LẬP TRÌNH WEB PHP NÂNG CAO

GV: Trần Thanh Tuấn

Laravel

Nội dung

- Cấu hình kết nối cơ sở dữ liệu
- Migrations
- Query Builder
- Seeding

Cấu hình kết nối cơ sở dữ liệu

- Laravel hỗ trợ kết nối với 04 HQT Cơ sở dữ liệu:
  - MySQL
  - PostgreSQL
  - SQLite
  - SQL Server

Cấu hình kết nối cơ sở dữ liệu

- Cấu hình kết nối cơ sở dữ liệu:
  - Tập tin: database.php trong thư mục config
  - Các hằng số trong tập tin .env

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=homestead
DB_USERNAME=root
DB_PASSWORD=
```

Nội dung

- Cấu hình kết nối cơ sở dữ liệu
- Migrations
- Query Builder
- Seeding

Migrations

- **Tạo / chỉnh sửa** lược đồ cơ sở dữ liệu
- Các tập tin migration được lưu trong thư mục database/migrations

Migrations

- Tạo migration:

```
php artisan make:migration <tên_file>
```

- Các tham số tùy chọn:
  - **--create="<tên_table>":** tạo table mới *(Schema::create)*
  - **--table="<tên_table>":** chỉnh sửa table *(Schema::table)*

Migrations

- Cấu trúc migration:
  - **Phương thức up:** thêm table, cột, chỉ mục mới

```
public function up()
{
        Schema::create(`linh_vuc', function (Blueprint $table) {
            $table->bigIncrements('id');
            $table->string(`ten');
            $table->boolean(`xoa')->default(0);
            $table->timestamps();
        });
}
```

Migrations

- Cấu trúc migration:
  - **Phương thức down:** đảo ngược các hoạt động được thực thi bởi phương thức up

```
public function down()
{
        Schema::drop(`linh_vuc’);
}
```

Migrations

- Thực thi migration :

```
php artisan migrate
```

- Rollback migration:
  - Hủy bỏ các migration mới nhất

```
php artisan migrate:rollback
```

  - Hủy bỏ n migration trước đó

```
php artisan migrate:rollback --step=<n>
```

  - Hủy bỏ tất cả migration

```
php artisan migrate:reset
```

Migrations

- Rollback kết hợp migrate:
  - Hủy bỏ tất cả migration và thực thi migrate lại

```
php artisan migrate:refresh
```

  - Hủy bỏ n migration và thực thi migrate lại

```
php artisan migrate:refresh --step=<n>
```

Migrations

- Xóa bảng và thực thi migrate:

```
php artisan migrate:fresh
```

Migrations

- Tạo bảng (table):

```
Schema::create('linh_vuc', function (Blueprint $table) {
        $table->bigIncrements('id');
        $table->string(`ten');
        $table->boolean(`xoa')->default(0);
        $table->timestamps();
});
```

Migrations

- Kiểm tra bảng / cột có tồn tại chưa?

```
if (Schema::hasTable('linh_vuc')) {
    //
}

if (Schema::hasColumn('linh_vuc', 'ten')) {
    //
}
```

Migrations

- Thiết lập các tùy chọn cho bảng:

Câu lệnh	Mô tả
$table->engine = 'InnoDB';	Specify the table storage engine (MySQL).
$table->charset = 'utf8';	Specify a default character set for the table (MySQL).
$table->collation = 'utf8_unicode_ci';	Specify a default collation for the table (MySQL).

Migrations

- Đổi tên bảng:

```
Schema::rename('<tên_cũ>', '<tên mới>');
```

- Xóa bảng:

```
Schema::drop('<tên_bảng>');

Schema::dropIfExists('<tên_bảng>');
```

Migrations

- Tạo cột (column):
  - Số nguyên tự động tăng (khóa chính):

Câu lệnh	Mô tả
$table->increments('id');	Auto-incrementing UNSIGNED INTEGER (primary key) equivalent column.
$table->tinyIncrements('id');	Auto-incrementing UNSIGNED TINYINT (primary key) equivalent column.
$table->smallIncrements('id');	Auto-incrementing UNSIGNED SMALLINT (primary key) equivalent column.
$table->mediumIncrements('id');	Auto-incrementing UNSIGNED MEDIUMINT (primary key) equivalent column.
$table->bigIncrements('id');	Auto-incrementing UNSIGNED BIGINT (primary key) equivalent column.

Migrations

- Tạo cột (column):
  - Số nguyên:

Câu lệnh	Mô tả
$table->integer('votes');	INTEGER equivalent column.
$table->tinyInteger('votes');	TINYINT equivalent column.
$table->smallInteger('votes');	SMALLINT equivalent column.
$table->mediumInteger('votes');	MEDIUMINT equivalent column.
$table->bigInteger('votes');	BIGINT equivalent column.
$table->unsignedInteger('votes');	UNSIGNED INTEGER equivalent column.
$table->unsignedTinyInteger('votes');	UNSIGNED TINYINT equivalent column.

Migrations

- Tạo cột (column):
  - Số nguyên:

Câu lệnh	Mô tả
$table->unsignedSmallInteger('votes');	UNSIGNED SMALLINT equivalent column.
$table->unsignedMediumInteger('votes');	UNSIGNED MEDIUMINT equivalent column.
$table->unsignedBigInteger('votes');	UNSIGNED BIGINT equivalent column.

Migrations

- Tạo cột (column):
  - Số thực:

Câu lệnh	Mô tả
$table->float('amount', 8, 2);	FLOAT equivalent column with a precision (total digits) and scale (decimal digits).
$table->double('amount', 8, 2);	DOUBLE equivalent column with a precision (total digits) and scale (decimal digits).
$table->decimal('amount', 8, 2);	DECIMAL equivalent column with a precision (total digits) and scale (decimal digits).
$table->unsignedDecimal('amount', 8, 2);	UNSIGNED DECIMAL equivalent column with a precision (total digits) and scale (decimal digits).

Migrations

- Tạo cột (column):
  - Chuỗi:

Câu lệnh	Mô tả
$table->char('name', 100);	CHAR equivalent column with an optional length.
$table->string('name', 100);	VARCHAR equivalent column with a optional length.
$table->text('description');	TEXT equivalent column.
$table->mediumText('description');	MEDIUMTEXT equivalent column.
$table->longText('description');	LONGTEXT equivalent column.
$table->lineString('positions');	LINESTRING equivalent column.
$table->multiLineString('positions');	MULTILINESTRING equivalent column.

Migrations

- Tạo cột (column):
  - Ngày giờ:

Câu lệnh	Mô tả
$table->date('created_at');	DATE equivalent column.
$table->dateTime('created_at');	DATETIME equivalent column.
$table->dateTimeTz('created_at');	DATETIME (with timezone) equivalent column.
$table->time('sunrise');	TIME equivalent column.
$table->timeTz('sunrise');	TIME (with timezone) equivalent column.
$table->timestamp('added_on');	TIMESTAMP equivalent column.
$table->timestampTz('added_on');	TIMESTAMP (with timezone) equivalent column.

Migrations

- Tạo cột (column):
  - Ngày giờ:

Câu lệnh	Mô tả
$table->timestamps();	Adds nullable created_at and updated_at TIMESTAMP equivalent columns.
$table->nullableTimestamps();	Alias of timestamps() method.
$table->timestampsTz();	Adds nullable created_at and updated_at TIMESTAMP (with timezone) equivalent columns.
$table->softDeletes();	Adds a nullable deleted_at TIMESTAMP equivalent column for soft deletes.
$table->softDeletesTz();	Adds a nullable deleted_at TIMESTAMP (with timezone) equivalent column for soft deletes.

Migrations

- Tạo cột (column):
  - Các kiểu dữ liệu khác:

Câu lệnh	Mô tả
$table->binary('data');	BLOB equivalent column.
$table->boolean('confirmed');	BOOLEAN equivalent column.
$table->json('options');	JSON equivalent column.
$table->jsonb('options');	JSONB equivalent column.
$table->rememberToken();	Adds a nullable remember_token VARCHAR(100) equivalent column.
$table->uuid('id');	UUID equivalent column.
$table->year('birth_year');	YEAR equivalent column.

Migrations

- Tạo cột (column):
  - Các kiểu dữ liệu khác:

Câu lệnh	Mô tả
$table->ipAddress('visitor');	IP address equivalent column.
$table->macAddress('device');	MAC address equivalent column.
$table->enum('level', ['easy', 'hard']);	ENUM equivalent column.
$table->set('flavors', ['strawberry', 'vanilla']);	SET equivalent column.

Migrations

- Tạo cột (column):
  - Các kiểu dữ liệu khác:

Câu lệnh	Mô tả
$table->point('position');	POINT equivalent column.
$table->multiPoint('positions');	MULTIPOINT equivalent column.
$table->geometry('positions');	GEOMETRY equivalent column.
$table->geometryCollection('positions');	GEOMETRYCOLLECTION equivalent column.
$table->polygon('positions');	POLYGON equivalent column.

Migrations

- Tạo cột (column):
  - Các kiểu dữ liệu khác:

Câu lệnh	Mô tả
$table->morphs('taggable');	Adds taggable_id UNSIGNED BIGINT and taggable_type VARCHAR equivalent columns.
$table->uuidMorphs('taggable');	Adds taggable_id CHAR(36) and taggable_type VARCHAR(255) UUID equivalent columns.
$table->multiPolygon('positions');	MULTIPOLYGON equivalent column.
$table->nullableMorphs('taggable');	Adds nullable versions of morphs() columns.
$table->nullableUuidMorphs('taggable');	Adds nullable versions of uuidMorphs() columns.

Migrations

- Các tùy chọn của cột:

Câu lệnh	Mô tả
->default($value)	Specify a "default" value for the column
->nullable($value = true)	Allows (by default) NULL values to be inserted into the column
->charset('utf8')	Specify a character set for the column (MySQL)
->collation('utf8_unicode_ci')	Specify a collation for the column (MySQL/PostgreSQL/SQL Server)
->autoIncrement()	Set INTEGER columns as auto-increment (primary key)
->comment('my comment')	Add a comment to a column (MySQL/PostgreSQL)
->unsigned()	Set INTEGER columns as UNSIGNED (MySQL)

Migrations

- Các tùy chọn của cột:

Câu lệnh	Mô tả
->after('column')	Place the column "after" another column (MySQL)
->first()	Place the column "first" in the table (MySQL)
->storedAs($expression)	Create a stored generated column (MySQL)
->useCurrent()	Set TIMESTAMP columns to use CURRENT_TIMESTAMP as default value
->virtualAs($expression)	Create a virtual generated column (MySQL)
->generatedAs($expression)	Create an identity column with specified sequence options (PostgreSQL)
->always()	Defines the precedence of sequence values over input for an identity column (PostgreSQL)

Migrations

- Cập nhật cột:
  - Thay đổi thuộc tính: sử dụng phương thức change()

```
Schema::table('users', function (Blueprint $table) {
    $table->string('name', 50)->change();
});
```

```
Schema::table('users', function (Blueprint $table) {
    $table->string('name', 50)->nullable()->change();
});
```

Lưu ý: chỉ các cột có kiểu bigInteger, binary, boolean, date, dateTime, dateTimeTz, decimal, integer, json, longText, mediumText, smallInteger, string, text, time, unsignedBigInteger, unsignedInteger và unsignedSmallInteger mới được thay đổi thuộc tính

Migrations

- Cập nhật cột:
  - Đổi tên cột:

```
Schema::table('users', function (Blueprint $table) {
    $table->renameColumn('from', 'to');
});
```

  - Xóa cột:

```
Schema::table('users', function (Blueprint $table) {
    $table->dropColumn('votes');
});
```

```
Schema::table('users', function (Blueprint $table) {
    $table->dropColumn(('votes', 'avatar', 'location'));
});
```

Migrations

- Cập nhật cột:
  - Xóa cột:

Câu lệnh	Mô tả
$table->dropMorphs('morphable');	Drop the morphable_id and morphable_type columns.
$table->dropRememberToken();	Drop the remember_token column.
$table->dropSoftDeletes();	Drop the deleted_at column.
$table->dropSoftDeletesTz();	Alias of dropSoftDeletes() method.
$table->dropTimestamps();	Drop the created_at and updated_at columns.
$table->dropTimestampsTz();	Alias of dropTimestamps() method.

Migrations

- Tạo chỉ mục:

Câu lệnh	Mô tả
$table->primary('id');	Adds a primary key.
$table->primary(['id', 'parent_id']);	Adds composite keys.
$table->unique('email');	Adds a unique index.
$table->index('state');	Adds a plain index.
$table->spatialIndex('location');	Adds a spatial index. (except SQLite)

Tên chỉ mục: tham số thứ 2 trong các phương thức trên
(nếu không có tham số thứ 2 thì tên chỉ mục được tạo thành từ tên bảng và tên cột)

```
$table->unique(`email', `unique_email');
```

Migrations

- Đổi tên chỉ mục:

```
$table->renameIndex('from', 'to')
```

- Xóa chỉ mục: bằng tên chỉ mục

Câu lệnh	Mô tả
$table->dropPrimary('users_id_primary');	Drop a primary key from the "users" table.
$table->dropUnique('users_email_unique');	Drop a unique index from the "users" table.
$table->dropIndex('geo_state_index');	Drop a basic index from the "geo" table.
$table->dropSpatialIndex('geo_location_spatialindex');	Drop a spatial index from the "geo" table (except SQLite).

Migrations

- Xóa chỉ mục: bằng tên cột

```
Schema::table('geo', function (Blueprint $table) {
    $table->dropIndex(['state']); // Drops index 'geo_state_index'
});
```

Migrations

- Độ dài của chỉ mục
  - Laravel sử dụng charset mặc định là utf8mb4
  - MySQL version > 5.7.7, MariaDB version > 10.2.2
  - Cấu hình độ dài chuỗi mặc định để tạo chỉ mục khi thực hiện migrate

```
public function boot()
{
    Schema::defaultStringLength(191);
}
```

AppServiceProvider.php

Migrations

- Khóa ngoại
  - Tạo khóa ngoại:

```
Schema::table('posts', function (Blueprint $table) {
    $table->unsignedBigInteger('user_id');

    $table->foreign('user_id')->references('id')->on('users');
});
```

```
$table->foreign('user_id')
      ->references('id')->on('users')
      ->onDelete('cascade');
```

Migrations

- Khóa ngoại
  - Xóa khóa ngoại:

```
$table->dropForeign('posts_user_id_foreign');
```

```
$table->dropForeign(['user_id']);
```

Migrations

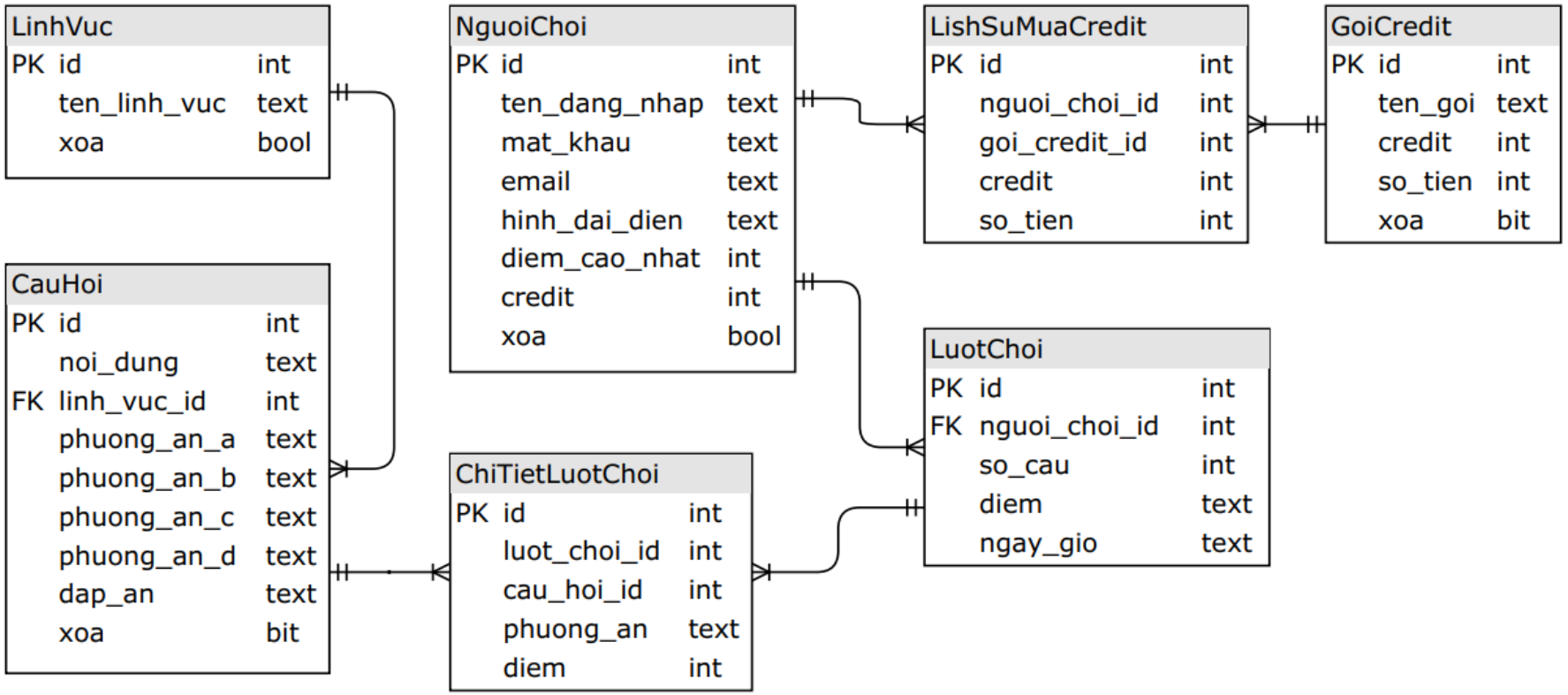
- Khóa ngoại
  - Bật tắt khóa ngoại trong migration

```
Schema::enableForeignKeyConstraints();
```

```
Schema::disableForeignKeyConstraints();
```

Bài tập

- Tạo lược đồ CSDL sau



Bài tập

- Tạo lược đồ CSDL sau

CauHinhDiemCauHoi		
PK	id	int
	thu_tu	int
	diem	int

CauHinhApp		
PK	id	int
	co_hoi_sai	int
	thoi_gian_tra_loi	int

QuanTriVien		
PK	id	int
	ten_dang_nhap	text
	mat_khau	text
	ho_ten	text
	xoa	bit

CauHinhTroGiup		
PK	id	int
	loai_tro_giup	int
	thu_tu	int
	credit	int

Nội dung

- Cấu hình kết nối cơ sở dữ liệu
- Migrations
- Query Builder
- Seeding

Query Builder

- Sử dụng phương thức DB::table() để bắt đầu truy vấn
- Lấy tất cả dữ liệu trong bảng:

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

- Lấy các dòng thỏa mãn điều kiện:

```
$user = DB::table('users')->where('name', 'John')->get();
```

- Lấy dòng đầu tiên thỏa mãn điều kiện:

```
$user = DB::table('users')->where('name', 'John')->first();
```

Query Builder

- Tìm theo **id**:

```
$user = DB::table('users')->find(3);
```

- Các hàm: **count**, **sum**, **max**, **min** và **avg**

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

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

$price = DB::table('orders')
            ->where('finalized', 1)
            ->avg('price');
```

Query Builder

- Kiểm tra có dòng dữ liệu trong kết quả truy vấn hay không?

```
DB::table('orders')->where('finalized', 1)->exists();

DB::table('orders')->where('finalized', 1)->doesntExist();
```

- Lấy dữ liệu từ các cột được chỉ định:

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

- Lấy các dòng dữ liệu không trùng nhau:

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

Query Builder

- Kiểm tra có dòng dữ liệu trong kết quả truy vấn hay không?

```
DB::table('orders')->where('finalized', 1)->exists();

DB::table('orders')->where('finalized', 1)->doesntExist();
```

- Lấy dữ liệu từ các cột được chỉ định:

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

- Lấy các dòng dữ liệu không trùng nhau:

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

Query Builder

- Mệnh đề WHERE

```
$users = DB::table('users')->where('votes', 100)->get();
```

```
$users = DB::table('users')
            ->where('votes', '>=', 100)
            ->get();
```

```
$users = DB::table('users')->where([
    ['status', '=', '1'],
    ['subscribed', '<>', '1'],
])->get();
```

Query Builder

- Mệnh đề WHERE

```
$users = DB::table('users')
            ->where('votes', '>', 100)
            ->orWhere('name', 'John')
            ->get();
```

```
$users = DB::table('users')
            ->whereBetween('votes', [1, 100])->get();
```

```
$users = DB::table('users')
            ->whereNotBetween('votes', [1, 100])
            ->get();
```

Query Builder

- Mệnh đề WHERE

```
$users = DB::table('users')
            ->whereIn('id', [1, 2, 3])
            ->get();
```

```
$users = DB::table('users')
            ->whereNotIn('id', [1, 2, 3])
            ->get();
```

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

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

Query Builder

- Mệnh đề WHERE

```
$users = DB::table('users')
            ->whereDate('created_at', '2016-12-31')
            ->get();
```

```
$users = DB::table('users')
            ->whereMonth('created_at', '12')
            ->get();
```

```
$users = DB::table('users')
            ->whereDay('created_at', '31')
            ->get();
```

```
$users = DB::table('users')
            ->whereYear('created_at', '2016')
            ->get();
```

Query Builder

- Mệnh đề WHERE

```
$users = DB::table('users')
            ->whereTime('created_at', '=', '11:20:45')
            ->get();
```

```
$users = DB::table('users')
            ->whereColumn('first_name', 'last_name')
            ->get();
```

```
$users = DB::table('users')
            ->whereColumn('updated_at', '>', 'created_at')
            ->get();
```

Query Builder

- Mệnh đề WHERE

```
$users = DB::table('users')
            ->whereColumn([
                ['first_name', '=', 'last_name'],
                ['updated_at', '>', 'created_at']
            ])->get();
```

Query Builder

- Mệnh đề WHERE

```
SELECT * FROM users WHERE name = 'John' and (votes > 100 or title = 'Admin')
```

```
DB::table('users')
        ->where('name', '=', 'John')
        ->where(function ($query) {
            $query->where('votes', '>', 100)
            ->orWhere('title', '=', 'Admin');
        })
        ->get();
```

Query Builder

- Ordering

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

```
$user = DB::table('users')
            ->latest() // sắp xếp theo cột created_at
            ->first();
```

```
$user = DB::table('users')
            ->latest('tên cột')
            ->first();
```

```
$randomUser = DB::table('users')
            ->inRandomOrder()
            ->first();
```

Query Builder

- Grouping

```
$users = DB::table('users')
            ->groupBy('account_id')
            ->having('account_id', '>', 100)
            ->get();
```

```
$users = DB::table('users')
            ->groupBy('first_name', 'status')
            ->having('account_id', '>', 100)
            ->get();
```

Query Builder

- Limit, & Offset

```
$users = DB::table('users')->skip(10)->take(5)->get();
```

```
$users = DB::table('users')
            ->offset(10)
            ->limit(5)
            ->get();
```

Query Builder

- Thêm dữ liệu

```
DB::table('users')->insert(
    ['email' => 'john@example.com', 'votes' => 0]
);
```

```
DB::table('users')->insert([
    ['email' => 'john@example.com', 'votes' => 0],
    ['email' => 'taylor@example.com', 'votes' => 0]
]);
```

```
$id = DB::table('users')->insertGetId (
    ['email' => 'john@example.com', 'votes' => 0]
);
```

Query Builder

- Cập nhật dữ liệu

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

```
DB::table('users')
   ->updateOrInsert(
      ['email' => 'john@example.com', 'name' => 'John'],
      ['votes' => '2']
   );
```

TS1: Điều kiện

TS2: Giá trị cập nhật

Nếu tồn tại dòng dữ liệu thỏa mãn <điều kiện> thì cập nhật giá trị (TS2), ngược lại thì thêm dòng dữ liệu mới (gộp TS1 và TS2)

Query Builder

- Xóa dữ liệu

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

```
DB::table('users')->where('votes', '>', 100)->delete();
```

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

Nội dung

- Cấu hình kết nối cơ sở dữ liệu
- Migrations
- Query Builder
- Seeding

Seeding

- Tạo dữ liệu mẫu
- Cập nhật / xóa dữ liệu trong bảng
- Các class Seed được lưu trong thư mục database/seeds

Seeding

- Tạo seeder:

```
php artisan make:seeder <tên Seeder>
```

Seeding

- Ví dụ:

```php
<?php

use Illuminate\Support\Str;
use Illuminate\Database\Seeder;
use Illuminate\Support\Facades\DB;

class UsersTableSeeder extends Seeder
{
    /**
     * Run the database seeds.
     *
     * @return void
     */
    public function run()
    {
        DB::table('users')->insert([
            'name' => Str::random(10),
            'email' => Str::random(10).'@gmail.com',
            'password' => bcrypt('password'),
        ]);
    }
}
```

Seeding

- Thực thi seeder
  - B1: Gọi thực thi các class Seeder trong class DatabaseSeeder

```php
<?php

use Illuminate\Support\Str;
use Illuminate\Database\Seeder;
use Illuminate\Support\Facades\DB;

class DatabaseSeeder extends Seeder
{
    /**
     * Run the database seeds.
     *
     * @return void
     */
    public function run()
    {
        $this->call([
                UsersTableSeeder::class,
                PostsTableSeeder::class,
                CommentsTableSeeder::class
        ]);
    }
}
```

Seeding

- Thực thi seeder
  - B2: Tạo lại **autoloader** của **Composer**

```
composer dump-autoload
```

  - B3: thực thi seeder (lớp DatabaseSeeder)

```
php artisan db:seed
```

Seeding

- Thực thi seeder
  - Thực thi seeder cho class Seeder được chỉ định

```
php artisan db:seed --class=<tên Seeder>
```

  - Thực thi seeder kèm với **migrate:refresh**

```
php artisan migrate:refresh --seed
```

Bài tập

- Viết các lớp Seeder để tạo dữ liệu mẫu (<= 5 dòng dữ liệu) cho bảng:
  - Lĩnh vực
  - Quản trị viên
  - Gói credit