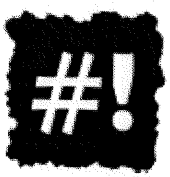


Preparador Informática

www.preparadorinformatica.com

TEMA 35. INFORMÁTICA

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

TEMA 35 INF: LA DEFINICIÓN DE DATOS. NIVELES DE DESCRIPCIÓN. LENGUAJES. DICCIONARIO DE DATOS.

1. INTRODUCCIÓN

Hoy en día el mayor activo de las organizaciones son los datos y su gestión eficaz y segura. Por ello, si analizamos la mayoría de los ámbitos de actividad, nos encontramos que la utilización de las bases de datos está ampliamente extendida (al registrarse en una web, al acudir a la consulta médica, al consultar el catálogo de productos de una tienda online, etc.). Las bases de datos y los datos contenidos en ellas, son imprescindibles para llevar a cabo multitud de acciones.

Para poder trabajar con bases de datos relacionales, lo primero que hay que hacer es definirlas. En el presente tema vamos a empezar definiendo y diferenciando los conceptos de base de datos y sistema gestor de base de datos y posteriormente nos adentraremos en la definición de datos, niveles de descripción y analizaremos el funcionamiento de los lenguajes de definición de datos (DDL). Para ello, se detallarán las órdenes para crear y borrar una base de datos relacional y para crear, borrar y modificar las diferentes tablas que la componen. Terminaremos el tema explicando el concepto y objetivos del diccionario de datos.

2. LA DEFINICIÓN DE DATOS

Podemos definir una **base de datos** como un conjunto, colección o depósito de datos almacenados en un soporte informático no volátil. Los datos están interrelacionados y estructurados.

Un **sistema gestor de base de datos o SGBD** es una colección de programas de aplicación que permite a los usuarios la creación y el mantenimiento de una base de datos, facilitando la definición, construcción y manipulación de la información contenida en ésta.

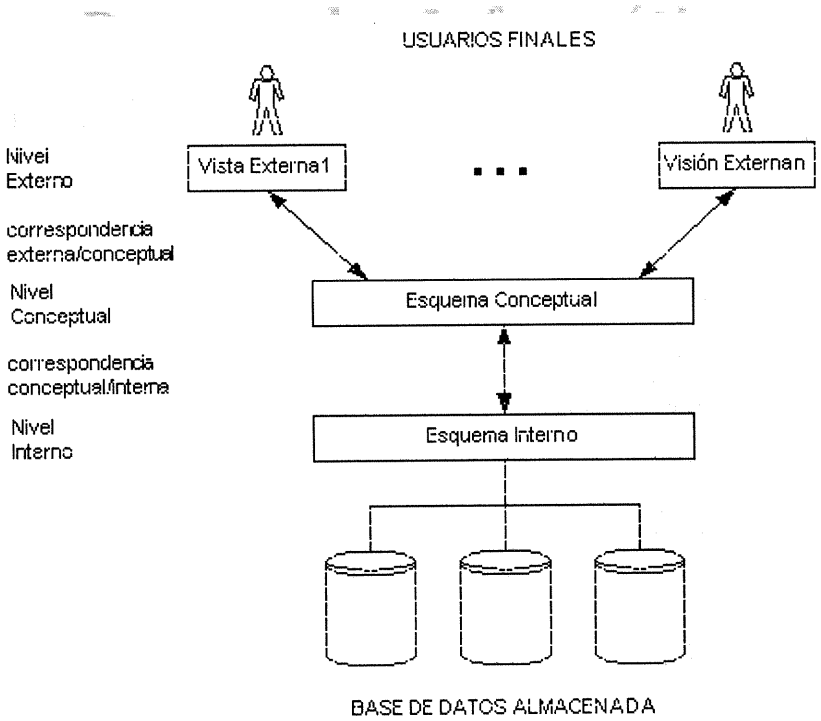
En la actualidad debido al gran volumen de datos que manejamos se requiere de sistemas gestores de bases de datos robustos, que nos permitan acceder y gestionar los datos de un modo eficaz y eficiente.

Un SGBD desarrolla tres funciones fundamentales como son las funciones de definición, manipulación y control de los datos. Me centraré en la función de definición de datos que es de lo que este tema trata.

La **función de definición de datos** debe permitir al diseñador de la BD especificar los elementos de los datos que la integran, su estructuras, las relaciones entre ellos..., en definitiva permite describir y definir los esquemas de la base de datos. Y esta definición de datos se hará a todos los niveles: interno (índices, características físicas de almacenamiento...), conceptual (tablas y restricciones propias de la BD), y externo (vistas...). Esta función se realiza mediante el **lenguaje de definición de datos** (DDL) propio de cada SGBD.

3. NIVELES DE DESCRIPCIÓN

En 1975, el comité ANSI-SPARC propuso una arquitectura de tres niveles de abstracción para los SGBD cuyo objetivo principal era separar los programas de aplicación de la base de datos física: nivel interno o físico, nivel lógico o conceptual y nivel externo o de visión del usuario.
El SGBD se encargará de hacer las correspondencias entre los tres niveles de esquemas.



3.1. NIVEL INTERNO O FÍSICO

Este es el nivel más bajo de abstracción y asociado a él se encuentra el esquema interno. En él se describe la estructura física de la base de datos a través de un esquema que detalla el sistema de almacenamiento de la base de datos y sus métodos de acceso. Es el nivel más cercano al almacenamiento físico. Los usuarios que trabajan a este nivel son los **diseñadores de la base de datos o los Administradores.**

3.2. NIVEL LÓGICO O CONCEPTUAL

Se describe con el esquema conceptual o esquema de la BD. Se obtiene a partir de los requerimientos de los usuarios potenciales del sistema de BD a implantar así que en este nivel se describe la estructura completa de la base de datos a través de un esquema que detalla las entidades, atributos, relaciones, operaciones de los usuarios y restricciones. A este nivel los usuarios que intervienen son los **programadores,** encargados de crear las estructuras lógicas necesarias para guardar la información.

3.3. NIVEL EXTERNO O DE VISIÓN DEL USUARIO

En este nivel se describen las diferentes vistas que los usuarios percibirán de la base de datos. Cada tipo de usuario o grupo de ellos verá sólo la parte de la base de datos que le interesa, ocultando el resto.

3.4. CORRESPONDENCIAS

En un SGBD basado en la arquitectura de tres niveles, cada grupo de usuarios hace referencia exclusivamente a su propio esquema externo. Por lo tanto, el SGBD debe transformar cualquier petición expresada en términos de un esquema externo a una petición expresada en términos del esquema conceptual, y luego, a una petición en el esquema interno, que se procesará sobre la BD almacenada. Si la petición es de una obtención de datos, será preciso modificar el formato de la información extraída de la BD, para que coincida con la vista externa del usuario. El proceso de transformar peticiones y resultados de un nivel a otro se denomina **correspondencia**.

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 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 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. LENGUAJES DE DEFINICIÓN DE DATOS (DDL)

4.1 DEFINICIÓN

Los lenguajes del SGBD son el Lenguaje de Definición de los Datos (DDL), Lenguaje de Control de Datos (DCL) y el Lenguaje de Manipulación de Datos (DML). Veamos en profundidad todo lo referente a los lenguajes de definición de datos:

Un **lenguaje de definición de datos** es un lenguaje proporcionado por el sistema de gestión de base de datos que permite a los usuarios de la misma llevar a cabo las tareas de definición de las estructuras que almacenarán los datos, así como de los procedimientos o funciones que permitan consultarlos. La definición de la estructura de la base de datos incluye tanto la creación inicial de los diferentes objetos que formarán la base de datos, como el mantenimiento de esa estructura.

Habitualmente, los DDL suelen ser lenguajes muy simples con una gramática muy sencilla de manera que permiten describir los datos con facilidad y precisión y sin necesidad de apoyarse en ningún lenguaje de programación.

Gracias a la utilización del DDL obtendremos un conjunto de tablas que se almacenarán en un archivo llamado **diccionario de datos**. En este diccionario de datos contiene "metadatos", es decir, datos sobre nuestra propia base de

datos. Cada vez que se añada, modifique o borre información sobre la estructura de la Base de Datos, el diccionario se actualizará para reflejar esos cambios.

4.2. TIPOS

Podemos considerar fundamentalmente dos tipos de lenguajes de definición de datos:

- **Interactivo**: Los lenguajes interactivos permiten tratar directamente con el sistema gestor de base de datos por medio de una consola.
- **Incrustado o embebido**: Estos lenguajes se utilizan dentro de un lenguaje anfitrión cuando éste requiera hacer operación contra el sistema gestor de base de datos.

En algunos lenguajes ambos tipos pueden ser solapados, por ejemplo, el SQL se puede usar como lenguaje de consulta interactivo o como lenguaje incrustado.

4.3. CARACTERÍSTICAS

Las características fundamentales de los DDL son:

- **Dependencia del SGBD:** El DDL depende tanto del modelo de base de datos como del SGBD concreto utilizado. En el DDL es donde radican las mayores diferencias entre los distintos SGBD relacionales.
- **Simplicidad:** son lenguajes sencillos, compuestos de pocas instrucciones aplicables a muchos objetos diferentes. Habitualmente se utilizan los mismos verbos para todos los objetos diferentes.
- Son **lenguajes de alto nivel**.
- **Distintos niveles de abstracción**: permiten la definición de datos a varios niveles: externo, lógico e interno.

4.4. SQL COMO DDL.

La primera fase del trabajo con cualquier base de datos comienza con sentencias DDL, puesto que antes de poder almacenar y recuperar información debemos definir las estructuras donde almacenar la información. Las estructuras básicas con las que trabaja SQL son las tablas. Las instrucciones DDL generan acciones que no se pueden deshacer, por eso es conveniente usarlas con precaución y tener copias de seguridad cuando manipulamos la base de datos.

a) Creación y eliminación de base de datos.

Básicamente, la creación de la base de datos consiste en crear las tablas que la componen. Crear una base de datos implica indicar los archivos y ubicaciones que se van a utilizar además de otras indicaciones técnicas y administrativas. Es obvio que todo esto sólo lo puede realizar si se tiene privilegio de Administrador.

Con el estándar de SQL la instrucción a usar sería ***Create Database***, pero cada SGBD tiene un procedimiento para crear las bases de datos. Crearíamos una base de datos con el nombre que se indique a continuación:

```
CREATE DATABASE NombredemiBasedeDatos;
```

En caso de que quisiéramos eliminar una base de datos usamos:

```
DROP DATABASE NombredemiBasedeDatos;
```

b) Creación de tablas.

Los objetos básicos con los que trabaja SQL son las tablas, que como ya sabemos es un conjunto de filas y columnas cuya intersección se llama celda. Es ahí donde se almacenarán los elementos de información, los datos que queremos recoger. Antes de crear la tabla es conveniente planificar algunos detalles:

- Qué nombre le vamos a dar a la tabla.
- Qué nombre le vamos a dar a cada una de las columnas.
- Qué tipo y tamaño de datos vamos a almacenar en cada columna.
- Qué restricciones tenemos sobre los datos.
- Alguna otra información adicional que necesitemos.

La sintaxis básica del comando que permite crear una tabla es la siguiente:

```
CREATE TABLE [esquema.] NombredeTabla (
columna1 Tipo_Dato [ModificadordeTipo],
columna2 Tipo_Dato [ModificadordeTipo],
...
columnaN Tipo_Dato [ModificadordeTipo]  ) ;
```

donde:

- *columna1, columna2, ..., columnaN son los nombres de las columnas que contendrá la tabla.*

- *Tipo_Dato indica el tipo de dato de cada columna. Podremos tener, entre otros, datos de tipo Char, number, integer, date y long.*
- *ModificadordeTipo/Restricciones: Una restricción es una condición que una o varias columnas deben cumplir obligatoriamente. Entre otras podemos destacar NULL (Admite valores nulos), NOT NULL (no permite valores nulos) y UNIQUE (no admite valores repetidos). Al definir la tabla podemos indicar cuál de las columnas constituye la clave principal (PRIMARY KEY) que es el cual nos permite distinguir entre varios registros de datos y cuáles son las claves secundarias, indicando el nombre del atributo y la tabla con el que se relaciona (REFERENCES FOREIGN KEY). Las restricciones CKECK exigen la integridad del dominio mediante la limitación de los valores que puede aceptar una columna.*

Ejemplo:

```
CREATE TABLE ALUMNOS (
      NUMERO_MATRICULA INT NOT NULL PRIMARY KEY,
      NOMBRE VARCHAR(15) NOT NULL,
      FECHA_NACIMIENTO DATE,
      DIRECCION VARCHAR(30),
      LOCALIDAD VARCHAR(15) );
```


c) Modificación de tablas.

a) Si queremos cambiar el nombre de una tabla:

```
RENAME TABLE NombreViejo TO NombreNuevo;
```

b) Si queremos añadir columnas a una tabla: las columnas se añadirán al final de la tabla.

```
ALTER TABLE NombreTabla ADD
( ColumnaNueva1 Tipo_Dato [ModificadordeTipo],
ColumnaNueva2 Tipo_Dato [ModificadordeTipo],
...
ColumnaNuevaN Tipo_Dato [ModificadordeTipo]  );
```

c) Si queremos eliminar columnas de una tabla: se eliminará la columna indicada sin poder deshacer esta acción. Además, se eliminan todos los datos que contenga la columna.

```
ALTER TABLE NombreTabla DROP COLUMN (Columna1 [, Columna2, …] );
```

d) Si queremos modificar columnas de una tabla: podemos modificar el tipo de dato y las propiedades de una columna. Todos los cambios son posibles si la tabla no contiene datos.

```
ALTER TABLE NombreTabla MODIFY (Columna1 Tipo_Dato
[ModificadordeTipo] [, columna2 Tipo_Dato [ModificadordeTipo]
...] );
```

e) Si queremos renombrar columnas de una tabla:

```
ALTER TABLE NombreTabla RENAME COLUMN NombreAntiguo TO
NombreNuevo;
```

f) Podemos añadir y eliminar las siguientes restricciones de una tabla: CHECK, PRIMARY KEY, NOT NULL, FOREIGN KEY y UNIQUE. Para añadir restricciones usamos la orden:

```
ALTER TABLE nombretabla ADD CONSTRAINT nombrerestricción ...
```

Para eliminar restricciones usamos la orden:

```
ALTER TABLE nombretabla DROP CONSTRAINT nombrerestriccion ...
```

d) Eliminación de tablas.

Podemos eliminar una tabla siempre y cuando contemos con privilegios para ello. En tal caso debemos escribir:

```
DROP TABLE NombreTabla [CASCADE];
```


CASCADE elimina las restricciones de integridad referencial que remitan a la clave primaria de la tabla borrada.

e) Creación de índices.

Crear índices ayuda a la localización más rápida de la información contenida en las tablas. No es aconsejable que se utilicen índices en campos de tablas pequeñas o campos que se actualicen con mucha frecuencia.

El diseño de índices es un tema bastante complejo para los Administradores de Bases de Datos, ya que una mala elección ocasiona ineficiencia y tiempos de espera elevados. Un uso excesivo de ellos puede dejar a la Base de Datos colgada simplemente con insertar alguna fila. Para crear un índice utilizaríamos la siguiente sentencia:

```
CREATE INDEX NombreIndice ON NombreTabla (Columna1 [, Columna2 ...]);
```

f) Eliminación de índices.

Para eliminar un índice es suficiente con poner la instrucción:

```
DROP INDEX NombreIndice;
```

g) Creación de vistas.

Las vistas permiten definir subconjuntos de datos formados con una o varias tablas y/o vistas. Tiene la apariencia de una tabla, pero ocupa menos espacio que éstas ya que solo se almacena la definición de la vista y no los datos que la forman. Gracias a las vistas podemos mostrar a los usuarios una visión parcial de los datos, de modo que podamos mostrar a los usuarios sólo aquellos datos que son de su interés.

Cuando creamos una vista lo que hacemos es aplicar una consulta sobre la base de datos. La diferencia es que la vista se queda almacenada y podrá ser reutilizada en el futuro. Para crear una vista usamos:

```
CREATE VIEW NombreVista AS Subconsulta;
```

h) Eliminación de vistas.

Las vistas también podrán ser eliminadas de la base de datos. Para ello escribimos lo siguiente:

```
DROP VIEW NombreVista;
```


5. DICCIONARIO DE DATOS

También se le conoce como catálogo del sistema. Es el lugar donde se almacena la información sobre la totalidad de los datos que forman la base de datos. Se trata de una metabase de datos; es decir, una base de datos que contiene información sobre otra base de datos.

Contiene las características lógicas de las estructuras que almacenan los datos, su nombre, descripción, contenido y organización.

Además, en él están contenidas, entre otras cosas, las restricciones de privacidad y acceso a los datos que han sido definidas en DDL y/o DCL.

Concretamente el diccionario de datos contiene:

- Descripción de los esquemas interno, conceptual y externo.

- Las restricciones de privacidad y acceso a los datos almacenados en la BD.
- Las reglas, normas o restricciones referentes a la seguridad de los datos.
- Programas de aplicación que utilizan.
- Las operaciones que realizan los usuarios en el sistema.

Los principales **objetivos** del diccionario de datos y razones por las cuales el diccionario de datos es importante son las siguientes:

- Manejar los detalles en sistemas muy grandes ya que es imposible de recordar todo lo referente a un sistema.
- Proporcionar una vista universal de cada elemento, asegurando que solamente haya un significado posible para cada dato.
- Documentar las características del sistema.
- Localizar errores en el sistema.
- Facilidades de análisis para determinar si son necesarias nuevas características o si están en orden los cambios de cualquier tipo.

6. CONCLUSIÓN

Un SGBD desarrolla tres funciones fundamentales como son las funciones de definición, manipulación y control de los datos. En este tema se ha presentado una visión global de la definición de datos. También se han descrito los diferentes niveles de descripción, los cuales fueron propuestos por el comité ANSI-SPARC con el objetivo principal de separar los programas de aplicación de la base de datos física. Además, se ha descrito el lenguaje de definición de datos (DDL), el cual se encarga de llevar a cabo las tareas de definición de las estructuras que almacenarán los datos y finalmente se han detallado los principales datos que contiene un diccionario de datos, así como sus principales objetivos.

7. BIBLIOGRAFÍA

- Date D.J.: **Introducción a los sistemas de bases de datos**. Editorial Addison-Wesley
- De Miguel A,y Piattini M:**Fundamentos y modelos de BBDD**. Edit. Ra-Ma
- Korth H. y Silberschatz: **Fundamentos de bases de datos**. Editorial McGraw-Hill