Analyzing the Yii error stack trace

When an error occurs, Yii can display the error stack trace along with the error. A stack trace is especially  
helpful when we need to know what really caused an error rather than just the fact that an error occurred.

Getting ready

1. Create a new yii2 -app-basic application by using the Composer package manager, as described  
   in the official guide at [http://www. yiiframework. c om/doc-2.0/guide -start-installation .html](http://www.yiiframework.com/doc-2.0/guide-start-installation.html).
2. Configure a database and import the following migration:

<?php

use yii\db\Migration;

class m160308\_093234\_create\_article\_table extends Migration  
{

public function up()

{

$this->createTable('{{%article}}', [

'id' => $this->primaryKey(),

'alias' => $this->string()->notNull(),

'title' => $this->string()->notNull(),

'text' => $this->text()->notNull(),

]);

}

public function down()

{

$this->dropTable('{{%article}}');

}

}

1. Generate an Article model using Yii.

How to do it...

Carry out the following steps:

1. Now we will need to create some code to work with. Create  
protected/controllers/ErrorController .php as follows:

<?php

namespace app\controllers;

use app\models\Article;  
use yii\web\Controller;

class ErrorController extends Controller  
{

public function actionIndex()

{

$article = $this->findModel('php');  
return $article->title;

}

private function findModel($alias)

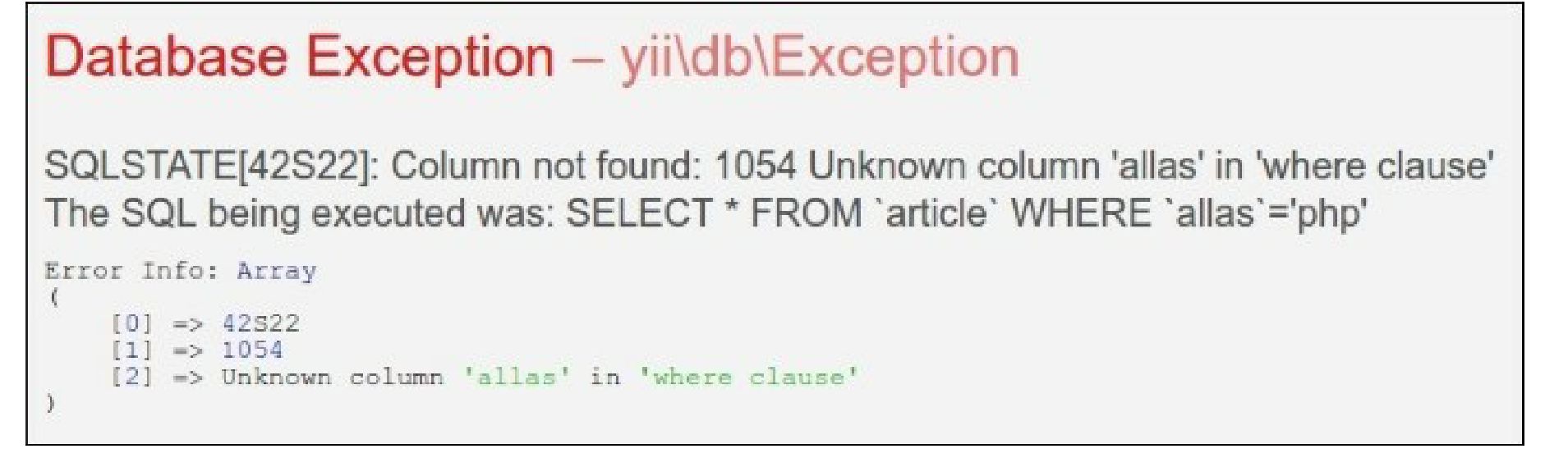
{

return Article::findOne(['allas' => $alias]);

}

}

2. After running the preceding action, we should get the following error:



3. Moreover, the stack trace shows the following error:



How it works...

From the error message, we know that we have no alias column in the database, but we have used it  
somewhere in the code. In our case, it is very simple to find it just by searching all the project files, but in a  
large project, a column can be stored in a variable. Moreover, we have everything to fix an error without  
leaving the screen where the stack trace is displayed. We just need to read it carefully.

The stack trace displays a chain of calls in the reversed order starting with the one that caused an error.  
Generally, we don’t need to read the whole trace to get what is going on. The framework code itself is  
tested well, so the probability of error is less. That is why Yii displays the application trace entries  
expanded and the framework trace entries collapsed.

Therefore, we take the first expanded section and look for alias. After finding it, we can immediately tell  
that it is used in ErrorController .ph p on line 19.

See also

* [In order to learn more about error handling, refer to http://www.yiiframework.com/doc-2.0/guide-](http://www.yiiframework.com/doc-2.0/guide-runtime-handling-errors.html)  
  rnntime-handling-errors.html
* The Logging and using the context information recipe