Адаптер к движку ElasticSearch

Это расширение является activerecord как обертка для интеграции ElasticSearch полнотекстового поиска в рамках yii2. Он позволяет работать с любыми данными модели и использовать шаблон ActiveRecord для извлечения и хранения записей в коллекциях ElasticSearch.

Подготовка

1. Создайте новое приложение с помощью диспетчера пакетов Composer, как описано в официальном руководстве по адресу  
   <http://www.yiiframework.com/doc-2.0/guide-start-installation.html>.   
   По русски <http://yiiframework.domain-na.me/doc/guide/2.0/ru/start-installation>.
2. Установите службы elasticsearch найденные на <https://www.elastic.co/downloads/elasticsearch>.
3. Установите расширение с помощью следующей команды:

Composer require yiisoft/yii2-elasticsearch

Как это сделать...

Установите новое соединение ElasticSearch в конфигурации приложения:

return [

//....

'components' => [

'elasticsearch' => [

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

'nodes' => [

['http\_address' => '127.0.0.1:9200'],

// configure more hosts if you have a cluster

],

],

]

];

Используем класс запроса

Класс Query можно использовать для низкоуровневого запроса записей из любой коллекции:

use \yii\elasticsearch\Query;

$query = new Query;

$query->fields('id, name')

->from('myindex', 'users')

->limit(10);

$query->search();

Можно также создать команду и запустить ее напрямую:

$command = $query->createCommand();

$rows = $command->search();

Используя ActiveRecord

Использование ActiveRecord является распространенным способом доступа к записям. Просто расширьте класс yii\elasticsearch\ActiveRecord и реализуйте метод attributes () для определения атрибутов ваших документов.

Например, можно написать модель клиента:

class Buyer extends \yii\elasticsearch\ActiveRecord  
{

public function attributes()

{

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

}

public function getOrders()

{

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

}

}

Затем напишите модель заказа:

class Order extends \yii\elasticsearch\ActiveRecord  
{

public function attributes()

{

return ['id', 'user\_id', 'date'];

}

public function getBuyer()

{

return $this->hasOne(Customer::className(), ['id' => 'buyer\_id']);

}

}

Можно переопределить index () и type (), чтобы определить индекс и тип, который представляет эта запись.

Ниже приведен пример использования:

$buyer = new Buyer();

$buyer>primaryKey = 1; // it equivalent to $customer->id = 1;

$buyer>name = 'test';

$buyer>save();

$buyer = Buyer::get(1);

$buyer = Buyer::mget([1,2,3]);

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

Можно использовать запрос DSL для определенных запросов:

$result = Article::find()->query(["match" => ["title" => "yii"]])->all();

$query = Article::find()->query([

"fuzzy\_like\_this" => [

"fields" => ["title", "description"],

"like\_text" => "Some search text",

"max\_query\_terms" => 12

]

]);

$query->all();

Вы можете добавить фасеты в поиск:

$query->addStatisticalFacet('click\_stats', ['field' => 'visit\_count']);

$query->search();

Использование ElasticSearch DebugPanel

Данное расширение содержит специальную панель для модуля yii2-debug. Это позволяет просматривать все выполненные запросы. Эту панель можно включить в файл конфигурации:

if (YII\_ENV\_DEV) {

// configuration adjustments for 'dev' environment  
$config['bootstrap'][] = 'debug';

$config['modules']['debug'] = [

'class' => 'yii\debug\Module',

'panels' => [

'elasticsearch' => [

'class' => 'yii\elasticsearch\DebugPanel',

],

],

];

$config['bootstrap'][] = 'gii';

$config['modules']['gii'] = 'yii\gii\Module';

}

Как это работает...

Расширение предоставляет низкоуровневый конструктор команд и высокоуровневую реализацию ActiveRecord для запроса записей из индекса ElasticSearch.

Использование модели расширения, очень похожа на базу данных и activerecord, как описано в главе 3, ActiveRecord, Model, and Database, кроме операторов join(), groupBy(), having(), and union() ActiveQuery.

**Примечание.** ElasticSearch по умолчанию ограничивает число возвращаемых записей десятью элементами. Будьте осторожны с ограничениями если вы используете отношения с опцией via().

Смотрите так же

* Дополнительные сведения о расширении см. в разделе:

° [https://github.com/yiisoft/yii2-elasticsearch/hlob/master/docs/guide/RFADMF.md](https://github.com/yiisoft/yii2-elasticsearch/blob/master/docs/guide/README.md)  
o [http://www.viiframework.com/doc-2.0/ext-elasticsearch-index.html](http://www.yiiframework.com/doc-2.0/ext-elasticsearch-index.html)

* Вы также можете посетить официальный сайт расширения по адресу <https://www.elastic.co/products/elasticsearch>.
* Для получения дополнительной информации о запросе DSL, вы можете посетить:

° <http://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-match-query.html>  
o [http://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-flt-quervhtml](http://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-flt-query.html)

* Для использования ActivRecord см. в главе 3, ActiveRecord, Model, and Database