

Tema 1: Introducción a las bases de datos.

1.1. Evolución histórica de las bases de datos.

Las bases de datos empezaron a utilizarse a partir de la década de los 70 del siglo XX. Antes de emplearse las bases de datos para almacenar la información se utilizaban archivos o ficheros, por lo que vamos a comenzar explicando el concepto de archivo o fichero.

La parte del ordenador en la que se almacena información se denomina memoria y podemos hablar de dos tipos de memoria fundamentalmente: la memoria principal o memoria RAM y la memoria secundaria.

La memoria RAM es de tipo volátil, lo que quiere decir que la información contenida en ella desaparece al desconectarse el ordenador. Por este motivo, se hace necesaria la existencia de una memoria secundaria en la cual permanezca la información, aunque se apague el ordenador.

La información depositada en la memoria secundaria está organizada en archivos o ficheros, por lo que podríamos decir que un fichero consiste en un conjunto de bytes almacenados de forma organizada en un dispositivo de almacenamiento secundario (disco, CD, DVD, pen drive, etc.).

En los ficheros de datos la información se almacena en unas unidades llamadas registros, cada uno de los cuales a su vez consta de varios campos. Así, por ejemplo, en una empresa que se dedica a la comercialización de productos o servicios se podría disponer de un fichero de clientes. Este fichero constaría de muchos registros, uno por cada uno de los clientes de que dispone la organización. A su vez, cada registro se descompondría en varios campos. Por ejemplo, podríamos almacenar por cada cliente su NIF, nombre, apellidos, dirección, localidad, provincia, e-mail y teléfono. Cada una de estas unidades de información es un campo. En la figura 1 se representa la estructura de registro Cliente y en la figura 2, un ejemplo de información almacenada en el fichero de clientes.

NIF	Nombre	Apellidos	Dirección	Localidad	Provincia	e-mail	Teléfono
-----	--------	-----------	-----------	-----------	-----------	--------	----------

Figura 1: Registro Cliente

12345678C	Ana	Gil Ruiz	Avda. , 5 2º A	Getxo	Vizcaya	agr@r.es	948789989
88776655X	Luis	Gómez García	Mayor, 20	Madrid	Madrid	lgg@r.es	911111111
...
00000000A	María	Pérez España	Menor, 15	Estepona	Málaga	mbe@r.es	666868900

Figura 2: Contenido del fichero Clientes

Vamos a ver a continuación la forma tradicional de gestionar y almacenar los datos mediante el empleo de ficheros. Supongamos una empresa que necesita mantener información acerca de los clientes a los que atiende y acerca de los productos que vende. Sobre los clientes y los productos será necesario realizar diferentes tratamientos, para cada uno de los cuales se dispondrá de una aplicación.

Pues bien, cada una de estas aplicaciones dispondrá de su propio conjunto de ficheros que contendrán los datos necesarios, los cuales estarán organizados de acuerdo con la forma en que son tratados por la correspondiente aplicación. Los ficheros son diseñados, por tanto, para una determinada aplicación y, en consecuencia, la organización de los datos en los ficheros es totalmente dependiente de la aplicación que los trata.

De esta forma, si es necesario cambiar la estructura de alguno de los ficheros, será también necesario modificar la aplicación que los usa. Por otro lado, si es necesario cambiar una aplicación, casi con total seguridad será necesario modificar el número de ficheros que utiliza, su organización, los campos, etc.

Por otro lado, es casi seguro que existirá una alta redundancia de los datos, es decir, que existirán datos repetidos en diversos ficheros, por ejemplo, podrían estar los nombres de los clientes repetidos en varios ficheros. Esto implica que se está ocupando memoria de manera innecesaria, pero tiene otro inconveniente mucho más importante, que es la posibilidad de que se generen inconsistencias por el siguiente motivo: si un dato está repetido en varios ficheros, una modificación del mismo se debe llevar a cabo sobre varios ficheros, de manera que si no se hace así, pueden generarse inconsistencias, es decir, puede ocurrir que el mismo dato tenga diferentes valores en distintos ficheros. Supongamos el caso de que la dirección de un cliente esté almacenada en varios ficheros. Pues bien, si un cliente modifica su dirección, habrá que actualizarla en varios ficheros y si olvidamos actualizarla en uno de ellos, llegará el momento en que no sepamos cuál es su dirección real: la almacenada en un fichero o la guardada en el otro.

Esta manera de trabajar recibe el nombre de sistema orientado al proceso porque la manera en que están almacenados y organizados los datos es totalmente dependiente del proceso que se lleva a cabo con ellos.

Las desventajas de los sistemas tradicionales que trabajaban con ficheros desembocaron en la aparición del concepto de base de datos.

Una base de datos se puede definir como (Piattini et al, 2006): Una colección o depósito de datos integrados con redundancia controlada y con una estructura que refleje las interrelaciones y restricciones existentes en el mundo real. Los datos, que han de ser compartidos por diferentes usuarios y aplicaciones, deben mantenerse independientes de estas, y su definición y descripción únicas para cada tipo de dato, han de estar almacenadas junto con los mismos. Los procedimientos de actualización y recuperación, comunes y bien determinados, habrán de ser capaces de conservar la seguridad (integridad, confidencialidad y disponibilidad) del conjunto de los datos.

1.2. Ventajas e inconvenientes de las bases de datos.

La sustitución de un conjunto de ficheros por una base de datos proporciona las siguientes ventajas:

- Independencia de los datos respecto a los tratamientos y viceversa: esto quiere decir que un cambio en los tratamientos o programas no va a conllevar una modificación en la base de datos, un nuevo diseño de la misma. Por otra parte, la realización de modificaciones sobre la base de datos, como inclusiones o modificaciones de informaciones o cambios en la manera en que se almacenen los datos, no va a implicar casi nunca una modificación en los programas que acceden a los datos de la base de datos.
- Consistencia de los datos: eliminando o controlando las redundancias de datos se reduce el riesgo de que haya inconsistencias. Si la redundancia es mínima y está controlada por el sistema, como ocurre en las bases de datos, el propio sistema se encargará de garantizar que las copias se mantienen consistentes. A diferencia de lo que ocurre con los sistemas tradicionales de ficheros, en una base de datos no hay datos redundantes. Más bien, en una base de datos son solo redundantes o están repetidos únicamente aquellos datos que es imprescindible que lo estén para que en la base de datos se puedan reflejar las relaciones existentes entre los datos. Esto quiere decir que, por ejemplo, en una base de datos nunca aparecerá repetida la dirección de un cliente, pero sí puede aparecer su NIF repetido en varias tablas para reflejar, por ejemplo, que ese cliente ha realizado un pedido.
- Compartición de datos: en los sistemas de ficheros, estos pertenecen a las personas o departamentos que hacen uso de ellos. Cuando se trabaja con una base de datos, esta pertenece a la empresa y puede ser compartida por todos los usuarios que tienen autorización para ello. Para ello, cada usuario deberá conocer su nombre de usuario y

contraseña y en función del perfil del usuario podrá realizar determinadas operaciones sobre la base de datos. Será el administrador de la base de datos (DBA) el encargado de asignar a cada usuario un nombre de usuario y contraseña. Además, también será labor del DBA especificar los privilegios que un usuario tiene sobre los objetos de la base de datos: puede ocurrir que solo pueda consultar ciertas tablas, que otras tablas las pueda consultar y modificar, etc. Para facilitar esta tarea se suelen definir roles, los cuales agrupan un conjunto de privilegios sobre los objetos de la base de datos, de manera que si a un usuario se le asigna un rol, adquiere todos los privilegios del rol en cuestión.

- Mayor valor informativo: en la base de datos se almacenan los datos junto con las interrelaciones existentes entre ellos, por lo que el valor informativo del conjunto (de la base de datos) es mayor que la suma del valor informativo de los elementos individuales.
- Mejora en la accesibilidad a los datos: los sistemas gestores de bases de datos (en adelante, SGBD) incluyen lenguajes, como el SQL, que permiten a los usuarios con pocos conocimientos informáticos realizar consultas sobre los datos sin necesidad de escribir programas para tal fin. Analistas y programadores pueden generar programas para que los usuarios que desconozcan el lenguaje SQL puedan acceder a la base de datos y realizar operaciones sobre ella.
- Mejora en la integridad de los datos: la integridad de la base de datos, que se refiere a la consistencia y validez de los datos almacenados, se expresa mediante determinadas condiciones o restricciones que se deben cumplir, que suelen hacer referencia a los valores permitidos para un atributo. Pues bien, el subsistema de integridad del SGBD se encarga de asegurar que estas condiciones se cumplan. Esto se debe a que en una base de datos no solo se almacenan los datos propiamente dichos, sino también la semántica de los mismos.
- Control de la concurrencia: en algunos sistemas de ficheros, si hay varios usuarios que acceden simultáneamente a un mismo fichero, es posible que se pierda información o que afecte a la integridad de los datos. Sin embargo, los SGBD gestionan el acceso concurrente a la base de datos y permiten que no ocurra este tipo de problemas.
- Reducción del espacio de almacenamiento: la desaparición o disminución de la redundancia en las bases de datos conlleva una ocupación menor de memoria secundaria.

El trabajo con bases de datos se ha generalizado desde hace ya varias décadas debido a sus ventajas frente al trabajo con ficheros. No obstante, los sistemas de bases de datos también presentan algunos inconvenientes que se van a indicar a continuación:

- Instalación costosa: la implantación de un sistema de bases de datos puede implicar elevados costes en adquisición de hardware y software, entre los cuales es de destacar el coste de adquisición y mantenimiento del sistema gestor de la base de datos (SGBD).
- Personal especializado: va a ser necesaria la contratación de personal especializado que se encargue del diseño y administración de la base de datos.
- Falta de rentabilidad a corto plazo: la implantación de un sistema de base de datos, por los costes que conlleva tanto en personal como en equipos y por el tiempo que tarda en estar operativo, no resulta rentable a corto plazo, aunque sí lo sea a medio y largo plazo.

1.3. El sistema de gestión de la base de datos (SGBD/DBMS).

Un sistema de gestión de bases de datos (SGBD), en inglés *Database Management System* (DBMS), es una colección de programas que facilitan la labor de gestionar la base de datos en su conjunto. En general, como indican Oltra et al. (2006), “el SGBD se encargará de gestionar el correcto funcionamiento interno de la base de datos, en lo que se refiere al control de la concurrencia y de la integridad, además de facilitar a los usuarios la creación, el mantenimiento y, en ocasiones, el diseño de dicha base de datos”.

1.4. Anomalías del acceso concurrente.

Las bases de datos se emplean fundamentalmente en entornos multiusuario, lo que quiere decir que lo habitual es que una base de datos sea utilizada por varios usuarios, incluso simultáneamente. El uso concurrente o simultáneo de una base de datos por múltiples usuarios origina problemas de concurrencia. Así, mientras un usuario está modificando un registro de una tabla, no se debe permitir que otros usuarios realicen ninguna operación con dicho registro. Para llevar a cabo esto, existe un componente en el sistema gestor de la base de datos (SGBD) que se encarga de controlar que los accesos concurrentes sobre la base de datos no lleven a estados inconsistentes. Para ello, existen diferentes esquemas de control de concurrencia, como el basado en bloqueos.

Siguiendo a Silberschatz, Korth y Sudarshan (2002), un elemento de datos se puede bloquear en dos modos:

- En modo compartido: si una transacción bloquea un elemento de datos en modo compartido, solo lo puede leer, es decir, no lo puede escribir.
- En modo exclusivo: si una transacción obtiene un bloqueo sobre un elemento de datos en modo exclusivo, puede leerlo y escribirlo.

Cuando una transacción desea realizar una determinada operación sobre un elemento de datos, debe pedir un bloqueo de uno de estos dos tipos según la operación que se desee realizar. Esta petición es enviada al llamado gestor de bloqueos, de manera que la transacción solo podrá realizar la operación después de que le sea concedido el bloqueo solicitado.

Hay que tener en cuenta, como indican Silberschatz, Korth y Sudarshan (2002), que el modo compartido es compatible con otro modo compartido, pero no con el modo exclusivo. Esto quiere decir que puede haber varios bloqueos compartidos simultáneos pedidos por diversas transacciones sobre un mismo elemento de datos. Si sobre ese mismo elemento otra transacción solicita un bloqueo exclusivo, deberá esperar con anterioridad a que se liberen todos los bloqueos en modo compartido.

1.5. Administración de los datos y administración de bases de datos.

Una base de datos se emplea para almacenar la información que se maneja en una empresa, en una institución, etc. Por este motivo es necesario que exista una persona en la organización responsable de los datos que se manejan en la misma. Esta persona debe entender y conocer los datos que se manejan en la organización y las necesidades de la organización en relación con dichos datos. Es responsabilidad de este individuo, al que nos podemos referir como administrador de los datos, decidir cuáles son los datos que se deben almacenar en la base de datos y establecer políticas para mantener y manejar estos datos de la mejor manera posible.

Por otro lado, debe existir otra figura en la empresa, que es el administrador de la base de datos, que se va a encargar de implementar informáticamente las decisiones tomadas por el administrador de datos. El administrador de la base de datos (DBA) debe ser, por tanto, un individuo con los conocimientos informáticos suficientes como para crear la base de datos en un determinado SGBD e implementar los controles necesarios para hacer cumplir las políticas establecidas por el administrador de datos. El administrador de la base de datos también tendrá otras responsabilidades, a las que se hará referencia en la sección 1.11.

1.6. Niveles de arquitectura: interno, conceptual y externo.

Uno de los objetivos de un SGBD es evitar a los usuarios los detalles relativos a la forma en que los datos se almacenan y se mantienen, por lo que el administrador de la base de datos debe describir la estructura de los datos en varios niveles que conforman lo que se conoce como arquitectura de los sistemas de bases de datos. La arquitectura más estandarizada es la que cumple con los requerimientos de la normativa ANSI/X3/SPARC, que establece que la arquitectura de una base de datos debe poseer tres niveles de abstracción:

- Nivel físico: es el nivel más bajo de abstracción, en el que se describe cómo se almacenan físicamente los datos: el tamaño de los bloques de datos, los métodos de direccionamiento, los índices, etc.
- Nivel lógico o conceptual: en este nivel se describe a nivel lógico la totalidad de los datos que van a ser almacenados en la base de datos mediante la especificación de las entidades (por ejemplo, clientes, pedidos y artículos), atributos o propiedades de las entidades (por ejemplo, NIF, nombre, dirección y teléfono del cliente; referencia y fecha del pedido, etc.), relaciones entre las entidades, restricciones de integridad (limitaciones en los valores de los atributos) y restricciones de confidencialidad (permisos de acceso de los usuarios a los datos). Este nivel y el anterior son empleados solo por el administrador de la base de datos.
- Nivel externo o de vistas: muchos usuarios no tienen por qué trabajar con toda la información almacenada en la base de datos, pues precisan solo una parte de ella. Para dar una respuesta adecuada a esta situación se define para cada usuario una vista externa o subesquema de la base de datos, que será por tanto la visión que de la base de datos tiene cada usuario. Una vista será un subconjunto de la estructura lógica global de la base de datos definida en el anterior nivel.

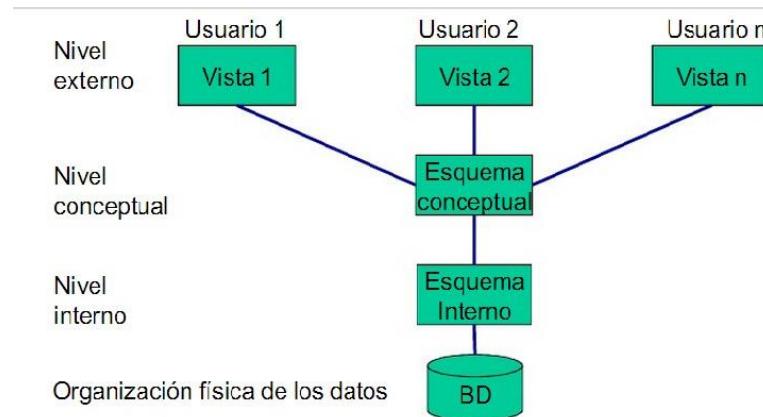


Figura 3: Niveles de la arquitectura ANSI/X3/SPARC

1.7. Modelos de datos. Clasificación.

Cuando queremos crear una base de datos, tenemos como objetivo representar de manera organizada en un dispositivo de almacenamiento una parte de la realidad que nos rodea, con el fin de poder trabajar con esa información de manera más rápida y eficiente que si lo tuviésemos que hacer de forma manual. Pues bien, esa parte de la realidad que deseamos modelar mediante una base de datos es lo que se denomina universo del discurso (UD).

Para poder crear una base de datos se emplean como herramienta los denominados modelos de datos. Podemos definir un modelo de datos como un conjunto de símbolos, conceptos y reglas que nos permiten representar los datos que se van a almacenar en una base de datos. El resultado de la aplicación de un modelo de datos, es decir, la plasmación de la parte de la realidad para la cual deseamos crear la base de datos (UD) mediante el empleo de un determinado modelo de datos, da lugar a lo que se denomina un esquema. Existen varios tipos de modelos de datos aplicables en distintos momentos a lo largo del proceso de creación de una base de datos (modelos conceptuales, modelos lógicos y modelos físicos) dando lugar a diferentes tipos de esquemas (esquemas conceptuales, esquemas lógicos y esquemas físicos, respectivamente).

Los modelos de datos tienen una parte estática y una parte dinámica:

- La estática de un modelo de datos consta de elementos permitidos y elementos no permitidos o restricciones:
 - Los elementos permitidos no son los mismos para todos los modelos, pero la mayoría de ellos incluyen los siguientes:
 - Entidades.
 - Atributos o propiedades de las entidades.
 - Dominios o conjuntos de valores sobre los que se definen los atributos.
 - Relaciones o asociaciones entre objetos.
 - La manera de representar estos elementos depende del modelo, pero por lo general se hace en forma de grafo.
 - Elementos no permitidos o restricciones: se trata de lo que se pueden considerar ocurrencias no permitidas, es decir, ciertos valores que no se pueden almacenar en una base de datos.
- La dinámica de un modelo de datos consta de un conjunto de operadores que se definen sobre la estructura del correspondiente modelo. Los valores de los datos almacenados en

un esquema se llaman ocurrencia del esquema o estado de la base de datos en un instante de tiempo t_i (BD_i). Pues bien, la aplicación de una operación sobre una ocurrencia del esquema transforma a esta en otra ocurrencia.

$$O(BD_i) = BD_j$$

Por ejemplo, si tenemos guardados en una tabla de una base de datos los datos de cinco clientes, eso constituye una ocurrencia del esquema. Si añadimos un nuevo cliente, es decir, si realizamos la operación de inserción de un nuevo cliente, llegamos a otra ocurrencia del esquema o nuevo estado de la base de datos, en el que ya no hay cinco clientes almacenados, sino seis.

Se van a indicar a continuación los pasos secuenciales que es necesario llevar a cabo para crear una base de datos:

- Diseño conceptual: consiste en representar el UD usando un modelo de datos conceptual, obteniendo de esta forma lo que se denomina un esquema conceptual. Estos modelos son altamente semánticos e independientes del tipo de base de datos que se vaya a utilizar con posterioridad. Esto quiere decir que esta tarea se puede llevar a cabo aun desconociendo el SGBD que se vaya a utilizar en fases posteriores. El modelo de datos masivamente utilizado en la actualidad a nivel mundial para la realización de esta tarea es el modelo Entidad-Interrelación o modelo Entidad-Relación (modelo E-R).
- Diseño lógico: consiste en transformar el esquema conceptual obtenido en la fase anterior en un esquema lógico aplicando una serie de reglas de transformación dependientes del modelo lógico y, por lo tanto, del tipo de base datos que deseemos crear. Los modelos lógicos que se han venido empleando a lo largo de la historia para las bases de datos son, en orden cronológico, el modelo jerárquico, el modelo en red, el modelo relacional y el modelo orientado a objetos.
- Diseño físico: consiste en transformar el esquema lógico obtenido en la fase anterior en un esquema físico, lo que requiere crear en un SGBD concreto (MySQL, Oracle, PostgreSQL, etc) todos los elementos de que consta la base de datos: dominios, tablas, restricciones, índices, etc.

Existen varios tipos de modelos de datos aplicables en distintos momentos a lo largo del proceso de creación de una base de datos (modelos conceptuales, modelos lógicos y modelos físicos). Estos modelos se emplean en las fases del diseño de bases de datos llamadas diseño conceptual, diseño lógico y diseño físico, respectivamente.

1.7.1. Modelos de datos conceptuales.

Los modelos de datos conceptuales son los primeros que se emplean en el proceso de creación de una base de datos. Estos modelos son altamente semánticos e independientes del tipo de base de datos que se pretenda crear.

En estos modelos se describen los datos del universo del discurso tal y como los captamos en el mundo real, esto es, alejados de su implementación en el ordenador en un SGBD concreto.

El modelo de datos conceptual más empleado en la actualidad y utilizado desde hace ya más de 30 años es el modelo Entidad-Interrelación, más conocido como modelo Entidad-Relación o modelo E-R. Este fue propuesto por Peter Chen en el año 1976. Posteriormente, otros autores realizaron aportaciones al modelo propuesto por Chen, dando lugar a lo que se conoce como el modelo extendido Entidad-Relación. Otro modelo conceptual importante en la actualidad es el modelo orientado a objetos, que surgió posteriormente y que difiere con respecto al modelo E-R en que en él no solo se representan los elementos de datos, sino también las operaciones o métodos aplicables sobre esos elementos de datos.

1.7.2. Modelos de datos lógicos.

Los modelos de datos lógicos se emplean para crear un esquema lógico que represente la estructura de la base de datos que se va a crear. El esquema lógico se crea a partir del esquema conceptual obtenido en la fase previa de diseño conceptual y para ello, se aplican sobre el esquema conceptual una serie de reglas de transformación que son diferentes dependiendo del tipo de base de datos que se vaya a utilizar. Así, cronológicamente han existido a lo largo de la historia las bases de datos jerárquicas, en red, las relacionales y las orientadas a objetos, por lo que podemos hablar de cuatro modelos de datos lógicos: el modelo jerárquico, el modelo en red, el modelo relacional y el orientado a objetos. El más empleado en la actualidad es el modelo relacional.

El modelo relacional es posterior a los modelos jerárquico y en red y fue desarrollado por Codd en 1970. En el modelo relacional se emplean tablas para la representación lógica de los datos y las relaciones entre ellos.

Se llama tupla a cada fila de la tabla y campo o atributo a cada columna de la tabla. Una clave es un atributo o conjunto de atributos que identifica de manera única a cada tupla. En la siguiente figura se representa la información que se podría almacenar en una base de

datos relacional que contiene información sobre los departamentos de que consta una empresa y los empleados que trabajan en ella.

Departamento			Empleado		
NumDep	NomDep	LocDep	NomEmp	SalEmp	NumDep
1	Recursos humanos	Bilbao	Luis Sánchez	2000 €	1
2	Central	Madrid	María Sol	2300 €	1
			Lucía Rodríguez	3200 €	2

Figura 4: Esquema de una base de datos relacional con datos.

El modelo relacional es el modelo más empleado en la actualidad, por lo que es el que se va a estudiar en detalle en el presente curso. Algunos SGBD relaciones comerciales son Oracle, SQL Server, MySQL, PostgreSQL, etc.

1.7.3. Modelos de datos físicos.

La última fase del diseño de una base de datos es el diseño físico, que consiste en crear en un sistema gestor de bases de datos (SGBD) concreto todos los elementos de que consta la base de datos. Si se trata de una base de datos relacional, como la mayoría de las que se emplean hoy en día, implicaría crear tablas, índices, vistas, disparadores, etc.

La creación de todos estos elementos se puede llevar a cabo de dos maneras: empleando asistentes con facilidades gráficas o mediante el empleo del lenguaje de definición de datos (DDL) que proporcione el SGBD que se esté utilizando. Por esto, en función del SGBD que se emplee, esta tarea se llevará a cabo de distinta forma.

Una base de datos relacional consta siempre de una o varias tablas, que son los elementos más importantes de que consta una base de datos. Para crear estas tablas mediante el DDL SQL se emplea la instrucción CREATE TABLE. Para cada tabla será necesario indicar cada uno de los atributos de que consta. A su vez, por cada uno de estos será necesario especificar:

- El nombre del atributo.
- El tipo de dato del atributo (numérico entero, numérico real, cadena de caracteres de longitud fija, cadena de caracteres de longitud variable, fecha, etc.).
- Las restricciones asociadas al atributo, si es el caso (clave primaria, clave ajena, valor único, restricciones de usuario, etc.).

1.8. Independencia de los datos.

Una de las ventajas de las bases de datos en relación con los sistemas tradicionales de ficheros es que proporciona independencia de los datos. Esta independencia se puede definir a dos niveles:

- Independencia física: se refiere a la posibilidad de modificar el esquema físico de la base de datos (ubicación de la base de datos, índices, etc.) sin que se tengan que volver a escribir los programas de aplicación que actúan sobre la base de datos. Las modificaciones en el nivel físico son necesarias a veces con el fin de mejorar el rendimiento del sistema.
- Independencia lógica: se refiere a la posibilidad de modificar el esquema conceptual de la base de datos sin necesidad de reescribir los programas de aplicación. A veces es necesario realizar modificaciones en el esquema conceptual, como añadir o eliminar atributos a una entidad o crear nuevas entidades. Esta independencia es más difícil de conseguir que la independencia física.

1.9. Lenguaje de definición de datos.

El SGBD debe proporcionar los medios necesarios para que el administrador de la base de datos pueda especificar los elementos de datos que la integran (las tablas), su estructura (atributos de las tablas) y las relaciones que existen entre ellos, las reglas de integridad y de confidencialidad, así como las características de tipo físico y las vistas de los usuarios. Esta función, que se lleva a cabo mediante el empleo de un lenguaje de definición de datos (DDL: *Data Definition Language*) debe suministrar, por tanto, los medios necesarios para definir las estructuras física, lógica global y externas, correspondientes a cada uno de los niveles de la arquitectura ANSI/X3/SPARC. Cuando se emplea un DDL para definir los elementos que integran la base de datos, se deben definir objetos como tablas, vistas, índices, disparadores, procedimientos, funciones, etc. Los DDL dan la posibilidad de crear estos elementos (mediante sentencias del tipo CREATE), modificar su definición (mediante sentencias del tipo ALTER) y eliminarlos (mediante sentencias DROP).

1.10. Lenguaje de manipulación de datos.

El SGBD debe proporcionar los medios necesarios para permitir a los usuarios consultar y actualizar los datos almacenados en la base de datos. La actualización de una base de datos puede implicar tres tipos de operaciones:

- Inserción o adición de nuevos datos, por ejemplo, añadir los datos de un nuevo artículo que vende la empresa.
- Borrado o eliminación, por ejemplo, eliminar los datos de un artículo que la empresa ha dejado de vender.
- Modificación, por ejemplo, el cambio del precio de un determinado artículo.

Esta función de manipulación se llevará a cabo por medio de un lenguaje de manipulación de datos (DML: *Data Manipulation Language*). Estos lenguajes deben permitir, por tanto, realizar operaciones de consulta (SELECT), inserción (INSERT), borrado (DELETE) y modificación (UPDATE).

Según la posibilidad de emplear el DML de manera independiente o no, podemos hablar de lenguajes huésped, autocontenidos o duales. Los DML huésped son aquellos cuyas instrucciones de manipulación de datos deben embeberse en otro lenguaje de programación (lenguaje anfitrión). Los DML autocontenidos, por su parte, son lenguajes autosuficientes que pueden ser empleados por usuarios con pocos conocimientos de programación para, desde un terminal y de un modo interactivo, acceder a la base y manipular los datos almacenados en ella sin necesidad de apoyarse en un lenguaje de programación. Los lenguajes, como el SQL, que pueden operar como huésped o como autocontenido, reciben el nombre de lenguajes duales.

1.11. El administrador de la base de datos (DBA). Funciones.

El administrador de la base de datos (DBA) es el máximo responsable del correcto funcionamiento de la base de datos y tendrá encomendado el diseño de la base de datos, asegurar la confidencialidad de los datos y las siguientes funciones:

- Definir procedimientos de respaldo y recuperación: en caso de que alguna porción de la base de datos sufra algún daño (por un fallo humano, un fallo en el equipo, etc.), es necesario que sea posible reparar los datos implicados con un mínimo de retraso y afectando lo menos posible al resto del sistema. Así, por ejemplo, la disponibilidad de los

datos no dañados no debería verse afectada. Pues bien, el DBA en estos casos debe poner en marcha planes de recuperación adecuados, para lo cual una de las tareas que deberá llevar a cabo periódicamente es la realización de copias de seguridad con el fin de poder restaurarlas en estos casos.

- Supervisar el desempeño y responder a cambios en los requerimientos: el DBA es responsable de organizar el sistema de modo que sea lo más eficiente posible. Para ello, deberá monitorizar el sistema con el fin de comprobar y realizar los ajustes adecuados cuando el rendimiento no llegue al deseado y/o cuando haya modificaciones en los requisitos de los usuarios.

1.12. Estructura general de la base de datos. Componentes funcionales.

Los componentes principales de un SGBD son los que se exponen a continuación:

- Herramientas de gestión: como indican Oltra et al. (2006), todos los SGBD disponen de herramientas de gestión para poder crear las bases de datos, manipularlas, modificar su diseño, crear usuarios y asignar permisos, etc.
- Herramientas de programación: estas herramientas posibilitan la creación de programas que accedan a los datos y los manipulen para su uso por parte de usuarios que no puedan o deban trabajar directamente con el SGBD.
- Lenguajes: los SGBD proporcionan lenguajes que se pueden clasificar del siguiente modo:
 - Lenguajes de definición de datos, ya explicados en la sección 1.9.
 - Lenguajes de manipulación de datos, ya explicados en la sección 1.10.
 - Lenguajes de control de datos (DCL): estos deben permitir al administrador de la base de datos realizar tareas como crear, eliminar y modificar usuarios, conceder y retirar permisos sobre los distintos objetos de la base de datos, realizar y restaurar copias de seguridad, etc.
- El diccionario de datos: contiene toda la información sobre los datos almacenados en la base de datos. Así, contendrá las definiciones de todos los objetos de la base de datos (tablas, vistas, índices, disparadores, procedimientos, funciones, etc.), información acerca de restricciones de integridad, información sobre privilegios y roles de los diferentes usuarios de la base de datos, información sobre los accesos a los objetos, etc.

1.13. Arquitectura de sistemas de bases de datos.

Según la ubicación de los componentes de un SGBD, podemos hablar en general de dos tipos de sistemas: los centralizados y los distribuidos.

En los sistemas centralizados todo el SGBD se encuentra ubicado en una sola máquina.

En los sistemas distribuidos el SGBD se divide en diversas partes, cada una de las cuales puede estar ubicada en un ordenador diferente. Este tipo de sistemas presenta diferentes variantes, de las cuales las más usuales son la arquitectura cliente / servidor y las bases de datos distribuidas.

1.13.1. Arquitectura cliente / servidor.

Los sistemas de bases de datos con arquitectura cliente / servidor presentan el SGBD dividido en dos partes:

- El servidor, que es la parte principal del SGBD, la que gestiona la base de datos. El servidor, que normalmente contiene los datos de la base de datos, se encarga de aceptar las peticiones de los clientes, procesarlas y devolver los resultados.
- El cliente, que es la parte que utilizan los usuarios y las aplicaciones.

Normalmente hay una máquina servidora y varios clientes conectados en red, donde los clientes solicitan servicios al servidor. El software de gestión de datos, es decir, el que lleva a cabo la manipulación de los datos, reside normalmente en el servidor. Por otro lado, el software de interacción con el usuario y el de desarrollo suelen encontrarse en los clientes.

1.13.2. Bases de datos distribuidas.

Una base de datos distribuida es aquella en la que los datos están repartidos entre diferentes máquinas. Se trata en realidad de un conjunto de máquinas o nodos conectados entre sí mediante una red, en el que cada nodo es un sistema de base de datos en sí mismo (con una UCP, un SGBD y una serie de terminales), pero los diferentes nodos han convenido en trabajar conjuntamente con el fin de que un usuario de cualquier máquina pueda tener acceso a los datos de cualquier otra como si todos los datos estuvieran almacenados en el propio nodo del usuario.

Puede ocurrir que algunos datos estén repetidos en diferentes máquinas, en cuyo caso se dice que la información está replicada. Incluso podría darse el caso de que toda la base de datos estuviese replicada.

El principio fundamental de las bases de datos distribuidas es que desde el punto de vista del usuario, un sistema distribuido debe ser idéntico a un sistema no distribuido o centralizado, es decir, los usuarios deben poder comportarse igual que si el sistema no fuese distribuido. Todos los problemas de los sistemas distribuidos (ubicación de la información, réplicas, etc.) deben ser transparentes al usuario. Esto quiere decir que, por ejemplo, la consulta de datos no locales (ubicados en otro nodo) se debe poder llevar a cabo igual que si los datos fuesen locales.

BIBLIOGRAFÍA

Oltra, F.; Albert, J.; Vericat, A. *Operaciones con bases de datos ofimáticas y corporativas*. McGraw-Hill, Madrid, 2006.

Piattini, M.; Calvo-Manzano, J.; Cervera, J.; Fernández, L. *Análisis y diseño detallado de aplicaciones informáticas de gestión*, Ra-ma, Madrid, 2007.

Silberschatz, A.; Korth, H. F.; Sudarshan, S. *Fundamentos de bases de datos*, McGraw-Hill, Madrid, 2014.