UMI.CMS + Elasticsearch

Оптимизация скорости работы сайта

Толпекин Дмитрий

Iris Digital

Iris Digital

Elasticsearch: возможности

- Поиск
- Подсказки
- Фильтры и сортировки
- Агрегация

Этапы интеграции UMI.CMS и Elasticsearch

1. Использование официального клиента Elasticsearch
2. Создание индекса
3. Индексация данных
4. Модификация индекса без простоя
5. Поиск

Использование официального клиента Elasticsearch

1. Скачаваем и устанавливаем Composer

```
curl -s http://getcomposer.org/installer | php
```

2. Подключаем официальный клиент

```
php composer.phar require elasticsearch/elasticsearch
```

3. Создаем клиент

```
<?php
use Elasticsearch\ClientBuilder;
require 'vendor/autoload.php';
$client = ClientBuilder::create()->build();
```

Создание индекса

```
<?php
$params = [
    'index' => 'index_v1',
    'body' => [
        'properties' => [
            'object_id' => [
              ‘type’ => ‘is_active’,
              ‘index’ => ‘not_analyzed’
            ],
            'object_id' => [
              ‘type’ => ‘long’,
              ‘index’ => ‘not_analyzed’
            ],
            'parent_id' => [
              ‘type’ => ‘long’,
              ‘index’ => ‘not_analyzed’
            ],
            'module' => [
              ‘type’ => ‘string’,
              ‘index’ => ‘not_analyzed’
            ],
            'method' => [
              ‘type’ => ‘string’,
              ‘index’ => ‘not_analyzed’
            ],
            ...
        ]
    ]
];

$response = $client->indices()->create($params);
```

Индексация данных

Используем события UMI.CMS для индексации контента

Отслеживаемые события:

- systemCreateElement
- systemModifyElement
- systemMoveElement
- systemSwitchElementActivity
- systemDeleteElement
- systemRestoreElement
- systemVirtualCopyElement
- systemCloneElement
- systemKillElement

Индексация данных

Обработка событий:

```
/ /events.php
new umiEventListener('systemModifyElement', 'elasticsearch', 'updateIndex');
new umiEventListener('systemSwitchElementActivity', 'elasticsearch', 'updateIndex');
new umiEventListener('systemCreateElement', 'elasticsearch', 'updateIndex');
new umiEventListener('systemMoveElement', 'elasticsearch', 'updateIndex');
new umiEventListener('systemDeleteElement', 'elasticsearch', 'updateIndex');
new umiEventListener('systemRestoreElement', 'elasticsearch', 'updateIndex');
new umiEventListener('systemModifyPropertyValue', 'elasticsearch', 'updateIndexOnModifyProperty');
new umiEventListener('systemVirtualCopyElement', 'elasticsearch', 'updateIndexOnCloneOrVirtualCopy');
new umiEventListener('systemCloneElement', 'elasticsearch', 'updateIndexOnCloneOrVirtualCopy');
new umiEventListener('systemKillElement', 'elasticsearch', 'updateIndexOnKill');

/ /class.php
public function updateIndex(iUmiEventPoint $event)
{
  if ($event->getMode() !== 'after') {
    return false;
  }

  /* @var iUmiHierarchyElement $page */
  $page = $event->getRef('element');

  if (!$this->isValidPage($page)) {
    return false;
  }

  $this->processPages($page, 'update');
  return true;
}
...
```

Индексация данных

Обработка событий:

```
public function processPages(page, $operation)
{
  $client = $this->getConnection();
  $umiHierarchy = umiHierarchy::getInstance();
  $regedit = regedit::getInstance();
  $index = (string) $regedit->getVal("//modules/elasticsearch/index_name");

  switch ($operation) {
    case 'update':
      $data = array();
      $data['is_deleted'] = $page->getIsDeleted();
      $data['is_active'] = $page->getIsActive();
      $data['is_visible'] = $page->getIsVisible();
      $data['object_id'] = $page->getObjectId();
      $data['module'] = $page->getModule();
      $data['method'] = $page->getMethod();
      $data['type_id'] = $page->getTypeId();
      $data['parent_id'] = $page->getParentId();
      $params = [
        'index' => $index,
        'type' => 'page',
        'id'  =>  $page->id,
        'body' => $data
      ];
      $client->index($params);
    break;
    ...
  }
}
```

Модификация индекса без простоя

1. Создаем новый индекс index_v<n> с необходимыми изменениями
2. Переиндексируем данные, используя новый индекс
3. Изменяем псевдоним индекса

```
$params['body'] = [
  'actions' => [
      'remove' => [
        'index' => 'index_v<n-1>',
        'alias' => 'index_actual'
      ],
      'add' => [
        'index' => 'index_v<n>',
        'alias' => 'index_actual'
        ]
      ]
];
$client->indices()->updateAliases($params);
```

Поиск

Полнотекстовый поиск с фильтрацией:

```
$params = [
    'index' => 'index_actual',
    'type' => 'pages',
    'body' => [
        'query' => [
            'match' => [
                '_all' => [
                    'query' => 'search query',
                    'operator' => 'and'
                ]
            ]
        ],
        'filter' => [
            'bool' => [
                'must' => [
                    'term' => [
                        'is_active' => true
                    ],
                    'term' => [
                        'is_deleted' => false
                    ]
                ]
            ]
        ]
    ]
];

$results = $client->search($params);
```

Поиск

Агрегация:

```
$params = [
    'index' => 'index_actual',
    'type' => 'pages',
    'body' => [
        'query' => [
            'match_all' => []
        ],
        'filter' => [
            'bool' => [
                'must' => [
                    'term' => [
                        'is_active' => true
                    ],
                    'term' => [
                        'is_deleted' => false
                    ]
                ]
            ]
        ],
       'aggs' => [
            'propduct-count' => [
                'terms' => [
                    'field' => 'object_id',
                    'size' => 0
                ]
            ]
        ]
    ]
];
$results = $client->search($params);
```

Спасибо!

Толпекин Дмитрий

IrisDigital.ru

info@irisdigital.ru

Iris Digital