TEMA 35

LA DEFINICIÓN DE DATOS. NIVELES DE DESCRIPCIÓN. LENGUAJES. DICCIONARIO DE DATOS.

ÍNDICE

1. INTRODUCCIÓN

La función de descripción o definición debe permitir al administrador de la base de datos especificar los elementos que la integran, su estructura y las relaciones que existen entre ellos, las reglas de integridad de semántica, los controles a efectuar antes de autorizar el acceso a la base, etc., así como las características de tipo físico y las vistas lógicas de los usuarios.

Esta función, realizada por el lenguaje de descripción o definición de datos (DDL) propio de cada SGBD, debe suministrar los medios para definir las tres estructuras de datos (externa, lógica global e interna), especificando las características de los datos en cada uno de estos tres niveles.

A nivel interno, se ha de indicar el espacio (volúmenes, cilindros y pistas) reservado para la base de datos, la longitud de los campos o elementos de datos, su modo de representación (binario, decimal, alfanumérico, punto fijo o flotante, etc.). Además, se deben poder definir caminos de acceso como punteros, índices, etc.

Para las estructuras externa y lógica, la función de descripción ha de proporcionar los instrumentos para la definición de las entidades y su identificación, atributos de las mismas, interrelaciones entre ellas, autorizaciones de acceso, restricciones de integridad, etc. Las descripciones de las estructuras lógicas de los usuarios han de estar referidas a la estructura lógica global. El SGBD, además de suministrar facilidades de descripción, se ocupará de la función de correspondencia o transformación (mapping) de las estructuras lógicas externas orientadas a los usuarios en la estructura lógica global, y de la relación entre ésta y la estructura física.

2. LENGUAJES DE LOS SGBD

Las distintas funciones que ha de cumplir un SGBD hacen necesario disponer de diferentes lenguajes y procedimientos que permitan la comunicación con la base de datos; unos están orientados hacia la función (definición o manipulación), y otros a diferentes tipos de usuarios o de procesos.

Los lenguajes de los SGBD habrán de tener un enfoque distinto según el tipo de usuario, ya que, indudablemente, las exigencias de un programador o del administrador no tienen nada que ver con lo que puede necesitar un usuario no informático.

La estructura y sintaxis del lenguaje va a depender de cada SGBD. Las normas Codasyl proponen especificaciones concretas de la sintaxis para los lenguajes de descripción y manipulación que corresponden a un modelo de datos en red. En el caso de los SGBD relacionales, el SQL es un estándar muy extendido, y para los modelos en red existe el estándar de ANSI NDL/ANS, basado en las normas Codasyl, pero que a diferencia del SQL ha tenido una incidencia comercial baja.

2.1. Lenguaje de definición de datos (DDL) y lenguaje de manipulación de datos (DML)

Lenguaje de definición de datos (DDL):

Este es el lenguaje usado para definir la estructura lógica de la base de datos. Es decir, para crear el esquema y los subesquemas. Por extensión, se denomina DDL al software que interpreta este lenguaje.

- Permite especificar un **esquema de la base de datos**.
- Al compilar se generan tablas que se almacenan en el **diccionario de datos** (fichero que contiene “metadatos”).
- El lenguaje de almacenamiento y definición de datos permite especificar: La estructura de almacenamiento y los métodos de acceso.

La definición del esquema consiste en la enumeración de los registros que van a existir en la base de datos, especificando para cada uno de ellos los campos de que consta y los conjuntos de que forma parte. Para describir cada registro es necesario especificar:

- Nombre del registro.
- Descripción de campos. Consistente, para cada campo, en:
  - Nombre del campo
  - Tipo (numérico, alfabético, alfanumérico o binario).
  - Tamaño en bytes ocupado por el campo.
  - Restricciones en los valores que puede tomar.

Lenguaje de manipulación de datos (DML):

Es el lenguaje usado para acceder a la base de datos, para leer, escribir o modificar información. Las sentencias de DML se pueden utilizar de forma interactiva o bien desde algún lenguaje de alto nivel.

- Permite a los usuarios manejar o tener acceso a los datos organizados según un modelo apropiado.
- Su **objetivo** será lograr una interacción eficiente entre las personas y el sistema.
- Se pueden distinguir dos tipos: De procedimientos y sin procedimientos.
- El lenguaje de consultas se encarga de recuperar la información.

2.2. Lenguaje de definición de datos (DDL) en SQL

Cualquier lenguaje de bases de datos está formado por un DDL y por un DML. Existe un lenguaje considerado como estándar para manipular bases de datos relacionales: SQL (Structured Query Language). Estudiaremos el DDL y el DML que utiliza el SQL. Las características principales de SQL son su sencillez, su carácter estándar y ser un lenguaje declarativo o no procedural (para que SQL realice una acción concreta no se le dice cómo tiene que hacerla, sino lo que queremos obtener).

El **lenguaje de definición de datos** (**DDL**) *proporciona órdenes para definir, eliminar y modificar tablas, así como para crear índices y vistas*. Las órdenes SQL más importantes del DDL son:

CREATE TABLE. Define el esquema de la base de datos. Crea una tabla con los campos y tipos indicados. Su sintaxis es la siguiente:

CREATE TABLE nombre_tabla (n_campo1, t_campo1, ..., n_campoN, t_campoN)

n_campo es el nombre de un campo y t_campo es su tipo

ALTER TABLE. Con esta orden se pueden añadir nuevos campos a una tabla ya existente, modificar o borrar los campos de la tabla. Su sintaxis es la siguiente:

ALTER TABLE nombre_tabla ADD (n_campo, t_campo)
ALTER TABLE nombre_tabla MODIFY (n_campo, t_campo)
ALTER TABLE nombre_tabla DROP n_campo

DROP TABLE. Borra el esquema de la base de datos, es decir, destruye tanto los datos contenidos en la tabla como la estructura de la tabla. Su sintaxis es:

DROP TABLE nombre_tabla

3. NIVELES DE DESCRIPCIÓN DE DATOS

El administrador de la base de datos ha de disponer de instrumentos que le permitan definir los datos con facilidad y precisión, especificando sus distintas estructuras. Esto es lo que se denomina *lenguaje de definición de datos*.

Así pues, el SGBD tendrá que facilitar medios para describir la estructura lógica global, para hacer las especificaciones relativas a la estructura interna y para declarar las estructuras externas que sean requeridas para el desarrollo de las diferentes aplicaciones.

Es decir, como hay diferentes puntos de vista, es preciso describir los datos de diferentes formas:

- Desde el punto de vista del programador de aplicaciones: **Lenguaje para subesquemas** (Externo).
- Desde el punto de vista lógico global. **Lenguaje para esquemas** (Conceptual).
- Desde el punto de vista físico**: Lenguaje interno** (Físico).

El sistema de gestión de bases de datos utiliza las descripciones de datos para derivar los registros lógicos globales de los registros físicos y para derivar los registros requeridos por los programas de aplicación a partir de los registros lógicos globales.

3.1. Lenguajes para subesquemas (Vista externa). Estructuras externas.

El SGBD debe poner a disposición de los usuarios los medios que les permitan recuperar o actualizar los datos contenidos en la base, de acuerdo con la visión lógica o estructura externa (**vista**) que precise de cada aplicación.

En muchos sistemas de bases de datos, el lenguaje que utiliza el programador para describir sus datos difiere del que emplea el administrador. Un lenguaje común para describir datos desde el punto de vista del programador es el de la división de datos de Cobol. Todo programa escrito en Cobol contiene una descripción de los datos que emplea. Esa descripción concierne sólo a los datos para ese programa y se compila junto con las restantes partes del programa. En Cobol, como en otros lenguajes de programación, no existe el concepto de la definición de datos como operación independiente. Los enunciados de la división de datos son siempre parte de un programa determinado.

Sería útil que la descripción del programador tuviese la misma forma que la descripción de los mismos datos por parte del administrador. Desgraciadamente, en algunos de los sistemas de bases de datos, estas dos descripciones se escriben de forma muy variada. Ambas descripciones se compilan ya listas para el uso y los tipos de ítem de datos, tipos de segmento y otros tipos de datos están correlacionados.

En bastantes organizaciones, cuando las funciones del administrador de la base de datos están bien especificadas, es éste el responsable de la definición de las vistas externas, y el programador sólo tiene que ocuparse de llamarlas desde su propio programa.

Los lenguajes autocontenidos suelen incluir facilidades para la descripción de datos que se desean recuperar o actualizar. En este caso, no es habitual definir las vistas externas por el administrador, sino que el SGBD proporciona, dentro del lenguaje autocontenido, los medios para describir de forma muy sencilla esta estructura.

Es el nivel en el cual trabaja el usuario individual. Los usuarios pueden ser o bien programadores de aplicaciones o usuarios finales.

Cada usuario dispone de un lenguaje. En el caso de un programador de aplicaciones dicho lenguaje puede ser un lenguaje de alto nivel para manejar la base de datos y si la base de datos no lo permite, se utilizará un lenguaje propio del sistema de B.D. En el caso de ser un usuario final será o bien un lenguaje de consulta, por ejemplo el SQL o algún lenguaje de aplicación basado en menús.

Los lenguajes de programación deben incluir un sublenguaje de datos, es decir, un subconjunto del lenguaje total que se ocupe de manera específica de los objetos y operaciones de la base de datos. Se dice que el sublenguaje de datos (DSL) está inmerso dentro del lenguaje anfitrión correspondiente.

En principio, cualquier sublenguaje de datos es en realidad una combinación de por lo menos dos lenguajes subordinados:

> D.D.L.: Lenguaje de definición de datos: con el cual es posible definir o declarar los objetos de la base de datos.
> D.M.L.: Lenguaje de manipulación de datos: con el que es posible manipular o procesar dichos objetos.

Cuando el DSL es indistinguible del lenguaje anfitrión se dice que está fuertemente acoplado y si se pueden separar con nitidez se dice que están débilmente acoplados.

A la vista individual de cada usuario se denomina **vista externa**. La vista externa está formada por el conjunto de ocurrencias de los registros externos.

Toda vista externa se define mediante un esquema externo que es la definición de los tipos de registros externos en esa vista externa.

3.2. Lenguajes para esquemas (Vista conceptual). Estructura lógica global.

La mayoría de los SGBD tienen sus propios lenguajes para definir los esquemas utilizados. Cualquier lenguaje de programación, no posee capacidad suficiente para definir la variedad de relaciones que existen eventualmente en los esquemas.

El administrador deberá poder asignar nombres a los campos, a los agregados de datos, a los registros, etc., establecer sus longitudes y características, así como las relaciones entre estos elementos;

especificar los identificadores, e indicar las restricciones semánticas que se han de aplicar a los diferentes objetos descritos (entidades, atributos e interrelaciones).

En esta descripción no deberían incluirse informaciones tendentes a determinar ningún tipo concreto de estructura física que obligue a cambiar dicha descripción ante alteraciones de las características del soporte físico o de los caminos de acceso.

El nivel conceptual está formado por las vistas conceptuales, que son la representación de toda la información contenida en la base de datos.

Las vistas conceptuales se componen de varias ocurrencias de varios tipos de registro conceptual. Esta vista se define por medio de un esquema conceptual, el cual incluye definiciones de cada uno de los tipos de registro conceptual.

El esquema conceptual se define mediante el **DDL conceptual**. El DDL externo debe ser distinto al DDL conceptual. Si se logra hacer totalmente independiente el DDL conceptual a los datos, el nivel externo también lo será.

Es de esperar que las definiciones en el esquema conceptual incluyan muchas más características, como son las verificaciones de seguridad y de integridad.

3.3. Lenguaje interno (Físico). Estructura interna.

En un SGBD en el cual fuesen totalmente independientes las estructuras física y lógica global, y que consiguiese automáticamente la optimización en el almacenamiento y recuperación de los datos, el SGBD podría encargarse, a partir de la estructura lógica global, de definir la estructura interna adecuada sin ninguna intervención ajena al propio sistema, para lo cual habría que suministrar al SGBD las informaciones precisas, como volúmenes, crecimiento previsto, tipos de registros más accedidos, relación de actualizaciones y consultas, etc.

Se puede mejorar considerablemente el rendimiento de la base de datos si el administrador especifica características respecto a la estructura física que tiendan a conseguir una máxima eficiencia en el almacenamiento y en la recuperación de los datos de la base.

Bastantes SGBD tienen sólo dos niveles de descripción de datos, uno para las vistas externas y otro en el que se especifican tanto las características lógicas como las físicas. Esto supone un inconveniente, ya que, cualquier cambio de tipo físico que se produzca obligará a redefinir y compilar de nuevo el esquema.

El nivel interno es una representación de bajo nivel de toda la base de datos, se compone de varias ocurrencias de varios tipos de registro interno.

La vista interna se define mediante el esquema interno, el cual no sólo define los diversos tipos de registros almacenados sino también especifica cuáles índices hay, representación de los campos almacenados, secuencia física que van a seguir los registros almacenados, etc. El esquema interno se define mediante el lenguaje de definición de datos **DDL interno**.

3.4. Diferencias entre las distintas descripciones

La descripción de los datos de subesquema puede diferir de la descripción del esquema en lo siguiente:

- El subesquema puede omitir la descripción de ciertos datos que se hallan en el esquema. Puede omitir ítems de datos, agregados, registros o ficheros completos.
- Las características de los ítems de datos pueden alterarse. Por ejemplo, un ítem de datos que está en BCD en el subesquema y en binario puro en el esquema.
- Puede ser diferente la secuencia de los ítems de datos en un agregado de datos, o en un registro, o en otra subdivisión.
- Puede ser diferente la secuencia de registros de un fichero.
- Los vectores pueden definirse como matrices multidimensionales.
- El esquema puede definir relaciones distintas de las que aparecen en el subesquema.
- Pueden describirse diferentes tipos de registros, compuestos por ítems de datos que se toman de otros tipos de registro.
- Pueden describirse distintos tipos de agregados de datos compuestos por ítems de datos tomados de otros tipos de agregado o de registro.
- Pueden modificarse los códigos de seguridad.

3.5. Correspondencias

Se distinguen dos tipos de correspondencias que se encargan de conectar los tres niveles de una base de datos.

Conceptual - Interna: Se encarga de conectar la vista conceptual con la base de datos almacenada y se encarga de especificar como se representan los registros y campos en el nivel interno. Si se modifica la definición de la estructura de almacenamiento esta correspondencia deberá modificarse también de para que no varíe el esquema conceptual.

Externa - Conceptual: Se encarga de conectar la vista externa con la vista conceptual y de conectar los distintos campos del nivel externo y con los del nivel conceptual. También se encarga de mantener la correspondencia en el caso de que varios registros o campos conceptuales se correspondan con uno o más registros o campos externos.

4. INDEPENDENCIA

En un lenguaje para la descripción de esquemas no deberían existir enunciados que se refieran a la organización física del almacenamiento, porque esos enunciados destruirían la independencia de datos y sería necesario modificar las descripciones lógicas cada vez que se modificase la organización física.

De la misma forma, los lenguajes para la descripción de subesquemas tendrían que ser independientes del lenguaje utilizado en la descripción del esquema. El administrador de datos debería tener completa libertad para servir a los usuarios de datos con la mayor eficiencia posible, modificando los esquemas y los métodos de acceso cuando dicha modificación reporte ventajas, pero esos cambios no deberían exigir modificaciones en los programas de aplicación ni en los subesquemas.

Un área en la que es difícil separar completamente las vistas lógica y física de los datos es la del direccionamiento y exploración de la base de datos. La descripción lógica deberá incluir probablemente algún enunciado acerca del acceso a la información y los requerimientos de la exploración para que puedan ser cumplidos eficientemente, pero no debe especificar técnicas del caso, porque éstas cambian.

La manera en que el sistema localiza un registro o realiza una búsqueda no debería interesar al diseñador de la base de datos lógica, por lo menos en teoría. Esta cuestión concierne al diseñador de la distribución del almacenamiento físico.

4.1. Independencia de datos

Se dice que una aplicación es **dependiente de los datos** si es imposible alterar la estructura de almacenamiento o la técnica de acceso sin afectar a la aplicación. En un sistema de bases de datos no es recomendable tener aplicaciones dependientes de los datos, al menos por dos razones:

1. Cada aplicación puede requerir una vista diferente de los mismos datos.

2. El administrador de la B.D. ha de tener libertad de modificar la estructura de almacenamiento o las técnicas de acceso (o las dos cosas) para adaptarlos al cambio de los requerimientos sin tener que modificar las aplicaciones ya existentes.

Se va a buscar la **independencia de datos** a tres niveles:

1. **Nivel de campo almacenado:** Mínima cantidad de información que se almacena reconocible con un nombre.

2. **Nivel de registro almacenado:** Es un conjunto de campos almacenados relacionados entre sí, que cuenta con su propio nombre.

3. **Nivel de fichero almacenado:** Es un conjunto (con nombre) de todas las ocurrencias de un tipo de registro almacenado correspondientemente.

Registro lógico = Registro que ve el usuario.

Registro físico = Registro tal y como se almacena en la base de datos.

El campo lógico puede ser igual o no al campo almacenado, por tanto, se puede buscar la independencia de datos basándose en este concepto, que es la materialización, que puede ser de dos formas:

a) Directa: Campo lógico = campo almacenado.
b) Inversa: Campo lógico campo almacenado.

Ejemplo:

Registros físicos: 

Registro lógico: 

A nivel de fichero almacenado lo que se debe es prever el medio físico en el que se va almacenar porque una base de datos es dinámica.

5. DICCIONARIO DE RECURSOS DE INFORMACIÓN

Debe existir una verdadera arquitectura de datos que facilite la integración.

El núcleo de dicha arquitectura será el **diccionario de recursos de información** (**DRI**), que contendrá las descripciones de todos los datos que constituyen el sistema de información.

Las finalidades principales de los DRI son precisamente ayudar a la gestión de la información como recurso y conseguir la integración de la semántica de las aplicaciones de forma centralizada, con lo que se aumentan los beneficios estratégicos, operacionales y de control de las empresas.

Los recursos informáticos de las instituciones se gestionan almacenando, administrando y controlando los denominados **metadatos**, esto es, los datos que definen y describen los datos de la institución. El término más difundido para estos metadatos es el de **Diccionario de Datos** (**DD**).

5.1. Tipos de almacenes de datos: Diccionario de Datos y Directorio de Datos

Hay que distinguir entre los conceptos de diccionario y directorio de datos, que, aunque muy relacionados entre sí, cumplen finalidades distintas en el sistema de información.

Diccionario de datos: Reúne la información sobre los datos almacenados en la base de datos, como descripciones (narrativas y técnicas), significado, estructuras, consideraciones de seguridad, edición y uso de aplicaciones, etc., es decir, lo que los usuarios necesitan para comprender el significado de los datos y poder recuperarlos y manejarlos adecuadamente. El diccionario de datos tiene, por tanto, una finalidad de descripción lógica de los datos y está orientado hacia el usuario.

Directorio de datos: Es un subsistema del SGBD, encargado de describir dónde y cómo se almacenan los datos de la base, el modo de acceso y otras características físicas de los datos, atendiendo de este modo las peticiones de los programas y procesos. Un directorio de datos contiene, en definitiva, las especificaciones necesarias para pasar de la representación externa de los datos a su representación interna. Ha de estar siempre en un formato legible para la máquina. Su objetivo principal es transmitir al sistema la información necesaria para poder acceder a los datos contenidos en la base. El directorio, a diferencia del diccionario, es un instrumento del SGBD, y está orientado a facilitar a éste la información que necesita para su funcionamiento.

La complejidad del diccionario de datos se ha hecho cada vez mayor, ampliando el número de funciones que se le asignaban originalmente, así como el entorno operativo en el que se utilizaban. En los años setenta aparecieron varios paquetes de soporte lógico de este tipo: DB/DC, DATADICTIONARY, DATAMANAGER, ADR/DATADICTIONARY, LEXICON, etc.

Algunos de estos paquetes realizan tanto las funciones de diccionario como las de directorio, denominándose entonces **Diccionario/Directorio de datos** (**D/DD**).

Un sistema de base de datos efectivo permitirá el crecimiento y la modificación de la base de datos sin comprometer la integridad de los datos. El Diccionario/Directorio de datos ayuda a cumplir los objetivos permitiendo que las definiciones de datos se mantengan separadas de los datos. Esto permite que se hagan cambios en las definiciones de datos sin que tenga efecto en los datos almacenados. Por ejemplo, el subdiagrama usado por un programa en particular puede modificarse sin afectar de ninguna manera a los datos almacenados. Otros beneficios proporcionados por el D/DD incluyen:

- Las estructuras físicas de almacenamiento pueden cambiarse sin que afecte al programa que usa los datos.
- Las contraseñas y otras medidas de seguridad pueden estar almacenadas en el D/DD para facilitar el control sobre el acceso a los datos.
- La definición centralizada de los datos permite fáciles informes sobre el estado de la base de datos: Quién es responsable de varios elementos de datos, qué controles se les aplica, y qué programas y usuarios están accediendo a los datos.

Para producir estos beneficios, el D/DD incluye usualmente las siguientes características:

- Un lenguaje para definir las entradas en el D/DD.
- Un lenguaje de manipulación para añadir, eliminar y modificar las entradas en el D/DD.
- Métodos para validar entradas en el D/DD.
- Medios para producir informes con relación a los datos contenidos en el D/DD.

Un desarrollo importante en los SGBD relacionales es la práctica común de almacenar el directorio como un conjunto de relaciones. Esto permite el uso de los lenguajes de manipulación de datos de los SGBD para consultar, actualizar y mantener el diccionario de datos. La siguiente figura muestra el fragmento del diccionario de datos usado por DB2, un SGBD relacional de IBM:

TABLAS_SISTEMA	NOMBRE	CREADOR	CANTIDADCOL
	PRODUCTO	JHANSEN	3
	MFR	JHANSEN	4
COLUMNAS_SISTEMA	NOMBRE	NOMBRETB	TIPOCOL
	IDPROD	PRODUCTO	ENTERO
	DESCRPROD	PRODUCTO	CARÁCTER
	IDMFR	MFR	ENTERO
	PAÍS	MFR	CARÁCTER

También se han construido y comercializado herramientas conocidas por el nombre de CASE (Computer Aided Software Engineering), que contienen un diccionario llamado enciclopedia o **repositorio**, donde se almacenan los datos generados durante el ciclo de vida de un sistema de información: esquemas, grafos, matrices, información relativa a la gestión de proyectos, gestión de configuraciones, etc.

Recientemente, ha aparecido un nuevo concepto, el **diccionario de recursos de información** (**DRI**), que pretende ser el eslabón final en la evolución de los almacenes de datos. El DRI constituye el depósito integrado de todos los datos sobre la organización, automatizados o no, que son utilizados para efectuar las labores de planificación, control y operación que permitan a la empresa cumplir sus objetivos. Éstos engloban de algún modo las capacidades y funciones de todos los almacenes de datos anteriores.

El catálogo de los SGBD Relacionales se puede considerar como un diccionario/directorio de datos, pero de bajo contenido semántico. En este caso, el DRI es una base de datos más que se suele denominar **metabase**. En este enfoque se accede a la metabase con el mismo lenguaje que a las demás bases de datos (SQL en los SGBD) y será mucho más fácil de extender que un diccionario separado de las otras bases de datos.

Para Codd (1990), uno de los requisitos para que un SGBD pueda considerarse como verdaderamente relacional es que soporte un catálogo dinámico en línea en el que la descripción de la base de datos se represente como cualquier otro dato, permitiendo a los usuarios autorizados aplicar el mismo lenguaje relacional tanto a la descripción de la base de datos como a los datos regulares.

6. BIBLIOGRAFÍA

Gary W. Hansen
Diseño y Administración de Bases de Datos
Prentice Hall, 1998

Alberto Prieto
Introducción a la Informática
Mc Graw-Hill, 2ª edición, 1997

Alfonso Ureña López
Fundamentos de Informática
Ra-ma, 1997