Preventing SQL injections

SQL injection is a type of code injection that uses vulnerability at the database level and allows you to execute arbitrary SQL, allowing malicious users to carry out actions such as deleting data or raising their privileges.

In this recipe, we will see examples of vulnerable code and fix them.

Getting ready

1. Create a new 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. Execute the following SQL:

DROP TABLE IF EXISTS 'user';

CREATE TABLE 'user' (

'id' int(11) unsigned NOT NULL AUTO\_INCREMENT,

'username' varchar(100) NOT NULL,

'password' varchar(32) NOT NULL,

PRIMARY KEY ('id')

);

INSERT INTO 'user'('id','username','password') VALUES (

'1','Alex','202cb962ac59075b964b07152d234b70');

INSERT INTO 'user'('id','username','password') VALUES (

'2','Qiang','202cb962ac59075b964b07152d234b70');

1. Generate a User model using Gii.

How to do it...

1. First, we will implement a simple action that checks whether the username and password that came from a URL are correct. Create app/controllers/Sqlcontroller. php:

<?php

namespace app\controllers;

use app\models\User; use Yii;

use yii\base\Controller; use yii\base\Exception; use yii\helpers\ArrayHelper; use yii\helpers\Html;

/\*\*

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

class SqlController extends Controller {

protected function renderContentByResult($result)

{

if ($result) {

$content = "Success";

} else {