Laravel 8 Has One Through Eloquent Relationship Tutorial

May 3, 2021 by Sanjay Kumar

Table of Contents

Reading Time: 7 minutes 4,887 Views

Laravel eloquent relationship is a very important feature which connects one or more tables in a chain. This is the substitute of joins in laravel.

Laravel provides these following relationships –

- **One To One**
- **One To Many**
- **Many to Many**
- One To Many (Inverse) / Belongs To
- **Has One Through**
- **Has Many Through**

Eloquent relationships are defined as methods on your Eloquent model classes. Inside this article we will see the concept of laravel 8 Has One through Eloquent relationship.

This article will give you the detailed concept of about implementation of Has One through relationship in laravel.

Person -> Broker -> Home

Let's get started.

Installation of Laravel Application

Laravel Installation can be done in two ways.

- Laravel Installer
- By using composer

Laravel Installer

To install Laravel via Laravel installer, we need to install it's installer first. We need to make use of composer for that.

```
$ composer global require laravel/installer
```

This command will install laravel installer at system. This installation is at global scope, so you type command from any directory at terminal. To verify type the given command –

```
$ laravel
```

This command will open a command palette of Laravel Installer.

To create ad install laravel project in system,

```
$ laravel new blog
```

By using composer

Alternatively, we can also install Laravel by Composer command **create-project**.

If your system doesn't has composer Installed, **Learn Composer Installation Steps**.

Here is the complete command to create a laravel project-

```
$ composer create-project --prefer-dist laravel/laravel blog
```

After following these steps we can install a Laravel application into system.

To start the development server of Laravel –

```
$ php artisan serve
```

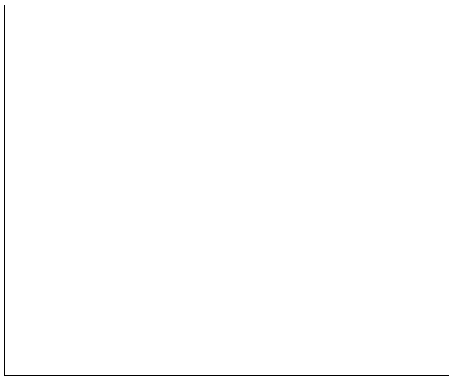
This command outputs –

Starting Laravel development server: http://127.0.0.1:8000

Assuming laravel already installed at system.

Create Database & Connect

To create a database, either we can create via Manual tool of PhpMyadmin or by means of a mysql command.



```
CREATE DATABASE laravel_app;
```

To connect database with application, Open **.env file** from application root. Search for **DB_** and update your details.

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

Create Migrations

We need few migration files –

- People migration to store people data
- Broker migration to store brokers
- Home Migration to store homes data

Open project into terminal and run these artisan commands.

```
$ php artisan make:migration create_people_table

$ php artisan make:migration create_brokers_table

$ php artisan make:migration create_homes_table
```

It will create two migration files 2021_05_03_162204_**create_people_table.php**, 2021_05_03_162246_**create_brokers_table.php,** and 2021_05_03_162253_**create_homes_table.php** at location **/database/migrations**.

Open 2021_05_03_162204_**create_people_table.php** and write this complete code into it.

Migration #1

```php
<?php

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

class CreatePeopleTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('people', function (Blueprint $table) {
            $table->id();
            $table->string("name", 120);
            $table->string("email", 120)->unique();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
```

Open 2021_05_03_162246_**create_brokers_table.php** and write this code into it.

Migration #2

```
<?php

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

class CreateBrokersTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('brokers', function (Blueprint $table) {
            $table->id();
            $table->string("name", 120);
            $table->string("email", 120)->unique();
            $table->unsignedBigInteger('person_id');
            $table->timestamps();
            $table->foreign('person_id')->references('id')->on('people');
        });
    }

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

Open 2021_05_03_162253_**create_homes_table.php** and write this complete code into it.

Migration #3

```
<?php

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

class CreateHomesTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
```

```
        Schema::create('homes', function (Blueprint $table) {
            $table->id();
            $table->string("title", 155);
            $table->text("location");
            $table->unsignedBigInteger('broker_id');
            $table->timestamps();
            $table->foreign('broker_id')->references('id')->on('brokers');
        });
    }

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

Run Migrations

Next, we need to create tables inside database.

```
$ php artisan migrate
```

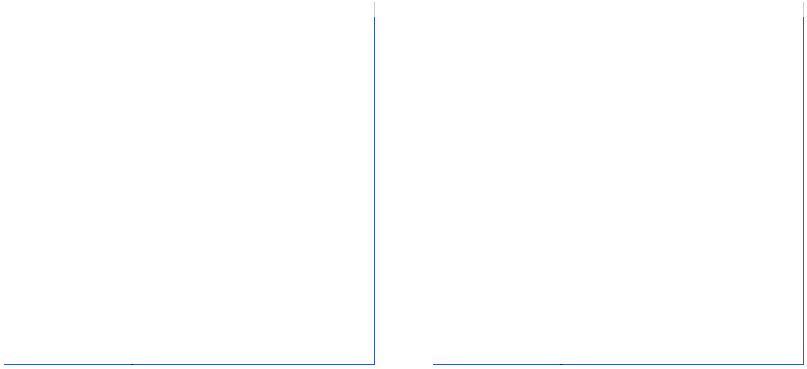
This command will create tables inside database.

Create Model

Next, we need to create three models. Back to terminal and run these artisan commands.

```
$ php artisan make:model Person
$ php artisan make:model Broker
$ php artisan make:model Home
```

These commands will create three files **Person.php, Broker.php** & **Home.php** at /app/Models folder.



Open **Person.php** and write this complete code into it.

Person.php

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use App\Models\Home;
use App\Models\Broker;

class Person extends Model
{
    use HasFactory;

    public function homeInformation()
    {
        return $this->hasOneThrough(Home::class, Broker::class);
    }
}
```

$this->hasOneThrough(Home::class, Broker::class); This line is implementing the relationship i.e Has One Through

Rest for all models like for **Broker.php** and **home.php** code will be same as default code.

Sample Data Insertion

Open mysql and run these queries to insert dummy data into people, brokers and homes table.

Data for People Table

```
#PeopleTable
--
-- Dumping data for table `people`
--

INSERT INTO `people` (`id`, `name`, `email`, `created_at`, `updated_at`) VALUES
(1, 'Sanjay Kumar', 'sanjay@gmail.com', NULL, NULL),
(2, 'Ashish Kumar', 'ashish@gmail.com', NULL, NULL);
```

Data for Brokers Table

```
#BrokersTable
--
-- Dumping data for table `brokers`
--

INSERT INTO `brokers` (`id`, `name`, `email`, `person_id`, `created_at`, `updated_at`) VALUES
(1, 'Broker 1', 'broker1@gmail.com', 1, NULL, NULL),
(2, 'Broker 2', 'broker2@gmail.com', 2, NULL, NULL);
```

Data for Homes Table

```
#HomesTable
--
-- Dumping data for table `homes`
--

INSERT INTO `homes` (`id`, `title`, `location`, `broker_id`, `created_at`, `updated_at`) VALUES
(1, 'Home 1', 'Location 1, Street 1, Sample', 2, NULL, NULL),
(2, 'Home 2', 'Location 2, Street 2, Sample ', 1, NULL, NULL);
```

Calling Method at Controller

Open any controller say **SiteController.php** file, we have created one method in which we used model method as a property.

```
#Controller
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\Person;

class SiteController extends Controller
{
    // To get home detail of a person
    public function getHome()
    {
        return Person::find(1)->homeInformation;
    }
}
```

Person::find(1)->homeInformation; It will return the home information of person by it's ID.

Create Routes

Open **web.php** from /routes folder and add this route into it.

```
web.php
# Add this to header
use App\Http\Controllers\SiteController;

Route::get('home-information', [SiteController::class, 'getHome']);
```

Application Testing

Open project to terminal and type the command to start development server

```
$ php artisan serve
```

Get Home Information by Person ID – http://127.0.0.1:8000/home-information

If you liked this article, then please subscribe to our **YouTube Channel** for PHP & it's framework, WordPress, Node Js video tutorials. You can also find us on **Twitter** and **Facebook**.

Find More on Laravel 8 Articles here

- How to Create Multi Language Website in Laravel 8
- How To Read XML File in Laravel 8 – Example
- How To Upload And Save XML Data in Laravel 8
- Laravel 8 Ajax Post Request Tutorial
- Laravel 8 Authentication using Jetstream with Inertia Js
- Laravel 8 Authentication using Jetstream with Livewire
- Laravel 8 Authentication with Breeze Tutorial
- Laravel 8 Clear Cache of Route, View & Config
- Laravel 8 Cron Job Task Scheduling Tutorial
- Laravel 8 DataTable Ajax Pagination with Search And Sort
- Laravel 8 Firebase Push Notification Tutorial
- Laravel 8 Form Validation Methods
- Laravel 8 Installation Guide – PHP Framework
- Laravel 8 Layouts And Views Complete Guide
- Laravel 8 Routing Tutorial Step by Step Guide
- Laravel 8 Send Mail using Gmail SMTP Server
- Laravel 8 Send Push Notification to Android Using Firebase
- Laravel 8 Send Push Notification to IOS Using Firebase
- Laravel 8 Stub Customization

Related Posts:

- Laravel 8 Has Many Through Eloquent Relationship Tutorial
- Laravel 8 One to One Eloquent Relationship Tutorial
- Laravel 8 One to Many Eloquent Relationship Tutorial
- Laravel 8 Many to Many Eloquent Relationship Tutorial
- Eloquent Mutators and Accessors in Laravel 8 Tutorial
- Laravel 8 Store Log Of Eloquent SQL Queries

Laravel 8

< How to Get Logged in User Data in Laravel 8 Tutorial

> Laravel 8 Has Many Through Eloquent Relationship Tutorial

Find Us on Youtube

Online Web Tutor Youtube

Advertisement

Popular Posts

Sorry. No data so far.

Categories

- Blogging (1)
- CakePHP 4 (11)
- CodeIgniter 4 (156)
- Education (1)
- jQuery & Javascript (39)
- Laravel 8 (152)
- MySQL (10)
- Node Js (15)
- PHP Miscellaneous (33)

Recent Posts