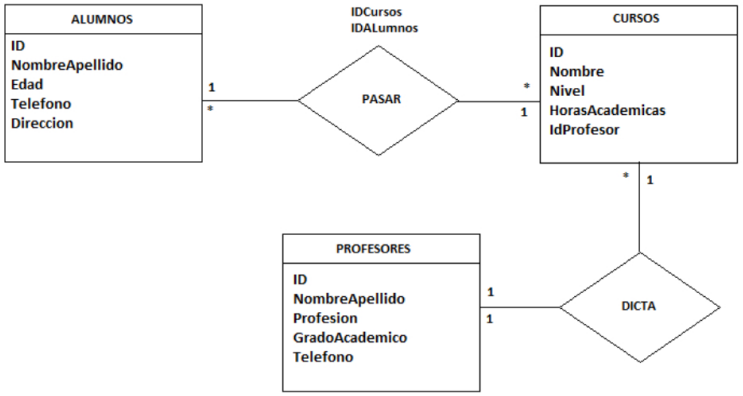
# DATU BASEAK LARAVELEN



Lehenengo gauza egingo duguna, datu base bat sortzea izango da, proiektu honetarako, gure phpAdmin-en datu base bat "**eskola**" izeneko datu basea sortuko dugu, momentuz inolako taularik gabe. Bakarrik databasea sortuko dugu, gainontzekoa, taulak, datuak etab. Artisanekin egingo dugu.

Orain, Laraveleri, esan behar diogu zein den erabiliko dugun data basea, horretarako, erroko direktorioan, badago fitxategi bat "**.env**" izenekoa. **DB_DATABASE**=eskola

Gure DB-ak 3 taula izango ditu, hurrengo eskeman ikusi dezakezuenez, baina, momentuz guk "Alumnos" taularekin hasiko dugu.




## Migrazioa

1. Behin hori eginda, **migrazioa** egingo dugu. Honi ezker, Laravelek guk adierazitako taulak sortuko ditu gure datu basean sql sententziak erabili barik. Hau guztia **Artisani** ezker oso errez egingo dugu.

Hau modeloarekin edo modelo barik egin daiteke. Erosoena modeloarekin egitea izango da, baina modu honetan, Laraveleko konbentzioaren arabera egingo da, hau da, taula modeloaren izena letra larriz eta singularrean eta aldiz taularen izena letra txikiz eta pluralean izango da, baina noski, ingelerazko plurala erabiliko du.

Modeloa Alumno bada, taula alumnos izango da, baina Profesor izatekotan profesors izango litzateke, eta euskararen kasuan ere Ikasle, ikasles izango litzateke. Ahal den einean Laraveleko konbentzioak jarraituko ditugu.

Gogoratu migrazioa egiterakoan beti dagokigun direktorioan egon behar dugula kokatuta.

- **Migrazioa + modeloa:**

```
php artisan make:Model Alumno -m
```

- **Migrazioa bakarrik** sortu nahi izango bagenu eran egingo dugu:

```
php artisan make:migration create_alumnos_table
```

  Aurreko migrazioan jarritako izena konbentzio bat da, modu honetan, migrazioa sortzerakoan, up() metodoaren barruan create metodoa sortuko du, eta honen barruan ikasles taula.

```
php artisan make:migration alumnos
```

  Horrela bakarrik jartzekotan, ez lituzke metodo horiek sortuko.

```
PS C:\xampp\htdocs\laravel2324\eskola2324> php artisan make:Model Ikasle -m

  INFO  Model [C:\xampp\htdocs\laravel2324\eskola2324\app/Models/Ikasle.php] created successfully.

  INFO  Migration [C:\xampp\htdocs\laravel2324\eskola2324\database\migrations/2023_10_26_171809_create_ikasles_table.php
] created successfully.

PS C:\xampp\htdocs\laravel2324\eskola2324>
```

Konprobatuko dugu zelan migrazioa ondo sortu duen, horretarako bagoaz: *database > migrations > 2023_10_26_171809_create_alumnos_table.php* fitxategia zabaltzera.

Momentuz data basean ez da ezer sortu.

2. Gure php honetan sartuko gara eta taula zelakoa izango den adieraziko dugu. Ikusten dugunez, Laravelek, up metodoan kodea sortu du jada, kasu honetan ikasles taula bat sortuko luke bi eremurekin, id eta timestamps eramuak. Aldatuko dugu hori guk nahi dugun datuekin:

```
        Schema::create('alumnos', function (Blueprint $table) {
            $table->id();
            $table->string('nombre', 75);
            $table->integer('edad');
            $table->string('telefono', 35)->nullable();
            $table->string('direccion', 75)->nullable();
            $table->timestamps();
        });
```

Orain bai, artisanekin, migrazioa **exekutatuko** dugu:

```
php artisan migrate
```

Ondo egin duen ala ez jakiteko, datu basera sartuko gara. Fijatu zeitzuk diren sortutako eremuak.

**OHARRA:** Ikusi dezakegunez, migrazioa sortzerakoan, Laravelek 3 zutabe sortuko ditu, lehenengoa *id* eta beste biak timestams-en bidez sortuko dituen beste bi, *created_at* eta *updated_at*. Bi zutabe hauek, datuak sortu eta aldatu egiten dituenean eguneratuko egiten ditu, datu horiek sartu edo aldatu diren data eta orduarekin.

Hori nahi ez izatekotan, migrazioan ezabatu beharko dugu, baina gogoratu ere, modeloa sortzerakoan eremu hori false jartzen:

```
public $timestamps = false;
```

Ikusten dugunez, migrazioa sortzerakoan, gure Migration klaseak bi funtzio izango ditu, up() eta down(), lehenengoan taula sortuko da guk esandako eramuekin. Aldiz down funtzioari deituko zaio migrazioa desegin nahi izatekotan, hori dela eta funtzio honetan soilik taula ezabatuko da.

## Seeder

Seeder-ak Laraveleko konponente batzuk dira datu basean datuak sartu ahal izateko. Hauek normalean garapen prozesuan datuak sartzeko edo aplikazioren hasieran behar diren datuak sartzeko balio dute.

**Zelan egingo genuke hau?**

Gure direktorioan egonda, artisaneko sententzia hau idatziko genuke IkasleSeeder sortzeko:

```
php artisan make:seeder AlumnoSeeder
```

database > seeders-en sortuko du fitxategia.

Fitxategian, gure modeloa zein den adieraziko diogu, horretarako Ikasle klasearen erreferentzia sortuko dugu:

```
use App\Models\Alumno;
```

$ikasle aldagaia sortu, Ikasle klasea ezarriz eta bere propietate guztiak beteko ditugu irudian agertzen den moduan, ikusten duzuenez gure datu baseko taulak objeto moduan erabiliko ditugu.

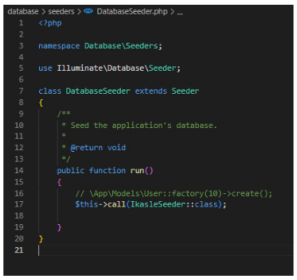
```php
use Illuminate\Database\Seeder;
use App\Models\Ikasle;


class IkasleSeeder extends Seeder
{
    /**
     * Run the database seeds.
     *
     * @return void
     */
    public function run()
    {
        //Ikasle klasean datuak sartuko ditugu
        $ikasle = new Ikasle();
        $ikasle->izen_abizen = 'Andoni Egaña';
        $ikasle->adina = 18;
        $ikasle->telefonoa = '658741122';
        $ikasle->helbidea = 'Pozas, 33';
        $ikasle->save();

        $ikasle = new Ikasle();
        $ikasle->izen_abizen = 'Iñaki Larrinaga';
        $ikasle->adina = 20;
        $ikasle->telefonoa = '658775532';
        $ikasle->helbidea = 'Rodriguez Arias, 54';
        $ikasle->save();

        $ikasle = new Ikasle();
        $ikasle->izen_abizen = 'Garazi Pascual';
        $ikasle->adina = 18;
        $ikasle->telefonoa = '699236859';
        $ikasle->helbidea = 'Ornilla, 76';
        $ikasle->save();

    }
```




Artisanek, Seederak exekutatzerakoan, beti **DatabaseSeeder** klasea deituko du; beraz, guk gure seederra exekutatzeko, klase horretako *run()* metodotik gure klasea "deituko" (instanciar) dugu.

Bat baino gehiago deitu daiteke metodo honetatik, eskuz idatzitako ordenean exekutatko litzateke.

```
$this->call(AlumnoSeeder::class);
```

Behin hori guztia eginda, gure datu basean datuak kargatzeko, seeder-a exekutatuko dugu:

```
php artisan db:seed
```

## Factories

*Dokumentazioa: https://laravel.com/docs/10.x/seeding*

Seederrak aukera onak dira oso datu gutxi dituguneko, baina badugu beste aukera bat, frogetarako datu asko behar dituguneko, horretarako **Factoriak** ditugu, garapenerako pentsatutako klase batzuk dira hauek.

Factoriak nahiko errezak dira erabiltzeko, hurrengo pausuak jarraitu behar dira:

1. IkasleFactory deitutako klase bat sortuko dugu:
   ```
   php artisan make:factory AlumnoFactory
   ```
2. Ikusten dugunez, database > factories karpetan fitxategi at sortu du IkasleFactory.php
3. Definition funtzio barruan, taulako eremu bakoitzeko balio aleatorio bat emango diogu, hau Faker konponentari ezker egin dezakegu.
4. **Faker**, PHP libreria bat da, datu aleatorioak sortzen dituena, sortu ditzakegu, izenak, helbidek, telefonoan etab...

   Faker sortu ahal dituen datu guztiak ikusteko esteka honetan sartu
5. Gure adibidearekin jarraituz, guk horrelako erregistro bat sortuko genuke:

```php
lass IkasleFactory extends Factory

    /**
     * Define the model's default state.
```

```
     * @return array
     */
    public function definition()
    {
        return [
            // Ikasle taulan datuak sartzeko erregistroa
            "izen_abizen"=>$this->faker->firstName() . ' ' . $this->faker->LastName(),
            "adina"=>$this->faker->randomElement([18, 19, 20, 21]),
            "telefonoa"=>$this->faker->phoneNumber(),
            "helbidea"=>$this->faker->address()
        ];
    }
```

6. Baina orain, DatubaseSeeder-en adierazi behar diogu lehenengo Seederra deitu beharrean Factorya exekutatzea. Kodean ikusi daitekenez, adierazi diezaiokugu zenbat biderretan egingo duen:

```
use App\Models\Alumno;

public function run()    {
       // $this->call(AlumnoSeeder::class);
       Alumno::factory(10)->create();
    }
```

Behin hau eginda, seederra exekutatuko dugu berriro:

```
php artisan db:seed
```