Redis database driver

This extension allows you to use Redis key-value storage in any project on the Yii2 framework. It contains  
the Cache and Session storage handlers, as well as the extension, which implements the ActiveRecord  
pattern for access to the Redis database records.

Getting ready

1. Create a new application by using composer, as described in the official guide at  
   <http://www.yiiframework.com/doc-2.0/guide-start-installation.html>.
2. Install the storage: <http://redis.io>.
3. Install all migrations with the following command:

composer require yiisoft/yii2-redis

How to do it...

First of all, configure the Connection class in your configuration file:

return [

//....

'components' => [

'redis' => [

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

'hostname' => 'localhost',

'port' => 6379,

'database' => 0,

],

]

];

Direct usage

For low-level working with Redis commands, you can use the executeCommand method of the connection  
component:

Yii::$app->redis->executeCommand('hmset', ['test\_collection', 'keyl', 'vall', 'key2',

'val2']);

You can also use simplified shortcuts instead of executeCommand calls:

Yii::$app->redi

s->hmset('test\_collection', 'key1', 'val1', 'key2', 'val2')

Using ActiveRecord

For access to Redis records via the ActiveRecord pattern, your record class needs to extend from the  
yii\redis\ActiveRecord base class and implement the attributes() method:

class Customer extends \yii\redis\ActiveRecord  
{

public function attributes()

{

return ['id', 'name', 'address', 'registration\_date'];

}

public function getOrders()

{

return $this->hasMany(Order::className(), ['customer\_id' => 'id']);

}

}

A primary key of any model can be defined via the primaryKey() method, which defaults to id if not  
specified. The primary key needs to be placed in the attribute list if you do not manually specify it in the  
primaryKey() method.

The following is a usage example:

$customer = new Customer();

$customer->name = 'test';

$customer->save();

echo $customer->id; // id will automatically be incremented if not set explicitly  
// find by query

$customer = Customer::find()->where(['name' => 'test'])->one();

How it works...

The extension provides a Connection component for low-level access to Redis storage records.

You can also use an ActiveRecord-like model with a limited set of methods (where(), limit(),  
offset(), and indexBy()). Other methods do not exist because Redis does not support SQL queries.

There are no tables in Redis, so you cannot define via relations via a junction table name. You can only  
define many-to-many relations via other hasMany relations.

For general information on how to use Yii’s ActiveRecord, please refer to Chapter 3, ActiveRecord,  
Model, and Database.

See also

* For more information about the extension’s usage, see:

° [https ://github.com/yiisoft/yii2-redis/blob/ma stpr/docs/guide/RF ADMF.md](https://github.com/yiisoft/yii2-redis/blob/master/docs/guide/README.md)  
o [http://www.viiframework.com/doc-2.0/ext-redis-index.html](http://www.yiiframework.com/doc-2.0/ext-redis-index.html)

* For information about Redis key-value storage, refer: <http://redis.io/documentation>
* Chapter 3, ActiveRecord, Model, and Database for ActiveRecord usage