BDI Tema 7. SQL: otros elementos.

Martín González Dios

30 de octubre de 2024

1. Transacciones

1.1. Definiciones importantes

- Transacción: acción o serie de acciones, llevadas a cabo por un usuario o por una aplicación, que leen y/o actualizan el contenido de la BD.
- Atomicidad: propiedad de una transacción que garantiza su indivisibilidad (o bien se realizan todas las acciones en su totalidad o bien la BD vuelve a su estado inicial como si no se realizara nada).

Los gestores de bases de datos funcionan, por defecto, realizando un COMMIT de confirmación después de cada instrucción (se denomina autocommit). Asumen, por tanto, que cada tarea lógica está formada por una única instrucción.
Commit work (compromiso): compromete la transacción actual; es decir, hace que las actualizaciones realizadas por la transacción pasen a ser permanentes en la BD.
Rollback work (retroceso): provoca el retroceso de la transacción actual; es decir, deshace todas las actualizaciones realizadas por las sentencias de la transacción.
Una vez que la transacción ha ejecutado commit work, sus efectos ya no se pueden deshacer con rollback work.

1.2. Violación de la restricción de integridad durante una transacción

Existe la cláusula initially deferred para especificar la restricción, la cual comprueba al final de la transacción y no en pasos intermedios si hay violación de la restricción de integridad.
Las restricciones pueden especificarse de manera alternativa con deferrable, de manera predeterminada se comprueba inmediatamente, pero puede diferirse si se desea. Para las restricciones declaradas como diferibles, la ejecución de la instrucción set contrains lista-resticciones deferred como parte de una transacción hace que se difiera la comprobación de las restricciones especificadas hasta el final de esa transacción.

2. Tipos de datos y esquemas de SQL

2.1. Tipos fecha y hora en SQL

- **date**. Fecha del calendario que contiene el año (con cuatro dígitos), el mes y el día del mes. ´2001-04-25´
- **time**. La hora del día, en horas, minutos y segundos. También es posible almacenar la información del huso horario junto a la hora especificando time with timezone. ´09:30:00´
- **timestamp**. Una combinación de date y time. ambién se almacena información sobre el huso horario si se especifica with timezone. ´2001-04-25 10:29:01.45´

Se pueden usar expresiones de la forma cast e as t para conver- tir una cadena de caracteres (o una expresión de tipo cadena de caracteres) e al tipo t, donde t es de tipo date, time o timestamp.

SQL permite realizar operaciones de comparación sobre todos los tipos de datos que se han mencionado, así como operaciones aritméticas y de comparación sobre los diferentes tipos de datos numéricos. SQL también proporciona un tipo de datos denominado **interval** y permite realizar cálculos basados en fechas, horas e intervalos.

2.2. Valores predeterminados

create table estudiante
(ID varchar (5),
nombre varchar (20) not null,
nombredept varchar (20),
totcréditos numeric (3,0) **default** 0,
primary key (ID));

2.3. Índices

Un índice sobre un atributo de una relación es una estructura de datos que permite al sistema de bases de datos encontrar de forma eficiente aquellas tuplas de la relación que tienen un determinado valor del atributo, sin tener que explorar todas las tuplas de la re- lación.
create **index** estudianteIDíndice **on** estudiante(ID);

2.4. Tipos de datos para objetos grandes

Tipos de datos para objetos de gran tamaño para los datos de caracteres (**clob**) y para los datos binarios (**blob**). Las letras ≪lob≫ de estos tipos de datos significan ≪objeto grande≫ (large object). Por ejemplo, se pueden declarar los atributos: revista clob(10KB) imagen blob(10MB) película blob(2GB)

2.5. Tipos definidos por los usuarios

Hay 2 formas; la primera se denomina **alias de tipos** (distinct types). La otra forma, denominada **tipos de datos estructurados**, permite la creación de tipos de datos complejos.
La cláusula create type puede utilizarse para definir tipos de datos nuevos. Por ejemplo, las sentencias:
create type Euros as numeric(12,2) final;
create type Libras as numeric(12,2) final;
Como consecuencia del control riguroso de los tipos de datos, la expresión (department.budget+20) no se aceptaría, ya que el atributo y la constante entera 20 tienen diferentes tipos de datos. Los valores de un tipo de datos pueden convertirse (cast) a otro dominio como se muestra a continuación:
cast (departamento.presupuesto to numeric(12,2))

Dominio: capaz de añadir restricciones de integridad a un tipo subyacente. Por ejemplo, se podría definir el tipo de dominio DEuros de la manera siguiente:
create domain DEuros as numeric(12,2) not null;

Existen 2 diferencias entre los tipos de datos y los dominios:

- Sobre los dominios se pueden especificar restricciones, como not null, y valores predeterminados para las variables del tipo de dominio, pero no se pueden especificar restricciones ni va- lores predeterminados sobre los tipos de datos definidos por los usuarios.
- Los dominios no tienen tipos estrictos. En consecuencia, los va- lores de un tipo de dominio pueden asignarse a los de otro tipo de dominio, siempre y cuando los tipos subyacentes sean com- patibles.

La cláusula **check** permite al diseñador del esquema especificar un predicado que todos los atributos declarados deben satisfacer en este dominio.

create domain SueldoAnual numeric(8,2)
constraint valorSueldoTest
check(value >= 29000.00);

La cláusula constraint valorSueldoTest es opcional, y se usa para dar el nombre valorSueldoTest a la restricción.

2.6. Extensiones a la creación de tablas

Extensión create table like para esta tarea:
create table tempProfesor like profesor;

Si se omite la cláusula **with data**, la tabla se crea pero no se rellena con datos.
La sentencia anterior **create table... as** se parece mucho a la sentencia **create view** y ambas se definen mediante consultas. La diferencia más importante es que el contenido de la tabla se inserta cuando se crea, mientras que el contenido de una vista siempre refleja el resultado actualizado de la consulta.

2.7. Esquemas, catálogos y entornos

Los sistemas de bases de datos ofrecen una jerarquía de tres niveles para los nombres de las relaciones. El nivel superior de la jerarquía consta de **catálogos**, cada uno de los cuales puede contener **esquemas**. Los objetos de SQL, como las **relaciones** y las vistas, están contenidos en esquemas.

3. Autorización

La autorización en bases de datos permite asignar distintos privilegios a los usuarios para interactuar con partes específicas de la base de datos. Estos **privilegios** incluyen lectura, inserción, actualización y borrado de datos.
La norma de SQL incluye los privilegios **select**, **insert**, **update** y **delete**. El privilegio **all privileges** se puede utilizar como forma abreviada de todos los privilegios que se pueden conceder.
La instrucción grant se utiliza para conceder autorizaciones. La forma básica de esta sentencia es:
grant ¡lista de privilegios¿
on ¡nombre de relación o de vista¿
to ¡lista de usuarios o de roles¿;

La autorización **select** sobre una relación es necesaria para leer las tuplas de una relación.
La autorización **update** sobre una relación permite a los usuarios actualizar cualquier tupla de la relación.
La autorización **insert** sobre una relación permite a los usuarios insertar tuplas en la relación.
La autorización **delete** sobre una relación permite a los usuarios borrar tuplas de una relación.
El nombre de usuario **public** hace referencia a todos los usuarios actuales y futuros del sistema. Por tanto, los privilegios concedidos a public se garantizan de manera implícita a todos los usuarios actuales y futuros.
Para revocar una autorización se emplea la sentencia revoke. Su forma es casi idéntica a la de grant:
revoke <lista de privilegios>
on <nombre de la relación o nombre de la vista>
from <lista de usuarios o de roles>;

3.1. Roles

El concepto de **roles** permite asignar permisos específicos a grupos de usuarios en función de sus responsabilidades, facilitando la gestión de autorizaciones.
Los permisos se asignan a roles con la instrucción grant, como si fueran usuarios, y los roles pueden asignarse a otros roles, creando una jerarquía de permisos.

3.2. Autorización sobre vistas

Permite restringir el acceso a datos específicos en la base de datos. Por ejemplo, se puede crear una vista profesorGeología que muestre solo información del departamento de Geología para los miembros de la dirección, sin darles acceso a toda la tabla de profesor. El sistema verifica que el usuario tenga los permisos necesarios en la tabla original.

3.3. Transferencia de privilegios

Permite a un usuario que recibe una autorización otorgarla a otros usuarios, pero esto solo es posible si el privilegio se concede con la opción **with grant option**. Por ejemplo, el privilegio select se puede otorgar a Amit en la tabla departamento con esta opción para que Amit pueda concederlo a otros.

grant select on departamento to Amit **with grant option**;

El administrador de la base de datos, como creador del objeto, conserva siempre todos los permisos, incluidos los de delegación. Las transferencias de autorizaciones pueden representarse mediante un grafo (**grafo de autorización**), donde cada nodo es un usuario y una arista indica que un usuario ha concedido permisos a otro. Un usuario tiene un permiso si existe un camino en el grafo desde el administrador hasta él.

3.4. Revocación de privilegios

Permite retirar permisos concedidos a un usuario. Cuando se revoca un permiso, puede ocurrir una **revocación en cascada**, que elimina los permisos de otros usuarios si dependían del permiso inicial revocado. Este es el comportamiento por defecto, aunque se puede usar **restrict** para evitarlo y cancelar la revocación si se afecta a otros usuarios.

SQL también permite revocar solo la capacidad de otorgar permisos, sin quitar el permiso en sí, usando **revoke grant option for**. Para casos complejos, como cuando roles jerárquicos otorgan privilegios, SQL permite el uso de **granted by current_role**, para asegurar que un rol intermedio (como decano) no pierda permisos clave al revocar un rol específico.