Defining and using multiple DB connections

Multiple database connections are not used very often for new standalone web applications. However, when you are building an add-on application for an existing system, you will most probably need another database connection.

From this recipe, you will learn how to define multiple DB connections and use them with DAO, Query Builder, and Active Record models.

Getting ready

1. Create a new application using the Composer package manager, as described in the official guide at

<http://www.yiiframework.com/doc-2.0/guide-start-installation.html>.

1. Create two MySQL databases named dbi and db2.
2. Create a table named post in dbi, as follows:

DROP TABLE IF EXISTS 'post';

CREATE TABLE IF NOT EXISTS 'post' (

'id' INT(10) UNSIGNED NOT NULL AUTO\_INCREMENT,

'title' VARCHAR(255) NOT NULL,

'text' TEXT NOT NULL,

PRIMARY KEY ('id')

);

1. Create a table named comment in db2, as follows:

DROP TABLE IF EXISTS 'comment';

CREATE TABLE IF NOT EXISTS 'comment' (

'id' INT(10) UNSIGNED NOT NULL AUTO\_INCREMENT,

'text' TEXT NOT NULL,

'post\_id' INT(10) UNSIGNED NOT NULL,

PRIMARY KEY ('id')

);

How to do it...

1. We will start with configuring the DB connections. Open config/main. php and define a primary connection as described in the official guide:

'db' => [

'connectionString' =>'mysql:host=localhost;dbname=db1',

'username' => 'root',

'password' => '',

'charset' => 'utf8',

],

1. Copy it, rename the db component to db2, and change the connection string accordingly. Also, you need to add the class name as follows:

'db2' => [

'class'=>'yii\db\Connection ',

'connectionString' => 'mysql:host=localhost;dbname=db2',

'username' => 'root',

'password' => '',

'charset' => 'utf8',

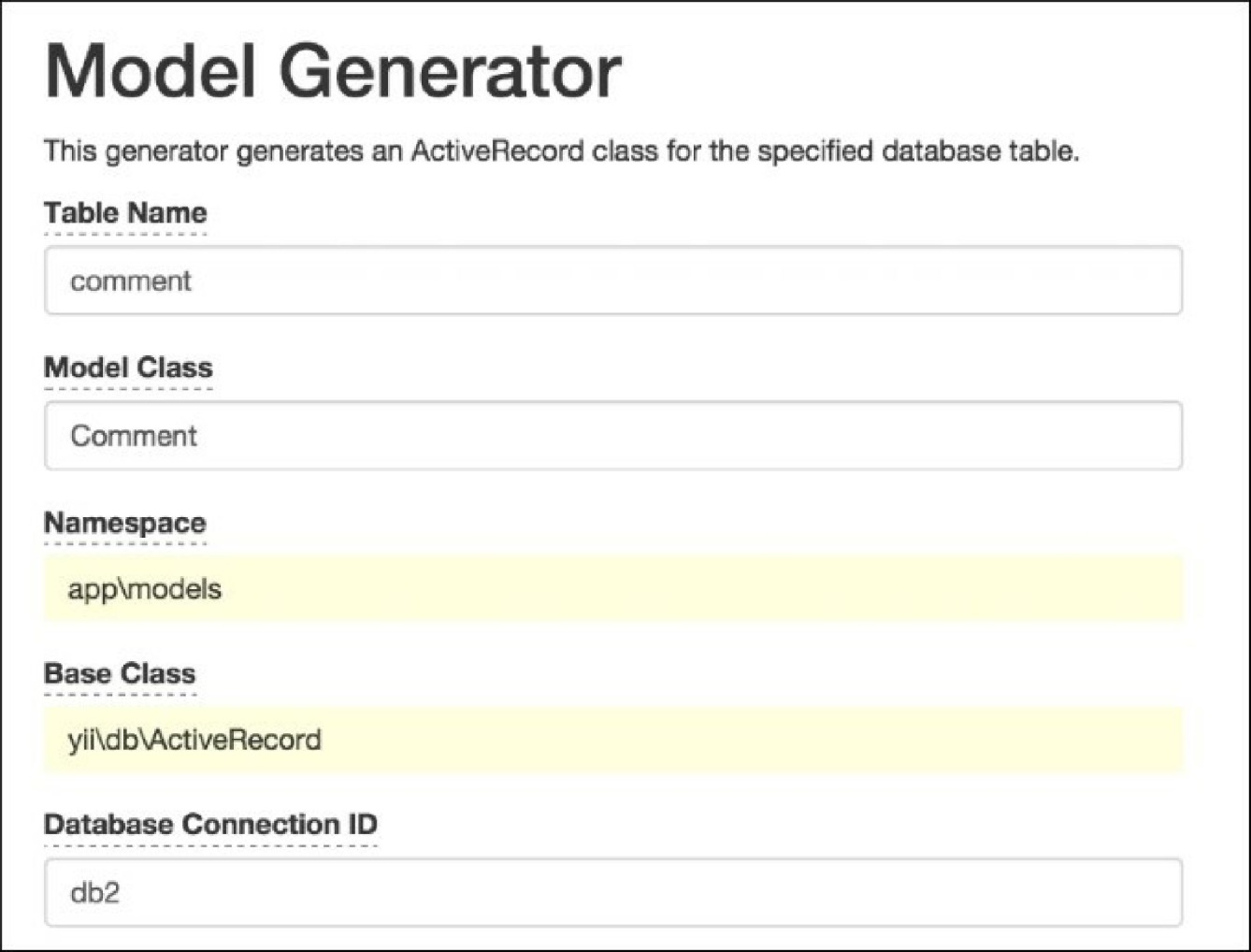
],

1. That is it. Now you have two database connections and you can use them with DAO and Query Builder, as follows:

$rows1 = Yii::$app->db->createCommand($sql)->queryAll();

$rows2 = Yii::$app->db2->createCommand($sql)->queryAll();

1. Now, if we need to use Active Record models, we first need to create the Post and Comment models with Gii. You can select an appropriate connection for each model. Set the db2 for database connection ID when you create the Comment model, as shown in the following screenshot:

5. Now you can use the Comment model as usual and create controllers/ DbController.php, as follows:

<?php

namespace app\controllers;

use app\models\Post;

use app\models\Comment;

use yii\helpers\ArrayHelper; use yii\helpers\Html;

use yii\web\Controller;

/\*\*

* Class DbController.
* @package app\controllers \*/

class DbController extends Controller {

public function actionIndex()

{

$post = new Post();

$post->title = 'Post #'.rand(1, 1000);

$post->text = 'text';

$post->save();

$posts = Post::find()->all(); echo Html::tag('h1', 'Posts');

echo Html::ul(ArrayHelper::getColumn($posts, 'title'));

$comment = new Comment();

$comment->post\_id = $post->id;

$comment->text = 'comment #'.rand(1, 1000);

$comment->save();

$comments = Comment::find()->all(); echo Html::tag('h1', 'Comments');

echo Html::ul(ArrayHelper::getColumn($comments, 'text'));

}

}

6. Run db/index multiple times and you should see records added to both databases, as shown in the following screenshot:





How it works...

In Yii, you can add and configure your own components through the configuration file. For nonstandard components such as db2, you have to specify the component class. Similarly, you can add db3, db4, or any other component, for example, facebookApi. The remaining array key/value pairs are assigned to the component’s public properties, respectively.

There’s more.

Depending on the RDBMS used, there are additional things we can do to make it easier to use multiple databases.

**Cross-database relations**

If you are using MySQL, it is possible to create cross-database relations for your models. In order to do this, you should prefix the Comment model’s table name with the database name, as follows:

class Comment extends \yii\db\ActiveRecord {

//...

public function tableName()

{

return 'db2.comment';

}

//...

}

Now, if you have a comments relation defined in the Post model relations method, you can use the following code:

$posts = Post::find()->joinWith('comments')->all();

See also

[For further information, refer to http://www.yiiframework.com/doc-2.0/guide-db-dao.html#creating-db-](http://www.yiiframework.com/doc-2.0/guide-db-dao.html%23creating-db-connections) connections.