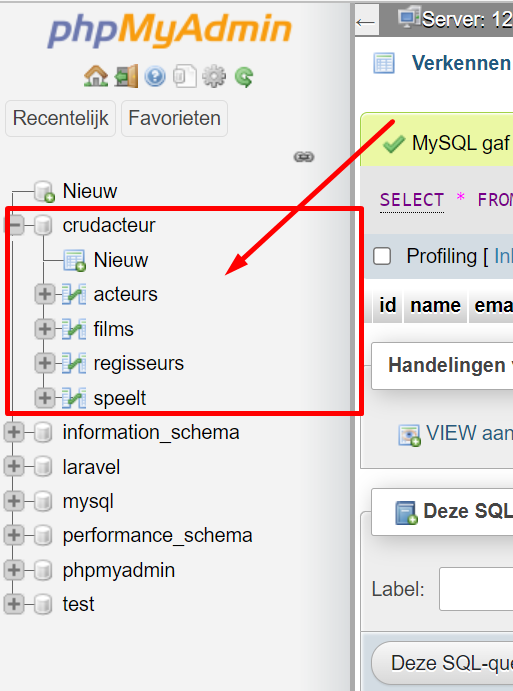
Opdracht 2 : we gaan een migration bestand maken voor acteur.

We gaan even spieken hoe de database eruit ziet bij crud acteurs.

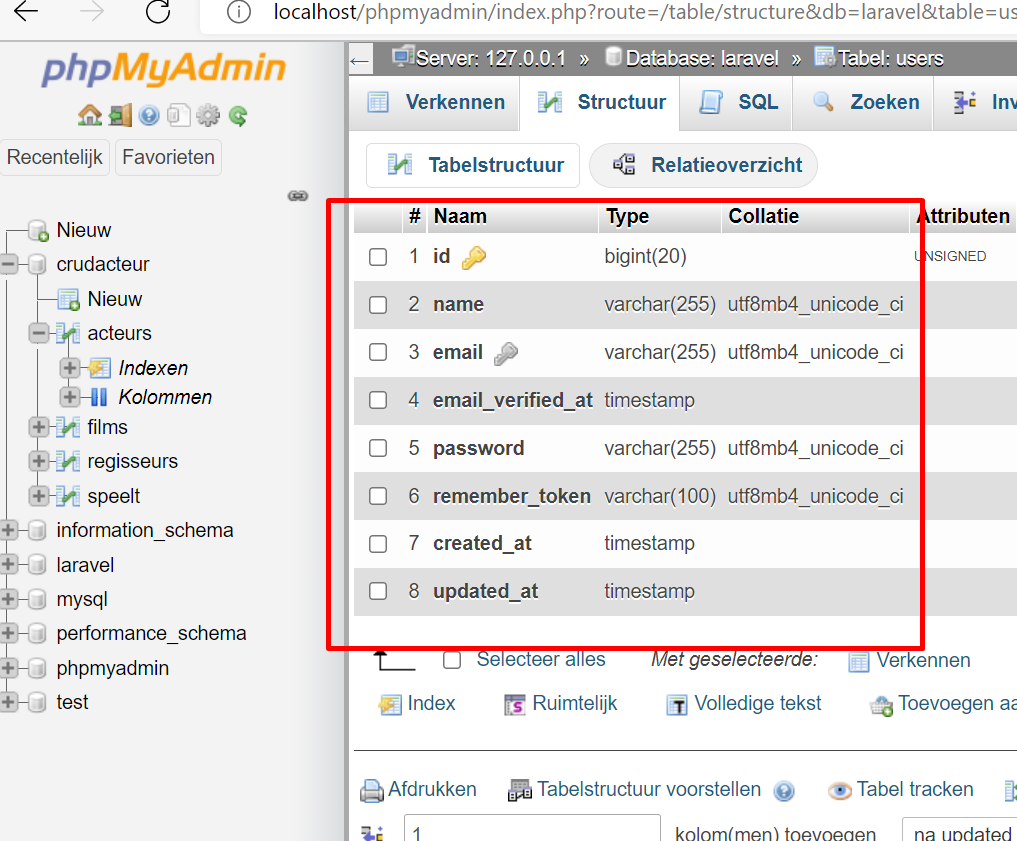
Er zit namelijk een export van de database.

Mochten we geen export hebben dan moeten we wel een database schema hebben. Duidelijk toch?

We openen phpMyAdmin (heel goed)



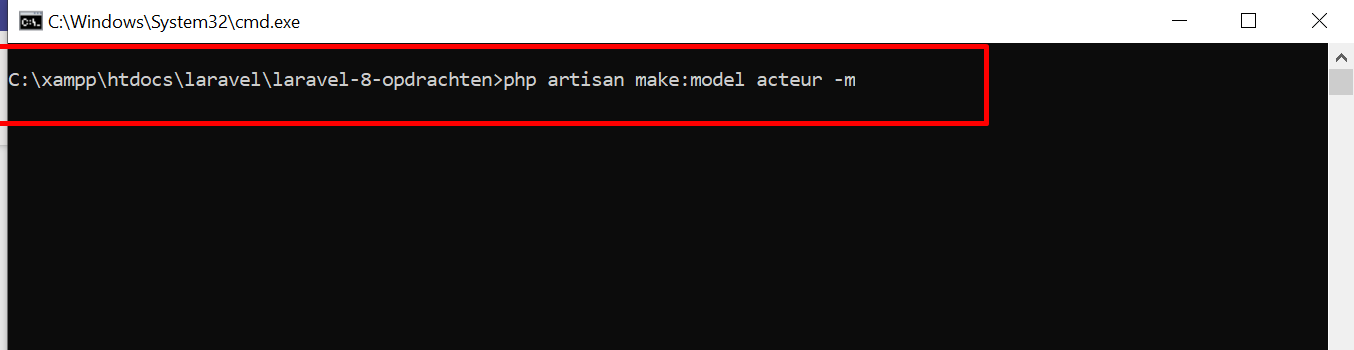
De bekijken de structuur van tabel acteur



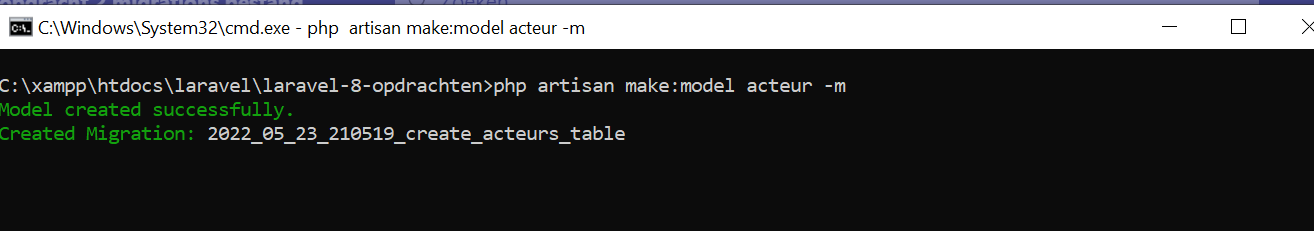
Open een command terminal

Voer hier in de opdracht:

php artisan make:model acteur –m



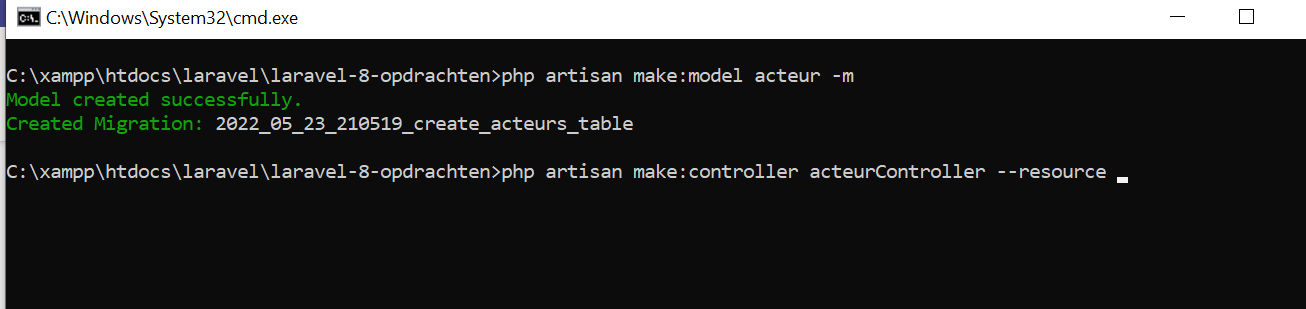
Als het goed gaat zie je een soortgelijk scherm:



We gaan nu ook gelijk een resourceController hiervan maken.

Wat was dit ook alweer? Een resourceController gaat er voor zorgen dat de crud acties (create, update, delete en insert) correct uitgevoerd worden.

php artisan make:controller acteurController –resource



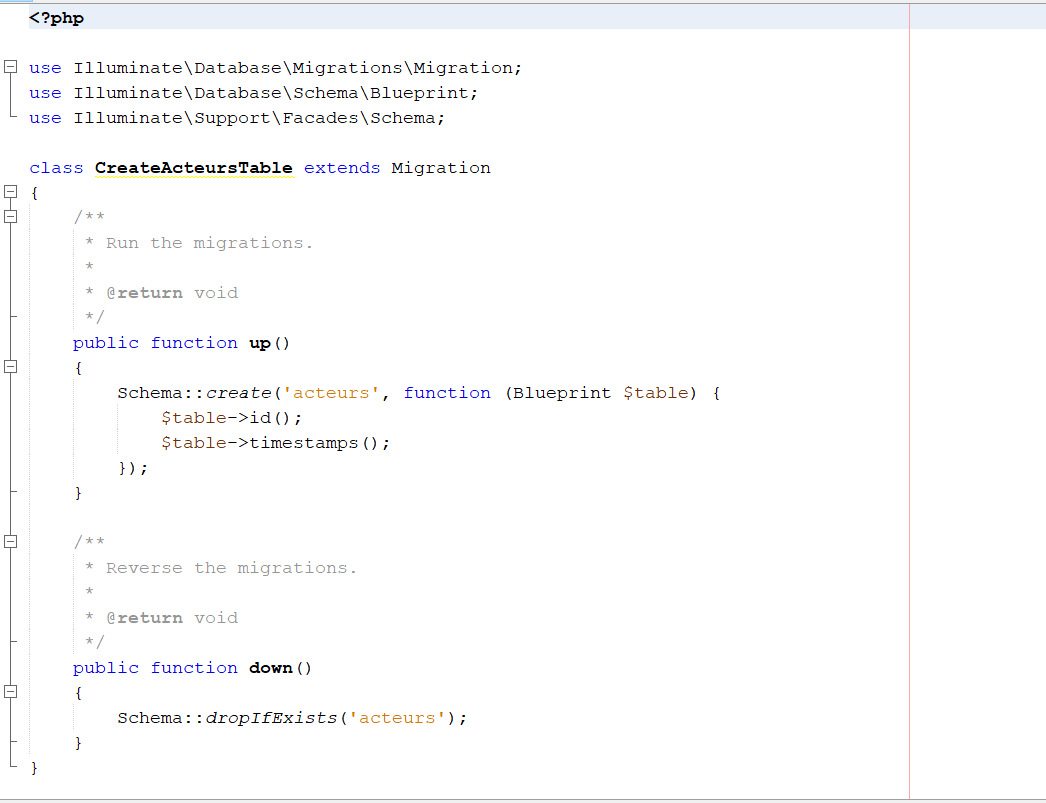
De migrationbestand aanpassen:

Dit gaat er voor zorgen dat de tabel correct wordt opgezet.

Waar zit dit? In je project folder --> database -> migrations \ <timestamp+> . Create\_acteurs\_table.php

(als je nu nog in je terminal kijken, dan zul je zien dat artisan je ook wijst welke bestand er aangemaakt is om verder aan te passen)

Op hiervoor de migration bestand voor acteur:



Welke kolommen en typen kan Laravel aan?

Command Description

$table->bigIncrements('id'); Incrementing ID using a "big integer" equivalent.

$table->bigInteger('votes'); BIGINT equivalent to the table

$table->binary('data'); BLOB equivalent to the table

$table->boolean('confirmed'); BOOLEAN equivalent to the table

$table->char('name', 4); CHAR equivalent with a length

$table->date('created\_at'); DATE equivalent to the table

$table->dateTime('created\_at'); DATETIME equivalent to the table

$table->decimal('amount', 5, 2); DECIMAL equivalent with a precision and scale

$table->double('column', 15, 8); DOUBLE equivalent with precision, 15 digits in total and 8 after the decimal point

$table->enum('choices', array('foo', 'bar')); ENUM equivalent to the table

$table->float('amount'); FLOAT equivalent to the table

$table->increments('id'); Incrementing ID to the table (primary key).

$table->integer('votes'); INTEGER equivalent to the table

$table->longText('description'); LONGTEXT equivalent to the table

$table->mediumInteger('numbers'); MEDIUMINT equivalent to the table

$table->mediumText('description'); MEDIUMTEXT equivalent to the table

$table->morphs('taggable'); Adds INTEGER taggable\_id and STRING taggable\_type

$table->nullableTimestamps(); Same as timestamps(), except allows NULLs

$table->smallInteger('votes'); SMALLINT equivalent to the table

$table->tinyInteger('numbers'); TINYINT equivalent to the table

$table->softDeletes(); Adds deleted\_at column for soft deletes

$table->string('email'); VARCHAR equivalent column

$table->string('name', 100); VARCHAR equivalent with a length

$table->text('description'); TEXT equivalent to the table

$table->time('sunrise'); TIME equivalent to the table

$table->timestamp('added\_on'); TIMESTAMP equivalent to the table

$table->timestamps(); Adds created\_at and updated\_at columns

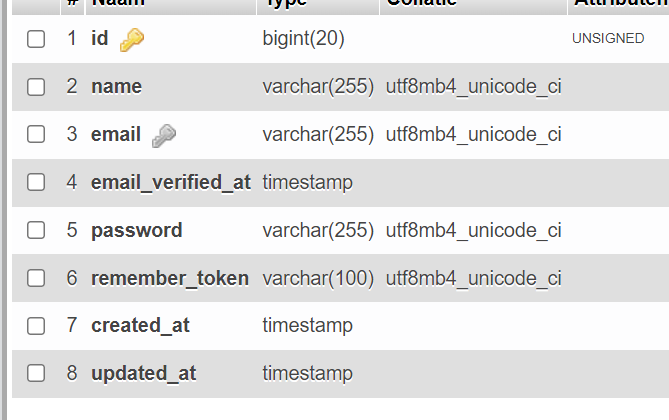
$table->rememberToken(); Adds remember\_token as VARCHAR(100) NULL

->nullable() Designate that the column allows NULL values

->default($value) Declare a default value for a column

->unsigned() Set INTEGER to UNSIGNED

Dit is wat we moeten beschrijven:

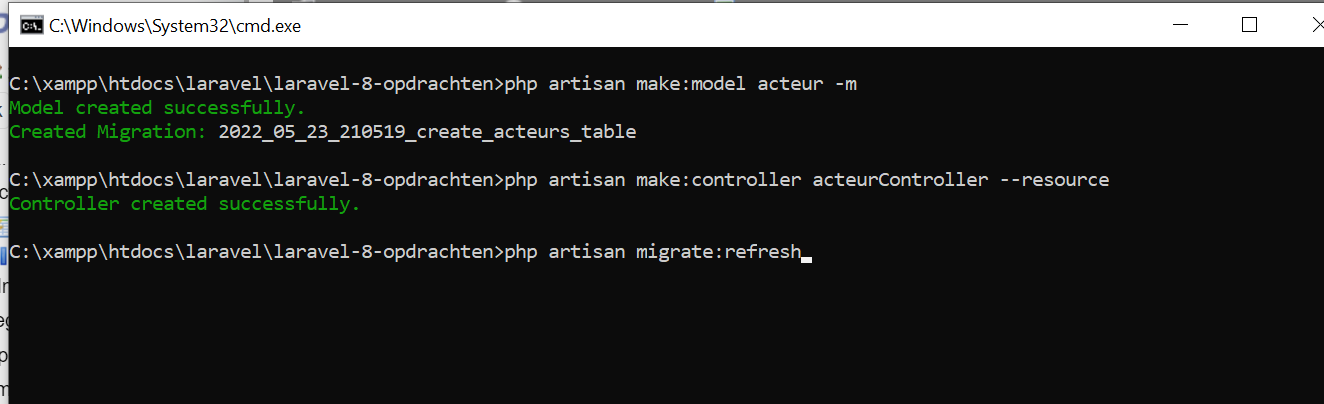


Je moet in je migrationbestand dus alleen nog toevoegen of aanpassen welke kolommen je nodig hebt.

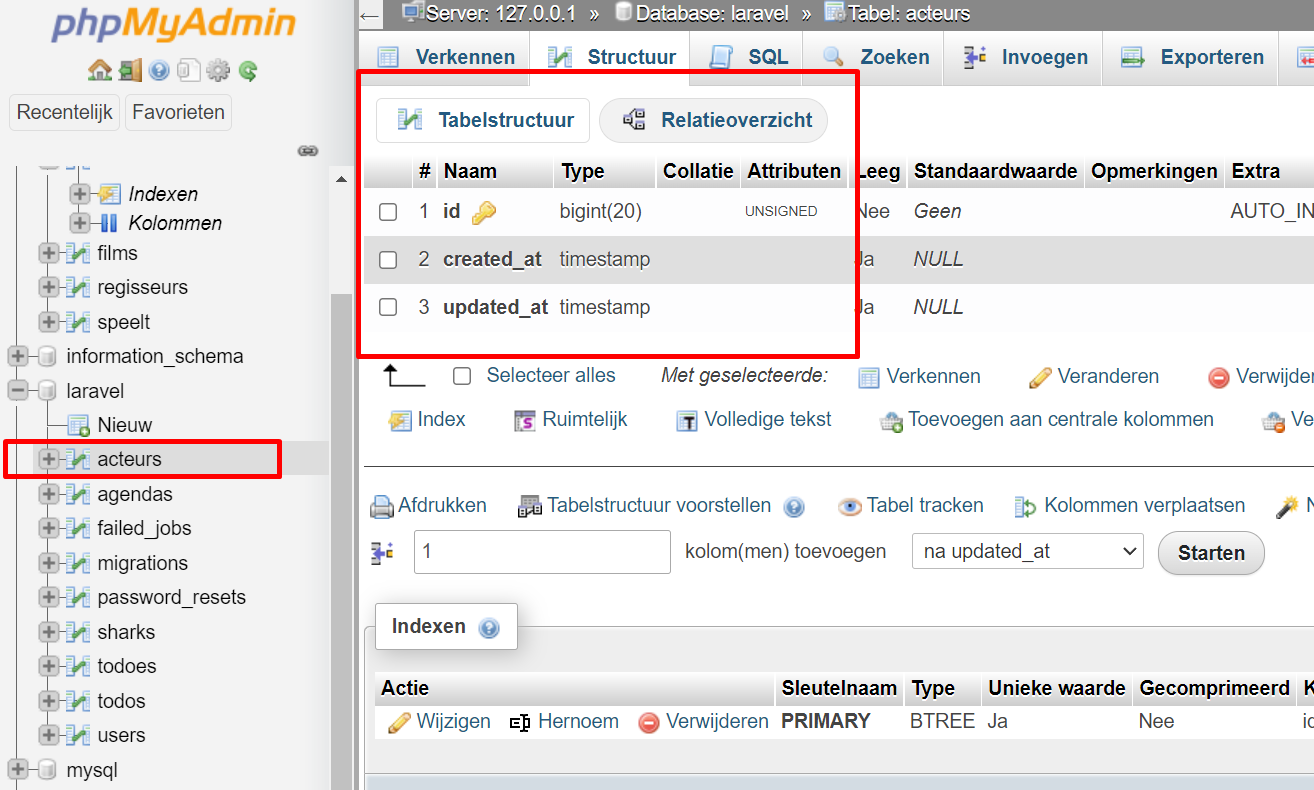
Vraag: maakt de volgorde eigenlijk iets uit?

Denk je klaar te zijn, voer dan de volgende commando uit:

>php artisan migrate:refresh



Kijk nu even in je phpMyadmin:



Pas je migrationbestand aan:

public function up()

{

Schema::create('acteurs', function (Blueprint $table) {

$table->id();

$table->string('ACHTERNAAM', 16);

$table->string('VOORNAAM', 15);

$table->timestamps();

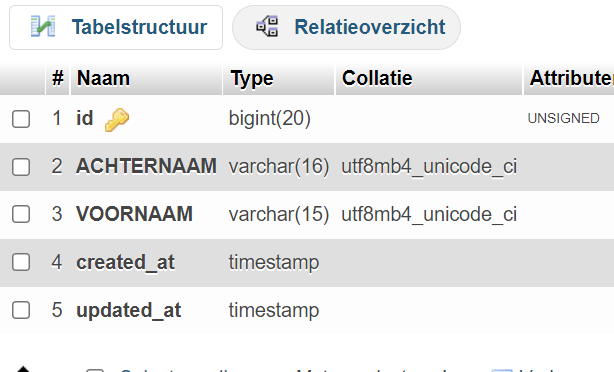
});

}

Voer de volgende commando uit:

>php artisan migrate:refresh

Kijk nog een keer in je phpmyadmin:



BELANGRIJK om te onthouden.

Gebruik ALTIJD een 'id' kolomnaam voor je primary key. Doe je dit niet dan gaat het later fout met joins.

Daarvoor geldt dus wel: bij het omzetten van een database moet je kijken naar je nieuwe situatie.

Maar verder is met de tabel die we nu aangemaakt hebben zo goed als identiek.

Binnen Laravel komen we verder wel uit.