Temas Generales para la preparación de la Oposición al Cuerpo Superior de Estadísticos del Estado.

Cuerpo Superior de Estadísticos del Estado
Especialidad de Estadística-Ciencia de Datos.

Almacenamiento y modelos de datos

Tema 1. El entorno en una base datos.

AUTOR: **Antonio J. Sánchez-Padial**

Asociación Profesional de Cuerpos Superiores de Sistemas y Tecnologías de la Información de las Administraciones Públicas

Creación: Junio 2021

ÍNDICE

1 Introducción

Un objetivo principal de un sistema de bases de datos es proporcionar a los usuarios una abstracción de los datos, ocultando detalles sobre su almacenamiento y manipulación. Así, el punto de partida del diseño de una base de datos será una descripción abstracta y general de los requisitos de información de la organización que deben ser representados en la base de datos.

Además, la base de datos será un recurso compartido, cada usuario puede requerir una vista distinta de los datos almacenados en la base de datos. Para satisfacer estas necesidades, la arquitectura de la mayoría de los Sistemas de Gestión de Bases de Datos (SGBD) comerciales disponibles hoy día se basan en la arquitectura conocida como arquitectura ANSI-SPARC. En ese tema se presentan varias características arquitectónicas y funcionales de los SGBD.

2 La arquitectura ANSI-SPARC de tres niveles

La arquitectura ANSI-SPARC fue propuesta en 1975 por el Standard Planning and Requirements Committee (SPARC) del American National Standards Institute (ANSI). Esta propuesta propone tres niveles de abstracción en los que podemos describir los datos de una base de datos. Estos niveles son: el nivel **externo**, el **conceptual**, y el **interno**. El nivel **externo** define la forma en que los datos son percibidos por los usuarios. Por su parte, el nivel **interno** define la manera en que el SGBD y el sistema operativo perciben y manejan los datos, usando ficheros y diversas estructuras de datos. El **nivel conceptual** proporciona el enlace y la independencia necesaria entre los niveles internos y externos.

El objetivo de esta arquitectura es separar la vista de cada usuario de la base de datos de la manera en que la base de datos funciona físicamente. Hay diversas razones que hacen deseable esta separación:

- Cada usuario debe acceder a los mismos datos, pero puede necesitar una vista particular de los mismos. Cada usuario debería ser capaz de modificar la manera en que él o ella ven los datos, y estos cambios no deberían afectar al resto de usuarios.
- Los usuarios no deberían acceder directamente con los detalles del almacenamiento físico de los datos, tales como los índices. En otras palabras, la interacción de un usuario con la base de datos debería ser independiente de los detalles de implementación del almacenamiento de esta.
- El administrador de la base de datos (DBA) debería ser capaz de modificar las estructuras de almacenamiento sin afectar las vistas de los usuarios.
- La estructura interna de la base de datos no debería ser afectada por los cambios en aspectos físicos del almacenamiento, tales como un traslado a un nuevo dispositivo de almacenamiento.
- Un administrador debería ser capaz de modificar la estructura conceptual de la base de datos sin que esto afectara a todos los usuarios.

o Nivel externo

En el nivel externo encontramos múltiples vistas de la base de datos adaptadas a las necesidades de cada uno de los usuarios. Estas vistas externas incluyen únicamente aquellos elementos de la base de datos: entidades, atributos, y relaciones que están dentro del ámbito de interés del usuario en el "mundo real". Aunque la base de datos puede incluir otros conceptos fuera de su ámbito de tu interés, el usuario no será consciente de ello desde su vista de la misma.

Además de incluir distintos contenidos, distintas vistas pueden ofrecer distintas representaciones de la misma información almacenada en la base de datos. Así, mientras un usuario puede ver a los clientes en formato (*apellidos, nombre*), otro usuario puede verlos en formato (*nombre apellidos*). Algunas vistas pueden incluir también información derivada o calculada. Tal vez nuestro sistema necesite conocer la edad de los clientes. Almacenar la edad en la base de datos suele ser una mala idea, ya que necesitaría la actualización constante de datos conforme llega la fecha del cumpleaños de cada uno. Lo usual es que almacenemos un dato invariable, como la fecha de nacimiento. Las vistas pueden mostrar la edad del cliente a cada usuario del SGBD cuando sea necesario mostrar este valor.

Las vistas pueden incluso mostrar información combinada a partir de los datos almacenados en distintas entidades de la base de datos.

o Nivel conceptual

El nivel conceptual es el nivel intermedio en la arquitectura ANSI-SPARC de tres niveles. Este nivel contiene toda la estructura lógica de la base de datos al completo. Aunque los usuarios corrientes no pueden acceder a este nivel, los administradores de la base de datos (DBA) sí que trabajan frecuentemente en este nivel. El nivel conceptual incluye una visión completa de todas las necesidades de información de la organización, separadas de los requisitos tecnológicos de su almacenamiento. En este nivel se representan:

- todas las entidades, sus atributos, y sus relaciones,
- todas las restricciones sobre los datos,
- información de carácter semántico sobre los datos,
- información relativa a la seguridad y la integridad de los datos.

Todas las vistas del nivel externo se basan el la información del nivel conceptual, bien porque se encuentre en él de modo explícito, bien porque pueda ser calculada o derivada a partir de la información que hay en él. La descripción que se hace de esta información en este nivel excluye los detalles sobre el almacenamiento. Por ejemplo, en el nivel conceptual encontraremos información sobre si un valor es real o entero, pero no sobre el número de bytes que ocupará en el disco duro.

o Nivel interno

El nivel interno es la representación lógica de la base de datos en el ordenador, teniendo en cuenta cualquier consideración sobre la implementación real para optimizar su funcionamiento y el espacio de almacenamiento utilizado. Incluye estructuras de datos en memoria, y también ficheros en dispositivos de almacenamiento. Debe comunicarse directamente con el sistema operativo para guardar datos en los dispositivos, crear índices, recuperar datos, etc. Algunas de las tareas que realiza son:

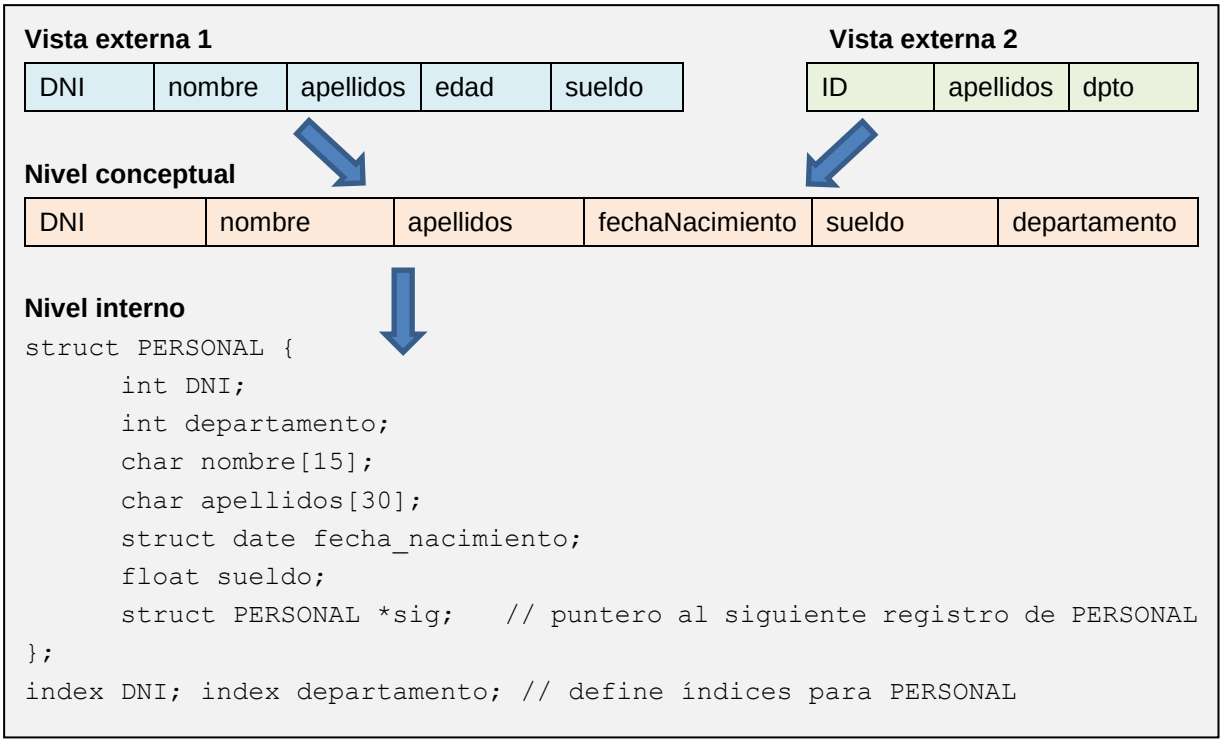
- reserva de espacio de almacenamiento para los datos y sus índices,
- registro de descripciones para el almacenamiento (incluyendo los tamaños de los elementos almacenados),
- registro de la ubicación de los elementos,
- compresión de datos y uso de técnicas de encriptación y seguridad

Por debajo del nivel interno hay un **nivel físico** que es gestionado por el sistema operativo bajo las instrucciones del SGBD. Sin embargo, la frontera entre las responsabilidades del SGBD y del sistema operativo en el nivel físico no están claramente delimitadas. Algunos SGBDs utilizan muchos de los métodos de acceso del sistema operativo, mientras que otros solo usan los métodos más básicos y crean los suyos propios para la organización de los archivos. El nivel físico por debajo del SGBD está compuesto por elementos que solo conoce el sistema operativo, tales como si los campos de los registros se almacenan en bytes continguos en el disco duro o no, y otros detalles como estos.

o Esquemas, mapeos e instancias

La descripción general de la base de datos se llama **esquema** de la base de datos. Hay tres tipos diferentes de esquema en la base de datos que se definen conforme a los niveles de abstracción de la arquitectura de tres niveles que estamos discutiendo. En el nivel superior, tendremos múltiples **esquemas externos** (también llamados **subesquemas**) que corresponden con las vistas de la base de datos. En el nivel conceptual, tendremos el **esquema conceptual**, que es una descripción exhaustiva de todas las entidades, atributos, y relaciones junto con las restricciones de integridad. En el nivel más bajo de abstracción, tendremos el **esquema interno**, que contiene una descripción precisa del modelo interno, con la definición de los registros almacenados, los métodos de representación, los campos de datos, los índices y estructuras de almacenamiento utilizadas. Cada base de datos tiene un único esquema conceptual y un único esquema interno.

El SGBD se encarga de las relaciones entre estos tres tipos de esquemas. Se encarga también de asegurar la consistencia entre ellos; comprobando que cada esquema externo se deriva del esquema conceptual, y mapea cada esquema externo con el esquema interno utilizando la información del esquema conceptual. La correspondencia entre el esquema conceptual y el esquema interno se realiza a través del **mapeo interno/conceptual**. Mediante este mapeo el SGBD es capaz de encontrar el registro o combinación de registros adecuados en el soporte físico que corresponden con un **registro lógico** en el esquema conceptual, junto con cualquier restricción de integridad que deba asegurarse sobre este registro lógico. También permite resolver cualquier diferencia en los nombres de las entidades, los nombres de los atributos o su orden, los tipos de datos, etc. que hubiera que resolver. De igual modo, la correspondencia entre cada esquema externo y el esquema conceptual se realiza a través del **mapeo externo/conceptual**. Mediante este mapeo el SGBD puede encontrar la parte relevante del esquema conceptual para cada nombre o elemento del esquema externo.



El recuadro superior muestra un ejemplo de los diferentes niveles. Hay dos vistas externas distintas para el personal: un incluye el documento de identidad (DNI), el nombre (nombre), los apellidos (apellidos) la edad y el sueldo; la segunda contiene un identificador (ID), los apellidos (apellidos) y el departamento en el que trabaja. Estas dos vistas están basadas en la información que existe en el nivel conceptual. Es destacable que no existe un atributo en el nivel conceptual para la edad, si no que existe una fechaNacimiento. EL SGBD se encargará de realizar las transformaciones oportunas cuando se requiera este valor, o de mantener la correspondencia entre el atributo ID de la vista externa 2 y el atributo DNI del nivel conceptual. Estas correspondencias se mantienen en el mapeo externo/conceptual. De igual modo el nivel conceptual está mapeado en el nivel interno. En este caso, el nivel interno tiene el aspecto de un registro en un lenguaje de programación de alto nivel. La estructura contiene un puntero, sig, a través del cual todos los registros de PERSONAL se mantienen en una lista enlazada. Obsérvese que además de las diferencias de nombres entre los niveles, también hay diferencias en el orden que se definen los atributos.

Es importante distinguir entre la descripción de la base de datos y la base de datos en sí. La descripción de la base de datos es el **esquema** de la base de datos. El esquema es creado durante el proceso de diseño y no suele cambiar a menudo. Sin embargo, los datos en la base de datos sí suelen cambiar frecuentemente, por ejemplo, cada vez que ingresamos dinero en nuestra cuenta bancaria o pagamos con una tarjeta asociada de crédito. Llamamos **instancia** de la base de datos a los datos que contiene una base de datos en cualquier momento determinado. Diferentes instancias de la base de datos pueden estar asociadas a un mismo esquema.

○ Independencia de los datos

Uno de los objetivos principales de esta arquitectura de tres niveles es proporcionar la **independencia de los datos**, que significa que los niveles superiores no se verán afectados por los cambios producidos en los niveles inferiores. Existen dos tipos de independencia: **lógica** y **física**.

La independencia lógica de los datos es la inmunidad de los esquemas externos a los cambios producidos en el esquema conceptual. Estos cambios, como la creación o supresión de nuevas entidades o atributos, debería poder realizarse sin tener que modificar los esquemas externos, incluso sin tener que modificar las aplicaciones que los utilizan. Naturalmente, si estos cambios han sido producidos para un usuario, este deberá ser consciente de ellos y sus vistas tendrán los cambios oportunos, pero lo importante es que este cambio no debería afectar al resto de usuarios.

La independencia física de los datos es la inmunidad del esquema conceptual a los cambios realizados en el esquema interno. De manera análoga, estos cambios, como el uso de una organización diferente de los ficheros o diferentes dispositivos de almacenamiento, modificaciones de los índices o los algoritmos de *hash*, deberían poder realizarse sin que cambien los esquemas externos ni el esquema conceptual. A menudo, estos cambios se realizan para mejorar el rendimiento de la base de datos.

3 Lenguajes de bases de datos

Un lenguaje de datos tiene dos partes: un Lenguaje de Definición de Datos (LDD, o *DDL, Data Definition Language*) y un Lenguaje de Manipulación de Datos (LMD, o *DML*,*Data Management Language*). Usamos el LDD para definir el esquema de la base de datos , y usamos el LMD tanto para leer el contenido de la base de datos como para modificarlo. Estos lenguajes también son llamados *sublenguajes de datos* porque no tienen todos los elementos de los lenguajes tradicionales, tales como sentencias condicionales o bucles, que sí proporcionan los lenguajes de programación de alto nivel. Muchos SGBD ofrecen algún mecanismo para utilizar su sublenguaje de datos en un lenguaje de programación de alto nivel como COBOL, Fortran, Pascal, Ada, C, C++, C#, Java o Visual Basic. Para compilar el fichero incrustado, se sustituyen los comandos del sublenguaje de datos por llamadas a funciones. Realizado este preprocesamiento del fichero, ya puede ser compilado, y linkado con un librería específica del SGBD que contiene las funciones reemplazadas, con lo que ya puede ser ejecutado cuando sea requerido. La mayoría de los sublenguajes pueden además ser ejecutados en un modo interactivo desde un terminal.

o El Lenguaje de Definición de Datos (LDD)

El LDD es el lenguaje en el que se define el esquema de la base de datos. Por tanto, es utilizado por el administrador de esta (DBA) para definir la entidades, atributos y sus relaciones, así como las restricciones de integridad y de seguridad que sean necesarias para el sistemas de información. Puede ser creado tanto para crear elementos, como para modificar o eliminar los existentes. No es posible manipular los datos usando el LDD. El resultado de la compilación de las sentencias LDD es un conjunto de tablas que se almacenará, de algún modo, en un conjunto de ficheros. Llamamos **catálogo del sistema** a este conjunto. El catálogo del sistema está compuesto por **metadatos**, que son datos que describen los elementos de la base de datos, de forma que facilita el acceso y la manipulación de estos elementos. Los metadatos contienen la definición de todos los elementos que puedan ser de interés para el usuario, así como otros que pueden ser necesarios para el correcto funcionamiento del SGBD. Normalmente, el SGBD consultará este catálogo antes de acceder realmente a la información de la base de datos. Los términos **diccionario de datos** y **directorio de datos** se utilizan como sinónimos de este catálogo, si bien el término diccionario de datos se refiere más frecuentemente a un componente software del SGBD.

Teóricamente, podríamos tener distintos LDD para uno de los esquemas descritos en la arquitectura de tres niveles: es decir, un LDD para los esquemas externos, un LDD para el esquema conceptual, y un LDD para el esquema interno. Sin embargo, esto no suele ser así en la práctica, y normalmente hay un único LDD que sirve para realizar especificar, al menos, los esquemas externos y el esquema conceptual.

o El Lenguaje de Manipulación de Datos (LMD)

El LMD es el lenguaje que permite realizar las operaciones básicas de manipulación sobre los datos de la base de datos. Algunas de estas operaciones son las siguientes:

- inserción de nuevos datos,
- modificación de los datos almacenados,
- recuperación de datos desde la base de datos,
- eliminación de datos de la base de datos.

Una de las responsabilidades de un SGBD es dar soporte a la ejecución de sentencias en LMD de los usuarios, de forma que tenga lugar la manipulación sobre los datos que estos pretenden. Estas manipulaciones van a tener repercusiones en los niveles externos, conceptual e interno. En el nivel interno las manipulaciones se realizan a través de la ejecución de funciones complejas de bajo nivel diseñadas para un acceso eficiente a los datos. Sin embargo, en los niveles superiores, el objetivo principal se centra en la facilidad de uso.

Llamamos **lenguaje de consulta** a la parte del LMD centrada en la recuperación de información. Un lenguaje de búsqueda puede definirse como un lenguaje especializado de alto nivel. El término “consulta” (*query* en inglés) se suele reservar oara indicar un comando expresado en el lenguaje de

consulta. Normalmente, los término “lenguaje de consulta” y “LMD” se usan de manera indistinta, aunque existen diferencias técnicas entre ambos conceptos.

Clasificamos los LMD según las técnicas de recuperación de la información que utilizan. Así, hablamos de dos tipos de LMD: **procedurales** y **no procedurales**. La principal diferencia es que los lenguajes procedurales describen *cómo* debe obtenerse la respuesta a la consulta a partir de la información en la base de datos, mientras que en los LMDs no procedurales se describe únicamente *qué* salida se espera conseguir. Los LMDs procedurales suelen tratar los registros de la base de datos de manera individual, mientras que los no procedurales operan sobre conjuntos de registros.

▪ LMDs procedurales

Con un LMD procedural, un usuario, o más frecuentemente, un programador, especifica tanto los datos que quiere conseguir como la manera de hacerlo. Esto significa que el usuario debe expresar todas las operaciones de acceso a datos que sean necesarias. Normalmente, un LMD procedural recupera un registro, lo procesa, en función del resultado obtenido en este proceso, busca un nuevo registro que será procesado de manera similar, y así sucesivamente. Este proceso continúa hasta que todos los datos solicitados en la consulta han sido obtenidos. Los LMD procedurales suelen estar acoplados a un lenguaje de programación de alto nivel que posee las abstracciones para facilitar la iteración y manejar la lógica del sistema. Los LMDs de los sistemas de bases de datos en red y jerárquicos son, normalmente, procedurales.

▪ LMDs no procedurales

Los LMD no procedurales permitan expresar los datos solicitados con único comando de consulta o actualización de datos. En estos lenguajes, el usuario especifica que datos requiere sin describir los pasos que hay que seguir para encontrarlos. El SGBD traduce las sentencias del LMD en uno o varios procedimientos que manipulan los conjuntos registros necesarios, de forma que el usuario no tiene porque conocer la implementación interna de las estructuras de datos ni que algoritmos son necesarios para encontrar y tal vez transformar los datos. Esto proporciona una gran independencia a los usuarios. Los lenguajes no procedurales también son denominados *lenguajes declarativos*, Los SGBDs relacionales suelen contar con algún LMD no procedural para la manipulación de datos, tales como SQL o QBE (Query-By-Example). Normalmente los LMDs no procedurales son más fáciles de aprender y de usar que los LMDs procedurales, ya que el SGBD se encarga de parte del trabajo en lugar del usuario.

○ Lenguajes de 4ª generación (4GL)

Aunque no existe una definición bien consensuada sobre qué son los **lenguajes de 4ª generación**, podemos entender que esta expresión se refiere a lenguajes de propósito específico. Una operación que puede necesitar unos cientos de líneas de código en un lenguaje de tercera generación (3GL), como C++, normalmente necesitará solo unas pocas en un lenguaje 4GL.

Mientras que los lenguajes 3GL son procedurales, los 4GL son no procedurales: el usuario indica *qué* necesita obtener, y no *cómo* obtenerlo*.* El lenguaje de 4GL depende para su funcionamiento de una serie de componentes de alto nivel, las herramientas de cuarta generación. El usuario no tiene que definir los pasos a seguir para realizar una tarea, si no que indica ciertos parámetros a las herramientas que utiliza para obtenerla respuesta. Se estima que el uso de los lenguajes de cuarta generación puede mejorar la productividad en un orden de magnitud (10x), pagando el coste de limitar el espectro de problemas o tareas que pueden ser tratados con la herramienta. Los ámbitos típicos de los lenguajes 4GL son:

- lenguajes de presentación, como los lenguajes de consulta o los generadores de informes,
- lenguajes especializados, como las hojas de cálculo o los lenguajes de bases de datos,
- generadores de aplicaciones, que definen datos, inserciones, modificaciones y borrado de datos de una base de datos para construir aplicaciones,
- lenguajes de muy alto nivel como los generadores de código.

Ya hemos visto SQL y QBE como lenguajes 4GL. A continuación, se describen brevemente los cuatros ámbitos expuestos.

- ## Generadores de formularios

Un generador de formularios es un programa interactivo diseñado para crear fácilmente entradas de datos y formularios electrónicos. El generador de formularios permite al usuario definir el aspecto del formulario en la pantalla, qué información mostrará, o en qué parte de la pantalla se mostrará. Puede también permitir escoger el color de los diversos elementos y otras características visuales, como escritura en negrita, subrayada, parpadeante, etc. Los mejores generadores de formularios permiten la creación de atributos derivados, bien mediante operadores aritméticos o mediante agregados, así como especificar chequeos y comprobaciones que debe cumplir los datos que provienen del formulario.

- ## Generadores de informes

Un **generador de informes** es una aplicación que permite crear informes a partir de los datos almacenados en una base de datos. Se parece a un lenguaje de consulta en que permite al usuario hacer preguntas a la base de datos y recuperar información desde esta para su informe. Sin embargo, los generadores de informes proporcionan mucho más control sobre el aspecto de la salida. Podemos dejar que el generador de informes decida automáticamente el aspecto del informe, o podemos crear nuestros propios informes personalizados usando las herramientas y comandos del generador.

Hay dos tipos principales de generadores de informes: basados en lenguaje o visuales. En el primer caso, se introduce un comando en un sublenguaje para definir qué datos deben ser mostrados en el informe y la disposición de estos datos en dicho informe. En el segundo caso, es posible definir esas mismas características del informe mediante una aplicación similar a un generador de formularios.

- ## Generadores de gráficas

Un **generador de gráficas** es una aplicación que recupera datos de una base de datos y los representa en una gráfica, mostrando tendencias y relaciones entre los datos. Normalmente, permita a los usuarios crear gráficos de barras, de tarta, diagramas de puntos y de líneas, y otros.

- ## Generadores de aplicaciones

Un **generador de aplicaciones** es una aplicación para crear programas que sirvan de interfaz con una base de datos. El uso de un generador de aplicaciones puede reducir el tiempo necesario para diseñar y construir una aplicación software completa. Los generadores de aplicaciones suelen consistir en una serie de módulos predefinidos que proporcionan las principales funciones que usan la mayoría de los programas. Estos módulos, normalmente escritos en un lenguaje de alto nivel, forman una librería de funciones a modo de catálogo del que escoger. El usuario especifica *qué* debe hacer su programa; y el generador de aplicaciones decidirá *cómo* deben ser realizadas esas tareas.

4 Modelos de datos y modelado conceptual

Hemos tratado como se utiliza el Lenguaje de Definición de Datos (LDD) para escribir el esquema de una base de datos. De hecho, se escribe en el LDD de cada SGBD en particular. Sin embargo, este lenguaje LDD es un lenguaje de muy bajo nivel como para que los usuarios comunes de una organización puedan entenderlo. Por eso necesitamos una descripción de alto nivel del esquema, a la que llamamos **modelo de datos**.

Un modelo es una representación de los objetos y sucesos del mundo real, y sus vínculos. En él, se recogen los aspectos esenciales de una organización, ignorando las características accidentales. El modelo de datos representa a la organización en sí, utilizando una serie de abstracciones para describir y manipular los datos, las relaciones entre los datos, y las restricciones que estos datos deben cumplir. El modelo se construye a partir de un conjunto de conceptos básicos y notaciones que permiten a los diseñadores de bases de datos comunicarse sin ambigüedades con los usuarios del sistema, de modo que recoge con precisión su conocimiento sobre la organización. Podemos distinguir tres elementos en un modelo de datos:

1. **Una parte estructural**, que consiste en un conjunto de reglas para la construcción de bases de datos
2. **Una parte manipulativa**, que define los tipos de operaciones que se permitirán sobre los datos (incluyendo las operaciones necesarias para la modificación y recuperación de estos datos, y para la modificación de la estructura de la base de datos)
3. **Un conjunto de restricciones de integridad**, que aseguran que los datos son correctos

El objetivo de un modelo de datos es representar los datos y hacerlos comprensibles. Podemos distinguir tres tipos de modelos de datos, siguiendo la arquitectura ANSI-SPARC de tres niveles:

1. Un modelo de los datos externos, que representa la vista que tiene de la organización cada usuario. A veces se denomina **Universo del Discurso**.
2. Un modelo de los datos conceptuales, que representa la vista lógica (o general) y que es independiente del SGBD
3. Un modelo de los datos internos, que representa el modelo conceptual de modo que es comprensible para el SGBD en cuestión.

o Modelos de datos basados en objetos

Los modelos de datos basados en objetos utilizan conceptos tales como entidades, atributos y relaciones. Una **entidad** es un objeto distinguible (persona, lugar, cosa, concepto, evento) de a organización que se quiere representar en la base de datos. Un **atributo** es una propiedad que describe algún aspecto del objeto que se quiere registrar, y una **relación** es una asociación entre entidades. Algunos de los tipos más comunes de modelos de datos basados en objetos son:

- Entidad – Relación (ER)
- Sémanticos
- Funcionales
- Orientados a objetos

El modelo Entidad – Relación es una de las principales técnicas de diseño de bases de datos. Los modelos de datos orientados a objetos extienden el concepto de entidad añadiendo las acciones que están asociadad con el objeto, su **comportamiento**, además de los atributos que describen su **estado**. Se dice que el objeto encapsula estado y comportamiento.

o Modelos de datos basados en registros

En un modelo basado en registros, la database está formada por una cantidad de registros de formato fijo, normalmente de distintos tipos. Cada tipo de registro tiene un número fijo de campos, normalmente de tamaño fijo. Hay tres tipos principales de modelos lógicos de datos basados en registros el **modelo**

de datos relacional, el **modelo de datos en red**, y el **modelo de datos jerárquico.** Los modelos de datos en red y jerárquico fueron desarrollados casi una década antes que el modelo relacional, por eso sus vínculos con el modelo tradicional de procesado de ficheros son más obvios.

▪ Modelo de datos relacional

El modelo de datos relacional se basa en el concepto matemático de relación. Tanto los datos como sus vínculos se representan mediante tablas, cada una con un número fijo de columnas con un nombre único.

Presidentes

Nombre	LugarNacimiento	AñoNacimiento	Siglas
Adolfo Suárez	Cebreros	1932	UCD
Leopoldo Calvo Sotelo	Madrid	1926	UCD
Felipe González	Sevilla	1942	PSOE
José María Aznar	Madrid	1953	PP
José Luis Rodríguez Zapatero	Valladolid	1960	PSOE
Mariano Rajoy	Santiago de Compostela	1955	PP
Pedro Sánchez	Madrid	1972	PSOE

PartidosPolíticos

Siglas	Nombre	Fecha Creación
UCD	Unión de Centro Democrático	1977
PSOE	Partido Socialista Obrero Español	1879
PP	Partido Popular	1989

En las tablas de ejemplo, tenemos parte del esquema relacional de una base de datos sobre la historia reciente de España. Se puede ver que el presidente Suárez nació en Cebreros en 1932; y también que militaba en UCD que fue fundada en 1977, como se ve en la segunda tabla. Se aprecia que hay una relación entre la tabla de Presidentes y la de PartidosPolíticos: cada presidente *milita* en un partido. Sin embargo, no hay ningún enlace explícito entre las dos tablas. Simplemente, sabemos que los valores del atributo Siglas en la tabla Presidentes, corresponden a los valores del atributo Siglas en la tabla PartidosPolíticos.

Puesto que se trata de un modelo lógico, solo requiere que la base de datos sea vista por los usuarios con este aspecto de tablas, en los niveles superiores del estándar ANSI-SPARC de tres niveles: los niveles externos y conceptual. La estructura física de la base de datos se mantiene independiente, por lo que puede ser implementada según una gran variedad de estructuras de almacenamiento.

▪ Modelo de datos en red

En el modelo en red, los datos se representan mediante colecciones de registros, y sus relaciones se representan mediante conjuntos. Comparado con el modelo relacional, las relaciones se modelan mediante conjuntos, que serán punteros en la implementación física. Los registros se organizan en una estructura de grafo,donde los registros son los nodos (también llamados segmentos), y los conjuntos son los vértices del grafo.

▪ Modelo de datos jerárquico

El modelo jerárquico es un tipo particular o restringido del modelo en red. Como en este, los datos se representan como una colección de registros, y sus relaciones, como conjuntos. Sin embargo, en el modelo jerárquico cada nodo puede tener solo un padre. El modelo jerárquico puede ser representado mediante un árbol, donde los registros son los nodos, y los conjuntos los vértices.

EJEMPLO / DIBUJO

Los modelos (lógicos) de datos basados en registros se usan para especificar la estructura general de la base de datos y una descripción de alto nivel de su implementación. Su mayor inconveniente es que no proporcionan mecanismos adecuados para expresar restricciones en los datos, mientras que los modelos de datos orientados a objetos carecen del soporte de una estructura lógica, si bien son más ricos en el plano semántico al permitir expresar restricciones sobre los datos.

La mayoría de los sistemas comerciales modernos están basados en el paradigma relacional, mientras que los sistemas primitivos se basaban en los modelos en red y jerárquico. En estos modelos, el usuario tenía que conocer detalles sobre la implementación física del sistema, mientras que el modelo relacional ofrece bastante independencia. Además, el modelo relacional ofrece una estrategia declarativa al procesamiento de la base de datos (se específica *qué* datos se quieren), mientras que los sistemas en red y jerárquicos tienen una estrategia navegacional (es decir, hay que expresar *cómo* recorrer los datos para encontrar lo que se busca).

o Modelos físicos de datos

El modelo físico de datos describe como se almacenan los datos en el ordenador, y representa información como estructuras de registros, o rutas de acceso

o Modelado conceptual

Desde el punto de vista de la arquitectura de tres niveles, el modelo conceptual es el alma de la base de datos. Da soporte a las vistas externas, y a su vez es soportado por el esquema interno. Sin embargo, el esquema interno no es más que la implementación física de ese modelo conceptual. El modelo conceptual debería ser una representación **completa y precisa** de las necesidades de información de la organización. Si no fuera así, alguna información importante podría faltar o ser incorrecta, y podría ser difícil implementar una o más de las vistas externas.

El **modelado conceptual** es el proceso de construcción del modelo del uso de la información en una compañía de manera independiente a los detalles sobre su implementación, como el SGBD que se usará, las aplicaciones que se construirán, los lenguajes de programación, o cualquier otra consideración física. El modelo conceptual se distingue del modelo lógico en su independencia de la implementación, mientras que el modelo lógico se basa en el conocimiento directo del SGBD utilizado. Es habitual construir el modelo lógico a partir del modelo conceptual.

5 Funciones de un sistema de gestión de bases de datos

Se muestran a continuación una serie de funciones y servicios que se podrían esperar de un SGBD.

- **(1) Almacenamiento, recuperación y modificación de datos**

Se trata de la parte fundamental de la base de datos. Conforme al modelo de tres niveles de la arquitectura ANSI-SPARC, el SGBD debe ocultar los detalles de implementación a los usuarios tanto como sea posible.

- **(2) Catálogo accesible**

Una de las características clave de la arquitectura ANSI-SPARC es la existencia de un **catálogo del sistema** que integra datos sobre los diversos esquemas, usuarios, aplicaciones, etc. El cátalogo debería ser accesible tanto a los usuarios, como al propio SGBD. En definitiva, este catálogo, también llamado diccionario de datos, es un repositorio de información que almacena una descripción de los datos en la base de datos. Llamamos **metadatos**, a estos "datos sobre los datos". La cantidad de información y las diversas maneras en que se puede usar, varían según el SGBD. Normalmente, el catálogo almacena:

- Nombres, tipos, y tamaños de los datos;
- Nombres de las relaciones;
- Restricciones de integridad sobre los datos;
- Nombres de los usuarios autorizados a acceder a los datos;
- A qué datos puede acceder cada usuario y los tipos de accesos que se le autorizan; como lectura, escritura, modificación o borrado.
- Los esquemas externos, internos y conceptuales, así como los mapas de correspondencia entre ellos.
- Estadísticas de uso, tales como frecuencia de determinadas transacciones o conteo del número de accesos realizados a cada objeto de la base de datos.

El catálogo del sistema del SGBD es uno de los componentes fundamentales del sistema. La mayoría de los servicios y sistemas que se describen en esta sección hace uso de este para llevar a cabo sus tareas. Algunos de los beneficios de contar con un catálogo son los siguientes:

- La información sobre los datos se recoge y almacena de un modo centralizado. Esto ayuda a mantener el control sobre los datos, considerados como un recurso.
- Es posible definir la semántica de los datos, lo que ayudará a otros usuarios a comprender el propósito de estos datos.
- Se simplifica la comunicación entre usuarios, ya que se almacenan las descripciones exactas de los datos. El catálogo del sistema puede ayudar a identificar qué usuario o usuarios tienen permisos para acceder a determinados datos del sistema.
- Es más fácil identificar repeticiones e inconsistencias en los datos al estar centralizados.
- Es posible registrar los cambios en la base de datos
- Se puede determinar el impacto de un cambio antes de que este tenga lugar, ya que el catálogo almacena información sobre todos los datos, sus relaciones y sus usuarios.
- Refuerza la seguridad
- Permite asegurar la integridad de la base de datos
- Proporciona información para la auditoría del sistema de información

Algunos autores distinguen entre el catálogo del sistema y el directorio de datos, especificando que el directorio de datos contiene información sobre cómo y dónde se almacenan los datos. La ISO ha adoptado un estándar para los diccionarios de datos llamado IRDS (Information Resource Dictionary System). IRDS es una herramienta *software* que sirve para control y documentar las fuentes de información de una organización. Proporciona las definiciones de las tablas incluidas en el diccionario de datos y de las operaciones que pueden ser realizadas sobre estas tablas.

- **(3) Soporte a transacciones**

Una transacción es un grupo de acciones, ejecutado por un único usuario o aplicación, que accede o modifica los contenidos de una base de datos. Un ejemplo clásico, es una transferencia bancaria entres dos cuentas corrientes de un banco. Las operaciones involucradas serán comprobar que hay suficientes fondos en la cuenta de origen, restar la cantidad transferida de la cuenta origen, y sumar la cantidad transferida a la cuenta destino. Estas tres operaciones deben realizarse como un todo, si el proceso fuera interrumpido la base de datos puede quedar en un estado **inconsistente.** El SGBD debe proporcionar un mecanismo para que todos los cambios dentro de una transacción se realicen por completo o que, en caso contrario, no sea ejecutado ninguno de ellos. Esta funcionalidad debe ser capaz de revertir la base de datos a un estado consistente, en caso de fallo de alguna transacción.

- **(4) Servicio de control de concurrencia**

Uno de los principales objetivos al utilizar un SGBD es permitir el acceso de múltiples usuarios a los datos compartidos de forma concurrente. El acceso concurrente no es difícil de gestionar si los usuarios solo realizan operaciones de lectura, ya que no es posible que las acciones de unos alteren los resultados de las operaciones de otros. Sin embargo, cuando dos o más usuarios acceden a la base de datos de manera simultánea, y al menos uno de ellos hace cambios en los dos, pueden tener interferencias que resulten en inconsistencias en los resultados.

Tiempo	T_1	T_2	Cuenta
t_1		read(cuenta)	100
t_2	read(cuenta)	cuenta = cuenta + 100	100
t_3	cuenta = cuenta + 100	write(cuenta)	200
t_4	write(cuenta)		200
t_5			200

En la tabla vemos dos transacciones, T_1 y T_2 , que suceden simultáneamente en una base de datos bancaria. En ambas se realiza un ingreso de 100 euros. Debido a un control incorrecto de la concurrencia el estado final de la cuenta es de solo 200 euros, en lugar de los 300 que deberían figurar. Cuando empieza la transacción T_1, lee el estado de la cuenta, que aún no ha sido actualizada por T_2. Ambas transacciones se ejecutan sin tener en cuenta el resultado de la otra, y cuando T_1 escribe su resultado en la base de datos, sobreescribe el valor calculado por T_2, con la consiguiente pérdida de 100 euros.

Por tanto, el SGBD debe proporcionar un mecanismo que asegure que la base de datos se actualiza correctamente cuando varios usuarios modifiquen la base de datos concurrentemente.

- **(5) Servicios de recuperación**

Hemos visto como en caso de una transacción fallida, el SGBD debería devolver la base de datos a un estado consistente. El fallo puede haber sido provocado por diferentes causas como un error del sistema, el fallo de un dispositivo físico, un bug en el software que haga que el SGBD se detenga, o tal vez un usuario detecte un error en el proceso de la transacción y decida ponerle fin, abortándola. Para todos estos casos, el SGBD debe proporcionar ese mecanismo capaz de recuperar la base de datos en caso que pudiera haberse dañado de alguna manera.

- **(6) Permisos y autorización**

No es difícil imaginar un ejemplo en que pudiéramos desear que no toda la información de una base de datos estuviera disponible para todos los usuarios. Por ejemplo, puede que solo los jefes de departamento puedan ver información relacionada con los salarios de su equipo, y que estos datos no estén disponibles para el resto de usuarios. Además, es probable que queramos proteger la base de datos de cualquier acceso no autorizado. Por tanto, el SGBD debe proporcionar un mecanismo que garantice que solo los usuarios autorizados pueden acceder a la base de datos.

- **(7) Apoyo a la comunicación de datos**

Un SGBD debería ser capaz de interoperar con el software y aplicaciones de comunicaciones. La mayoría de usuarios acceden a la base de datos desde ordenadores personales y estaciones de trabajo. A veces estos ordenadores estarán conectados directamente al servidor donde se ejecuta El SGBD. En otros casos, accederán desde puestos remotos y se comunican con el servidor de la base de datos a través de una red. En cualquier caso, el SGBD recibirá solicitudes en forma de **mensajes** y responderá a las mismas de un modo similar. Todas esas comunicaciones son manejadas por un Gestor de Comunicaciones de Datos (DCM, *Data Communication Manager*). Aunque el DCM no es parte del SGBD, es necesario que el SGBD sea capaz de comunicarse con una variedad de DCMs, de /cara a tener éxito comercial. Incluso los SGBDs para ordenadores personales deberían ser capaces de funcionar en una red de área local, de forma que una base de datos centralizada pueda ser compartda entre los usuarios. Esto no implica que la base de datos está distribuida por la red, si no que los usuarios serán capaces de acceder a la base de datos centralizada desde puestos remotos.

- **(8) Servicios de integridad**

Con el término “integridad de la base de datos” nos referimos a la corrección y consistencia de los datos almacenados: podemos considerarla como una protección adicional de la base de datos. Aunque la integridad está relacionada con la seguridad, tiene más implicaciones: la integridad está relacionada con la calidad misma de los datos. La integridad se suele expresar en forma de *restricciones* que son pruebas de consistencia que la base de datos debe cumplir en todo momento. Por ejemplo, podemos querer especificar una restricción para que ningún evaluador en una convocatoria de subvenciones pueda evaluar más de 5 solicitudes. El SGBD tendrá que realizar esta comprobación cada vez que se asigne una solicitud a un evaluador, e impedir que la asignación tenga lugar si se supera este límite.

Estas 8 funciones fueron establecidas originalmente por Codd (1982). Hoy día, resulta razonable esperar que un SGBD también proporcione las siguientes dos funciones:

- **(9) Servicios de soporte a la independencia de datos**

El SGBD debe incluir utilidades que faciliten la independencia de los programas de la estructura real de la base de datos. Normalmente la independencia se consigue mediante un mecanismo de vistas o subesquemas. La independencia física de los datos es más fácil de conseguir: hay una serie de cambios que se pueden hacer en el nivel físico de la base de datos sin afectar al resto de niveles. Sin embargo, la independica lógica de los datos es más complicada. La suma de una nueva entidad, atributo o relaciñon es más fácil de gestionar, pero no su eliminación. En algunos sistemas, no se permite la modificación de los elementos existentes en la estructura lógica.

- **(10) Herramientas**

Se trata de herramientas software que facilitan el trabajo del administrador de la base de datos (DBA). Algunas de estas herramientas trabajan en el nivel externo y por tanto pueden ser creadas por el DBA.. Otras funcionan en el nivel interno de la base de datos, y por tanto solo pueden ser creadas por la compañía creadora del SGBD. Algunos ejemplos de este tipo de herramientas son:

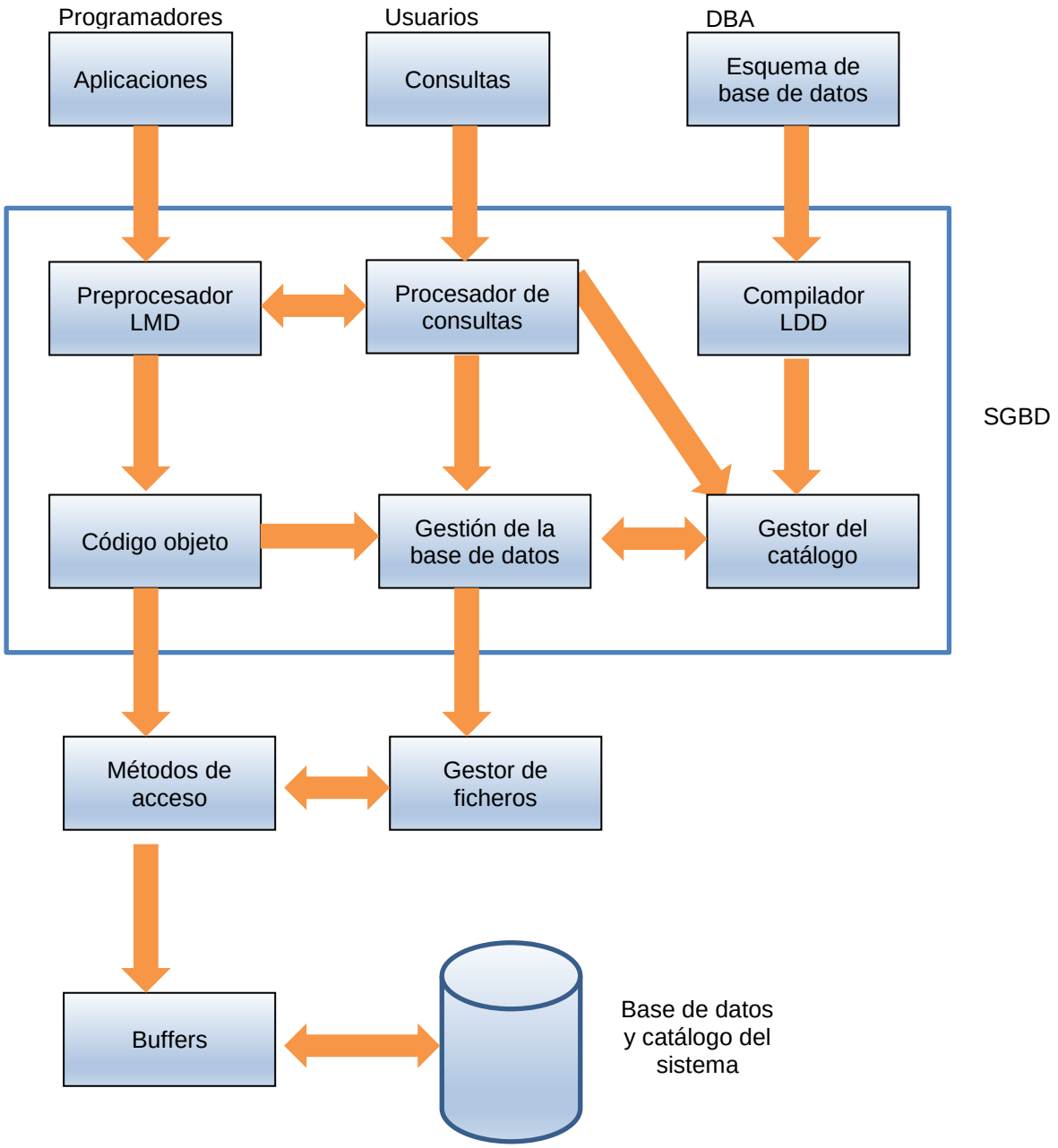
- Herramienta de importación, para cargar datos a la base de datos desde ficheros planos, y herramientas de exportación, para hacer volcados de la base de datos a ficheros planos.
- Herramientas de monitorización, para vigilar el funcionamiento y uso de la base de datos
- Programas de análisis estadísticos, para examinar el rendimiento y las estadísticas de uso
- Herramientas de reorganización de índices, permiten reorganizar los índices en caso de desbordamiento
- Recolección de basura, permite borrar físicamente registros de los dispositivos de almacenamiento, para consolidar el espacio liberado, y tenerlo disponible para nuevos registros si fuera necesario.

6 Componentes de un sistema de gestión de bases de datos (detalles)

Los SGBD son sistemas software muy complejos y sofisticados de cara a proporcionar todas las funciones descritas arriba. No es posible generalizar su estructura de componentes, ya que esta varía de un SGBD a otro. Sin embargo, es práctico visualizar estos componentes y sus relaciones cuando intentamos comprender el funcionamiento de una base de datos.

Un SGBD está dividido en varios componentes software (o *módulos*) con una responsabilidad distinta. Algunas de las funciones del SGBD son proporcionadas por el sistema operativo. Sin embargo, las funcionalidades del sistema operativo suelen ser muy básicas, y el SGBD debe proporcionar servicios enriquecidos a partir de dichas funciones. Por tanto, una parte importante del diseño de un SGBD es analizar la relación e interfaz entre el SGBD y el sistema operativo.

El diagrama inferior muestra los principales componentes de un SGBD, junto a sus interfaces con otros componentes software, como las aplicaciones o las consultas de usuario o los métodos de acceso (técnicas de gestión de ficheros que permiten la recuperación de registros de datos).



- *Procesador de consultas*. Es uno de los principales componentes del SGBD que transforma las consultas en una serie de instrucciones de bajo nivel dirigidas al gestor de la base de datos.
- *Gestor de la base de datos*. El gestor de la base de datos se comunica con consultas y programas de aplicación controlados por los usuarios. Después de aceptar una consulta examina los esquemas externos y conceptual para averiguar qué registros del nivel conceptual son necesarios para satisfacer la consulta. A continuación realiza las llamadas necesarias al gestor de ficheros para realizar la solicitud.
- *Gestor de ficheros*. El gestor de ficheros trabaja sobre los ficheros de almacenamiento y gestiona el espacio de almacenamiento en el disco. Crea y mantiene la lista de estructuras de datos e índices definidos en el esquema interno. Puede calcular qué fichero y posición en el mismo le corresponde a los registros. Sin embargo, el gestor de ficheros no manipula directamente la entrada y salda física de los datos. En realidad, le pasa las solicitudes adecuadas a los métodos de acceso, que se encargan bien de leer bien de escribir e el buffer del sistema (o *caché)*
- *Preprocesador LMD*. Este módulo convierte las sentencias DML (que fueron añadidas a los programas de aplicación) en llamadas a funciones del lenguaje de alto nivel de la aplicación. El preprocesador DML interactúa con el procesador de consultas para generar el código adecuado.
- *Compilador LDD*. El compilador LDD transforma las sentencias LDD en conjuntos de tablas que contienen metadatos. Estas tablas se almacenan a continuación en el catálogo del sistema y su información de control se almacena en las cabeceras de los ficheros de datos.
- *Gestor del catálogo*. El gestor del catálogo controla el acceso al catálogo del sistema y lo mantiene actualizado. Casi todos los componentes del SGBD tienen necesidad de acceder al catálogo del sistema.

Los principales componentes del gestor de la base de datos:

- *Control de autorización.* Este modulo confirma si cada usuario tiene permiso para llevar a cabo las operaciones que solicite
- *Procesador de comandos*. Una vez se ha confirma que el usuario tiene autorización, se le pasa el control al procesador de comandos.
- *Chequeador de integridad*. Aquellas operaciones que modifican la base de datos deben ser comprobadas por el chequeador de integridad, de modo que se asegure que satisfacen todas las restricciones de integridad necesarias.
- *Optimizador de consultas*. Este módulo busca la estrategia óptima para la ejecución de la operación.
- *Gestor de transacciones*. Este módulo controla la ejecución de las operaciones que forman parte de la transacción.
- *Planificador.* Este módulo planifica las tareas concurrentes sobre la base de datos de forma que se evite cualquier conflicto entre ellas. Controla el orden relativo en que se deben ejecutar las transacciones.
- *Gestor de recuperación*. Este módulo facilita que la base de datos esté siempre en un estado consistente a pesar de los posibles fallos. Es responsable tanto de aprobar como de abortar las transacciones.
- *Gestor del buffer*. Este módulo transfiere los datos entre la memoria principal y el almacenamiento secundario, como un disco duro. A veces se usa el término *gestor de datos* para referirse a la combinación de los gestores de buffer y de recuperación. También se usa el nombre *gestor de caché* para referirse a este gestor del búfer.

Además de estos módulos, cada implementación de un SGBD puede necesitar otras estructuras a nivel físico, incluyendo ficheros de datos y de índices o el catálogo del sistema.

7 RESUMEN ESQUEMÁTICO

La arquitectura de bases de datos ANSI-SPARC propone tres niveles de abstracción: externo, conceptual e interno. El nivel externo consiste en las vistas de los usuarios de la base de datos. El nivel conceptual es la visión institucional de la base de datos: recoge todos las necesidades de la base de datos sin tener en cuenta los detalles de implementación. El nivel interno es la representación de la base de datos desde el punto de vista del sistema informático que la alberga. En él se define como se representan los datos, como se almacenan los registros, qué punteros e índices son necesarios, etc.

El mapeo externo/conceptual indica como transformar las consultas y resultados entre estos dos niveles. De igual modo existirá un mapeo interno/conceptual.

El esquema de la base de datos es una representación de su estructura. Hay tres tipos distintos de esquemas que corresponden con los tres niveles de la arquitectura ANSI-SPARC. Gracias a la independencia de datos los esquemas de distinto nivel son inmunes a los cambios en los niveles inferiores. Se distingue entre la independencia lógica de los datos, que se refiere a la independencia entre los niveles exterior y conceptual, y la independencia física de los datos, cuando hablamos de los niveles interior y conceptual.

Hay dos tipos de lenguajes de datos: los lenguajes de definición de datos (LDD) y los lenguajes de manipulación de datos (LMD). Con el LDD se puede definir el esquema de una base de datos, mientras que el LMD se utiliza tanto para modificar como para acceder a la información almacenada. La parte del LMD que se utiliza para recuperar información también se llama lenguaje de consulta.

Un modelo es un conjunto de elementos con los que definimos un conjunto de datos, las operaciones que se pueden realizar sobre los mismos, y un conjunto de restricciones de integridad que deben cumplir estos datos. Existen tres tipos principales de modelos: modelos basados en objetos, modelos basados en registros, y modelos físicos de datos. Los dos primeros se utilizan para definir los datos en los niveles externo y conceptual; mientras que el último se utiliza a nivel interno.

Ejemplo de modelos basados en objetos son el modelo entidad relación, el modelo semántico, el modelo funcional y los modelos orientados a objetos. Ejemplos de modelos basados en registros son el modelo relacional, el modelo jerárquico y el modelo de estrella.

El modelado conceptual es el procedimiento por el que creamos una visión detallada de la arquitectura de una base de datos sin tener en cuenta los detalles relacionados con su implementación, como el SGBD que se utilizará, los lenguajes de programación elegidos para las aplicaciones, etc. El diseño del esquema conceptual es una fase crítica para el éxito de un sistema. Vale la pena dedicar el tiempo y esfuerzo necesario para obtener un modelo conceptual de calidad.

Un SGBD multiusuario debe ofrecer funcionalidades y servicios como: almacenamiento de datos, recuperación y actualización; un catálogo accesible a los usuarios; capacidad de transacciones; capacidad de trabajo concurrente; y facilidades de recuperación; sistemas de autorización; capacidad de comunicaciones de datos; control de integridad de los datos; servicios de soporte a la independencia de datos; y otras utilidades.

El catálogo del sistema es uno de los componentes fundamentales de un SGBD. En se almacenan los “datos sobre los datos” o metadatos. Este catálogo debería ser accesible para los usuarios. El Sistema de Diccionario de Recursos de Información (IRDS, *Information Resource Dictionary System*) es un estándar que define lo s métodos de acceso al diccionario de datos. Gracias a este estándar es posible compartir y transferir diccionarios de datos entre distintos sistemas.

7 GLOSARIO

Catálogo del sistema

Estructura de información en un SGBD (por ejemplo, un fichero) que recoge la definición del esquema de la base de datos: tablas, atributos, restricciones de integridad, etc.

Diccionario de datos

Catálogo del sistema. También se usa en otros contextos de computación, por lo que se considera un término más genérico que el Catálogo del sistema.

Directorio de datos

Catálogo del sistema.

Esquema

Conjunto de elementos que define la estructura de una base de datos.

Esquema conceptual

Esquema global de una base de datos definido con independencia de los detalles de implementación.

Esquema externo

Esquema de la base de datos desde el punto de vista de un usuario del sistema. Sinónimo de subesquema.

Esquema interno

Esquema de una base de datos desde el punto de vista del sistema informático (ordenador, sistema operativo) que le da soporte.

Independencia de datos

Propiedad por la que los esquemas o modelos de datos no se ven afectados por los cambios realizados en niveles inferiores.

Inconsistencia

Estado de una base de datos en el que la información recogida en esta no corresponde con las acciones llevadas a cabo sobre la misma. Puede deberse a la ejecución parcial de acciones, o a una falta de control de la concurrencia.

Instancia

Información recogida en una base de datos en un momento determinado.

Mapeo interno/conceptual

Es la correspondencia utilizada para transformar los elementos del modelo interno en el modelo conceptual y viceversa, en consultas y resultados.

Mapeo externo/conceptual

Es la correspondencia utilizada para transformar los elementos del modelo externo en el modelo conceptual y viceversa, en consultas y resultados.

Metadato

Datos que describen los datos, por ejemplo, su tamaño o lugar de almacenamiento. Se guardan en el catálogo del sistema.

Modelo de datos

Esquema de la base de datos.

Nivel conceptual

En la arquitectura ANSI-SPARC de tres niveles es el nivel intermedio, en el que se define la base de datos de manera global sin tener en cuenta los detalles relativos a su implementación.

Nivel externo

Nivel superior de la arquitectura ANSI-SPARC, en el que se representa la base de datos desde el punto de vista de cada uno de sus usuarios.

Nivel físico

Nivel interno

Nivel interno

Nivel inferior de la arquitectura ANSI-SPARC. En él se concibe la base de datos desde el punto de vista de su implementación final.

Subesquema

Esquema externo.

8 BIBLIOGRAFÍA BÁSICA

T. Connolly y C.Begg. Database systems: a practical approach to design,implementation, and management (6th ed.). Capítulo 2.