Curso Introducción a SQL Espacial sobre PostGIS

Structured Query Language (SQL)

Comandos SQL de estructura

formaSIG

La plataforma de aprendizaje en SIG Libre

sigte SERVEI DE SISTEMES D'INFORMACIÓ GEOGRÀFICA I TELEDETECCIÓ Universitat de Girona

UdGFormació FUNDACIÓ UNIVERSITAT DE GIRONA: INNOVACIÓ I FORMACIÓ

Edita: Servicio de SIG y Teledetección (SIGTE) de la Universitat de Girona

Año: 2014

Contenidos elaborados por: Toni Hernández Vallès

Comandos SQL de definición de estructura

Los llamados comandos SQL de estructura permiten crear, modificar y eliminar el esqueleto de nuestras bases de datos que dará cabida a nuestros datos. La inserción y manejo de esos datos está en manos de otro tipo de comandos que veremos más adelante.

La estructura de una base de datos comprende la definición de tablas, columnas, tipos de datos, relaciones entre tablas, índices, etc. Aunque el uso de los comandos de estructura es muy importante para gestionar bases de datos, en este curso vamos a centrar nuestro interés en la explotación que podemos hacer de nuestras bases de datos y no tanto en la optimización de su estructura.

CREATE TABLE

Este es el comando que vamos a utilizar para crear nuevas tablas en nuestras bases de datos.

Sintaxis:

```
CREATE TABLE < nombre_tabla> (
<nombre_columna> <tipo_de_dato> [NOT NULL]
[, <nombre_columna> <tipo_de_dato> [NOT NULL]...]
);
```

Para describir la sintaxis de todos los comandos SQL hemos optado por las siguientes convenciones:

• Los comandos y palabras clave se muestran en MAYÚSCULAS.

• < > (menor que y mayor que) significa que debemos incluir el elemento que está dentro de estos dos símbolos. Así, en el ejemplo anterior debe haber, obligatoriamente, un nombre de tabla directamente después del comando y, además, también debe haber al menos una columna definida en la tabla.

• [ ] (Corchetes) indican que puede (o no) haber información dentro de los corchetes, es decir, los corchetes encierran partes opcionales del comando. En el ejemplo anterior, vemos que la palabra clave NOT NULL es opcional y que la descripción de la segunda columna (pero no la primera) es opcional.

• ... (puntos suspensivos) indican que esta parte del comando puede ser repetida tantas veces como sea necesario. En el ejemplo anterior vemos que la descripción de columna puede ser repetida tantas veces como queramos.

Como vemos las expresiones SQL pueden escribirse en diferentes líneas ya que sólo el punto y coma final (;) determina el final de la expresión. Podéis aprovechar esto para facilitar la lectura del comando en expresiones SQL complejas.

Veamos un ejemplo.

```
CREATE TABLE vendedor
(
vendedorId INT NOT NULL,
nombre CHARACTER VARYING(20) NOT NULL,
primer_apellido CHARACTER VARYING(20) NOT NULL,
segundo_apellido CHARACTER VARYING(20) NOT NULL,
fecha_nacimiento DATE
);
```

El comando anterior creará una tabla *vendedor* con cinco columnas. vendedorId de tipo entero (int), nombre, primer_apellido y segundo_apellido de tipo alfanumérico (hasta 20 caracteres, ninguno más), y fecha_nacimiento de tipo fecha. El tipo de dato definido en cada columna limita los valores que podemos insertar en cada caso. Por ejemplo, en la columna vendedorId solo podremos introducir valores numéricos enteros y en la columna fecha_nacimiento solo podremos introducir fechas.

La palabra reservada NOT NULL sirve para indicar que la columna vendedorId no puede aceptar valores nulos. Un valor nulo es un valor desconocido y hay que distinguirlo de un valor vacío. Por ejemplo las columnas primer_apellido y segundo_apellido aceptarán valores vacíos pero no valores nulos, es decir, si un vendedor no tiene segundo apellido entonces ese campo deberá contener un valor vacío, pero no nulo. En cambio la columna fecha_nacimiento sí acepta valores nulos, por lo que si desconocemos la fecha de nacimiento de alguno de los vendedores podremos insertar el valor nulo.

El comando CREATE TABLE crea la cabecera de la tabla, es decir, una tabla vacía en la que se han definido los campos y en la que luego podremos insertar o importar los datos.

Cada definición de campo debe tener obligatoriamente, al menos, dos detalles: el nombre de dicho campo y el tipo de datos que alberga. El nombre del campo puede ser cualquier cadena de texto que deseemos, aunque cada SGBD puede imponer algunas restricciones. Un ejemplo de restricción puede ser que no contenga espacios, acentos o que dicha cadena de texto sea limitada a ambos lados por “ (comillas) o por otro carácter clave.

El tipo de dato, por el contrario, no puede ser una cadena de texto de nuestra invención sino que debe hacer referencia a una de las palabras clave reservadas a este efecto. En el ejemplo

precedente hemos visto tres tipos de datos distintos (números enteros, cadenas de caracteres y fechas) pero existen muchos tipos de datos distintos, algunos comunes entre todos los SGBD y otros más particulares. Para más detalles sobre los tipos de datos puedes consultar en la lectura **Tipos de datos** de este tema.

ALTER TABLE

Utilizaremos el comando ALTER TABLE para modificar la estructura de una tabla que ya existe. Podemos añadir columnas (especificicando el tipo de datos que va a contener esa nueva columna), eliminarlas o modificarlas.

Sintaxis: Para añadir una columna:

ALTER TABLE < nombre_tabla>
ADD [COLUMN] (<nombre_columna> <tipo_de_dato> [**NOT NULL**]
[, <nombre_columna> <tipo_de_dato> [**NOT NULL**]...]);

Para añadir la columna *num_telefono,* de tipo alfanumérico, a la tabla vendedor utilizaremos el siguiente comando:

ALTER TABLE vendedor
ADD COLUMN (num_telefono **CHARACTER VARYING(20)**);

Sintaxis: Para borrar una columna:

ALTER TABLE <nombre_tabla>
DROP [**COLUMN**] (<nombre_columna>[, <nombre_columna>...]);

Para borrar la columna que hemos creado anteriormente:

ALTER TABLE vendedor
DROP COLUMN num_telefono;

La sintaxis para modificar una columna depende del tipo de modificación que se quiera llevar a cabo. En la documentación oficial de PostgreSQL se encuentran todas las posibles modificaciones. http://www.postgresql.org/docs/9.3/static/sql-altertable.html

DROP TABLE

El comando DROP TABLE se utiliza para eliminar una tabla de nuestra base de datos. Es importante tener en cuenta que el resultado del comando DROP TABLE es irreversible. Una tabla eliminada no puede ser restaurada a menos que dispongamos de una copia de seguridad.

Sintaxis:

DROP TABLE <nombre_tabla>;

sigte
SERVEI DE SISTEMES D'INFORMACIÓ GEOGRÀFICA I TELEDETECCIÓ
Universitat de Girona

UdGFormació
FUNDACIÓ UNIVERSITAT DE GIRONA: INNOVACIÓ I FORMACIÓ

www.sigte.udg.edu/formasig

formasig@sigte.org