Làm việc với Database

1. Schema
2. Migrate
3. Seed
4. Query Builder
5. Eloquent - Model
6. Liên kết dữ liệu trong Laravel

Kết nối với cơ sở dữ liệu trong laravel

Mở file .env

```
DB_HOST=localhost
DB_DATABASE= Ten CSDL
DB_USERNAME= Ten nguoi dung
DB_PASSWORD= Mat khau
```

1. Schema

1.1 Tạo bảng

```
Schema::create('SanPham', function ($table) {
        $table->increments('id');          //Tự tăng, khóa chính
        $table->string('TenSanPham');  //Kiểu chuỗi
        $table->integer('Gia');            //Kiểu int
        $table->timestamps();              //Tự cập nhật thời gian
    });
```

Mở rộng

Câu lệnh	Mô tả
$table->primary('TenKhoaChinh');	Tạo khóa chính
$table->foreign('KhoaPhu')->references('KhoaChinh')->on('Bang');	Tạo khóa phụ
$table->unique('TênCột');	Rang buộc unique
$table->time();	Kiểu giờ
$table->dateTime();	Kiểu ngày, giờ
$table->date();	Kiểu ngày
$table->text();	Kiểu text
$table->float();	Kiểu float
$table->boolean();	Kiểu logic
$table->rememberToken();	Tạo Token

Điều kiện

Câu lệnh	Mô tả
->nullable();	Cho phép giá trị null
->default($value);	Gán giá trị mặc định cho cột
->unsigned();	Đặt unsigned cho integer

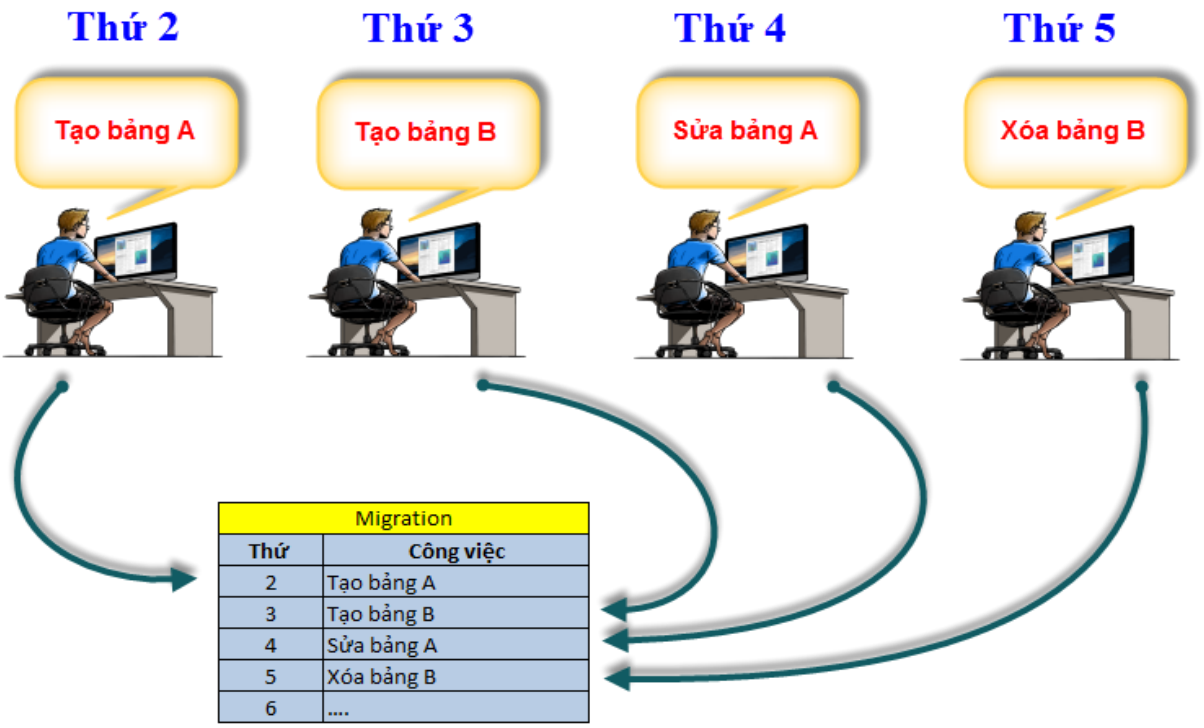
1.2 Sửa bảng

Câu lệnh	Mô tả
$table->dropColumn('TenCot');	Xóa cột trong bảng
Schema::rename($from, $to);	Đổi tên bảng

1.3 Xóa bảngss

Câu lệnh	Mô tả
Schema::drop('users');	Xóa bảng users
Schema::dropIfExists('users');	Xóa bảng users nếu bảng tồn tại

2. Migrate



Migrate dùng để tạo lên cấu trúc các bảng trong cơ sở dữ liệu. Ta có thể sử dụng migrate để tạo ra các bảng cũng như back up, restore lại theo ý muốn.

Các file migrate sẽ được lưu tại **database/migrations/**

Sử dụng migrate với cửa sổ cmd

php artisan make:migration TenMigrate	Tạo file migrate với artisan
php artisan migrate	Thực thi file migrate
php artisan migrate:rollback	Hủy bỏ việc thực thi của migrate trước
php artisan migrate:reset	Hủy bỏ hết công việc của migrate

php artisan migrate:fresh

Option

--create=TenBang	Migrate tạo bảng
--table=TenBang	Migrate chỉnh sửa bảng

Cấu trúc migration

```
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Migrations\Migration;
class CreateTable extends Migration
{
   public function up()
   {
       //đoạn lệnh khi thực hiện migrate
   }

   public function down()
   {
       //đoạn lệnh thực hiện khi Rollback.
   }
}
```

Tạo bảng với Schema

```
public function up()
{
        Schema::create('SanPham', function (Blueprint $table) {
           $table->increments('id');          //Tự tăng, khóa chính
           $table->string('TenSanPham');  //Kiểu chuỗi
           $table->integer('Gia');            //Kiểu int
           $table->timestamps();              //Tự cập nhật thời gian
        });
}
```

3. Seed

Seed là bộ dữ dữ liệu mẫu, nó giúp chúng ta quản lý dữ liệu trong bảng một cách thuận tiện, dễ dàng khôi phục lại khi cần thiết.

Các file seed được lưu tại thư mục **database/seeds/**

Tạo dữ liệu mẫu trong Seed.	Thực thi Seed.

use Illuminate\Database\Seeder;
use Illuminate\Database\Eloquent\Model;
class DatabaseSeeder extends Seeder
{
public function run()
{
DB::table('users')->insert([
'name' => str_random(10),
'email' => str_random(10).'@gmail.com',
'password' => bcrypt('secret'),
]);
}
}
	Mở cửa sổ cmd :
	php artisan db:seed

4. Query Builder

Có tác dụng thay thế cho các câu lệnh truy vấn thông thường bằng các phương trong lớp DB.

Ví dụ : **$users = DB::table('users')->get();** sẽ lấy toàn bộ dữ liệu trong bảng users ra và lưu vào $users

Lệnh này sẽ tương đương với lệnh truy vấn thông thường : ***SELECT * FROM users***

Các lệnh truy vấn

Lệnh truy vấn	Mô tả	Ví dụ
DB::table('users')	Chọn bảng trong cơ sở dữ liệu	DB::table('users')->get();
get()	Lấy dữ liệu trong bảng	DB::table('users')->get();
first()	Lấy một dòng dữ liệu đầu tiên từ kết quả truy vấn	DB::table('users')->where('name', 'John')->first();
value('tên cột')	Trả về dữ liệu của cột đã khai báo	DB::table('users')->where('name', 'Joh')->value('email');
select('tên cột 1')	Chọn tên cột cần truy vấn	DB::table('users')->select('name', 'email')->get();
addSelect('tên cột')	Thêm cột vào truy vấn trước đó với addSelect()	$query = DB::table('users')->select('name'); $users = $query->addSelect('age')->get();

DB::raw('Truy vấn')	Thêm lệnh truy vấn vào select()	DB::table('users')-> select(DB::raw('count(*) as userCount, status'))
join('bảng liên kết', 'cột liên kết 1', 'điều kiện', 'cột liên kết 2')	Lệnh Join bảng trong truy vấn	DB::table('users')->join('contacts', 'users.id', '=', 'contacts.user_id')->select('contacts.phone')->get();
where('cột 1', 'điều kiện' , giá trị)	Điều kiện where	DB::table('users')->where('votes', '=', 100)->get();
orwhere('cột 1', 'điều kiện' , giá trị)	Điều kiện hoặc	DB::table('users')->where('votes', '=', 100)->orwhere('age', '>=', '18')->get();
orderBy('tên cột', 'điều kiện')	Lệnh orderBy	DB::table('users')->orderBy('name', 'desc')->get();
groupBy('tên cột')->having(điều kiện)	Lệnh groupBy	DB::table('users')->groupBy('account_id')->having('account_id', '>', 100)->get();
skip(vị trí)->take(số lượng)	Giới hạn kết quả truy vấn Tương đương với LIMIT	DB::table('users')->skip(10)->take(5)->get();
avg('tên cột');	Lấy giá trị trung bình	DB::table('orders')->where('finalized', 1)->avg('price');
max('price');	Lấy giá trị max	DB::table('orders')->max('price');
count();	Lệnh đếm	DB::table('users')->count();

Lệnh update

Lệnh truy vấn	Mô tả	Ví dụ
update(['tên cột' => giá trị]);	Lệnh update	DB::table('users')->where('id', 1)->update(['votes' => 1]);
increment('tên cột',giá trị) decrement('tên cột',giá trị)	Tăng/giảm giá trị cột	DB::table('users')->increment('votes',4);

Lệnh insert

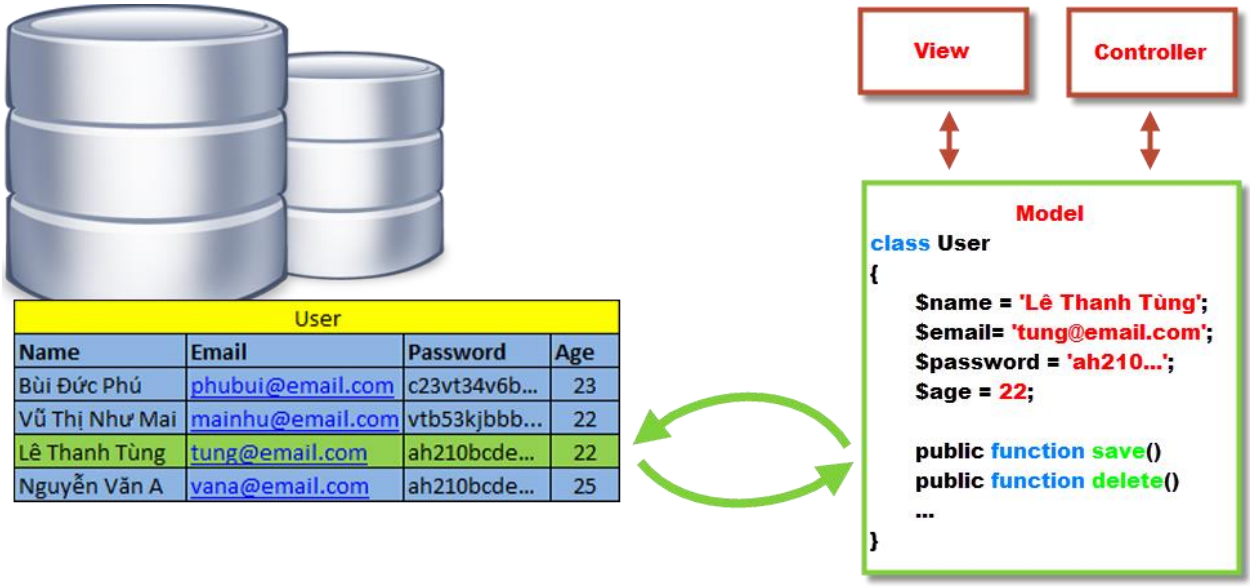
Lệnh truy vấn	Mô tả	Ví dụ
insert([mảng các bản ghi]);	Lệnh insert	DB::table('users')->insert(['email' => 'john@example.com', 'votes' => 0]);

Lệnh delete

Lệnh truy vấn	Mô tả	Ví dụ
delete();	Xóa dữ liệu	DB::table('users')->where('votes', '<', 100)->delete();
truncate();	Xóa tất cả dữ liệu trong bảng và đặt chỉ số tự tăng về 0	DB::table('users')->truncate();

5. Eloquent – Model

Model là một lớp dữ liệu, có cấu trúc giống với bảng trong cơ sở dữ liệu, dùng để xử lý dữ liệu ra vào trong bảng.



User

Name	Email	Password	Age
Bùi Đức Phú	phubui@email.com	c23vt34v6b...	23
Vũ Thị Như Mai	mainhu@email.com	vtb53kjbbb...	22
Lê Thanh Tùng	tung@email.com	ah210bcde...	22
Nguyễn Văn A	vana@email.com	ah210bcde...	25

5.1 Tạo model

Các file model sẽ được lưu tại thư mục App/

Tạo một model :

```
php artisan make:model TenModel
```

Tạo một model và migrate tương ứng với nó :

```
Php artisan make:model TenModel -m
```

Kết nối Model tới bảng trong cơ sở dữ liệu

Mã lệnh	Mô tả
protected $table = 'tên bảng';	Kết nối model với bảng trong cơ sở dữ liệu
public $timestamps = false;	Tắt/bật chế độ tự động quản lý 'created_at' và 'update_at'

Ví dụ

```
namespace App;
use Illuminate\Database\Eloquent\Model;

class User extends Model
{
    protected $table = 'user';
    public $timestamps = false;
}
```

5.2 Các phương thức trong model

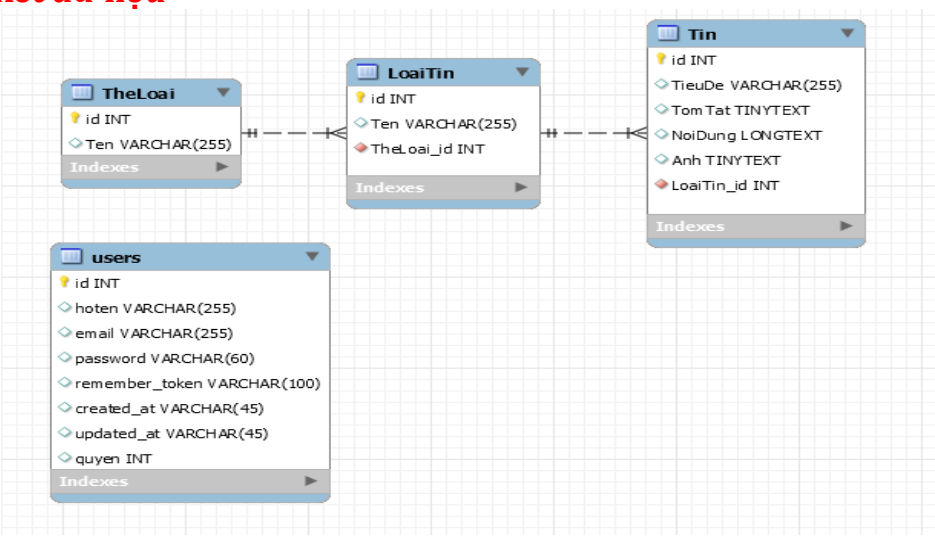
Một số phương thức hay sử dụng trong model

Mã lệnh	Mô tả
$user = new User(); echo $user->name;	Lấy giá trị thuộc tính của model
$user = User::all();	Lấy toàn bộ dữ liệu trong bảng
$user = User::find(giá trị khóa chính);	Tìm user theo khóa chính
$user->toJson();	Trả dữ liệu kiểu JSON
$user->save();	Lưu dữ liệu từ model vào bảng
$user->delete();	Xóa dữ liệu trong bảng
User::destroy(giá trị khóa chính);	Xóa dữ liệu bằng khóa chính trong bảng

Kết hợp model với query builder

```
$user = User::where('active', 1)->orderBy('name', 'desc')->take(10)->get();
```

6. Liên kết dữ liệu



Model là đại diện cho các bảng trong cơ sở dữ liệu, chính vì thế mà nó cũng có các liên kết với nhau.

Khai báo các liên kết tới các model khác.

Ví dụ : Liên kết một nhiều. Ta khai báo hàm TenLienKet() trong class model.

Khai báo	Sử dụng

public function TenLienKet()
{
return $this->hasMany('TenModel' , 'KhoaPhu', 'KhoaChinh');
}
	TenModel::TenLienKet()

Bảng liên kết

Liên kết	Hàm liên kết
Một – Một , Liên kết từ bảng cha tới bảng con.	**hasOne();**
Một – Một , Liên kết từ bảng con tới bảng cha.	**belongsTo();**
Một – Nhiều	**hasMany();**
Nhiều – nhiều	**belongsToMany();**
Liên kết qua bảng trung gian	**hasManyThrough();**