

06 Relaties, land en talen

Inleiding

We hebben het in de vorige les gehad over 1:1 relaties; elke land heeft precies één hoofdstad. Maar hoe zit het met een 1-op-veel (1:N) relatie?

In één land kunnen 1 of meerdere talen worden gesproken. In deze les gaan we uitvinden hoe dat werkt in Yii.

Wat leren we?

- We leren hoe we in Yii een 1:n relatie kunnen definiëren.
- We leren hoe we in Yii met de debug functie objecten kunnen inzien.

Talen in een land

In de *World* database staat een tabel *countrylanguage*. In deze tabel staat per land welke talen er worden gesproken. Per taal wordt ook aangegeven welk percentage van de bevolking deze taal spreekt.

De relatie tussen *country* en *countrylanguage* is 1:N. Namelijk in een land worden **1 of meer** talen gesproken.

We willen een overzicht maken dat er als volgt uit gaat zien:

[image-1594550267859.png](#)

In dit voorbeeld zien we dat in Indonesië 8 talen worden gesproken. We gaan ons overzicht aanpassen, zodat we per land alle talen die er worden gesproken netjes in de tabel worden weergegeven zoals hierboven in het voorbeeld is te zien.

Relatie in het model

Hoe werkt dit?

Bij de 1:1 relatie hierboven konden we via `$country->hoofdstad['Name']` de naam van de hoofdstad uit de gelinkte tabel opvragen. Als dit niet duidelijk is, lees dan het stuk over 1:1 relatie in [de vorige les](#) nog een keer goed door!

Bij een 1:N relatie werkt het hetzelfde als bij de 1:1 relatie. Dus in het model (*models/Country.php*) wordt de relatie beschreven alleen nu gebruiken we *hasMany* (in plaats van *hasOne*)

```
public function getCountrylanguages()
{
    return $this->hasMany(Countrylanguage::className(), ['CountryCode' => 'Code']);
}
```

Controleer of je het model hebt aangemaakt van *Countrylanguage* tabel.

De relatie moet je ook aanmaken, maar die wordt in het Country model aangemaakt. Elke country heeft namelijk een relatie met één of meer talen (*hasMany*).

Hoofd en kleine letters

Let even op de hoofd en kleine letters! We hebben de tabel in de database, de model file en de relatienaam. En allemaal hebben ze een andere spelling voor de hoofdletters!

Even alles op een rijtje:

tabel naam in database	Countrylanguage
model naam	CountryLanguage
relatienaam in Country.php	Countrylanguages

Relatie in de view

We kunnen nu vanuit de view (*views/country/overzicht.php*) via `$country->countryLanguages` de taal opvragen.

Als je de Gridview widget gebruikt, dan noem je het veld (in de Widget)
`country.countryLanguages`

Er is alleen één belangrijk verschil, omdat we geen 1:1 relatie hebben, maar een 1:N, krijgen we geen variabele terug maar een array. We moeten het dus met een loop door het array heen lopen.

Dit gaan we demonstreren met behulp van de debug functie die we in [les 1 \(optioneel\)](#) hadden toegevoegd. Als je dit nog niet hebt gedaan dan moet je dat nu doen.

Debugging

Ga nu naar je view en voeg bovenaan in de loop waarmee je het overzicht afdruckt, de regel `dd($countries)` toe. *dd* staat voor dump & die. Dus *dd()* laat de variabele zien en stopt met het uitvoeren van het programma.

```
<?php
[...
]echo "<table border=1>";
[...
]foreach ($countries as $country) {
[...
][...]// hier staat al code voor eerste kolommen
[...
][...]// voeg hier deze kolom toe
[]echo "<td>";
[]d($country->countrylanguages);
[]echo "</td>";
[...
}
...
```

Als we even de border van de tabel aanzetten (regel 3) dan is het iets makkelijker om de output te lezen.

Als je goed kijkt dan zie je in de rechter kolom een array van objecten. In deze Yii-objecten staat telkens een 'Language'.

[image.1616504143444.png](#)

Om de taal te af te drukken moeten we eerst het element array aanwijzen:

```
$country->countrylanguages[0]
$country->countrylanguages[1]
$country->countrylanguages[2]
$country->countrylanguages[3]
```

En van elk element moeten we de 'Language' hebben. Dat kun je in de dump (plaatje hierboven) goed zien.

```
$country->countrylanguages[0]->Language
$country->countrylanguages[1]->Language
```

```
$country->countrylanguages[2]->Language
```

```
$country->countrylanguages[3]->Language
```

Je weet natuurlijk niet hoeveel talen er in elk land worden gesproken. De lengte van het array is in dit voorbeeld 4, maar het kunnen er natuurlijk meer of minder zijn.

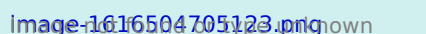
Opdracht

Let op de opdrachten hieronder volgen elkaar op. Er wordt telkens een stapje uitgevoerd.

Uiteindelijk willen we een overzicht waarin per land alle talen die daar worden gesproken, op volgorde worden afgedrukt (zie voorbeeld opdracht 4d). Dit doen we stapje-voor-stapje.

Opdracht 06a

Lees de uitleg hierboven goed door. Druk in de laatste kolom één taal af.



Nu hebben we één taal per land, maar in de meeste landen wordt meer dan één taal gesproken. We gaan nu met een loop door het array `$country->countrylanguages` heen. Gebruik hiervoor de volgende loop.

```
foreach( $country->countrylanguages as $taal) {  
    // vul zelf de juiste variable in om de taal af te drukken  
    echo .....;  
    echo " <br/>";  
}
```

Inleveren

Een schermafdruck yii-06a-jouw-naam met het scherm van jouw gehele browser waarin je het landenoverzicht laat zien en waarbij je één taal afdruckt.

Opdracht 6b

Gebruik de loop zoals hierboven is aangegeven en maak het volgende overzicht.

Image-1616505080100.png

Zorg ervoor (met CSS) dat de tabel er netjes uitgelijnd uit ziet zoals hierboven in het voorbeeld.

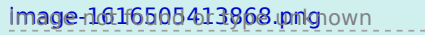
Je hebt nu een 1:N (één op meer) relatie gemaakt. In volgende lessen gaan we hier opnieuw mee aan de slag.

Inleveren

- Een schermafdruck yii-06b-jouw-naam.png waarin je je complete scherm laat zien waarin het landenoverzicht te zien is met alle talen zoals in het voorbeeld.

Opdracht 6c

In de Countrylanguages tabel staat naast de taal ook het percentage. Dit is het percentage van de bevolking dat deze taal spreekt. Zet dit percentage tussen haakjes achter elke taal. Voorbeeld:

 [Image 1616505413868.png](#)

Inleveren

- Een schermafdruck yii-06c-jouw-naam.png waarin je je complete scherm laat zien waarin het landenoverzicht te zien is met talen en percentages zoals in het voorbeeld.

Sorteren

Jammer dat de percentage niet gesorteerd zijn!

Pas de relatie aan in het Country model

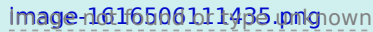
```
public function getCountrylanguages()
{
    return $this->hasMany(CountryLanguage::className(), ['CountryCode' => 'Code'])
        ->orderBy(['Countrylanguage.Percentage' => SORT_ASC]);
}
```

Regel 4 is erbij gekomen. **let op de ; !**

Als je nu het overzicht opnieuw toont dan zie je dat de talen nu oplopend (van laag naar hoog) zijn gesorteerd.

Opdracht 6d

Sorteer de talen op percentage aflopend van hoog naar laag. Op die manier komt de meest gesproken van een land bovenaan.

Image-1616506111435.png

Tip: gebruik Google om te zoeken hoe je de relatie in het model verandert zodat je **aflopend** kunt sorteren.

Nog een tip:

Als je in de database de relaties (via SQL) goed definieert dan worden de relaties in Yii door de model generator automatisch gemaakt. Het aanmaken van relaties in de databases valt buiten deze lessen en kan lastig zijn. Het is dus belangrijk dat je snapt hoe de relaties tussen tabellen in Yii worden gemaakt.

Inleveren

1. Een schermafdruck yii-06d-jouw-naam.png waarin je je complete scherm laat zien waarin het landenoverzicht te zien is zoals in het voorbeeld.
2. De code van de view van country. Voeg in jouw code commentaar toe waarin je jouw naam zet.

--

Revision #9

Created 17 June 2022 15:39:07 by Max

Updated 15 July 2022 09:53:11 by Max