



elastic



elasticsearch



logstash



kibana

О МНЕ И *tutu.ru*

Разрабатываю аналитические инструменты (clickstream, система АБ-тестов и т.д.)



О МНЕ И **tutu.ru**

Разрабатываю аналитические инструменты (clickstream, система АБ-тестов и т.д.)

Туту.ру — сервис путешествий №1 в России (данные кросс-медийной панели GfK Rus, дек. 2016).

Продаем туры, билеты на самолет, поезд и автобус, бронируем отели, рассказываем о расписании электричек.

900 тыс.

посетителей в день

2003

год основания

14 млн

посетителей в месяц

300

сотрудников



ИЩЕМ DATA-ENGINEER **tutu.ru**

Ищем в команду Data Engineer для решения интересных задач

Вопросы и резюме направлять на адрес:

leonova@tutu.ru

+7 (495) 787-52-05

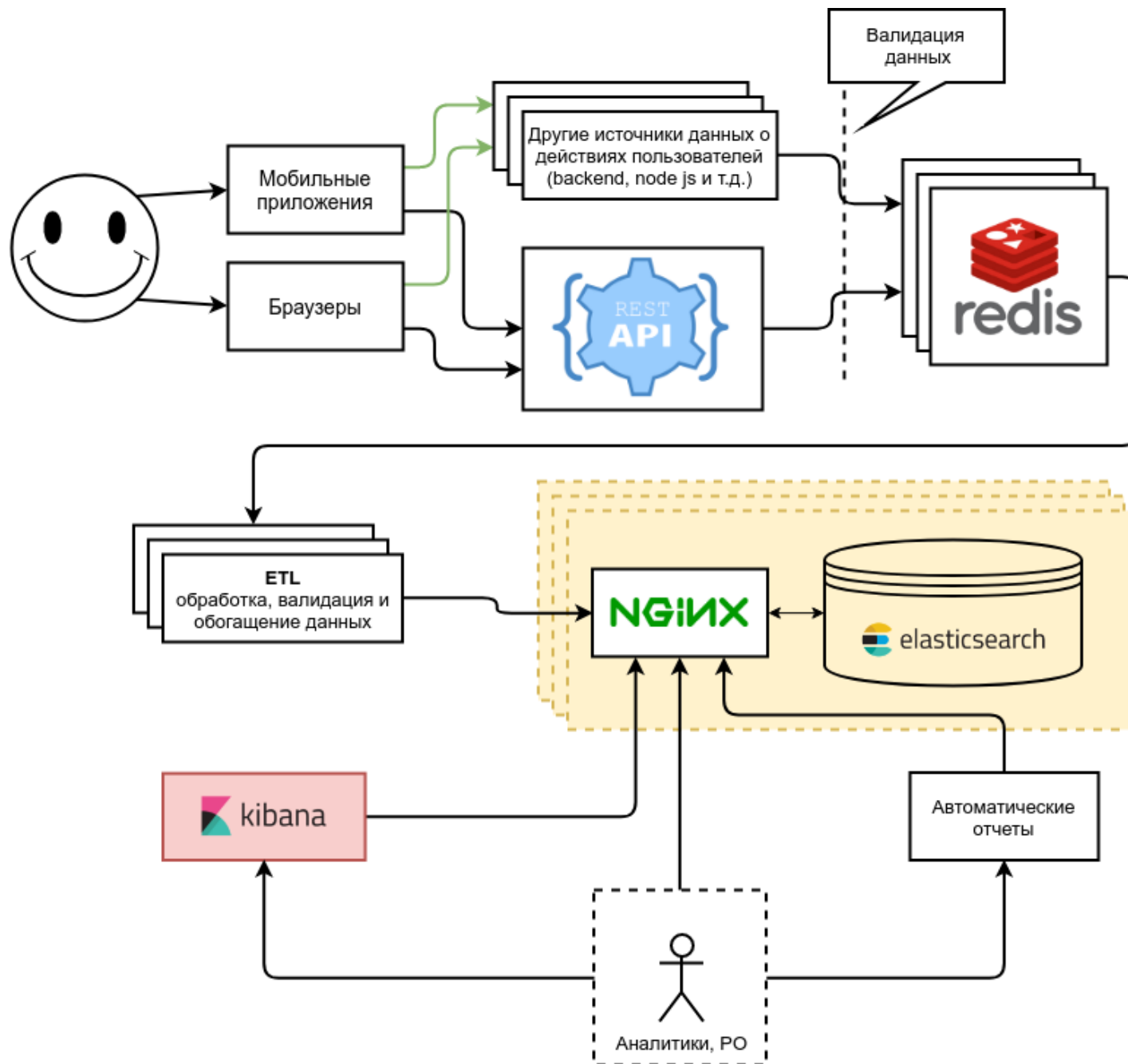


ЗАЧЕМ ЭТО ВСЕ?

- Готовый набор инструментов для построения data-платформы для среднего объема данных
 - Хранение
 - ETL
 - Визуализация
- Удобный и быстрый* доступ к данным (для OLAP)
- Отказоустойчивость
- Быстрое и гибкое масштабирование
- Простейшие агрегации без выгрузки данных
MapReduce для маленьких

НАШИ СИСТЕМЫ НА БАЗЕ ELK

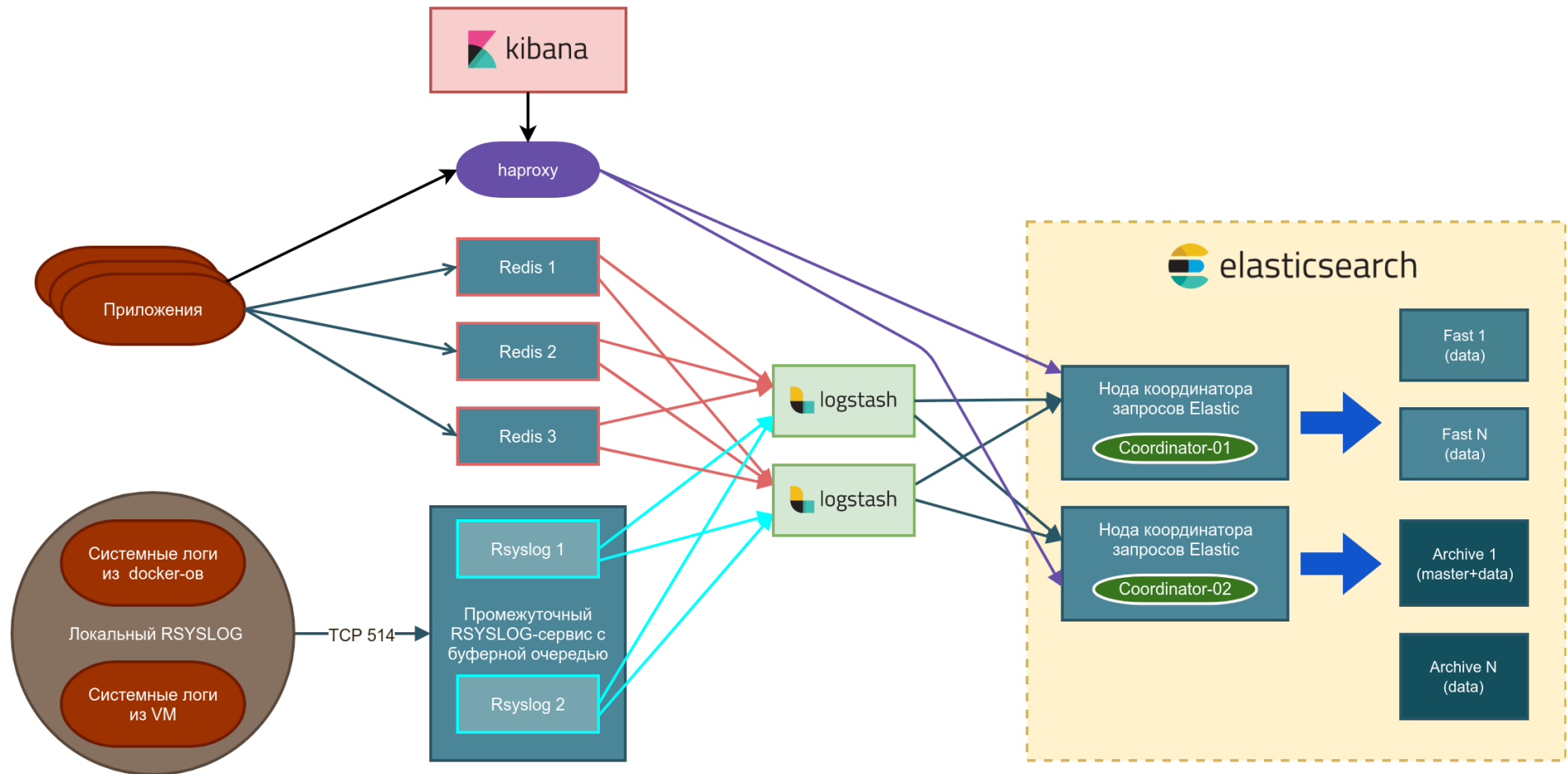
- UserWay (clickstream)
 - ~ 9TB
 - 14.5 млрд. документов
 - В среднем 20 т. записей в минуту, бывают пики до 40 т.



НАШИ СИСТЕМЫ НА БАЗЕ ELK

- UserWay (clickstream)
 - ~ 9TB
 - 14.5 млрд. документов
 - В среднем 20 т. записей в минуту, бывают пики до 40 т.
- Хранение логов со всех площадок
 - ~ 24TB, в пике 32TB
 - 17.5 млрд. документов
 - В среднем 300 т. записей в минуту, бывают пики до 500 т.
 - Не храним долго

PIPELINE ЛОГОВ



ELASTICSEARCH

- Распределенное решения для полнотекстового поиска ...

ELASTICSEARCH

- Распределенное решения для полнотекстового поиска ... а еще аналитическое хранилище

ELASTICSEARCH

- Распределенное решения для полнотекстового поиска ... а еще аналитическое хранилище
- Near Real Time (NRT)

ELASTICSEARCH

- Распределенное решения для полнотекстового поиска ... а еще аналитическое хранилище
- Near Real Time (NRT)
- Разработан на базе Lucene и написан на Java

ELASTICSEARCH

- Распределенное решения для полнотекстового поиска ... а еще аналитическое хранилище
- Near Real Time (NRT)
- Разработан на базе Lucene и написан на Java
- Аналоги: Solr, Sphinx

ВОЗМОЖНОСТИ

- CRUD
- Гибкие возможности для агрегации TimeSeries
- Хранение и поиск по гео-данным
- Работа с документами и параметрами хранилища через REST API
- Легкая вертикальная и горизонтальная масштабируемость

ОГРАНИЧЕНИЯ

- Нет транзакций
- Медленная консистентная запись
- Нет контроля связей
- Нет JOINов, в классическом понимании

ЗАПРОС КОТОРЫЙ УСКОРЯЕМ

```
SELECT count(session_id), count(e1.name), count(e2.name), count(o.id)
FROM session as s
  JOIN ua ON
    s.session_id=ua.session_id
    AND ua.ua_os_name='iOS'
  LEFT JOIN event as e1 ON
    s.session_id=e1.session_id
    AND e1.name='main'
  LEFT JOIN event as e2 ON
    e1.name IS NOT NULL
    AND s.session_id=e2.session_id
    AND e2.name='cabinet'
  LEFT JOIN order as o ON
    e1.name IS NOT NULL
    AND e2.name IS NOT NULL
    AND s.session_id=o.session_id
    AND o.ctime > DATE_SUB(CURDATE(), INTERVAL 2 DAY)
```

```

GET clickstream-2018.04.*/_search
{
  "query": {
    "bool": {
      "must": [
        {
          "term": {
            "type": {
              "value": "session"
            }
          }
        },
        {
          "has_child": {
            "type": "ua",
            "query": {
              "term": {
                "ua_os_name": {
                  "value": "iOS"
                }
              }
            }
          }
        }
      ]
    }
  },
  "size": 0,
  "aggs": {...на след слайде...}
}

```

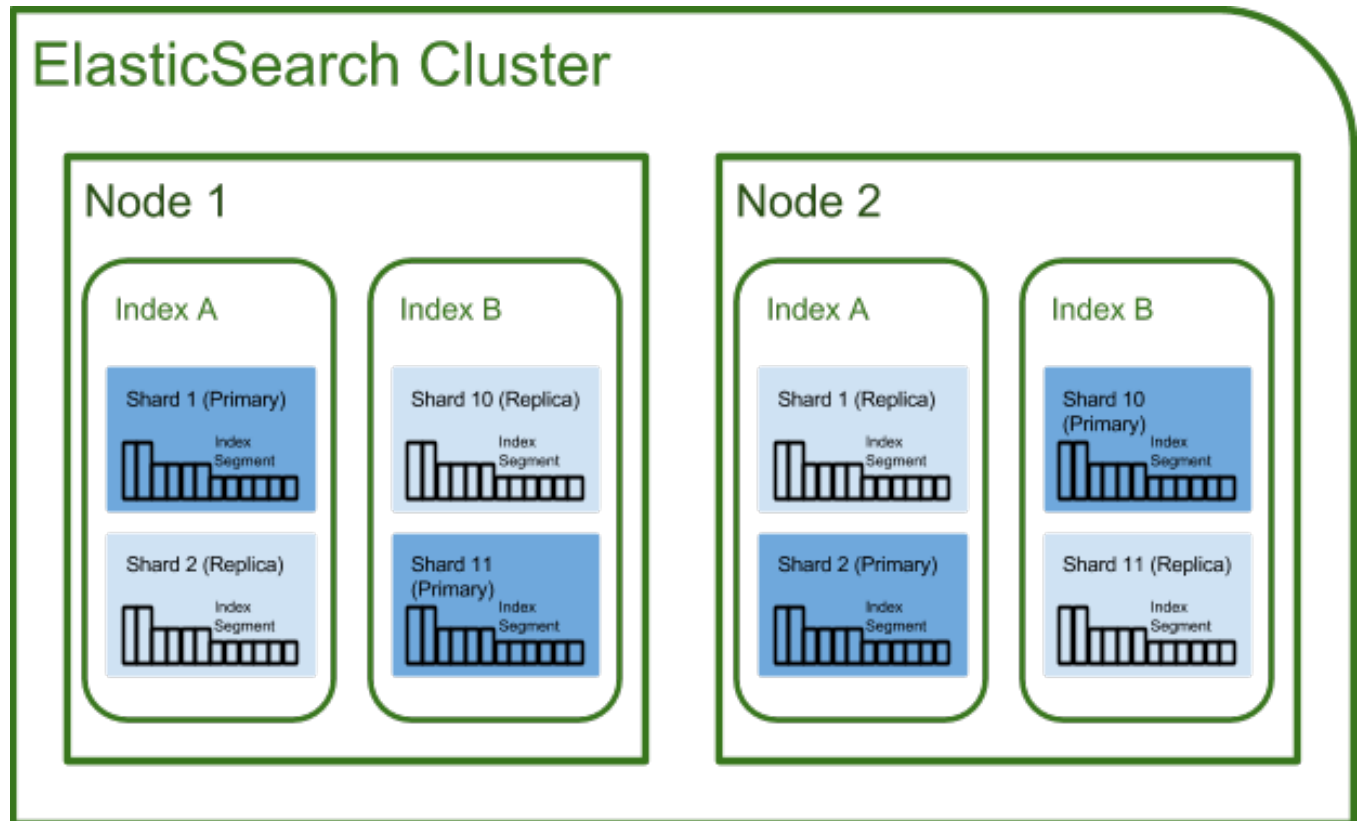
```

...
"aggs": {
  "name_agg_main": {
    "filter": {
      "has_child": {
        "type": "event",
        "query": {
          "term": {"name": "main"}
        }
      }
    },
    "aggs": {
      "name_agg_cabinet": {
        "filter": {
          "has_child": {
            "type": "event",
            "query": {
              "term": {"name": "cabinet"}
            }
          }
        },
        "aggs": {
          "name_agg_order": {
            "filter": {
              "has_child": {
                "type": "order",
                "query": {
                  "range": {
                    "ctime": {
                      "gte": "now-2d"
                    }
                  }
                }
              }
            }
          }
        }
      }
    }
  }
}

```

ОСНОВНЫЕ КОНЦЕПЦИИ

- Нода
- Кластер
- Индекс
- Mapping
- Шарды
- Реплики
- Сегменты



НОДЫ

- Ноды в кластере должны быть +/- одинаковыми
- Размер JVM HEAP оптимально держать не более 32 GB + 32 GB на кеш
- Возможно добавление дисков, но есть нюанс...
- Разные роли: master, data, coordinator
- Нечётное кол-во нод

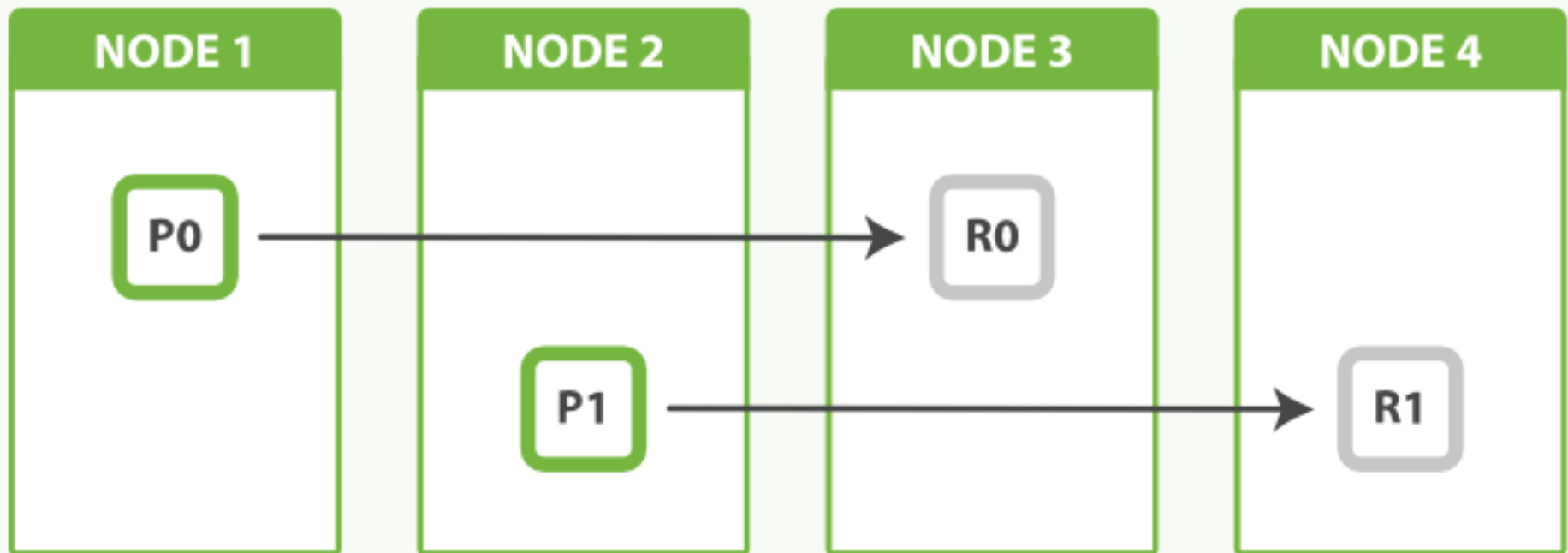
ИНДЕКСЫ

- В параметрах оперделяем:
 - Кол-во шард
 - Кол-во реплик
 - И многое другое :)
- Распределен по нодам
- На диске хранится отдельно от других индексов
- Держать открытым — накладные расходы

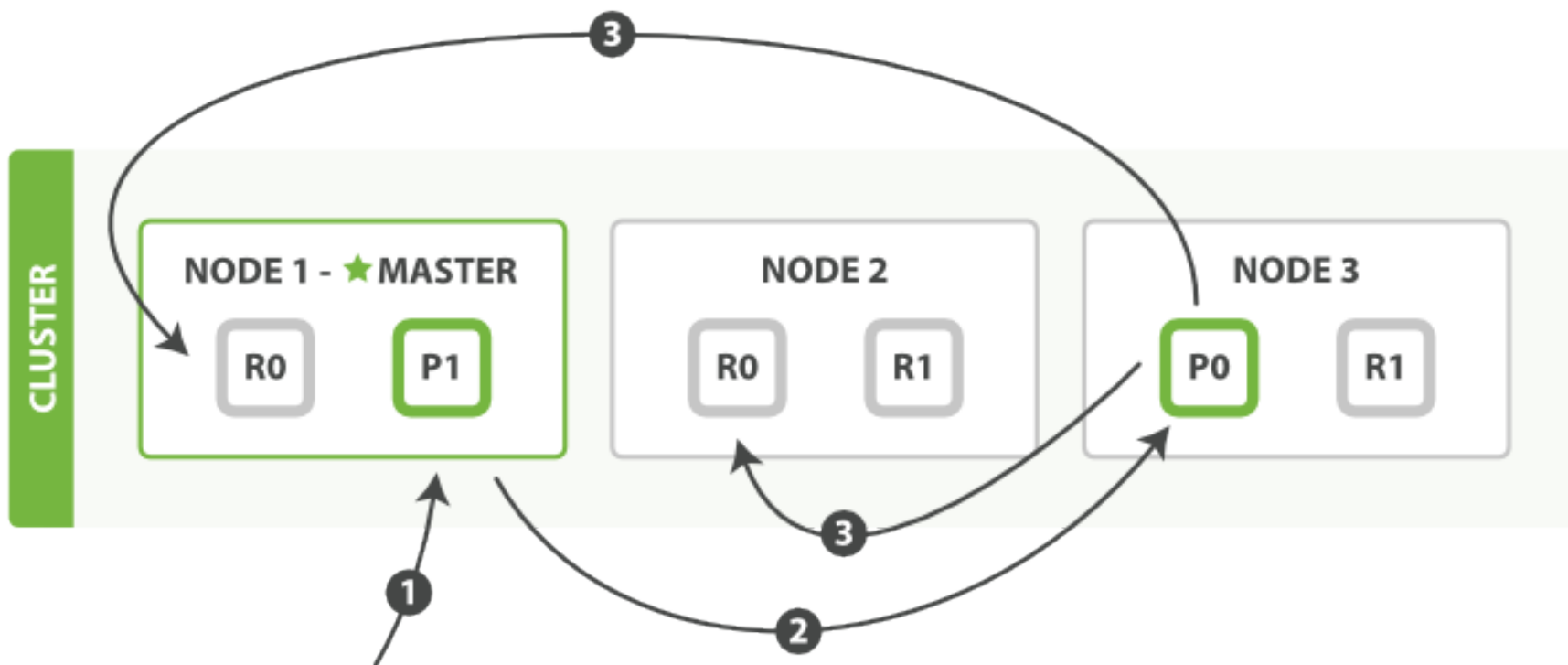
ШАРДЫ

- каждая шарда это индекс Lucene
- макс. кол-во документов ~2,15 млрд.
- кол-во шард задается при создани индекса
можно уменьшить через shrink API
- размер шард ~30-40GB

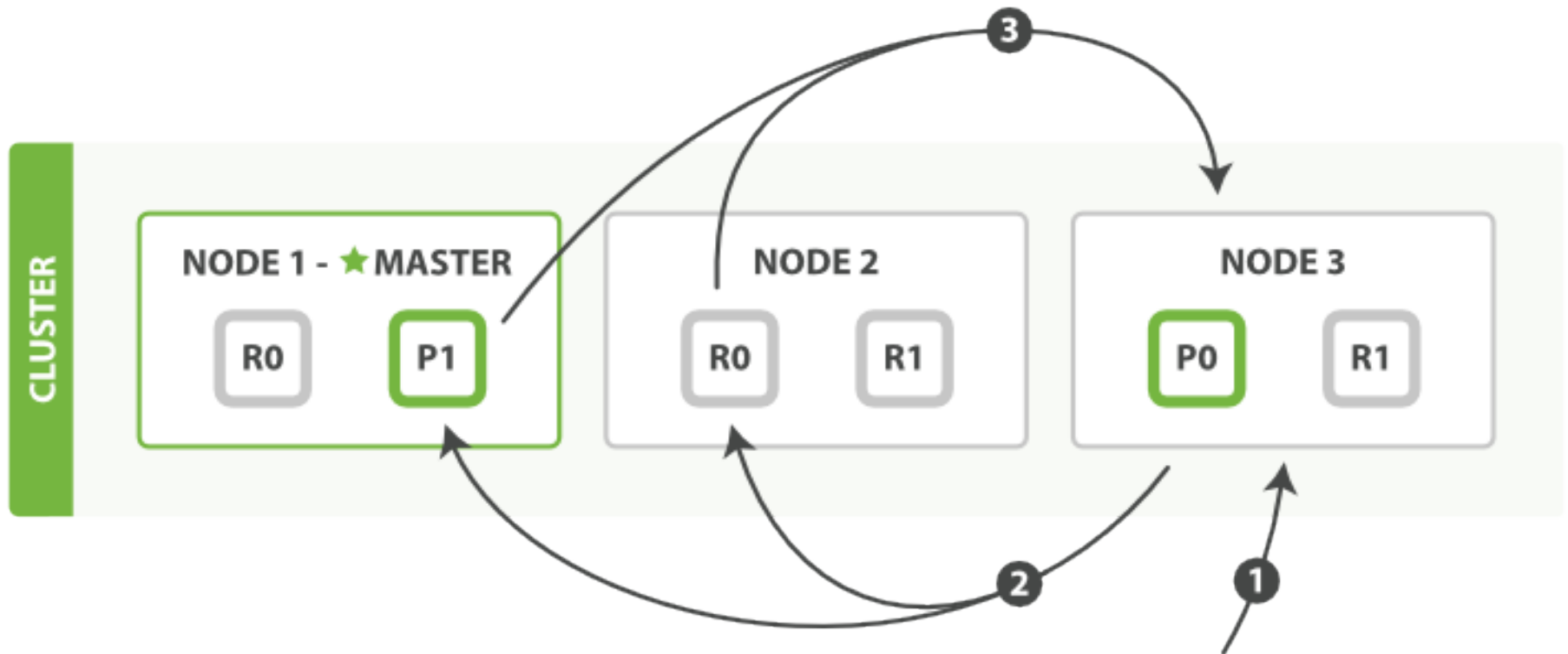
РЕПЛИКИ В КЛАСТЕРЕ



МАРШРУТ ЗАПРОСА НА ИНДЕКСАЦИЮ ДОКУМЕНТА



МАРШРУТ ЗАПРОСА ПОЛУЧЕНИЯ ДОКУМЕНТОВ



ИНДЕКСАЦИЯ ДОКУМЕНТОВ

- Создание
- Атомарные операции
- Сложные, но быстрые запросы: «обнови если ...»
- Удаление
 - ... а если найду?
- Bulk API
 - Размеры очередей thread pool
 - Самый быстрый способ удаления vs delete by query

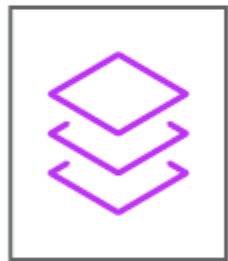
ЗАПИСЬ ДАННЫХ

- Запись в translog
- Создание или дополнение сегмента

<https://www.youtube.com/watch?v=YOkIKW9LJNY>

Inside a Shard

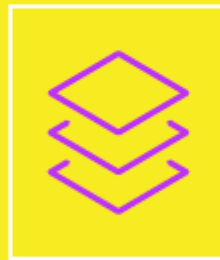
1. New docs
are indexed



TRANSLLOG



IN-MEMORY
BUFFER

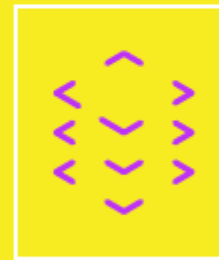


2. Refresh

TRANSLLOG



IN-MEMORY
BUFFER

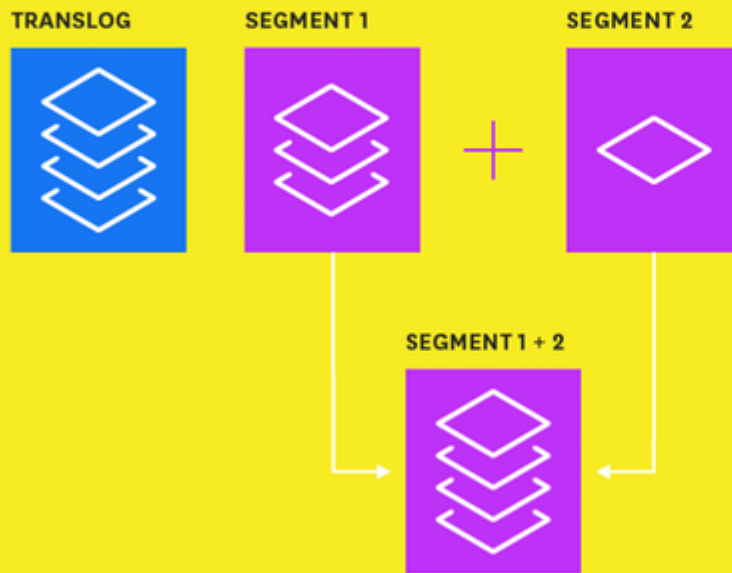


SEGMENT 1
(IN FILE SYSTEM
CACHE)

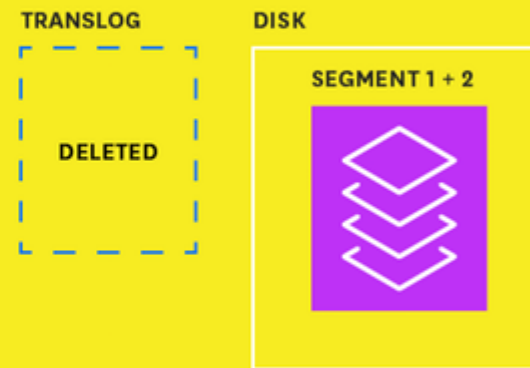


Inside a Shard

3. Over time, more segments are created and merged within the file system cache...



4. Flush



ЗАПИСЬ ДАННЫХ

- Запись в translog
- Создание или дополнение сегмента
<https://www.youtube.com/watch?v=YOklKW9LJNY>
- Refresh и flush
- Обновление или удаление записей без физического удаления
- Многопоточная отправка запросов

МАСШТАБИРУЕМОСТЬ И ОТКАЗОУСТОЙЧИВОСТЬ

- Реплики
- Кластеризация
- Шарда как минимальная единица
- Возможность динамического задания размещения индекса на одной или группе нод
- Hot-Warm-Cold
- Circuit breakers
 - лимит памяти на выполнение агрегаций
 - кол-во скриптовых запросов в ед. времени и т.д.

MAPPING И СХЕМА ДАННЫХ

- Список полей и их типы

```
{
  "clickstream-2018.04.16": {
    "mappings": {
      "_doc": {
        "dynamic": "strict",
        "properties": {
          "name": {
            "type": "keyword"
          },
          "order_id": {
            "type": "integer"
          },
          "price_total": {
            "type": "double"
          },
          "ip": {
            "type": "ip"
          }
        }
      }
    }
  }
}
```

MAPPING И СХЕМА ДАННЫХ

- Список полей и их типы
- Только добавление полей

MAPPING И СХЕМА ДАННЫХ

- Список полей и их типы
- Только добавление полей
- Типы индексаторов, стоп-слова, синонимы, лингвистические анализаторы...

MAPPING И СХЕМА ДАННЫХ

- Список полей и их типы
- Только добавление полей
- Типы индексаторов, стоп-слова, синонимы, лингвистические анализаторы...
- Строгость структуры

MAPPING И СХЕМА ДАННЫХ

- Список полей и их типы
- Только добавление полей
- Типы индексаторов, стоп-слова, синонимы, лингвистические анализаторы...
- Строгость структуры
- Связи, parent-child relationship
 - Хранение всех «детей» на одной шарде
 - Только «многие к одному»
 - Замедление запросов с использованием связей
 - Усложнение шардинга

MAPPING И СХЕМА ДАННЫХ

- Список полей и их типы
- Только добавление полей
- Типы индексаторов, стоп-слова, синонимы, лингвистические анализаторы...
- Строгость структуры
- Связи, parent-child relationship
- nested и object

MAPPING И СХЕМА ДАННЫХ

- Список полей и их типы
- Только добавление полей
- Типы индексаторов, стоп-слова, синонимы, лингвистические анализаторы...
- Строгость структуры
- Связи, parent-child relationship
- nested и object
- Формат дат, если хотим не по ISO, то определяем отдельно

```
{
  ...
  "ctime": {
    "type": "date",
    "format": "dd.MM.yyyy|dd.MM.yyyy HH:mm:ssZ|dd.MM.yyyy HH:mm:ssZZ|
              dd.MM.yyyy HH:mm:ss ZZZ|dd.MM.yyyy'T'HH:mm:ssZ|dd.MM.yyyy'T'HH:mm:ssZZ|
              dd.MM.yyyy'T'HH:mm:ss ZZZ|date_optional_time|epoch_millis"
  }
  ...
}
```

MAPPING И СХЕМА ДАННЫХ

- Список полей и их типы
- Только добавление полей
- Типы индексаторов, стоп-слова, синонимы, лингвистические анализаторы...
- Строгость структуры
- Связи, parent-child relationship
- nested и object
- Формат дат, если хотим не по ISO, то определяем отдельно
- Оптимизация индекса: `doc_values`, `_all`, `enabled`, `index=false` (`not_analyzed`)

MAPPING И СХЕМА ДАННЫХ

- Список полей и их типы
- Только добавление полей
- Типы индексаторов, стоп-слова, синонимы, лингвистические анализаторы...
- Строгость структуры
- Связи, parent-child relationship
- nested и object
- Формат дат, если хотим не по ISO, то определяем отдельно
- Оптимизация индекса: doc_values, _all, enabled, index=false (not_analyzed)
- Типы (буду удалены в 9 версии)

ЗАПРОСЫ

- SQL vs JSON

```
SELECT SUM(price_total)
FROM order
WHERE item_count!=1
```

```
GET clickstream-2018.04.*/_search
{
  "query": {
    "bool": {
      "must": [
        {
          "term": { "type": "order" }
        }
      ],
      "must_not": [
        {
          "term": { "item_count": 1 }
        }
      ]
    }
  },
  "aggs": {
    "sum_price": {
      "sum": { "field": "price_total" }
    }
  }
}
```


ЗАПРОСЫ

- SQL vs JSON
- Пока только JSON в далее появится SQL

ЗАПРОСЫ

- SQL vs JSON
- Пока только JSON в далее появится SQL
- Запросы к нескольким индексам или кластерам

ЗАПРОСЫ

- SQL vs JSON
- Пока только JSON в далее...
- Запросы к нескольким...
- Агрегации

```
GET clickstream-2018.04.*/_search
{
  "query": { "match_all": {} },
  "size": 0,
  "timeout": "10s",
  "aggs": {
    "name_agg_main": {
      "filter": { ... },
      "aggs": {
        "name_agg_cabinet": {
          "filter": { ... },
          "aggs": {
            "name_agg_order": {
              "filter": { ... }
            }
          }
        }
      }
    }
  }
}
```


ЗАПРОСЫ

```
{
  "query": { ... }
  "aggs": {
    "my_date_histo": {
      "date_histogram": {
        "field": "time",
        "interval": "hour",
        "format": "yyyy-MM-dd HH",
        "keyed": true,
        "time_zone": "+03:00"
      },
      "aggs": {
        "profit": {
          "scripted_metric": {
            "init_script": "params._agg.speed = []",
            "map_script": "params._agg.speed.add((doc.count_complete.value + doc.count_version_conflicts.value) / doc.running_time_in_millis);",
            "combine_script": "double result = 0; int count = 0; for (speed in params._agg.speed) { count++; result += speed; } return result / count",
            "reduce_script": "double result = 0; for (a in params._aggs) { if (a != null) { result += a } } return result",
            "params": {
              "_agg": {}
            }
          }
        }
      }
    }
  }
}
```

ЗАПРОСЫ

- SQL vs JSON
- Пока только JSON в далее появится SQL
- Запросы к нескольким индексам или кластерам
- Агрегации
- scroll

SCROLL



```
GET /clickstream-2018.04.16/_search?scroll=1m
{
  "query": {
    "match_all": {}
  },
  "size": 100
}
```

SCROLL

```
GET /clickstream-2018.04.16/_search?scroll=1m
{
  "query": {
    "match_all": {}
  },
  "size": 100
}
```

```
{
  "_scroll_id": "DnF1ZXJ5VGhlbkZldGNoIAA...",
  "took": 31,
  "timed_out": false,
  "_shards": { ... },
  "hits": { ... }
}
```

SCROLL

```
GET /clickstream-2018.04.16/_search?scroll=1m
{
  "query": {
    "match_all": {}
  },
  "size": 100
}
```

```
{
  "_scroll_id": "DnF1ZXJ5VGhlbkZldGNoIAA...",
  "took": 31,
  "timed_out": false,
  "_shards": { ... },
  "hits": { ... }
}
```

```
GET /_search/scroll
{
  "scroll" : "1m",
  "scroll_id": "DnF1ZXJ5VGhlbkZldGNoIAA..."
}
```


SCROLL

```
GET /clickstream-2018.04.16/_search?scroll=1m
{
  "query": {
    "match_all": {}
  },
  "size": 100
}
```

```
{
  "_scroll_id": "DnF1ZXJ5VGhlbkZldGNoIAA...",
  "took": 31,
  "timed_out": false,
  "_shards": { ... },
  "hits": { ... }
}
```

```
GET /_search/scroll
{
  "scroll" : "1m",
  "scroll_id": "DnF1ZXJ5VGhlbkZldGNoIAA..."
}
```

```
{
  "_scroll_id": "DnF1ZXJ5VGhlbkZldGNoIAA...",
  "took": 15,
  "timed_out": false,
  "terminated_early": true,
  "_shards": { ... },
  "hits": { ... }
}
```

ЗАПРОСЫ


- SQL vs JSON
- Пока только JSON в далее появится SQL
- Запросы к нескольким индексам или кластерам
- Агрегации
- scroll
- Оптимизация запросов через Profile API или в kibana

ЗАПРОСЫ

- SQL vs JSON
- Пока только JSON в далее появится SQL
- Запросы к нескольким индексам или кластерам
- Агрегации
- scroll
- Оптимизация запросов через Profile API или в kibana
- При ответе проверять все ли шарды ответили успешно

ПРИЕМЫ ИЗВЛЕЧЕНИЯ ДАННЫХ

- Получение слайса отправив запрос к одной шарде для аппроксимации
 - Запрос данных с шард без проксирования
 - Отключение сортировки и scoring
-



```
GET /_search?scroll=1m
{
  "sort": [ "_doc" ]
}
```

ПРОЕКТИРОВАНИЕ ИНДЕКСА

- «нарезка» данных по индексам
 - По времени
 - По ID родителей
 - Настройка миграции данных (комбинация двух подходов выше)
- Даты: создания на клиенте, дата попадания в систему и обновление записи
- Тестирование на ноде с одной шардой

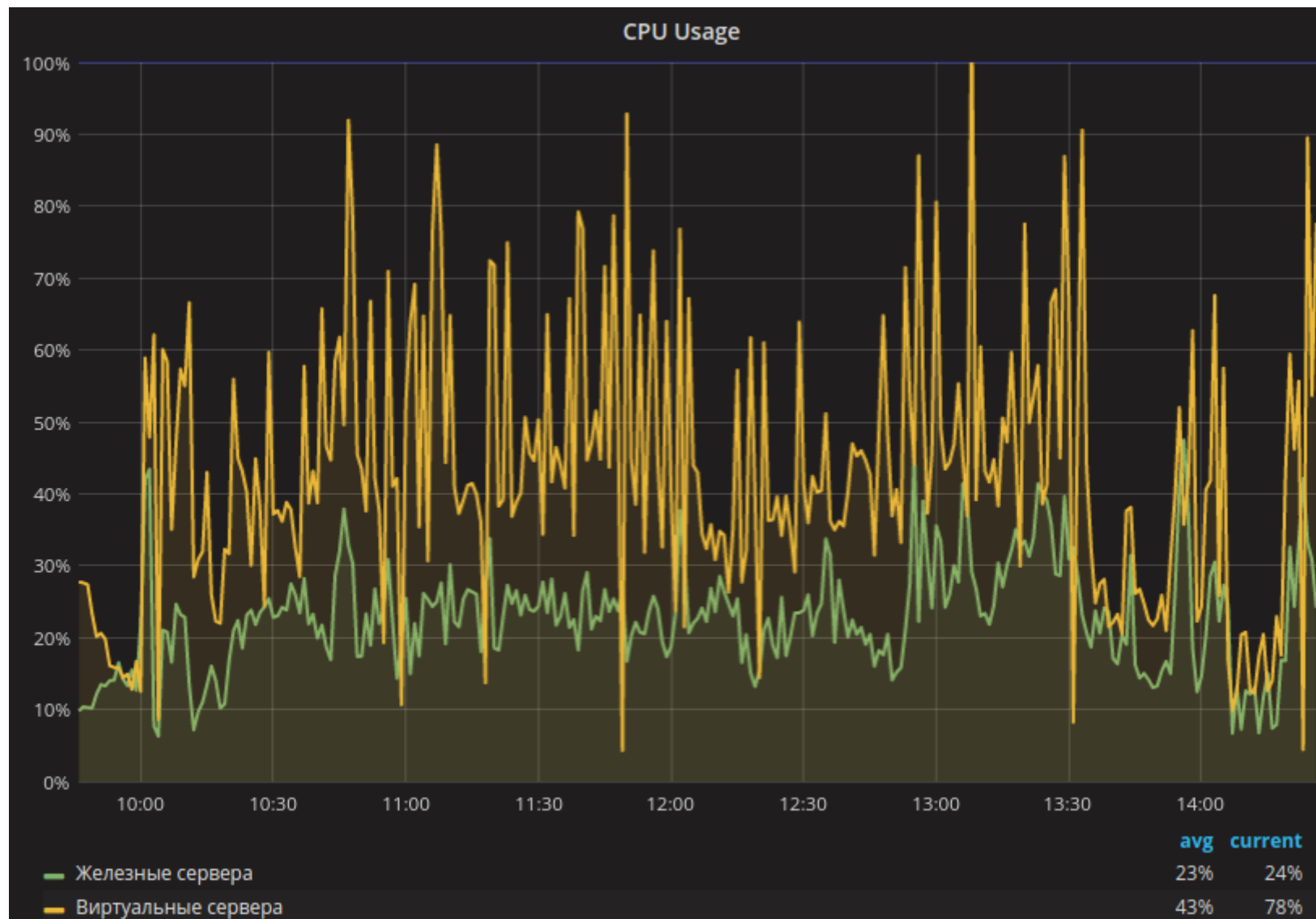
МОНИТОРИНГ

- Инструменты
 - X-Pack
 - graphite + grafana + es2graphite
- Лог входящих запросов
- Логировать активные задачи для каждой ноды
- Следить за размером сегментов
- Метрики
 - Основные метрики системы (LA, IO и т.д.)
 - HEAP + GC
 - Время обработки запроса (запись + чтение)
 - Кол-во открытых контекстов
 - Кол-во запросов на разные типы операций
 - Состояние очередей

МИГРАЦИЯ ДАННЫХ

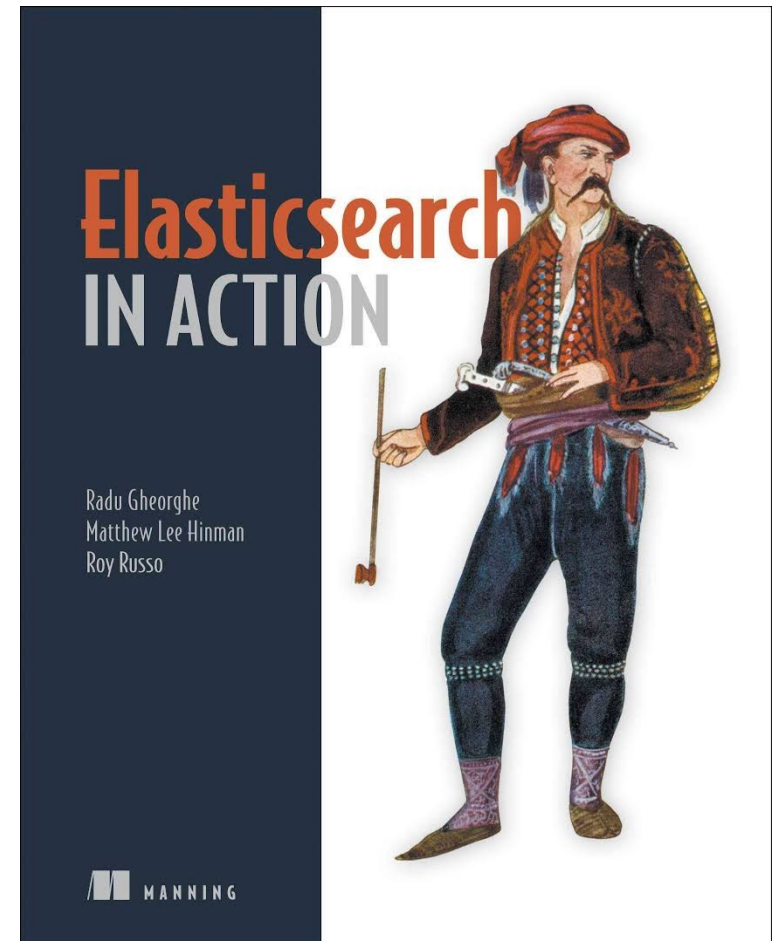
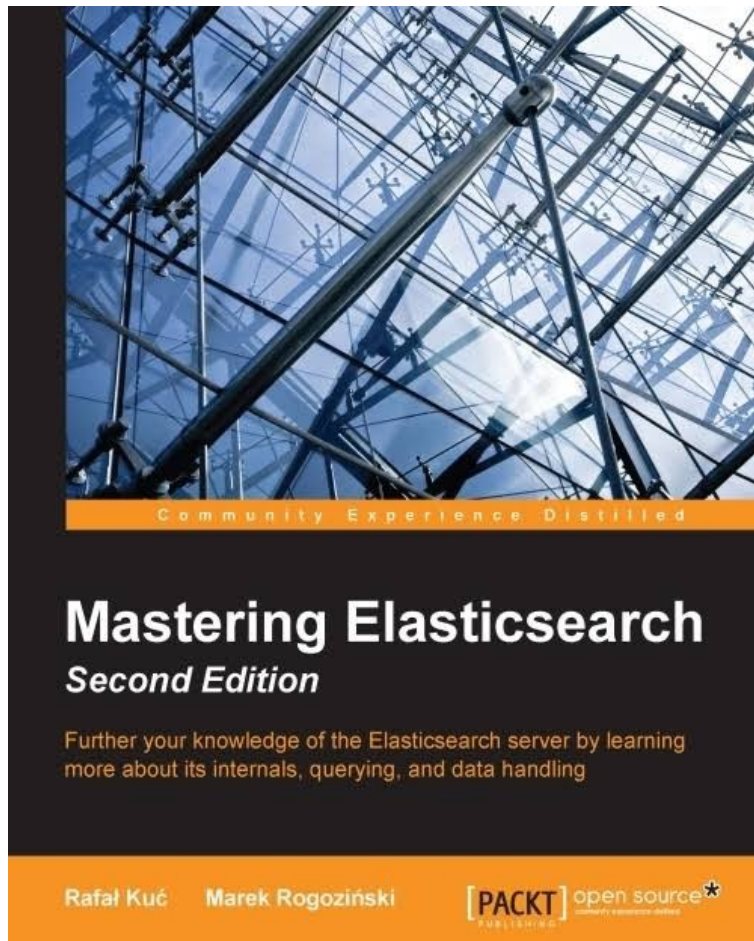
- Reindex с 5.0 поддержка удаленного получения данных
- Перегон шард через решардинг
- Snapshot/Restore
- elasticsearch-dump

ЖЕЛЕЗО VS ОБЛАКО



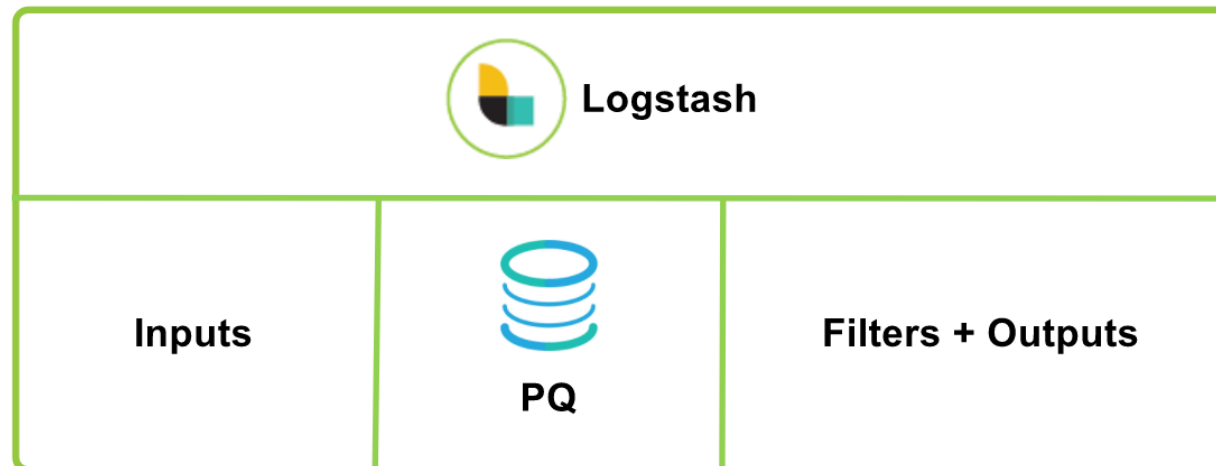
КНИГИ

Mastering Elasticsearch или ElasticSearch in Action



LOGSTASH

- Трансформация
- Валидация данных
- Обогащение
- Имеет большое количество плагинов
- Расширяем через плагины на ruby



КОНФИГУРАЦИЯ

```
input {
  file {
    path => "/tmp/access_log"
    start_position => "beginning"
  }
}

filter {
  grok {
    match => { "message" => "%{COMBINEDAPACHELOG}" }
  }
  date {
    match => [ "timestamp" , "dd/MMM/yyyy:HH:mm:ss Z" ]
  }
}

output {
  elasticsearch { hosts => ["elasticsearch_host:9200"] }
}
```

НАДЕЖНОСТЬ* LOGSTASH

- Кластеризации нет - «прячем» за балансировщики
- DLQ только для output elasticsearch
- Очередь на диске той же ноды
- В версиях до 6-ки имел «детские» болезни
- Потеря пачки событий с каждого воркера
- Не обработанные ошибка в filter
- Отсутствие exactly-once, но есть «но»

KIBANA

- Рисование гистограмм, пирогов и т.д.
- Фильтрация по сложным запросам
- Дашборды
- Мониторинг всего стека через X-Pack
- Встраиваемые визуализации
- Dev tool — консоль запросов и отладки запросов

СОБЕРЕМ PIPELINE

- Docker + docker-compose
- Задача «реализовать систему сбора и данных о действиях пользователя»
 - Реализовать endpoint куда приложение будет присылать данные о совершенных пользователем действиях
 - Сохранять в elasticsearch данные о каждом событии
 - Распарсить и сохранить в структурированном виде данные о устройстве клиента. Данные о клиенте пишутся отдельной записью с типом device
 - Каждый пользователь в нашем хранилище имеет часть данных которые попадают в нашу систему только когда пользователь совершает целевое действие, вот несколько из них:
 - Пользователь может иметь ранее совершенные заказы информацию по которым нам нужно долить в хранилище для анализа
 - Пользователь мог иметь историю визитов на наш сайт, которая сохранилась в access-логах нашего http сервера и нам хорошо бы получить эту историю тоже

О ЧЕМ УЗНАЛИ

- Из чего состоит стек ELK
- Области применения
- Собрали pipeline на ELK
- Как оптимально выбрать структуру данных
- Какие есть инструменты для поддержки
- Какие есть возможности для агрегации данных
- Пример архитектуры

ГДЕ ИСКАТЬ ИНФОРМАЦИЮ

- Документация <https://www.elastic.co/guide/index.html>
- Блог <https://www.elastic.co/blog>
- YouTube <https://www.youtube.com/user/elasticsearch/videos>