DISEÑO FÍSICO DE DATOS EN SQL

2C.1.- INTRODUCCIÓN Y OBJETIVOS

El diseño físico es el proceso de producir la descripción de la implementación de la base de datos en memoria secundaria, a partir del esquema lógico obtenido en la etapa anterior. Para especificar dicha implementación se debe determinar las estructuras de almacenamiento y escoger los mecanismos necesarios para garantizar un acceso eficiente a los datos. Puesto que el esquema lógico utiliza el modelo relacional, la implementación del diseño físico se realizará en SQL.

Al finalizar este capítulo se podrá:

- Traducir el esquema lógico de una base de datos dada a un conjunto de sentencias SQL de creación de tablas que la implementen fielmente.
- Acudir a los manuales del SGBD escogido para la implementación y obtener en ellos toda la información necesaria para llevar a cabo la implementación sobre el mismo (sintaxis del lenguaje, los tipos de datos, etc.).
- Escoger las organizaciones de ficheros más apropiadas en función de las que tenga disponible el SGBD que se vaya a utilizar.
- Decidir qué índices deben crearse con el objetivo de aumentar las prestaciones en el acceso a los datos.
- Diseñar las vistas necesarias para proporcionar seguridad y facilitar el manejo de la base de datos.

2C.2.- METODOLOGÍA DE DISEÑO

Mientras que en el diseño lógico se especifica qué se guarda, en el diseño físico se especifica cómo se guarda.
Para llevar a cabo esta etapa se debe haber decidido cuál es el SGBD que se va a utilizar, ya que el esquema físico se adapta a él. El diseñador debe conocer muy bien toda la funcionalidad del SGBD concreto y también el sistema informático sobre el que éste va a trabajar.
El diseño físico no es una etapa aislada, ya que algunas decisiones que se tomen durante su desarrollo, por ejemplo para mejorar las prestaciones, pueden provocar una reestructuración del esquema lógico. De este modo, entre el diseño físico y el diseño lógico hay una realimentación.
En general, el propósito del diseño físico es describir cómo se va a implementar físicamente el esquema lógico obtenido en la fase anterior. Concretamente, en el modelo relacional, esto consiste en:

- Obtener un conjunto de sentencias para crear las tablas de la base de datos y para mantener las restricciones que se deben cumplir sobre ellas.
- Determinar las estructuras de almacenamiento y los métodos de acceso que se van a utilizar para conseguir unas prestaciones óptimas.
- Diseñar el modelo de seguridad del sistema.

En los siguientes apartados se detallan cada una de las etapas que componen la fase del diseño físico.

2C.2.1.- TRADUCIR EL ESQUEMA LÓGICO

La primera fase del diseño físico consiste en traducir el esquema lógico a un esquema (físico) que se pueda implementar en el SGBD escogido. Para ello, es necesario conocer toda la funcionalidad que éste ofrece.

Sentencias de creación de las tablas

Las tablas se definen mediante el lenguaje de definición de datos del SGBD.
Para ello, se utiliza la información producida durante el diseño lógico: el esquema lógico y toda la documentación asociada (diccionario de datos). El esquema físico consta de un conjunto de tablas y, para cada una de ellas, se especifica:

- El *nombre.* Es conveniente adoptar unas reglas para nombrar las tablas, de manera que aporten información sobre el tipo de contenido. Por ejemplo, a las tablas de referencia se les puede añadir el prefijo o el sufijo REF, a las tablas que almacenan información de auditoría ponerles el prefijo/ sufijo AUDIT, a las tablas que sean de uso para un solo departamento, ponerles como prefijo/sufijo las siglas del mismo, etc.

- La *lista de columnas* con sus nombres. De nuevo resulta conveniente adoptar una serie de reglas para nombrarlas. Algunas reglas habituales son: poner el sufijo PK a las claves primarias (PRIMARY KEY), poner el sufijo FK a las claves ajenas (FOREIGN KEY), usar el nombre de la clave primaria a la que se apunta en el nombre de una clave ajena o el nombre de su tabla, usar el mismo nombre para las columnas que almacenan el mismo tipo de información (por ejemplo, si en varias tablas se guarda una columna con la fecha en que se ha insertado cada fila, usar en todas ellas el mismo nombre para dicha columna), etc. Además, para cada columna se debe especificar:

    - Su *dominio*: tipo de datos, longitud y restricciones de dominio (se especifican con la cláusula CHECK).
    - El *valor por defecto*, que es opcional (DEFAULT).
    - Si admite *nulos* o no (NULL/NOT NULL).

- La *clave primaria* (PRIMARY KEY), las *claves alternativas* (UNIQUE) y las *claves ajenas* (FOREIGN KEY), si las tiene.
- Las *reglas de comportamiento* de las claves ajenas (ON UPDATE, ON DELETE).

A continuación, se muestra un ejemplo de la creación de las tablas FACTURAS y LINEAS_FAC utilizando la especificación de SQL:

```
CREATE TABLE facturas (
codfac NUMERIC(6,0) NOT NULL,
fecha DATE NOT NULL,
codcli NUMERIC(5,0),
codven NUMERIC(5,0),
iva NUMERIC(2,0),
dto NUMERIC(2,0),
CONSTRAINT cp_facturas PRIMARY KEY (codfac),
CONSTRAINT ca_fac_cli FOREIGN KEY (codcli)
REFERENCES clientes(codcli)
ON UPDATE CASCADE ON DELETE SET NULL,CONSTRAINT ca_fac_ven FOREIGN
KEY (codven) REFERENCES vendedores(codven)
ON UPDATE CASCADE ON DELETE SET NULL,
CONSTRAINT ri_dto_fac CHECK (dto BETWEEN 0 AND 50)
);
```

```
CREATE TABLE lineas_fac  (
codfac NUMERIC(6,0) NOT NULL,
linea NUMERIC(2,0) NOT NULL,
cant NUMERIC(5,0) NOT NULL,
codart VARCHAR(8) NOT NULL,
precio NUMERIC(6,2) NOT NULL,
dto NUMERIC(2,0),
CONSTRAINT cp_lineas_fac PRIMARY KEY (codfac, linea),
CONSTRAINT ca_lin_fac FOREIGN KEY (codfac)
REFERENCES facturas(codfac)
ON UPDATE CASCADE ON DELETE CASCADE,
CONSTRAINT ca_lin_art FOREIGN KEY (codart)
REFERENCES articulos(codart)
ON UPDATE CASCADE ON DELETE RESTRICT,
CONSTRAINT ri_dto_lin CHECK (dto BETWEEN 0 AND 50)
);
```

Mantenimiento de restricciones y reglas de negocio

Las actualizaciones que se realizan sobre las tablas de la base de datos deben observar ciertas restricciones o producir determinadas consecuencias que imponen las reglas de funcionamiento de la empresa. Algunos SGBD proporcionan mecanismos que permiten definir restricciones y reglas, y vigilan su cumplimiento.

Un mecanismo para definir restricciones es la cláusula CONSTRAINT ... CHECK. Un ejemplo de ella se puede observar en las sentencias de creación de las tablas FACTURAS y LINEAS_FAC, sobre la columna dto. Otro mecanismo son los *disparadores* (TRIGGER), que también se utilizan para establecer reglas de negocio en las que se requiere la realización de alguna acción como consecuencia de algún evento.
Hay algunas reglas que no las pueden manejar todos los SGBD, como por ejemplo «a las 20:30 del último día laborable de cada año archivar los pedidos servidos y borrarlos». Para algunas reglas habrá que escribir programas de aplicación específicos. Por otro lado, hay SGBD que no permiten la definición de reglas, por lo que éstas deberán incluirse en los programas de aplicación.
Todas las reglas que se definan deben estar documentadas. Si hay varias opciones posibles para implementarlas, hay que explicar por qué se ha escogido la opción implementada.

2C.2.2.- DISEÑAR LA REPRESENTACIÓN FÍSICA

Uno de los objetivos principales del diseño físico es almacenar los datos de modo eficiente. Para medir la eficiencia hay varios factores que se debe tener en cuenta:

- *Rendimiento de transacciones.* Es el número de transacciones que se quiere procesar en un intervalo de tiempo.
- *Tiempo de respuesta.* Es el tiempo que tarda en ejecutarse una transacción. Desde el punto de vista del usuario, este tiempo debería ser el mínimo posible.
- *Espacio en disco.* Es la cantidad de espacio en disco que hace falta para los ficheros de la base de datos. Normalmente, el diseñador querrá minimizar este espacio.

Lo que suele suceder es que todos estos factores no se pueden satisfacer a la vez. Por ejemplo, para conseguir un tiempo de respuesta mínimo puede ser necesario aumentar la cantidad de datos almacenados, ocupando más espacio en disco. Por lo tanto, el diseñador deberá ir ajustando estos factores para conseguir un equilibrio razonable. El diseño físico inicial no será el definitivo, sino que habrá que ir monitorizándolo para observar sus prestaciones e ir ajustándolo como sea oportuno. Muchos SGBD proporcionan herramientas para monitorizar y afinar el sistema.

Analizar las transacciones

Para realizar un buen diseño físico es necesario conocer las consultas y las transacciones que se van a ejecutar sobre la base de datos. Esto incluye tanto información cualitativa, como cuantitativa. Para cada transacción, hay que especificar:

- La frecuencia con que se va a ejecutar.
- Las tablas y los atributos a los que accede la transacción, y el tipo de acceso: consulta, inserción, modificación o eliminación. Por ejemplo, los atributos que se modifican a menudo no son buenos candidatos para construir índices.
- Las restricciones temporales impuestas sobre la transacción. Los atributos utilizados en los predicados de la transacción pueden ser candidatos para construir estructuras de acceso.

Escoger las organizaciones de ficheros

El objetivo de este paso es escoger la organización de ficheros óptima para cada tabla. Por ejemplo, un fichero desordenado es una buena estructura cuando se va a cargar gran cantidad de datos en una tabla al inicializarla, cuando la tabla tiene pocas filas, también cuando en cada acceso se deben obtener todas las filas de la tabla, o cuando la tabla tiene una estructura de acceso adicional, como puede ser un índice.

Por otra parte, los ficheros dispersos (*hashing*) son apropiados cuando se accede a las filas a través de los valores exactos de alguno de sus campos (condición de igualdad en el WHERE). Si la condición de búsqueda es distinta de la igualdad (búsqueda por rango, por patrón, etc.), entonces la dispersión no es una buena opción.

Algunos SGBD proporcionan otras organizaciones alternativas a éstas. Las organizaciones de ficheros elegidas deben documentarse, justificando en cada caso la opción escogida.

Escoger los índices a crear y sus tipos

Los índices son estructuras adicionales que se utilizan para acelerar el acceso a las tablas en respuesta a ciertas condiciones de búsqueda. Hay que tener en cuenta que los índices conllevan un coste de mantenimiento que es necesario sopesar frente a la ganancia en prestaciones.

Cada SGBD proporcionará uno o varios tipos de índices entre los que escoger.

A la hora de seleccionar los índices a crear, se pueden seguir las siguientes indicaciones:

- Crear un índice sobre la clave primaria de cada tabla.
- La mayor parte de los SGBD relacionales crean un índice único de manera automática sobre la clave primaria de cada tabla porque es el mecanismo que utilizan para mantener la unicidad.
- No crear índices sobre tablas pequeñas. Si el SGBD ha creado índices automáticamente sobre este tipo de tablas, se pueden eliminar (DROP INDEX). Aquí conviene tener en cuenta que, en la mayor parte de los SGBD, no se permite eliminar un índice creado sobre una clave primaria a la que apunta una clave ajena, ya que este índice se utiliza para mantener la integridad referencial.
- Crear un índice sobre las claves ajenas que se utilicen con frecuencia en operaciones de JOIN.
- Crear un índice sobre los atributos que se utilizan con frecuencia para hacer restricciones WHERE (son condiciones de búsqueda).
- Crear un índice único sobre las claves alternativas que se utilizan para hacer búsquedas. Al igual que ocurre con las claves primarias, los SGBD suelen mantener la unicidad de las claves alternativas mediante un índice único que crean automáticamente.
- Evitar los índices sobre atributos que se modifican a menudo.
- Evitar los índices sobre atributos poco selectivos: aquellos en los que la consulta selecciona una porción significativa de la tabla (más del 15 % de las filas).
- Evitar los índices sobre atributos formados por tiras de caracteres largas.
- Evitar los índices sobre tablas que se actualizan mucho y que se consultan muy esporádicamente (tablas de auditoría o diarios). Si se han creado índices sobre este tipo de tablas, podría ser aconsejable eliminarlos.
- Revisar si hay índices redundantes o que se solapan y eliminar los que no sean necesarios.

2C.2.3.- DISEÑAR LOS MECANISMOS DE SEGURIDAD

Los datos constituyen un recurso esencial para la empresa, por lo tanto su seguridad es de vital importancia. Durante el diseño lógico se habrán especificado los requerimientos en cuanto a seguridad que en esta fase se deben implementar. Para llevar a cabo esta implementación, el diseñador debe conocer las posibilidades que ofrece el SGBD que se vaya a utilizar.

Diseñar las vistas de los usuarios

El objetivo de este paso es diseñar las vistas o esquemas externos de los usuarios, correspondientes a los esquemas lógicos de cada grupo de usuarios.
Cada esquema externo estará formado por tablas y vistas (VIEW) de SQL. Las vistas, además de preservar la seguridad, mejoran la independencia de datos, reducen la complejidad y permiten que los usuarios vean los datos en el formato deseado.

Diseñar las reglas de acceso

El administrador de la base de datos asigna a cada usuario un identificador que tendrá una contraseña asociada por motivos de seguridad. Para cada usuario o grupo de usuarios se otorgarán privilegios para realizar determinadas acciones sobre determinados objetos de la base de datos. Por ejemplo, los usuarios de un determinado grupo pueden tener permiso para consultar los datos de una tabla concreta, y no tener permiso para actualizarlos.