

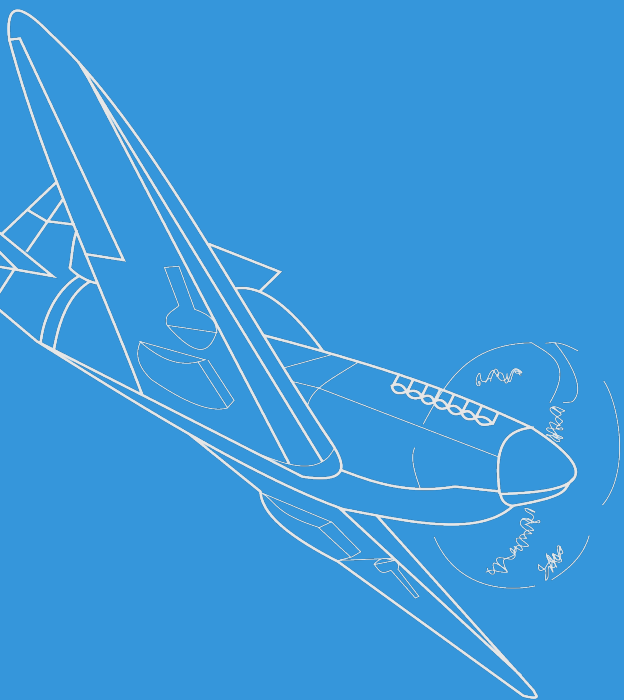
Урок №10

Поисковый движок и KV-хранилища в веб-приложении

на котором расскажут из чего состоит поиск, какие бывают KV-хранилища

Содержание занятия

1. elasticsearch
2. KV-хранилища
 - a. etcd
 - b. consul
 - c. vault
 - d. aerospike
 - e. redis
 - f. kafka



Поисковые платформы



elasticsearch



- Open source (1548 контрибьюторов на 27 ноября 2020 г.)
- Масштабируемость и отказоустойчивость
- Удобный API (Restfull API)
- Гибкие настройки
- Динамический маппинг
- Геопоиск
- СJK

Elasticsearch



- GitHub (поиск репозитория)
- Uber
- Microsoft (хранилище для MSN)
- stackoverflow
- ebay
- docker



- **Морфология**

Раздел грамматики, который оперирует формами слов.

- **Стемминг**

Приближённый эвристический процесс, в ходе которого от слов отбрасываются окончания в расчёте на то, что в большинстве случаев это себя оправдывает. (running -> run)

- **Нечеткий поиск**

По заданному слову найти в тексте или словаре размера n все слова, совпадающие с этим словом (или начинающиеся с этого слова) с учетом k возможных различий.



- **Лемматизация**

Точный процесс с использованием лексикона и морфологического анализа слов, в результате которого удаляются только флексивные окончания и возвращается основная, или словарная, форма слова, называемая леммой.
(ran -> run).

- N-грамма

n каких-то элементов. Это более абстрактное понятие.

- Стоп-слова

Что мы получаем из коробки



- Огромные возможности для поиска документа;
- Около 50 видов агрегаций на все случаи жизни (максимальное, минимальное, среднее);
- Гео-поиск;
- Подсказки (suggester);
- Гибкая работа и настройка всего, что есть в Elasticsearch;
- И ещё много чего!

Elasticsearch концепты сверху



- Нода
- Кластер
- Шард
- Реплика

Elasticsearch концепты внутри



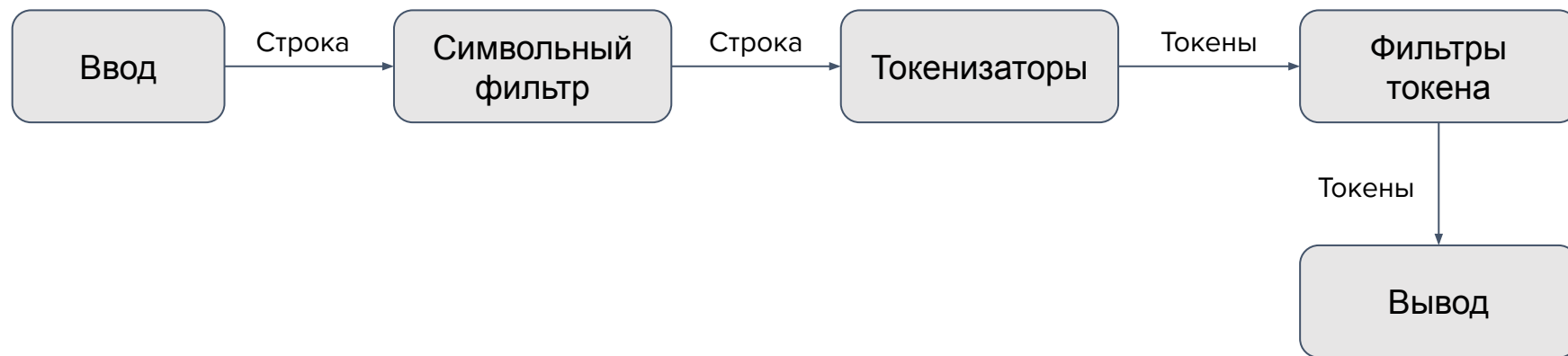
- Индекс
- Тип
- Документ
- Поле
- Отображение (mapping)
- Query DSL

Elasticsearch концепты внутри



Мир реляционных БД	Elasticsearch
База данных (Database)	Индекс (Index)
Таблица (Table)	Тип (Type)
Запись (Row)	Документ (Document)
Колонка (Column)	Поле (Field)
Схема (Schema)	Отображение (Mapping)
SQL	Query DSL

Цель - из входной фразы получить список токенов, которые максимально отражают её суть.



Расстояние Левенштейна



Минимальное количество операций вставки одного символа, удаления одного символа и замены одного символа на другой, необходимых для превращения одной строки в другую.

Цены операций могут зависеть от вида операции

- $w(a, b)$ — цена замены символа a на символ b
- $w(\epsilon, b)$ — цена вставки символа b
- $w(a, \epsilon)$ — цена удаления символа a

Частный случай задачи - Расстояние Левенштейна

- $w(a, a) = 0$
- $w(a, b) = 1$ при $a \neq b$ $w(\epsilon, b) = 1$
- $w(a, \epsilon) = 1$

Пример анализатора



```
PUT /your-index/_settings
```

```
{
```

```
"index": {
```

```
  "analysis": {
```

```
    "analyzer": {
```

```
      "customHTMLSnowball": {
```

```
        "type": "custom",
```

```
        "char_filter":
```

```
        ["html_strip"],
```

```
        "tokenizer": "standard",
```

```
        "filter": ["lowercase",
```

```
        "stop", "snowball"] ]]]}]
```



Ubuntu

...

```
apt install elasticsearch  
sudo -i service elasticsearch start
```

Нужно установить Java \geq version 7.

`http://localhost:9200/`

```
pip install elasticsearch
```

MacOS

```
brew install elasticsearch  
brew services start elasticsearch
```


Создание и заполнение индекса



```
pip install elasticsearch
```

Mappings



```
PUT my_index
{
  "mappings": {
    "_doc": {
      "properties": {
        "title": { "type": "text" },
        "name": { "type": "text" },
        "age": { "type": "integer" },
        "created": {
          "type": "date",
          "format": "strict_date_optional_time||epoch_millis"
        }
      }
    }
  }
}
```

Создание индекса



Создание индекса

PUT http://localhost:9200/blogs

```
{  
  "settings" : {  
    "index" : {  
      "number_of_shards" : 5, "number_of_replicas" : 3  
    }  
  }  
}
```

Создание и заполнение индекса



Заполнение индекса пачкой

POST http://localhost:9200/blogs/_bulk

```
{ "index":{"_index":"blogs", "_type":"posts", "_id":"10"} }  
{ "title":"Test1", "description":"First test description" }  
{ "index":{"_index":"blogs", "_type":"posts", "_id":"11"} }  
{ "title":"Test2", "description":"Second test description" }
```

или

POST http://localhost:9200/blogs/post/

```
{ "title":"Test3", "description":"Third test description" }
```

Получение результатов



Получение по id

GET <http://localhost:9200/blogs/posts/1>

Поиск по индексам index1,index2,index3 и по полю

GET http://localhost:9200/index1,index2,index3/_search

```
{  
  "query" : {  
    "match" : { "title": "test" }  
  }  
}
```

Поиск по определённому полю

GET http://localhost:9200/_search?q=name:central

Синтаксис запросов



+ signifies AND operation

| signifies OR operation

- negates a single token

" wraps a number of tokens to signify a phrase for searching

* at the end of a term signifies a prefix query

(and) signify precedence

~N after a word signifies edit distance (fuzziness)

~N after a phrase signifies slop amount

Внедряем в приложение. Вариант 1



```
from elasticsearch import Elasticsearch

es = Elasticsearch()

es.indices.create(index='my-index', ignore=400)

es.index(index="my-index", id=42, body={"any": "data", "timestamp":
datetime.now()})

{'_index': 'my-index',
 '_type': '_doc',
 '_id': '42',
 '_version': 1,
 'result': 'created',
 '_shards': {'total': 2, 'successful': 1, 'failed': 0},
 '_seq_no': 0,
 '_primary_term': 1}

es.get(index="my-index", id=42)['_source']
```

Внедряем в приложение. Вариант 2



```
from rest_framework_elasticsearch import es_views, es_pagination, es_filters
class BlogView(es_views.ListElasticAPIView):
    es_client = es_client
    es_model = BlogIndex
    es_pagination_class = es_pagination.ElasticLimitOffsetPagination
    es_filter_backends = (
        es_filters.ElasticFieldsFilter,
        es_filters.ElasticFieldsRangeFilter,
        es_filters.ElasticSearchFilter,
        es_filters.ElasticOrderingFilter,
        es_filters.ElasticGeoBoundingBoxFilter
    )
```


Внедряем в приложение. Вариант 2.

Продолжение



```
...
es_ordering = 'created_at'
es_filter_fields = (es_filters.ESFieldFilter('tag', 'tags'),)
es_range_filter_fields = (es_filters.ESFieldFilter('created_at'),)
es_search_fields = ( 'tags', 'title', )
es_geo_location_field = es_filters.ESFieldFilter('location')
es_geo_location_field_name = 'location'
```

Внедряем в приложение. Вариант 3



```
# documents.py

from django_elasticsearch_dsl import Document
from django_elasticsearch_dsl.registries import registry
from .models import Car

@registry.register_document
class CarDocument(Document):
    class Index:
        name = 'cars'
        settings = {'number_of_shards': 1,
                    'number_of_replicas': 0}

    ...
```

Внедряем в приложение. Вариант 3



... продолжение

```
class Django:
```

```
    model = Car # The model associated with this Document
```

```
    # The fields of the model you want to be indexed in
```

```
Elasticsearch
```

```
    fields = [
```

```
        'name',
```

```
        'color',
```

```
        'description',
```

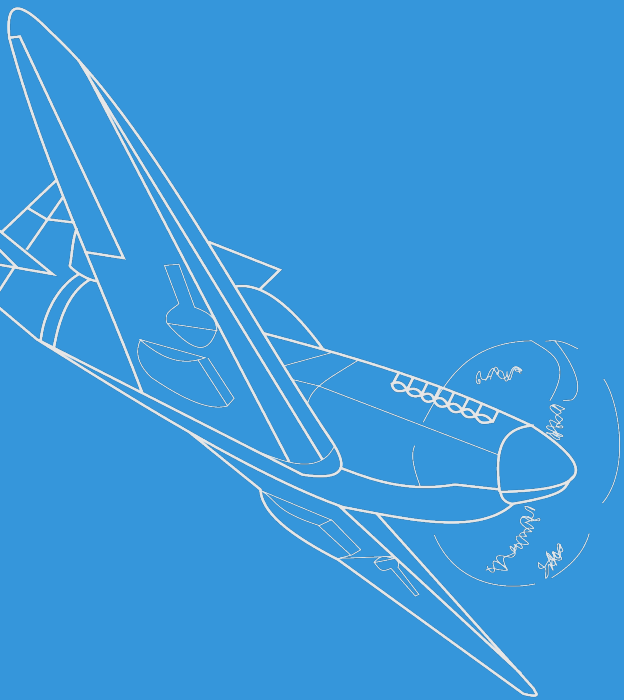
```
        'type',
```

```
]
```

Внедряем в приложение. Вариант 3



```
./manage.py search_index --rebuild  
s = CarDocument.search().filter("term", color="blue")[:30]  
qs = s.to_queryset()
```



KV-хранилища

etcd, consul, vault, aerospike, redis, memcached и даже kafka

Etcd — распределенное KV-хранилище, которая обеспечивает надежное хранение информации на кластере машин. Реализован через B+ дерево.





- Хранить настройки, метадату, флаги конфигурации;
- Алгоритм распределённого консенсуса Raft
 - Чёткое разделение фаз.
 - Явно выделенный лидер
 - Протоколы работы не могут содержать пропусков
- Распределенная блокировка по узлам кластера.
- B-tree индексы для ключей



- Это не in-memory база-данных, все пишется на диск
- HTTP/JSON API
- Подписку на изменение значений(Watch)
- Multi-version Concurrency Control
- Распределенные блокировки
- Транзакции
- Пользователи, роли

etcd. Ограничения



- Все сообщения на запись отправляются на узел-лидер
- Чтение может проходить на любом узле кластера
- Прежде чем сохранить данные большинство узлов должны подтвердить вставку



- Обнаружение служб — используя DNS или HTTP, приложения могут легко найти службы, от которых зависят.
- Состояние проверки работоспособности — может предоставить любое количество проверок работоспособности. Он используется компонентами обнаружения служб для маршрутизации трафика от нездоровых хостов.
- Хранилище ключей / значений — может использовать иерархическое хранилище ключей / значений Consul для любого числа целей, включая динамическую настройку, маркировку функций, координацию, выбор лидера и т. д.



- Развертывание нескольких центров обработки данных — Консул поддерживает несколько центров обработки данных. Он используется для создания дополнительных слоев абстракции для расширения до нескольких регионов.
- Веб-интерфейс — Консул предоставляет своим пользователям прекрасный веб-интерфейс, с помощью которого можно легко использовать и управлять всеми функциями в консуле.



Vault - хранилище секретов. Секреты хранятся в key-value виде. Доступ к хранилищу осуществляется исключительно через API.

- Все данные хранятся в зашифрованном контейнере. Получение самого контейнера не раскрывает данные.
- Гибкие политики доступа. Вы можете создать столько токенов для доступа и управления секретами, сколько вам нужно. И дать им те разрешения, которые необходимы и достаточны для выполнения работ.
- Возможность аудирования доступа к секретам. Каждый запрос к Vault будет записан в лог для последующего аудита.



- Поддерживается автоматическая генерация секретов для нескольких популярных баз данных (postgresql, mysql, mssql, cassandra), для rabbitmq, ssh и для aws.
- Поддержка шифрования-дешифрования данных без их сохранения. Это может быть удобно для передачи данных в зашифрованном виде по незащищённым каналам связи.
- Поддержка полного жизненного цикла секрета: создание/отзыв/завершение срока хранения/продление.



Aerospike – это распределенная база данных NoSQL и хранилище ключевых значений, созданная для удовлетворения потребностей современных приложений в веб-масштабе, обеспечивая стойкость и сильную согласованность без простоя.



Redis (от англ. remote dictionary server) — резидентная система управления базами данных класса NoSQL с открытым исходным кодом, работающая со структурами данных типа «ключ — значение». Используется как для баз данных, так и для реализации кэшей, брокеров сообщений.

1. Хранит базу данных в оперативной памяти;
2. Поддерживает репликацию данных с основных узлов на несколько подчинённых



Apache Kafka — распределённый программный брокер сообщений, проект с открытым исходным кодом, разрабатываемый в рамках фонда Apache.

- Сообщения (messages) записываются по разделам (partition) темы (topic) и хранятся в течении заданного периода
- Подписчики сами опрашивают Kafka на предмет наличия новых сообщений, и указывают, какие записи им нужно прочесть
- Позволяет централизовать сбор, передачу и обработку большого количества сообщений в непрерывных информационных потоках

Домашнее задание № 10



1. Написать функцию, которая будет считать расстояние Левенштейна между двумя словами;
2. Развернуть и наполнить тестовыми данными Elasticsearch;
3. Реализовать поиск по пользователям, продуктам (сущностям);
4. Реализовать метод API для поиска по указанным сущностям и создать страничку HTML с вёрсткой для поиска и отображения результатов.

Рекомендуемая литература

Введение в информационный поиск
Маннинг Кристофер Д., Рагхаван
Прабхакар
Elasticsearch для Python
Elasticsearch для Django



Для саморазвития (опционально)
Чтобы не набирать двумя
пальчиками

Спасибо за
внимание!

Антон Кухтичев



a.kukhtichev@mail.ru



[@toshunster](https://t.me/toshunster)