PHP Laravel Query Builder

1/ Cách để sử dụng được PHP Laravel Query Builder:

```
use Illuminate\Support\Facades\DB;
```

2/ Query đơn giản (Select)

```
$get = DB::table(<tên_table>)->get();
```

```
<?php
$get = DB::table('users')->get();
```

Tương đương câu truy vấn : "SELECT * FROM users"

```
public function getUsers() {
        $get = DB::table('users')->get();
        return $get;
}
```

SELECT COLUMN:

```
$get = DB::table('users')
        ->select('name','email')
        ->get();
```

SELECT LIMIT:

```
$get = DB::table('users')
        ->limit(8)
        ->get();
```

Ngoài ra, phương thức **take(int)** tương tự như method limit(int)

```
$get = DB::table('users')
        ->take(8)
        ->get();
```

SELECT OFFSET, LIMIT:

Phương thức **skip(int) và offset(int)**:

```
// Phương thức skip
$get = DB::table('users')
```

```
        ->skip(5)
        ->take(8)
        ->get();
// Phương thức Offset
$get = DB::table('users')
        ->offset(5)
        ->limit(8)
        ->get();
```

SELECT ORDER BY:

Sử dụng phương thức **orderBy(string Column, string "DESC/ASC")**

```
$get = DB::table('users')
                ->orderBy('id', 'desc')
                ->get();
```

3/ Điều kiện (Where)

Điều kiện bằng: where(string column, TYPE)

```
$get = DB::table('users')
                ->where('id', 10)
                ->get();
```

Lưu ý: nếu kết quả trả về 1 bản ghi sử dụng phương phương thức **get()** thay cho **first();**

```
$get = DB::table('users')
                ->where('id', 10)
                ->first();
```

Điều kiện nhỏ hơn, lớn hơn,…: where(column, operator, type)

```
$get = DB::table('users')
                ->where('id', '>=', 10)
                ->get();
```

Điều kiện nằm giữa giá trị: whereBetween(column, array(min, max))

```
$get = DB::table('users')
                ->whereBetween('age', [18, 30])
                ->get();
```

Điều kiện không nằm giữa giá trị: whereNotBetween(column, array(min, max))

```
$get = DB::table('users')
                ->whereNotBetween('age', [18, 30])
                ->get();
```

Điều kiện whereIn và whereNotIn: whereIn(column, array())

```
$get = DB::table('users')
                ->whereIn('age', [18, 19, 20, 21])
                ->get();
$get = DB::table('users')
                ->whereNotIn('age', [18, 19, 20, 21])
                ->get();
```

Nhiều điều kiện tương tự sử dụng toán tử and (Where and Where,…)

```
$get = DB::table('users')
                ->where('name', 'Teo')
                ->where('email', 'teo@gmail.com')
                ->where('age', 20)
                ->first();
```

Hoặc sử dụng mảng kết hợp cho kết quả tương tự:

```
$check = [
        'name'  => 'Teo',
        'email' => 'teo@gmail.com',
        'age'   =>  20
];
$get = DB::table('users')
                ->where($check)
                ->first();
```

Hoặc sử dụng mảng với dạng where tự chọn

```
$check = [
        ['age', '>=', 20],
        ['id', '<', 100]
];
$get = DB::table('users')
                ->where($check)
                ->get();
```

Nhiều điều kiện tương tự sử dụng toán tử or (Where and Where,…)

```
$get = DB::table('users')
                ->where('id', 10)
                ->orWhere('name', 'Teo')
                ->first();
```

4/ Gán (Join, Left Join,…)

- join(tables, column to check 1, operator, column to check 2)
- leftJoin(tables, column to check 1, operator, column to check 2)

```
$get = DB::table('Products')
                ->orderBy('proid', 'desc')
                ->join('Categorys', ' Categorys.catit', '=', ' Products.catid')
                ->get();
```

5/ Lấy thông tin sử dụng sau khi truy vấn

Truy xuất phần tử khi truy vấn bằng phương thức first();

```
$user = DB:table('users')
                ->where('id', 1')
                ->first();
// Lấy giá trị
$name = $user->name;
$age = $user->age;
$email = $user->email;
```

Duyệt toàn bộ mảng khi truy vấn bằng phương thức get();

```
$user = DB:table('users')
                ->get();
// foreach
foreach ($user as $infoUser) {
        // sử dụng $infoUser->tên_column
}
```

6/ Tính toán (Aggregates)

```
// Đếm toàn bộ số rows của table users
$users = DB::table('users')->count();
// Lấy ra giá trị max của column price trong table order
$price = DB::table('orders')->max('price');
// Lấy ra giá trị min của column price trong table order
$price = DB::table('orders')->min('price');
// Tính trung bình của column price trong table order
$price = DB::table('orders')->avg('price');
// Tính tổng votes của toàn bộ rows trong table users
$total = DB::table('users')->sum('votes');
```

7/ Thêm (Insert)

Insert:

```
DB::table('users')
        ->insert([
                'name'   => 'Teo',
                'email'  => 'teo@gmail.com',
                'phone'  => '0934999999',
                'age'    => 20
        ]);
```

Insert và trả về ID của dòng mới thêm:

```
$id = DB::table('users')
        ->insertGetId([
                'name'   => 'Teo',
                'email'  => 'teo@gmail.com',
                'phone'  => '0934999999',
                'age'    => 20
        ]);
```

Insert multiple rows

```
$insert = [
        ['name' => 'Teo', 'email' => 'teo@gmail.com', 'phone' => '0934', 'age' =>
20],
        ['name' => 'Ty', 'email' => 'ty@gmail.com', 'phone' => '4567',  'age' =>
21],
        ['name' => 'Tia', 'email' => 'tia@gmail.com', 'phone' => '1234', 'age' =>
21]
];
DB::table('users')->insert($insert);
```

8/ Sửa/Cập nhập (Update)

Update:

```
DB::table('users')
        ->where('id', 1)
        ->update([
                'name'   => 'Ty',
                'phone'  => '090xxxxxxx'
        ]);
```

Update with increment(column, by = 1), decrement(column, by = 1)

```
DB::table('users')->increment('age');
DB::table('users')->increment('age', 5);
DB::table('users')->decrement('age');
DB::table('users')->decrement('age', 5);
```

Update with increment/decrement(column, by = 1, assoc_array)

```
DB::table('users')
        ->where('id', 1)
        ->increment('age', 1, ['name' => 'Ty']);
```

9/ Xóa (Delete)

Xóa với điều kiện

```
DB::table('users')
        ->where('age', '<', 18)->delete();
```

Xóa toàn bộ rows

```
DB::table('users')->delete();
```

Xóa toàn bộ (Truncating)

Xóa kiểu truncate, khi nhập thông tin mới thì ID sẽ bắt đầu lại từ 1.

```
DB::table('users')->truncate();
```

https://laravel.com/docs/7.x/queries

PHP Laravel Eloquent ORM

1.Lấy tất cả các bản ghi

- QueryBuilder

```
$users = DB::table('users')->get();
```

- Eloquent ORM

```
$users = User::all();
```

2.Lấy một bản ghi theo id

- QueryBuilder

```
$users = DB::table('users')->where('id', 1)->first();
```

- Eloquent ORM

```
$users = User::find(1);
```

3.Lấy một trường của một bản ghi

- QueryBuilder

```
//lấy ra trường name có id user = 1
$name = DB::table('users')->where('id', 1)->value('name');
```

- Eloquent ORM

```
$name = User::where('id', 1)->value('name');
```

4.Lấy một trường của tất cả bản ghi

- QueryBuilder

```
//lấy ra trường name có id user = 1
$name = DB::table('users')->lists('name');
```

- Eloquent ORM

```
$users = Users::lists('name');
```

5.Lấy số lượng bản ghi cho phép

- QueryBuilder

```
//lấy ra 100 user
$users = DB::table('users')->chunk(100, function($users) {
    foreach ($users as $user) {
        //
    }
});
```

- Eloquent ORM

```
User::chunk(100, function ($users) {
     foreach ($users as $user) {
         //
     }
});
```

6.Insert

- QueryBuilder

```
addUser = DB::table('users')->insert(
    ['email' => 'test@gmail.com']
);
```

- Eloquent ORM

```
$user = new User;
$user->email = 'test@gmail.com';
$user->save();
```

7.Update

- QueryBuilder

```
editUser = DB::table('users')->where('id', 1)->update(['name' =>
'nameTest']);
```

- Eloquent ORM

```
$user = User::find(1);
$user->name ='nameTest';
$user->save();
```

8.Delete

- QueryBuilder

```
deleteUser = DB::table('users')->where('id', '=', '1')->delete();
```

- Eloquent ORM

```
// nếu là 1 câu truy vấn
$user = User::find(1);
$user->delete();
//hoặc nếu không phải 1 câu truy vấn
$user->destroy();
```

9.Aggregates

- QueryBuilder

```
$users = DB::table('users')->count();
$price = DB::table('orders')->max('price');
```

- Eloquent ORM

```
$count = User::where('name', '%a%')->count();
```

```
$max = Order::where('name', '%a%')->max('price');
```

https://laravel.com/docs/7.x/eloquent

PHP Laravel Framework

1. Câu lệnh tải Laravel 9.* thông qua Composer :

```
composer create-project laravel/laravel:^9.* project-name
```

2. Câu lệnh chạy server ảo trong Laravel :

```
php artisan serve
```

3. Câu lệnh để xem toàn bộ routes đang có trong PHP Framework Laravel :

```
php artisan route:list
```

4. Câu lệnh tạo controller trong Laravel :

```
php artisan make:controller UserController
```

5. Câu lệnh tạo controller với 7 hàm CRUD trong Laravel :

```
php artisan make:controller UserController --resource
```

6. Câu lệnh tạo Migration trong Laravel:

```
php artisan make:migration create_users_table
```

7. Câu lệnh chạy Migration:

```
php artisan migrate
```

8. Câu lệnh quay trở lại dựa trên dữ liệu đã ghi vào migrations table và chạy lại migration:

```
php artisan migrate:refresh
```

9. Câu lệnh xóa hết các bảng, không quan tâm về rollback và chạy lại migration:

```
php artisan migrate:fresh
```

10. Câu lệnh tạo file Seeder trong Laravel:

```
php artisan make:seeder UsersTableSeeder
```

11 Câu lệnh chạy file Seeder cụ thể với tên class trong Laravel:

```
php artisan db:seed --class=UsersTableSeeder
```

12. Câu lệnh chạy file DatabaseSeeder, có thể gọi tới nhiều file với class seeder cụ thể:

```
php artisan db:seed
```

13. Câu lệnh để xóa tất cả các bảng dữ liệu, sau đó chạy lại migration sau đó chạy file DatabaseSeeder, có thể gọi tới nhiều class seeder:

```
php artisan migrate:fresh --seed
```

14. Câu lệnh tạo Model trong PHP Framework Laravel :

```
php artisan make:model User
```

15. Câu lệnh kết hợp tạo Model và Controller trong Laravel :

```
php artisan make:model User -c
```

16. Câu lệnh kết hợp tạo Model, Controller và Migration trong PHP Framework Laravel:

```
php artisan make:model Category -mc
```

17. Câu lệnh kết hợp tạo Model, Controller + 7 hàm CRUD và Migration trong PHP Framework Laravel:

```
php artisan make:model User -mcr
```