SEP
7

SPRINT 3 - MEETING 49

Base de datos relacionales III

Grabación

Presentación

La información cobra sentido cuando la cruzamos

> *"Lo que necesitamos es entender el significado de los números, el contexto y luego de eso comprender todo junto, los números y la big data de la calidad de la información." — Tricia Wang.* **Etnógrafa tecnológica global.***

Ya conoces cómo manipular datos dentro de cada tabla individualmente. Pero las bases de datos cobran mucho más sentido cuando comienzan a realizarse **relaciones.** ¡Es momento de cruzar información! En esta bitácora veremos cómo relacionar información de diferentes tablas en una misma instrucción SQL.

Las **relaciones de bases de datos son asociaciones entre tablas** que contienen información. Existen varios tipos de relaciones dentro de las bases de datos: **uno a uno, uno a muchos** y **muchos a muchos**.

1. Uno a uno.

Por cada registro de la tabla principal puede existir un solo registro en la tabla relacionada (tabla que contiene la clave externa). Por ejemplo, observa las siguientes dos tablas, una de estudiante (Id, Nombre y Email) y otra de matrícula (Id, Nro de Matrícula e Id Persona):

Hacemos que un elemento de la tabla estudiante se relacione con un elemento de la tabla Matricula.

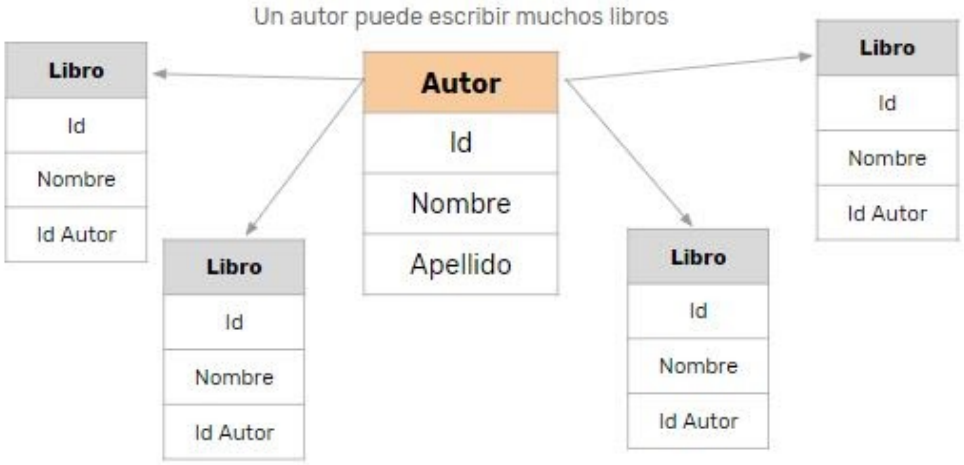
Esta relación en particular no es de las más utilizadas, ya que se puede simplificar simplemente si agregamos los campos de la tabla 2 a la tabla 1 para que contemple esos datos.

Retomando el ejemplo anterior:

2. Uno a muchos.

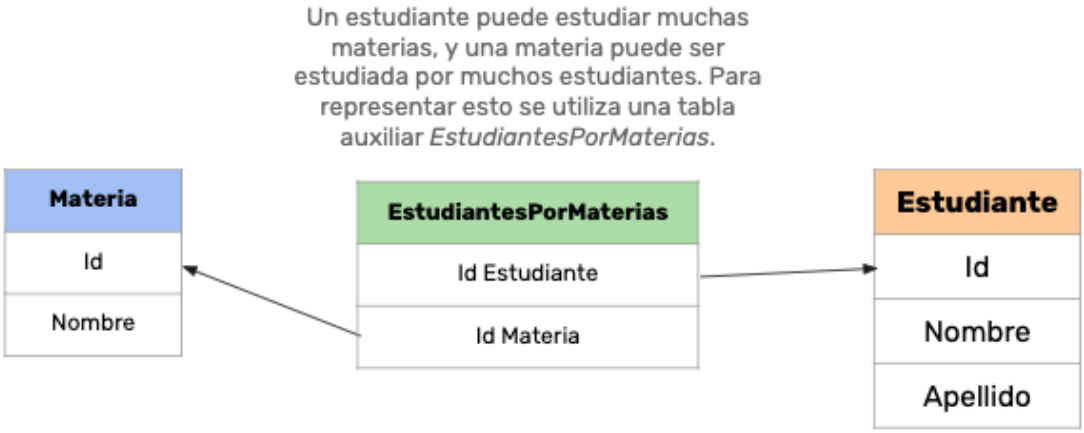
En esta relación, por cada registro de la tabla principal (tabla de la clave principal o lado uno de la

externa o lado infinito de la relación).



3. **Muchos a Muchos.**

Este caso sucede cuando un registro de una tabla está relacionado con más de un registro de la otra tabla y viceversa. Por ejemplo:



Comenzar a relacionar la información

El **diagrama de entidad de relación (DER)** es una herramienta muy útil para la planificación de la estructura de una base de datos. Como su nombre lo indica, vamos a tener las **entidades** y las **relaciones** entre ellas. Por ejemplo si tenemos tres entidades:

- Película

- Actores/Actrices

Las representamos de la siguiente manera:

En el diagrama que dibujamos en lápiz y papel podemos ver que película esta relacionado con géneros y actores/actrices.

Sabemos que una película tiene un solo género pero puede tener uno o muchos actores/actrices.Para la relación de género nos alcanza con tener un campo en nuestra tabla de películas llamado “genero_id” que hará referencia a campo id de la tabla géneros.

Ahora bien, para hacer la relación entre películas y sus actores no podemos utilizar la misma lógica debido a que no sabremos cuántos actores/actrices puede haber en una película.

Para esto podemos crear una tabla intermedia entre ambas entendidas que nos permita almacenar infinitos “actores/actrices”.

Esta tabla pongamos el nombre de las 2 entidades separadas por un guión bajo y estará compuesta por 2 campos: película*id, actor*id.

Ahora, que sabemos cómo indicar las relaciones, definimos las tablas que necesitamos.

Procesar CSV y Bulk insert a DB

A la hora de insertar datos en nuestra base de datos, tenemos la opción de ingresar los registros uno por uno pero qué pasa si es necesario procesar un gran volumen de datos para poder satisfacer un requerimiento ¿Qué hacemos ante esta situación? Para ello se utilizan los Bulk Insert, **proceso donde se insertan numerosos registros, generalmente cargados desde un archivo.**

Entonces supongamos que ya terminamos de desarrollar e implementar un sistema que debe almacenar grandes cantidades de productos. Sin embargo, hay algo importante que nos está faltando: data. Probablemente creamos un par de registros de prueba para poder testear las consultas a la BD, pero si queremos pasar a producción vamos a necesitar llenar las tablas con datos reales. Existen varias posibilidades una de ellas que es sentarnos varias horas delante de una consola SQL para ingresar y validar cada uno de los registros, que no sería la más divertida pero tenemos otra opción que es crear un panel de administración al que pueda acceder el cliente para cargar los productos uno a uno. ¡Pero existe una forma mucho más eficiente y poderosa para hacer esto!

Es muy común que para casos de grandes cantidades de elementos, el cliente posea algún tipo de archivo con todos los datos necesarios. Teniendo esto en cuenta, podemos crear un panel de administración que posea un input para subir un archivo con esta información y nosotros procesarlo y hacer los INSERTs en la base de datos de forma automática.

Un estándar bastante común es el formato **CSV (Coma Separated Values)** o su primo el **TSV (Tab Separated Values).** Estos archivos ofrecen una matriz de dos dimensiones que podría verse de la siguiente forma:

```
nombre, precio, precio_descuento, disponible
Hamburguesa Doble Quotes, 250, 199, true
Hamburguesa Singleton, 190, 180, true
Papas Fritas, 150, 150, true
Cerveza Tirada, 90, 90, false
```

Podemos identificar cada fila separada por un salto de línea \n como un registro y cada columna o propiedad está separada por , . Podemos hacer split sobre ese gran **string** y obtener un **array** fácilmente procesable.

El desafío de esta actividad es crear un algoritmo que procese un CSV con un listado de 1000 personas y las ingrese en la DB. Una vez creado el algoritmo base, un desafío adicional puede ser calcular la edad de la persona en base a su fecha de nacimiento y guardarla en una nueva columna. Puntos extra si agregamos validaciones a nuestro algoritmo de forma tal que no permita insertar personas sin apellido o que la fecha de nacimiento sea previa al 1800.

¡Prepárate para el próximo encuentro!

Profundiza

Te invitamos a conocer más sobre el tema de esta bitácora.

Herramientas

Programas necesarios para facilitar tu experiencia

Challenge

Te proponemos el siguiente desafío, ¿te animas?

MEETING 48

MEETING 50