

Добавляем поисковую строку в ваше приложение с Elasticsearch

Маленький google в вашем приложении

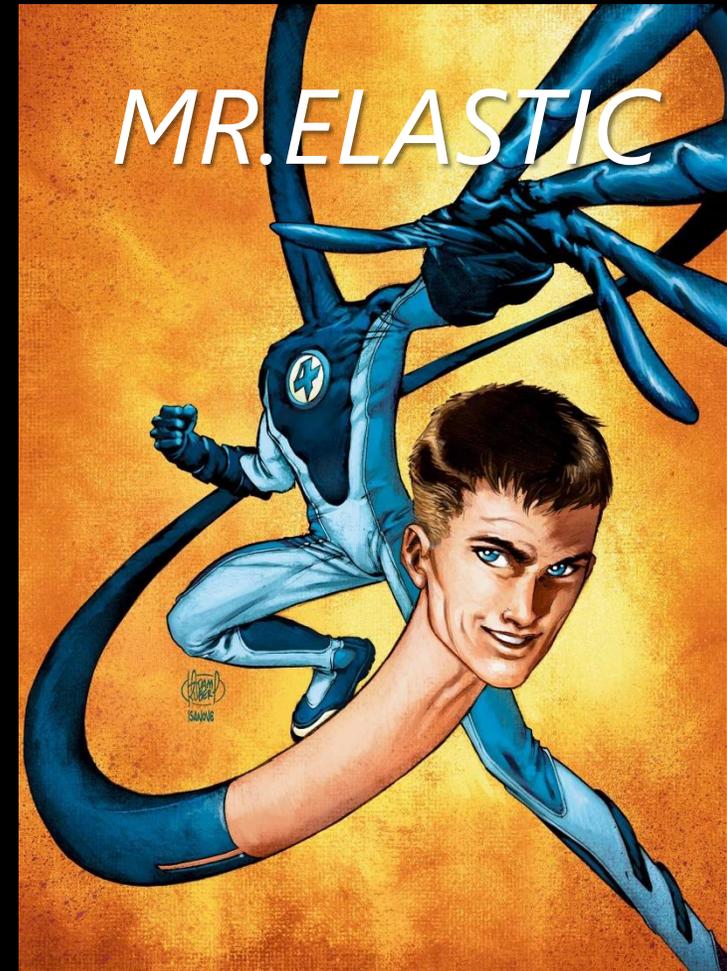
Шестаков Алексей

Инженер-Программист



КТО Я

- Backend
- Внутренняя разработка
- Пишу Контур.Стафф – внутреннюю социальную сеть контура



Что будет в этом докладе

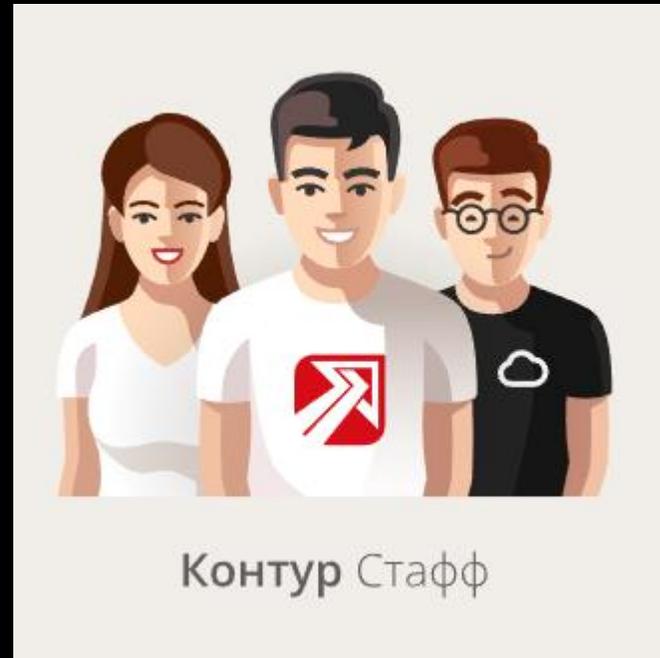
- История и Введение в Elasticsearch
- Примеры использования
- Теория
- Как сделать поиск умнее
- Как мы сделали наш поиск на Elasticsearch

Чего не будет в этом докладе

- Эксплуатация ES
- Использование ES для хранения логов
- ML (ака Машинное обучение) в поиске.

Контур.Стафф – Соцсеть Контура

- Пользователи ~ 9000
- Посты
- Сообщества >1000
- Мероприятия



Нам нужен был поиск

- Находить сотрудников
- Находить контент

- Поиск должен уметь разграничивать доступ

Почему у нас появился Elasticsearch

У нас тогда было и не подошло:

- SQL
- Mongo

Потом у нас появился поиск на:

- Lucene

Почему Elasticsearch > Lucene

- Lucene – Java библиотека
- Elasticsearch – REST сервис
- Elasticsearch добавляет поверх Lucene
 - Масштабируемость
 - Реплицируемость
 - Аналитические инструменты



Что такое Elasticsearch?

- Opensource
- Широко используемый
- Масштабируемый
- Быстрый
- Кастомизируемый

История Elasticsearch

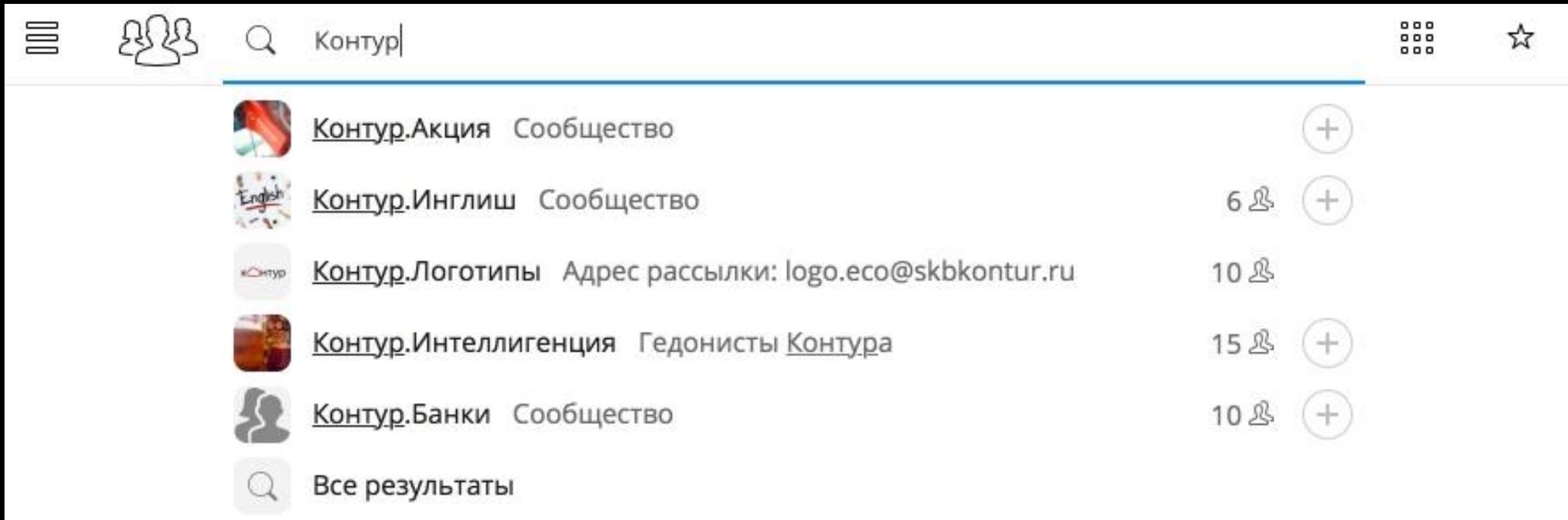


- Opensource
- Автор Shay Banon
- Начал проект Compass в 2004 г.,
- В 2010 г. как результат большого переписывания появился Elasticsearch
- В качестве основы используется **Lucene**



Поисковые подсказки

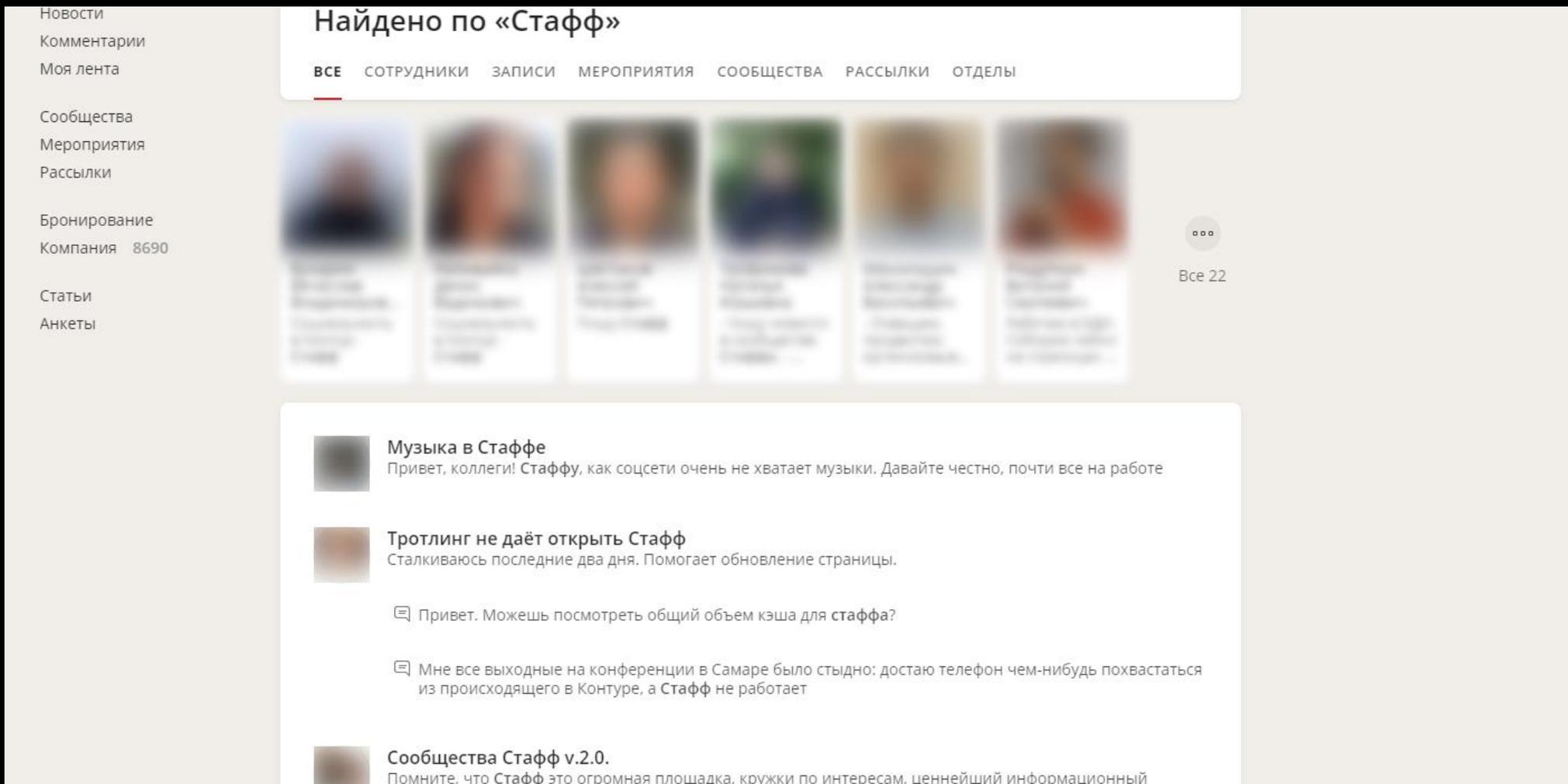
- Search-as-you-type
- Помогает сформировать поисковый запрос
- Переход в SERP или на сразу на результат



Поисковая подсказка в Стафф – точка входа на любой странице

SERP – Search Engine Result Page

- Страница с результатом поиска
- Снippet – часть документа, важная в контексте запроса



Поиск в Стафф

SERP в Стафф

Elasticsearch – не черная коробка

- Elasticsearch использует классическую теорию поиска
- Для того что бы качественно настроить поиск нужно понимать процесс обработки текста

Модель – Bag of Words

- Документом называется индексируемое текстовое поле
- Каждый документ режется на «слова» или «токены»
- Порядок не учитывается



Что такое «Токен»?

- «Токен» – минимальная единица поиска
- Обычно слово, но не всегда
- Поиск меньших частей «токена» возможен, но более затратен

Поисковый индекс

- «Предметный указатель»
- Запись в индексе:
 - Связка токена – и номера документов, где слово встречается
- Инвертированный индекс

Index

B-TREE-INSERT-NONFULL, 496
B-TREE-SEARCH, 492, 499 ex.
B-TREE-SPLIT-CHILD, 494
BUBBLESORT, 40 pr.
bucket, 200
bucket sort, 200–204

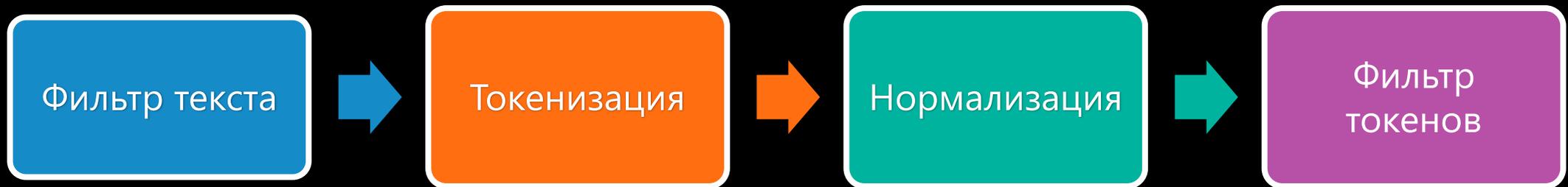
Инвертированный индекс - Частоты

Слово	Номера документов, где слово встречается и сколько раз
все	2 документ – 3 раза, 4 документ – 2 раза, 7 документ – 1 раз
всегда	2 документ – 3 раза, 7 документ – 1 раз
делать	2 документ – раза
дело	3 документ – раза

- Редкое слово (во всех документах) = важное в запросе
- Часто встречаемое слово (в документе) = важное

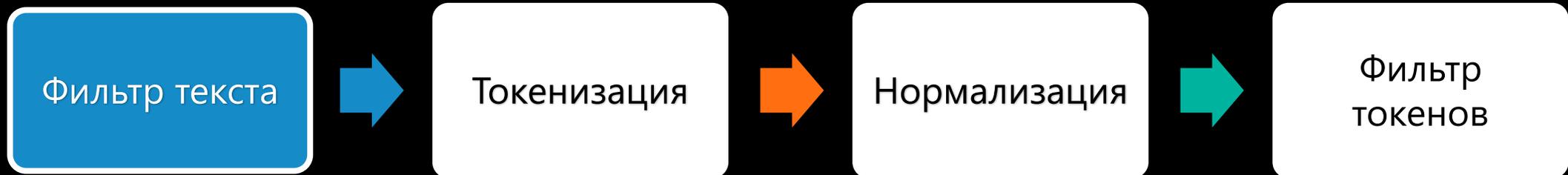
Процесс индексации

- Процесс построения индекса
- Управляя процессом индексации можно
 - Убрать лишнее из выдачи
 - Научить поиск находить больше



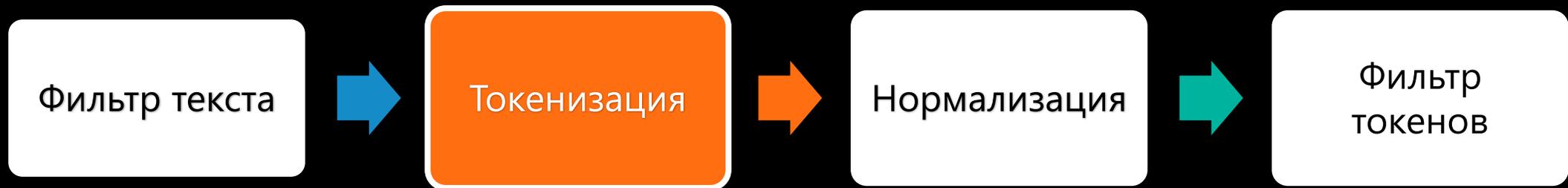
Фильтр текста

- Убираем из текста все что не должно искаться
- В терминах ES – «char_filter»
- Пример – «html_strip» - убирает HTML разметку



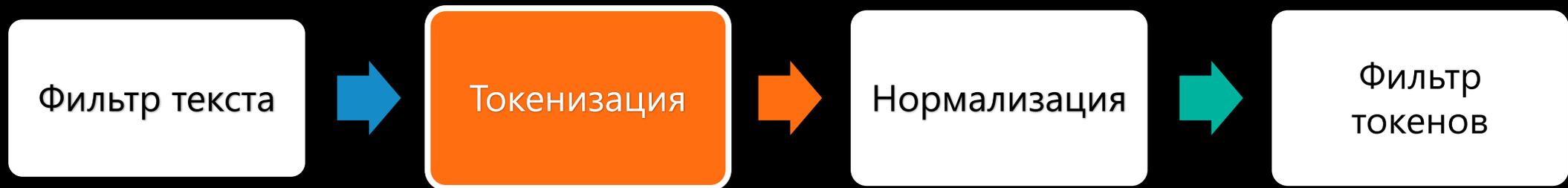
Токенизация

- Управляя токенизацией важно понять что мы хотим считать одним «словом»



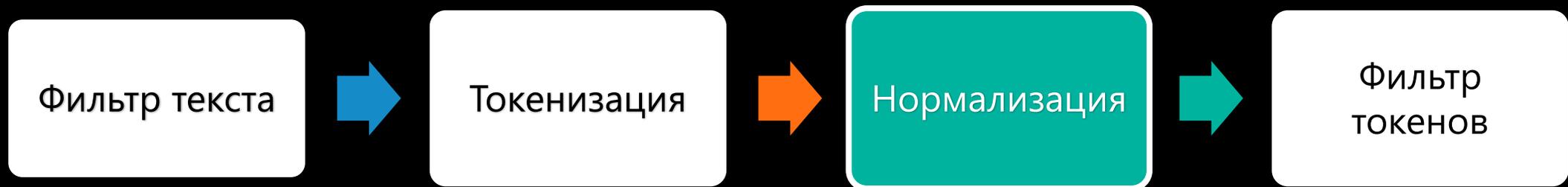
Токенизация

- От выбора токенов зависит будет ли находиться тот или иной вариант
- Мобильные номера, Номера автомобилей
 - + 7 (933) 34344322 → +7(933)34344322, (933)34344322, 34344322



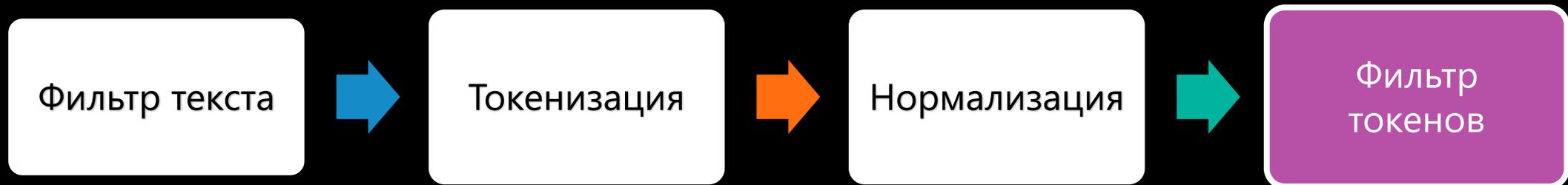
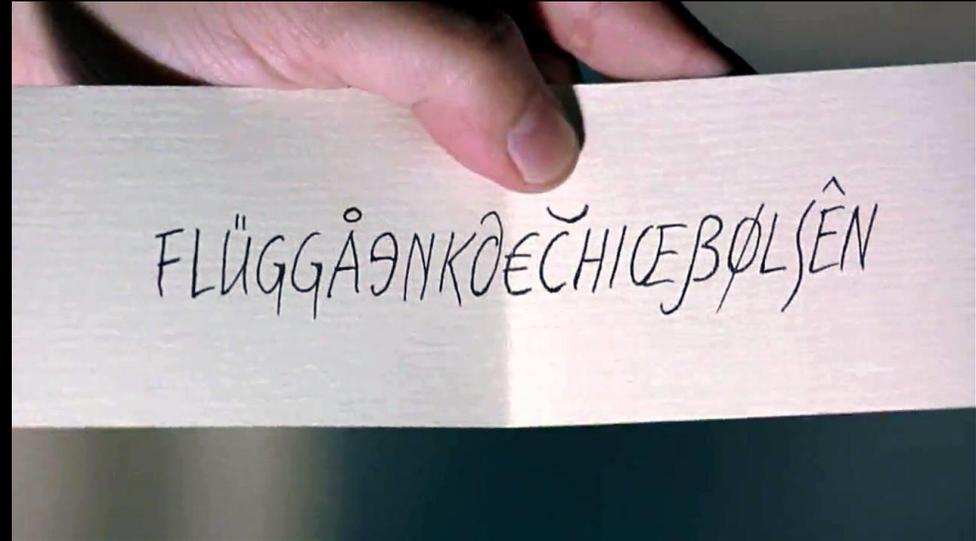
Нормализация

- Приводим схожие токены к одному
 - Хотим ли мы считать слова в разном падеже одним и тем же словом?
 - Мы теряем немного информации, но и будем находить больше



Фильтр токенов

- Не все токены нам интересны
 - Пример: предлоги, местоимения – «Стоп-слова»



Elasticsearch – Индексируем

POST localhost:9200/information/person/1

```
{  
  "user" : "Paul",  
  "lastname" : "Smith",  
  "description" : "Business Analyst",  
  
  "age" : 33  
}
```

Elasticsearch – Поиск

```
curl -X GET "localhost:9200/information/_search?pretty" -H 'Content-Type: application/json' -d'
```

```
{  
  "query" : {  
    "term" : { "user" : "Paul" }  
  }  
}
```

Еще варианты:

```
GET localhost:9200/_search?q=Paul
```

```
GET localhost:9200/_search?q=user:Paul
```

Elasticsearch – Поиск по префиксу

```
curl -X GET "localhost:9200/_search?pretty" -H 'Content-Type: application/json' -d'
```

```
{  
  "query": {  
    "prefix": {  
      "user": {  
        "value": "ki"  
      }  
    }  
  }  
}
```

Настраиваем анализаторы

- Управляет **процессом индексации** конкретного поля
- Можно создать свой если стандартный не подходит

Настраиваем анализаторы

PUT information {

 "settings": { "analysis": {

 "analyzer": {

"description_analyzer": {

 "type": "custom",

 "tokenizer": "standard",

 "char_filter": ["html_strip"],

 "filter": ["lowercase"]

 } } } }

Настраиваем маппинги

- Маппинг – аналог схемы в ES
- Для настройки важен
 - Тип поля
 - Анализатор

Настраиваем маппинги

- ES Автоматически настраивает поля и для текстовых полей выбирается анализатор "standard"
- Посмотреть настройки маппингов можно так:

```
curl -X GET "localhost:9200/my-index/_mapping?pretty"
```

Настраиваем маппинги

```
curl -X PUT "localhost:9200/information?pretty" -H 'Content-Type: application/json' -d'
```

```
{  
  "mappings": {  
    "properties": {  
      "age": { "type": "integer" },  
      "job_description": { "type": "text" , "analyzer": "description_analyzer" },  
      "user": { "type": "keyword" }  
    }  
  }  
}
```

Настраиваем маппинги

- ES может автоматически создавать маппинги
- Маппинги immutable
- Настраивать нужно при создании индексов

Строим правильно запрос поиска

```
curl -X GET "localhost:9200/_search?pretty" -H 'Content-Type: application/json' -d'
```

```
{
```

```
  "query": {
```

```
    "multi_match" : {
```

```
      "query" : "Что ищем",
```

```
      "name" : [ "user^2", "job_description", "lastname" ]
```

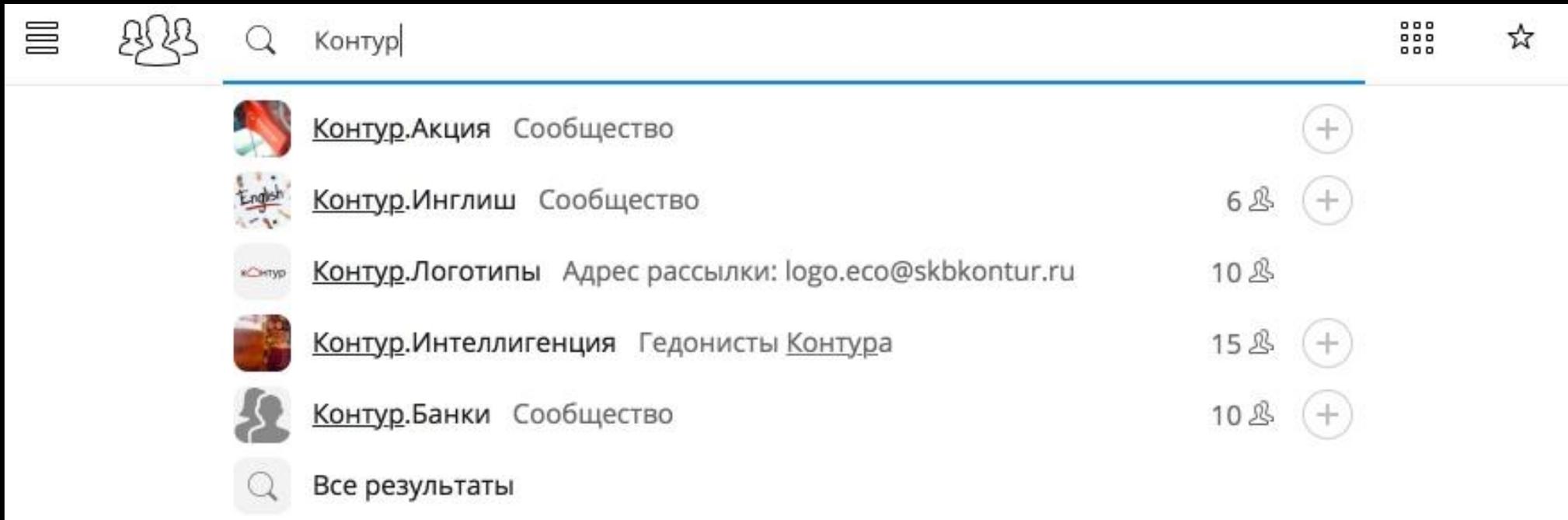
```
    }
```

```
  }
```

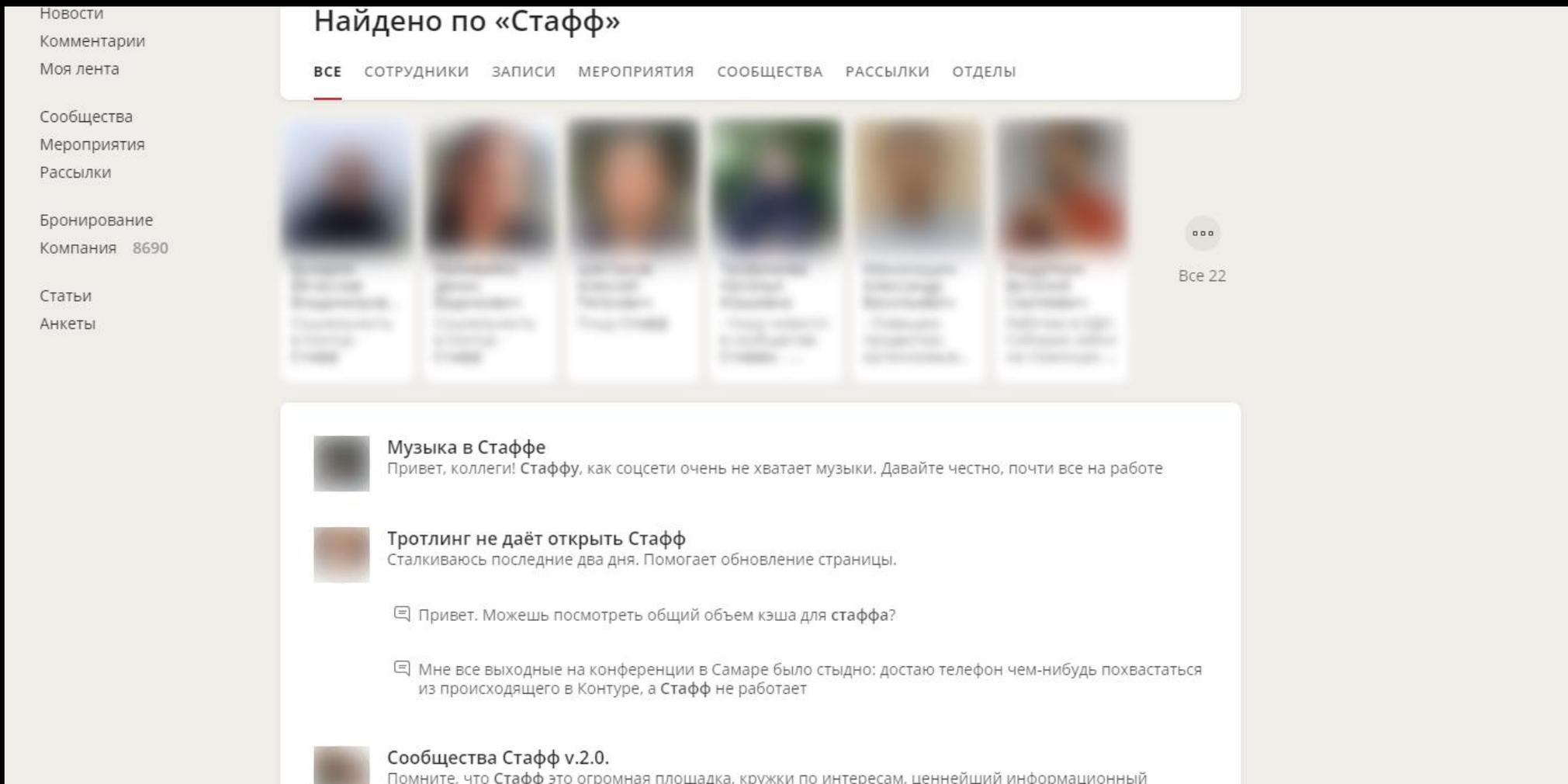
```
}
```

Контур.Стафф

- Использует продвинутые фишки ES
 - Скрипты ранжирования для управления результатом
 - Вложенные объекты
 - Релевантностью полей в запросе
- Настроены фильтры управления видимости
- Агрегирование



Поисковая подсказка в Стафф – точка входа на любой странице



Поиск в Стафф

Это и есть SERP Стафф

Шестаков Алексей

Найдено по «Шестаков Алексей»

ВСЕ СОТРУДНИКИ ЗАПИСИ



Шестаков Алексей Петрович

Программист

Отдел развития информационных ресурсов (ОРИР)



java.ural.Meetup @3

на Хабре: С докладами выступят: — Алексей Шестаков, — Владимир Лиля — Григорий Кошелев

Java Community · Григорий Кошелев · 9 сентября

Поиск в Стафф

Это и есть SERP Стафф

Участники и руководители

УЧАСТНИКИ 78 РУКОВОДИТЕЛИ 1

Шестаков

 Шестаков Алексей
Программист

Пригласить
Текст приглашения можно задать в [настройках сообщества](#)

ES – Используется как фильтр

Популярные темы

#марафонзнаний

#контур_интроверт

#впередкзнаниям

#вызов2019

#трекдня

#деньhg

#неоспоримые2года

#неоспоримые

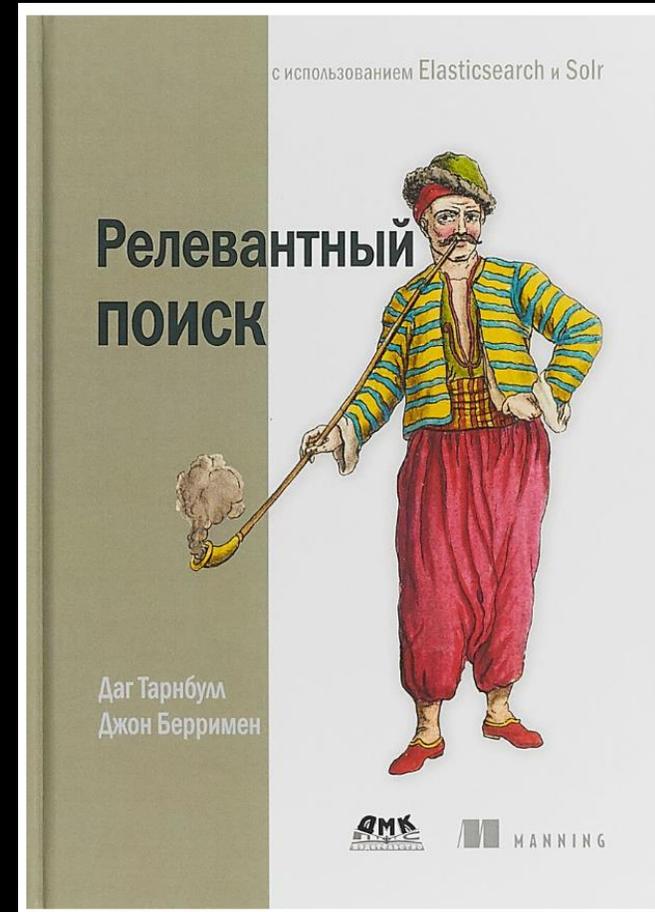
#алло

#еслинетактотак

Популярные теги за последнее время – выборка из ES

Литература

- Конкретные рецепты
- Практика использования ES
- Ориентация на ручное управление ранжированием без ML



Литература

- Хорошее введение в теорию информационного поиска
- Идеально если вы хотите глубже понимать ES либо написать свой аналог
- Новой редакции нет с 2008 г.
- Устарела в ML части.



Вопросы?



Алексей Шестаков

Инженер-Программист

shestakovap@skbkontur.ru

kontur.ru