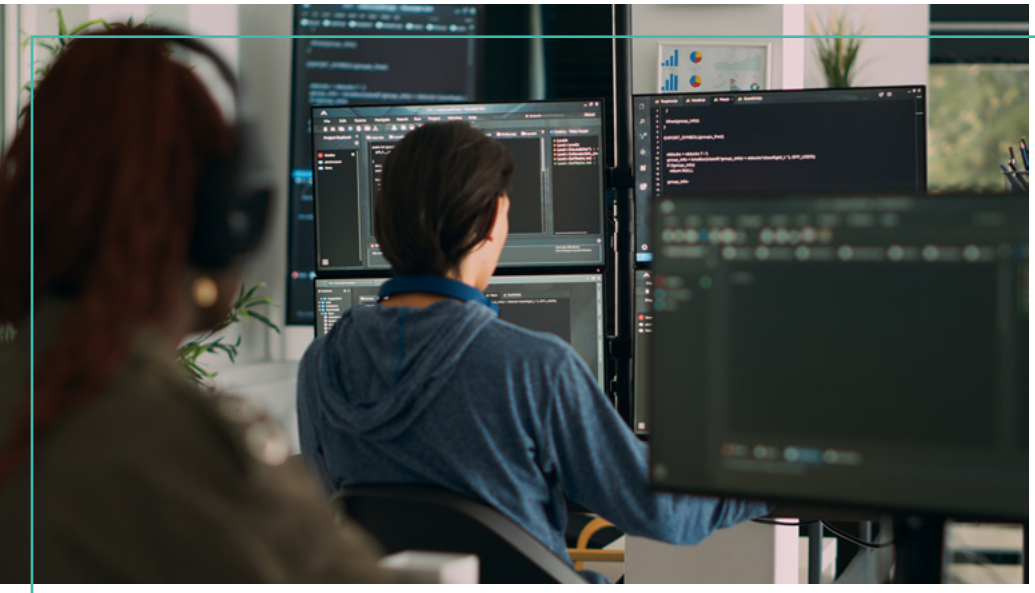
- **Modelo en la nube.** Con el crecimiento del cloud computing, muchas bases de datos se han migrado o diseñado específicamente para la nube. Estas bases de datos ofrecen escalabilidad automática, alta disponibilidad y administración simplificada. Ejemplos de bases de datos en la nube son **Google BigQuery**, **Amazon Aurora** y **Azure Cosmos DB**.

Por otro lado, también se pueden clasificar las bases de datos en dos tipos dependiendo de donde se almacena físicamente la información:

- **Centralizadas.** La base de datos reside en una única máquina. Suele ser el servidor de base de datos.

- **Distribuidas.** Las que más se usan en grandes organizaciones ya que tienen la información repartida en múltiples servidores que no tienen por qué estar cerca físicamente, de hecho, lo más normal es que estén alejados unos de otros. La principal característica es que los datos se replican y se sincronizan entre varios servidores. Esto mejora la disponibilidad y tolerancia a fallos. **CockroachDB** y **Apache Cassandra** son ejemplos de bases de datos distribuidas.

Actualmente, existen bases de datos que combinan características de varios modelos. Un ejemplo de esto son las bases de datos **NewSQL**, que combinan la escalabilidad de las bases de datos NoSQL con las garantías de consistencia y transacciones propias del modelo relacional.



1.3. Sistemas gestores de bases de datos

Cuando tenemos múltiples bases de datos activas como puede ser en una gran organización, se usa un *software específico* que nos ayuda con la administración y gestión de todas estas bases de datos. De nuevo citando a la RAE, su definición para este término es:

> "Conjunto de programas, lenguajes, etc., que permiten la definición, manipulación y administración de la información contenida en una base de datos y asegurar el mantenimiento de la integridad, la confidencialidad y la seguridad de dichos datos."

Existen multitud de opciones en el mercado para los SGBD, todos siguiendo el modelo subyacente de base de datos.

1.3.1. Componentes de un sistema gestor de base de datos

Si hablamos de manera general, los SGBD comparten los siguientes componentes:

- **Datos.** Es la estructura de la información y se almacena en los archivos del sistema operativo eficientemente.
- **Herramientas de acceso a los datos.** Estas herramientas suelen basarse en un lenguaje de programación mediante el cual, los usuarios con conocimientos técnicos pueden gestionar la información dependiendo siempre de los permisos otorgados, la estructura, etc. El lenguaje más usado para esto es SQL.
- **Utilidades.** Herramientas que se usarán de manera adicional como pueden ser las de backup, migración, etc.
- **Entornos gráficos.** Si no se está muy familiarizado con el lenguaje de las bases de datos, este es complicado de usar, por eso muchos SGBD cuentan con una interfaz gráfica como por ejemplo adminer que nos ayuda en la gestión de nuestras bases de datos.

Imagen 9. Adminer, ejemplo de interfaz gráfica para los SGBD

1.3.2. Funciones de un sistema gestor de base de datos

Aunque existen multitud de opciones y de modelos de SGBD, hay unas funciones que están presentes en todos:

- En todos es posible que la información que se almacena en los ficheros de la base de datos se pueda visualizar y modificar sin mucha complicación.
- Se garantiza la integridad de los datos.
- Ofrecer un lenguaje de programación que haga posible el tratamiento de los datos.
- Proveer el diccionario de datos.
- Solucionar conflictos nacientes del acceso concurrente a la información.
- Gestionar las transacciones de manera que se garantice las unidades de múltiples instrucciones que se relacionen entre sí.
- Incluir utilidades para la copia de seguridad.
- Proporcionar los mecanismos de seguridad necesario para que se pueda controlar el acceso y los privilegios al sistema.

1.3.3. Sistemas gestores de base de datos comerciales o libres

En la actualidad existen multitud de gestores de base de datos con diferentes características. Nos encontraremos con SGDB comerciales o libres, una característica que no representa si es gratuito o no, simplemente permite conocer quien está detrás, si una compañía, una comunidad o un grupo para utilizarlo quien quiera.

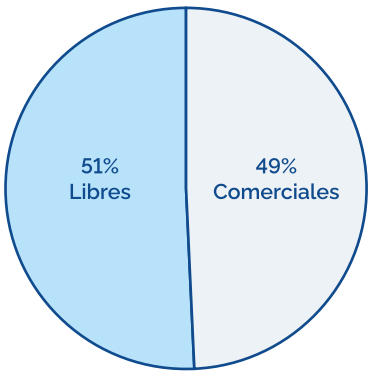


Imagen 10. Estadística de uso SGBD comerciales y libres a febrero de 2023

La elección de un SGDB dependerá de más factores, como puede ser: el sistema operativo donde puede funcionar, el coste, el volumen de datos, el tipo de soporte, etc.

A continuación se detalla un listado de los SGDB más usados:

Listado de SGBD Comerciales		
SGDB	**Tipo**	**Características**
Oracle	Objeto-relacional	+ De pago, existe una versión gratuita Express Edition con limitaciones. + Multiplataforma. + Alto rendimiento. + Gestión de la seguridad.
Microsoft SQL Server	Objeto-relacional	+ De pago, pero tiene versiones gratuitas con limitaciones en sus prestaciones. + Inicios solo para Windows, posteriormente para Linux o Docker. + Potente entorno gráfico. + Escalabilidad, estabilidad y seguridad.
Access	Relacional	+ Se incluye dentro del paquete de office. + Para sistemas operativos de Windows. + Facilidad de administración y uso. + Relativamente es la opción más asequible económicamente.

Listado de SGDB Libres		
SGDB	**Tipo**	**Características**
MySQL	Relacional	+ Comercial / Libre. + Gratuito. Posteriormente lo adquirió la compañía Oracle sacando licencias comerciales y de pago. + Multiplataforma. + Tiene un gran rendimiento. + Se integra sobre cualquier arquitectura.
PostgreSQL	Objeto-relacional	+ Gratuito. + Multiplataforma. + Es robusta, eficiente y estable. + Flexibilidad con los lenguajes de programación.
MongoDB	Orientado al documento	+ Comercial / Libre. + Tiene una licencia libre y gratuita. Se puede optar por licencias comerciales con diferentes precios. + Multiplataforma. + Es muy escalable.
Sqlite	Relacional	+ Gratuito. + Ocupa muy poco espacio. + Gran portabilidad y rendimiento. + No es adecuada para base de datos muy grandes.

La página web DB-Engines (db-engines.com) contiene información estadística del uso de SGBD y su popularidad. Es posible consultar cuales son las más usadas, distinguir por diferentes tipos de base de datos y distinguir entre comerciales y libres.

410 systems in ranking, February 2023

Rank Feb 2023	Rank Jan 2023	Rank Feb 2022	DBMS	Database Model	Score Feb 2023	Score Jan 2023	Score Feb 2022
1.	1.	1.	Oracle	Relational, Multi-model	1247.52	+2.35	-9.31
2.	2.	2.	MySQL	Relational, Multi-model	1195.45	-16.51	-19.23
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	929.09	+9.70	-19.96
4.	4.	4.	PostgreSQL	Relational, Multi-model	616.50	+1.65	+7.12
5.	5.	5.	MongoDB	Document, Multi-model	452.77	-2.42	-35.88
6.	6.	6.	Redis	Key-value, Multi-model	173.83	-3.72	-1.96
7.	7.	7.	IBM Db2	Relational, Multi-model	142.97	-0.60	-19.91
8.	8.	8.	Elasticsearch	Search engine, Multi-model	138.60	-2.56	-23.70
9.	↑ 10.	↑ 10.	SQLite	Relational	132.67	+1.17	+4.30
10.	↓ 9.	↓ 9.	Microsoft Access	Relational	131.03	-2.33	-0.23

Imagen 11. Ranking de SGDB en el mes de febrero de 2023

www.universae.com