Migraciones de bases de datos

Para poder trabajar con una base de datos en Laravel lo primero que tenemos que editar es el archivo .env y configurar los siguientes campos:

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=dwes
DB_USERNAME=super
DB_PASSWORD=123456

La base de datos debe estar creada, en caso de que no exista la aplicación no la creará y obtendremos los correspondientes errores cada vez que tratemos de acceder a ella.
La migración de la base de datos nos permitirá crear las tablas sobre las que trabajaremos en la aplicación. Esto nos permitirá estructurar nuestra base de datos desde la misma aplicación y además, si necesitamos volver a crear las tablas porque hemos trasladado la aplicación o hemos perdido la base de datos, podremos regenerar toda la estructura ejecutando un comando.

Como ejemplos de migraciones vamos a generar unas tablas similares a las de la tarea 3, aunque no vamos a generar toda la base de datos y en algunas columnas habrá algunas diferencias para poder ejemplificar mejor los conceptos de la migración.

Creación de una tabla

El primer paso consistirá en ejecutar un comando de Artisan:

```
sudo php artisan make:migration create_libros_table
```

Este comando nos crea un nuevo archivo en la carpeta database/migration, con la fecha actual y el nombre indicado (libros).

A continuación describimos la estructura de la tabla:

```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateLibrosTable extends Migration
{
```

```
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('libros', function (Blueprint $table) {
                  $table->increments('lib_id'); // Crea una clave
primaria autoincremental
            $table->string('lib_titulo'); // Columna tipo texto
            $table->integer('lib_paginas'); // Tipo entero
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('libros');
    }
}
```

Con esto hemos creado dos funciones: up, que crea la tabla y down que la borra.
Para finalizar la migración ejecutamos en la línea de comandos:

```
php artisan migrate
```

Si queremos deshacer la migración, escribimos:

```
php artisan migrate:rollback
```

Para añadir una columna a una tabla existente:

```
sudo php artisan make:migration add_autor_to_libros
```

Nótese que hay que usar sudo cada vez que implique la creación de un archivo, para hacer rollback no ha sido necesario. A continuación editamos el nuevo archivo:

```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
```

```
class AddAutorToLibros extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::table('libros', function (Blueprint $table) {
            $table->string('lib_autor')->nullable();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::table('libros', function (Blueprint $table) {
            $table->dropColumn('lib_autor');
        });
    }
}
```

Y volvemos a invocar la migración desde el intérprete de comandos.

Creación de una relación

A continuación, y tal como estaba en la base de datos de la tarea 3, vamos a crear una tabla para Autores. Tenemos que deshacer la columna que habíamos puesto como string, para ello hacemos rollback y eliminamos el archivo que la creaba (add_autor_to_libros) y escribimos en el intérprete de comandos:

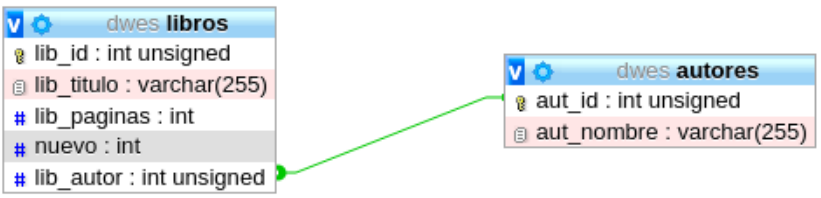
```
sudo php artisan make:migration create_autores_table_and_relationship
```

Ahora implementamos las funciones:

```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
```

```
class CreateAutoresTableAndRelationship extends Migration {
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up() {
        Schema::create('autores', function (Blueprint $table) {
            $table->increments('aut_id');
            $table->string('aut_nombre');
        });

        Schema::table('libros', function (Blueprint $table) {
            $table->integer('lib_autor')->unsigned()->nullable();

$table->foreign('lib_autor')->references('aut_id')->on('autores')-
>onDelete('set null'); // También podemos poner cascade o restrict
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down() {
        Schema::table('libros', function (Blueprint $table) {
            $table->dropForeign(['lib_autor']);
            $table->dropColumn('lib_autor');
        });

        Schema::dropIfExists('autores');
    }

}
```

Si ejecutamos *php artisan migrate* podemos ver que se ha creado la segunda tabla y también la relación:



Referencias:

https://richos.gitbooks.io/laravel-5/content/capitulos/chapter6.html
https://laravel.com/docs/9.x/migrations