OZON tech

Индексация в поисковой платформе Ozon

Денис Габайдулин, Search Runtime

Денис Габайдулин

- Staff software engineer в Ozon
- Ранее работал в OK (big data), Yandex (YQL)
- Более 10 лет опыта в Infrastructure Development

О чем доклад?

ozon{ech

- Архитектура и реализация индексации в движке O2*
- Архитектура записи документов в Apache Lucene

O2 — поисковая платформа Ozon
O2 = O_2 (кислород)

Поисковая платформа Ozon

Поисковая платформа Ozon

ozon tech

Текущий статус

Движок на базе **Apache Lucene**

Десятки разных индексов

Размеры индексов: от 10K **до 160M** документов

Размеры индексов: от 10Mb **до 500Gb**

Максимальное количество полей в документе: 1000+ 😕

Десятки тысяч RPS поисковых запросов в движок (без кэша)

Поисковая платформа Ozon

ozon tech

Текущий статус

Успешно прошла **два сезона распродаж**

Рост основного индекса **в 10+ раз**

RPS вырос **в 10+ раз**

OZON tech

Но есть же ElasticSearch!

Почему не ElasticSearch?

- Полный контроль над процессом ранжирования

- Полный контроль над процессами построения и репликации индекса

- Доступ к низкоуровневым оптимизациям на стадии поиска и фильтрации

Почему не ElasticSearch?

ozon{ech

Полный контроль над процессом ранжирования

- Многоуровневое ранжирование
- Хотим менять базовую формулу ранжирования L1 (BM25)
- Хотим использовать ML в L1, L2 (learning to rank)

Почему не ElasticSearch?

ozon{ech

Полный контроль над процессом ранжирования

- Многоуровневое ранжирование
- Хотим менять базовую формулу ранжирования L1 (BM25)
- Хотим использовать ML в L1, L2 (learning to rank)

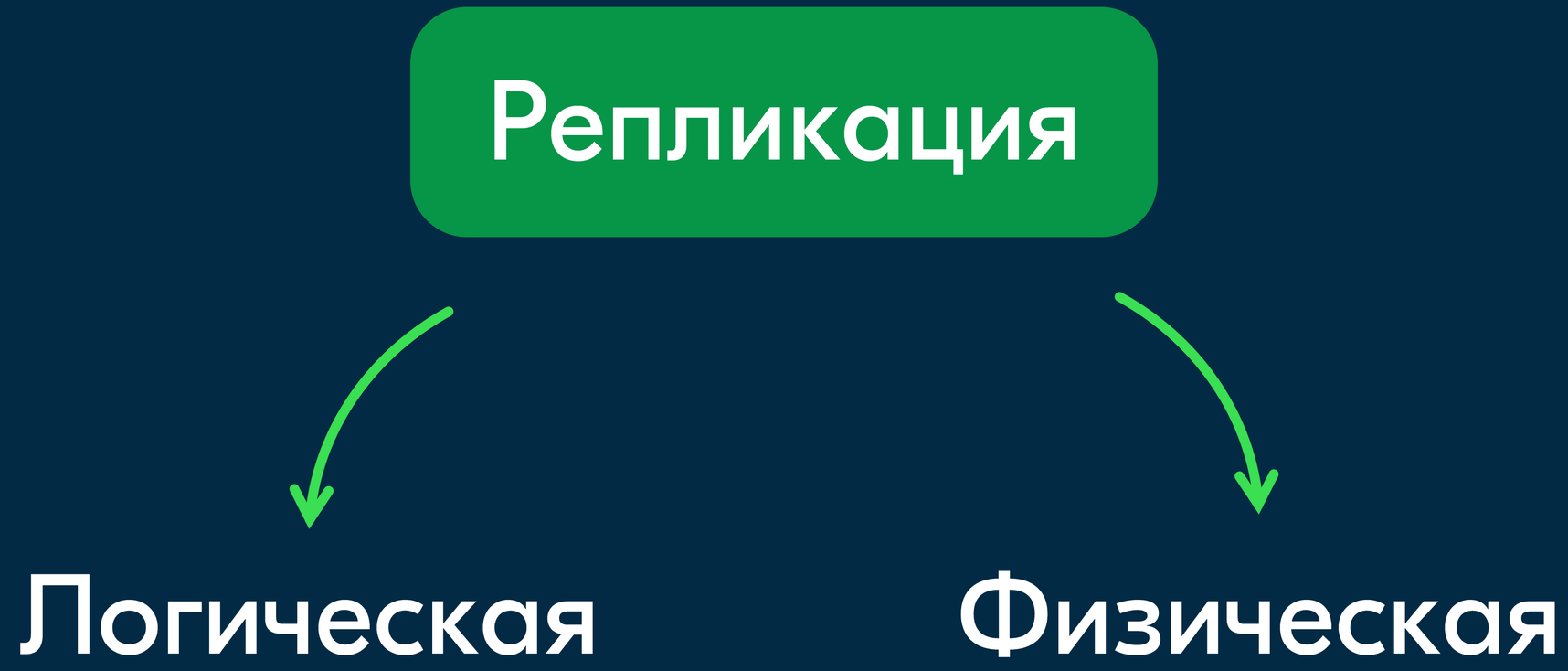
Почему не ElasticSearch?

ozon{ech

Полный контроль над процессом ранжирования

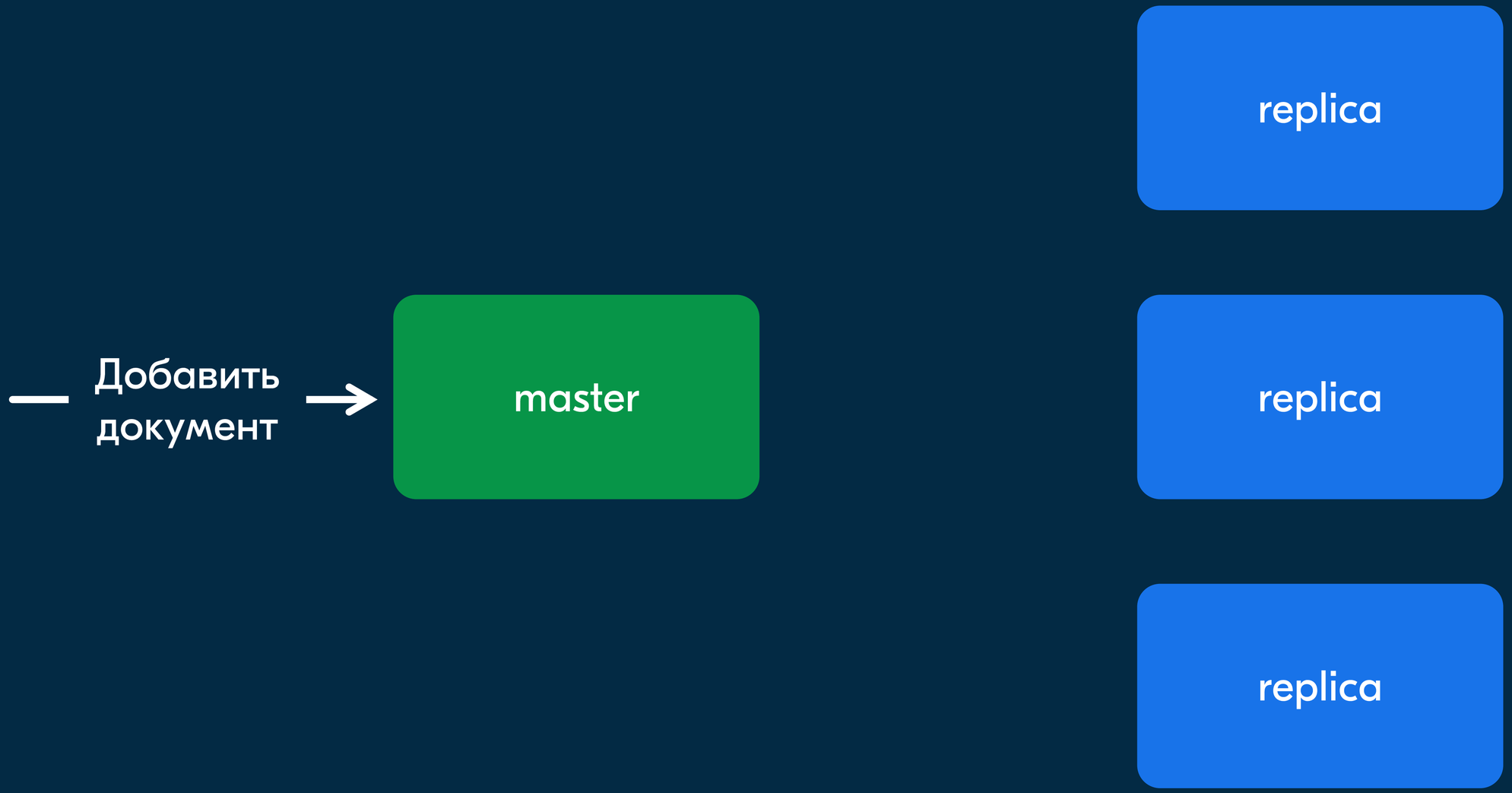
- Многоуровневое ранжирование
- Хотим менять базовую формулу ранжирования L1 (BM25)
- Хотим использовать ML в L1, L2 (learning to rank)

Почему не ElasticSearch?



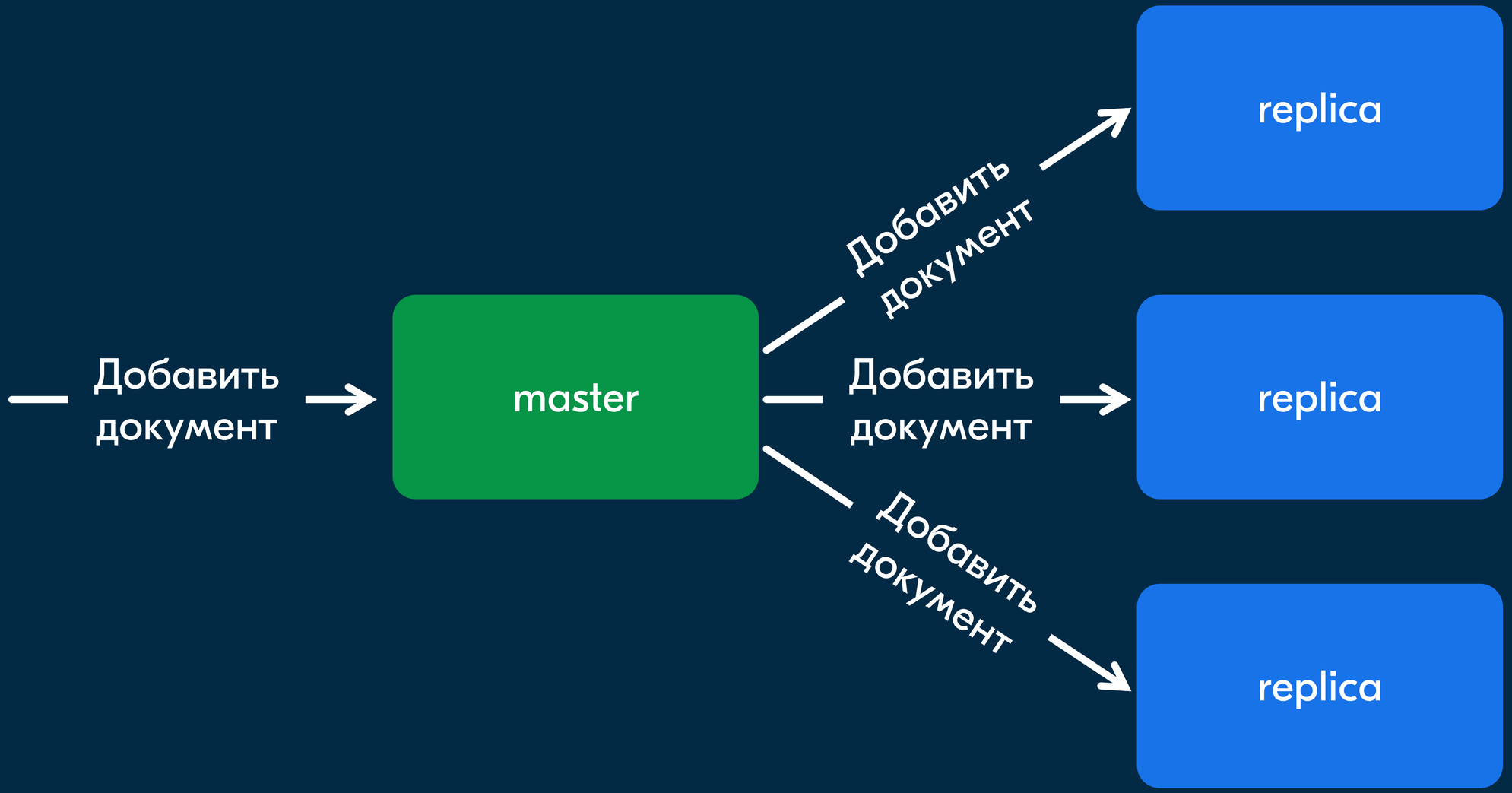
Почему не ElasticSearch?

Логическая репликация



Почему не ElasticSearch?

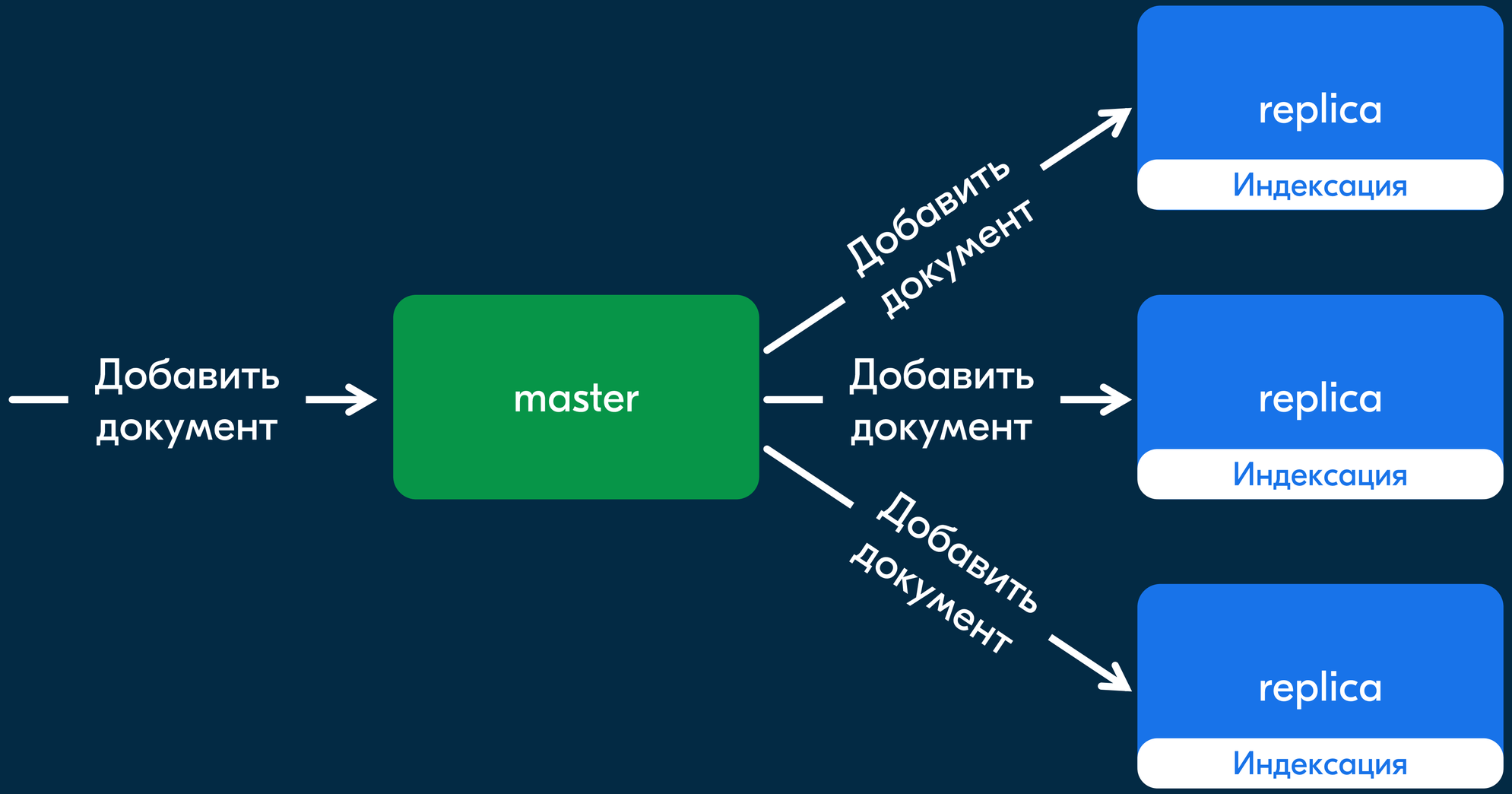
Логическая репликация



Почему не ElasticSearch?

ozon tech

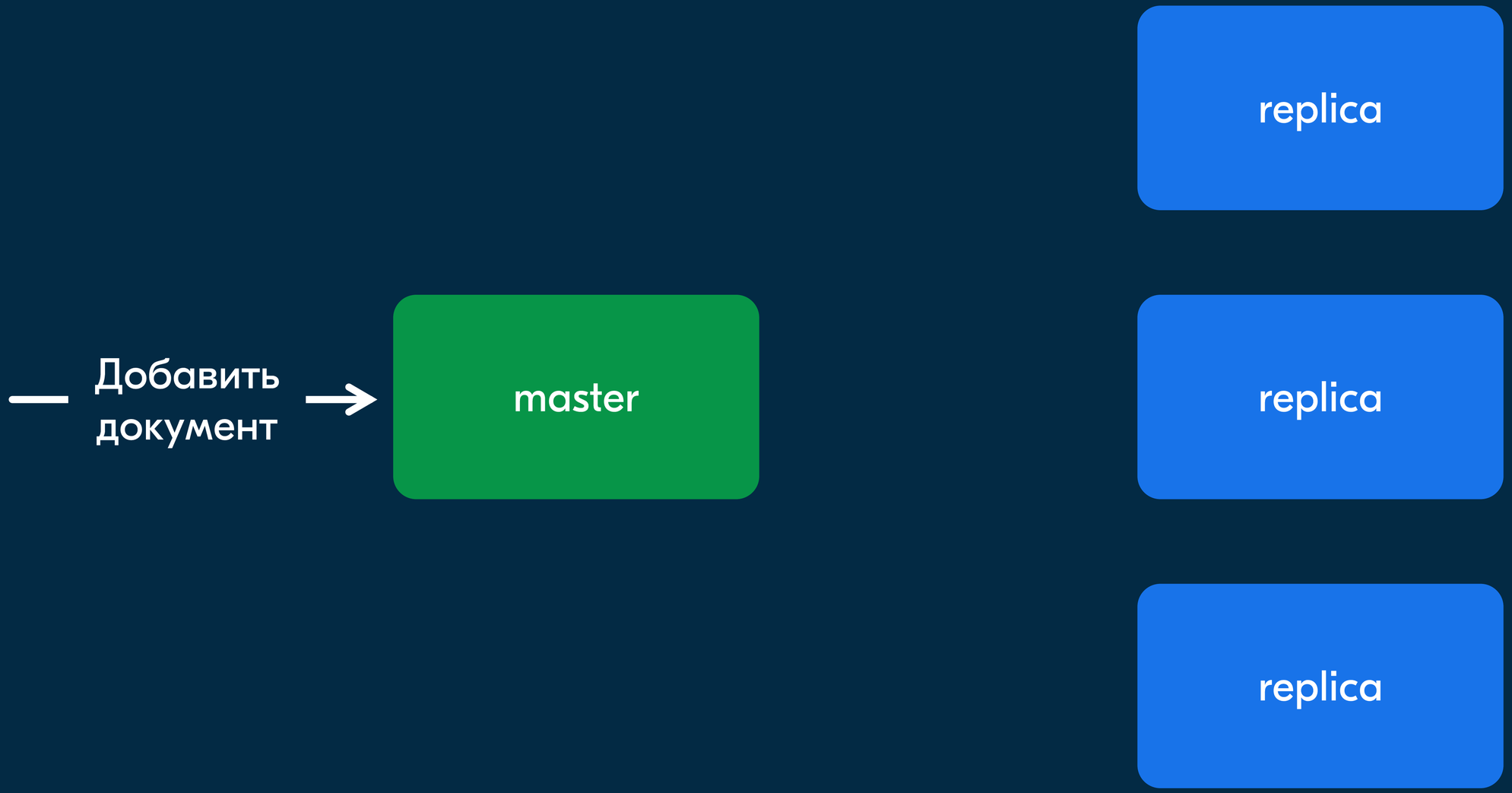
Логическая репликация



Почему не ElasticSearch?

ozon{ech

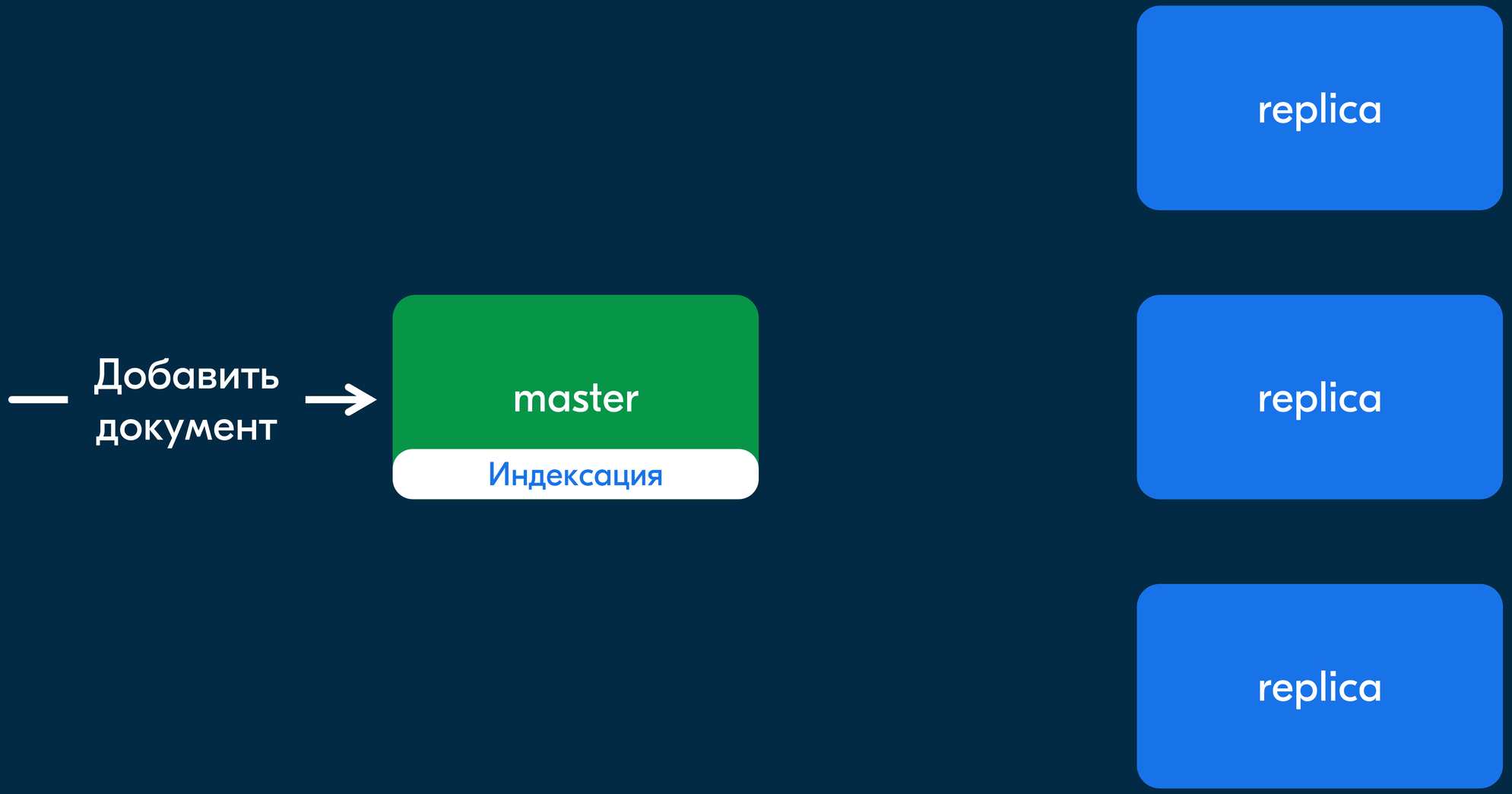
Физическая репликация



Почему не ElasticSearch?

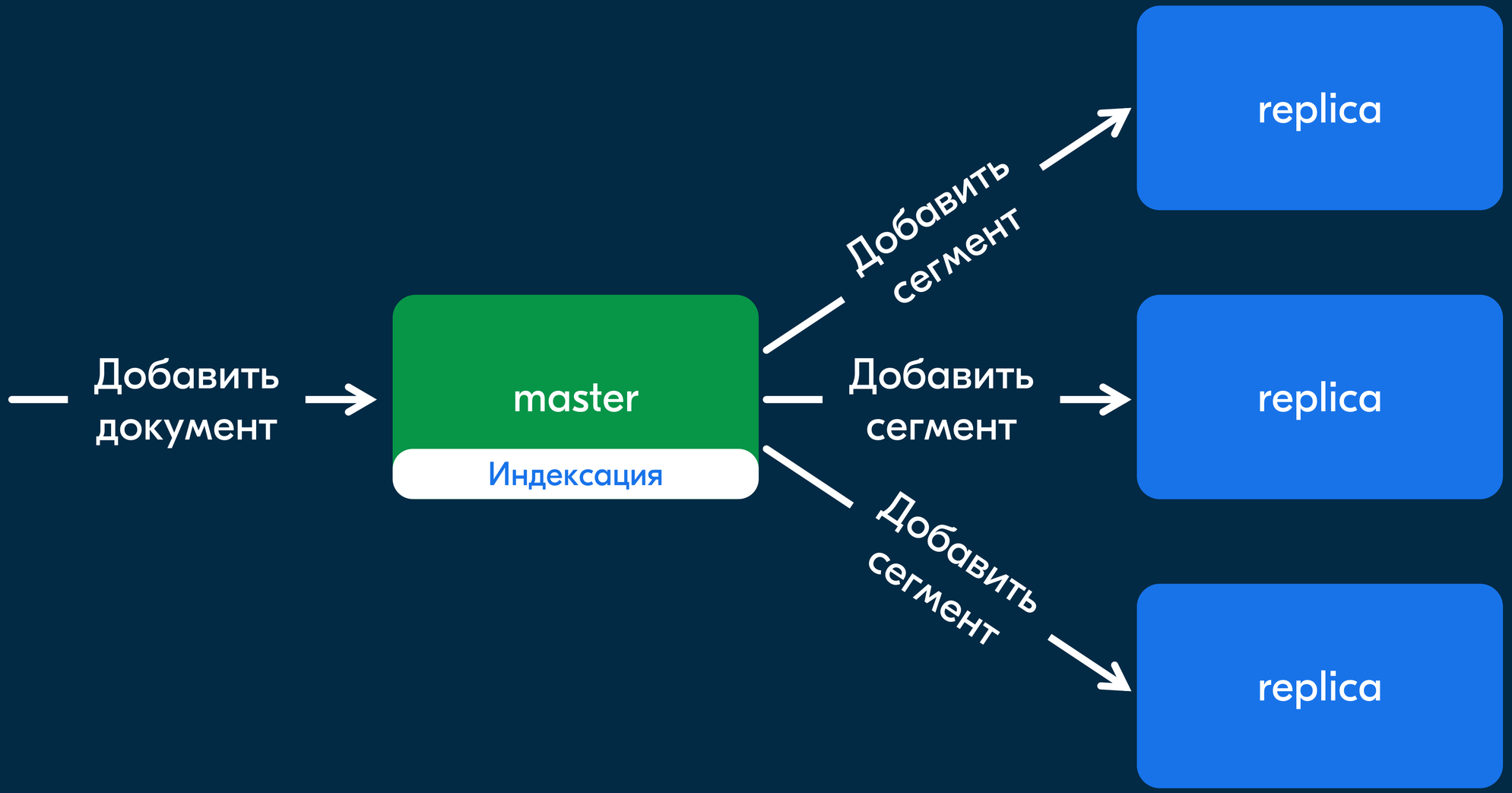
ozon tech

Физическая репликация



Почему не ElasticSearch?

Физическая репликация



Почему не ElasticSearch?

ozon{ech

Физическая репликация

Добавление готового сегмента
дешевле на порядки, чем индексация

Почему не ElasticSearch?

Физическая репликация

- Масштабируется на сотни поисковых реплик
- У нас **600+** реплик на данный момент

Почему не ElasticSearch?

ozon tech

Полный контроль над процессом построения индекса

- Response time во многом зависит от геометрии индекса
- Геометрия индекса — это размер и количество сегментов
- Управление геометрией индекса делается через merge policy
- В elastic нет возможности написать свою merge policy

Почему не ElasticSearch?

ozon tech

Полный контроль над процессом построения индекса

- Response time во многом зависит от геометрии индекса
- Геометрия индекса — это размер и количество сегментов
- Управление геометрией индекса делается через merge policy
- В elastic нет возможности написать свою merge policy

Почему не ElasticSearch?

ozon tech

Полный контроль над процессом построения индекса

- Response time во многом зависит от геометрии индекса
- Геометрия индекса — это размер и количество сегментов
- Управление геометрией индекса делается через merge policy
- В elastic нет возможности написать свою merge policy

Почему не ElasticSearch?

ozon{ech

Полный контроль над процессом построения индекса

- Response time во многом зависит от геометрии индекса
- Геометрия индекса — это размер и количество сегментов
- Управление геометрией индекса делается через merge policy
- В elastic нет возможности написать свою merge policy

Почему не ElasticSearch?

ozon{ech

Размеры сегментов (плохо )

Индекс с 6 файлами (сегментами)

Почему не ElasticSearch?

ozon tech

Размеры сегментов (плохо )

Поиск документов в 3-х потоках

Почему не ElasticSearch?

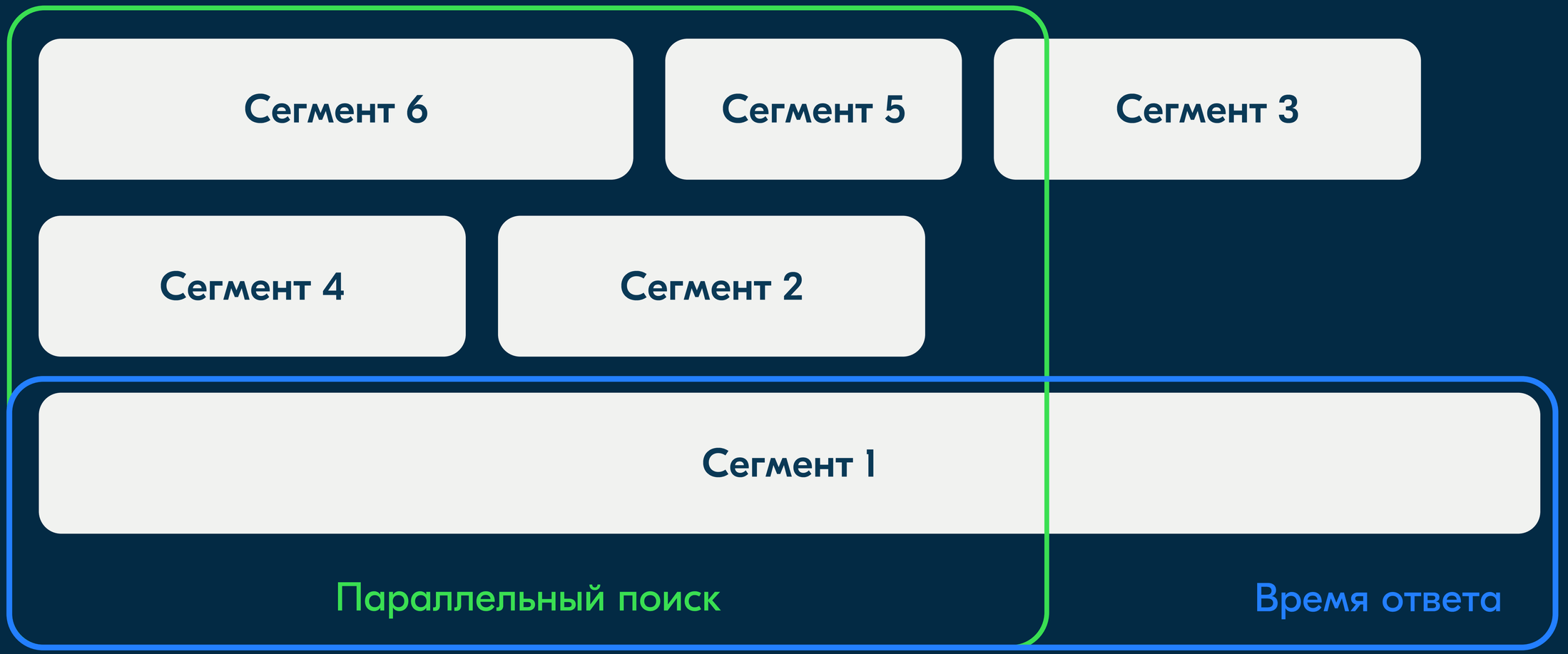
ozon{ech

Размеры сегментов (плохо )

Почему не ElasticSearch?

ozon{ech

Размеры сегментов (плохо )



Почему не ElasticSearch?

ozon{ech

Размеры сегментов (хорошо )

Почему не ElasticSearch?

ozon{ech

Эксплуатация

- ElasticSearch работал какое-то время (2017-2020 годы) [1]
- Время ответа в p95 было неприемлемо (1 секунда)
- O2 отвечает в p99: **250-300 ms**, p90: **90 ms**

[1] https://habr.com/ru/companies/ozontech/articles/667600/

Поисковая платформа Ozon

Таймлайн разработки движка O2

ozon tech

Индексация — это процесс построения специальных структур данных для поиска (inverted index, doc values, kd trees)

- На входе документы, на выходе поисковый индекс

O2 Master — это сервис, который отвечает за управление индексами и непосредственно индексацию

Архитектура O2 Master

Наша цель

- Уменьшить время построения индекса за счет утилизации всех доступных ядер и серверов
- Перестраивать индекс полностью не менее чем раз в сутки

Требования к сервису индексации

ozon{ech

- Вертикальное масштабирование
- Горизонтальное масштабирование
- Отказоустойчивость (отказ ноды, отказ ЦОД)
- Throughput важнее latency

Ключевая метрика

Количество проиндексированных документов в секунду

Или коротко:
скорость индексации

Индексация в Apache Lucene

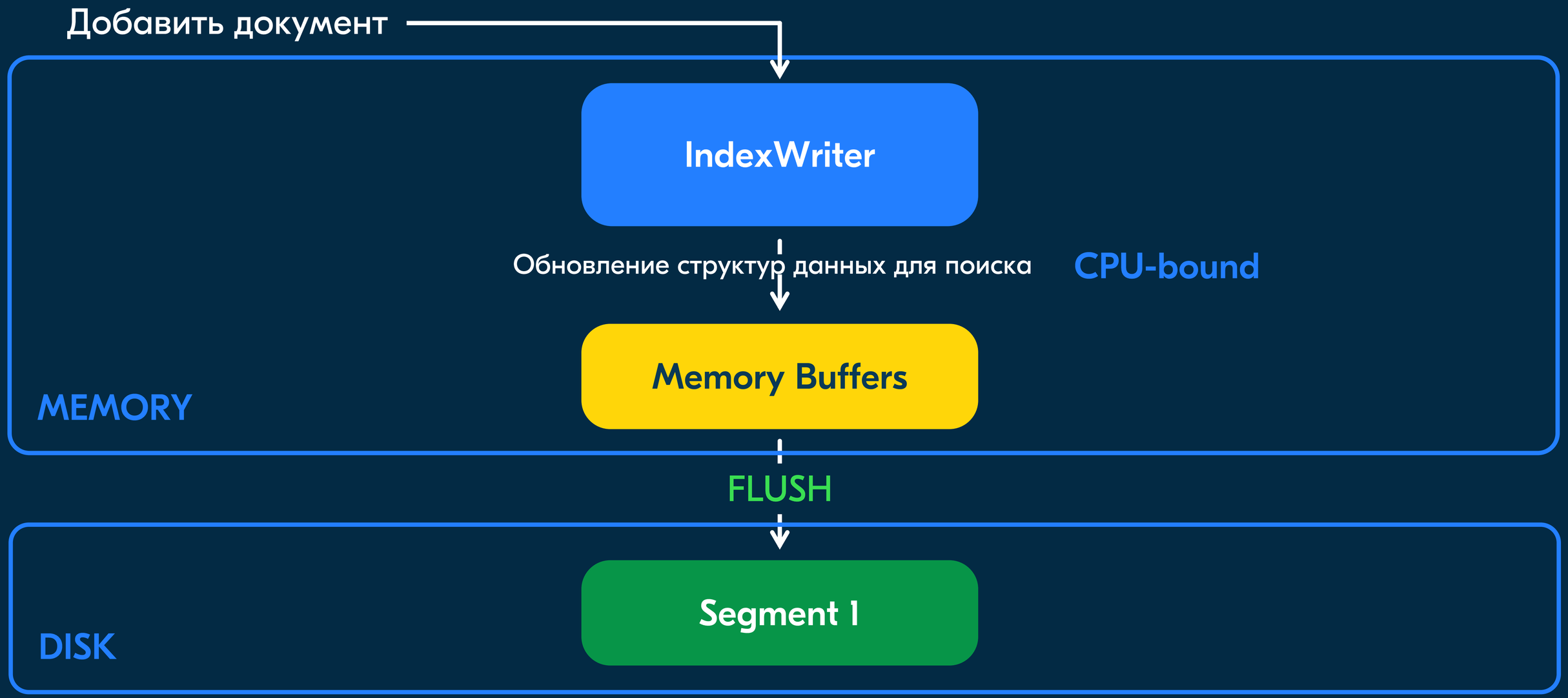
Архитектура индексации в Apache Lucene

Архитектура индексации в Apache Lucene

ozon tech

Архитектура индексации в Apache Lucene

ozon tech



Индексация

Ресурсы

- CPU-bound-задача

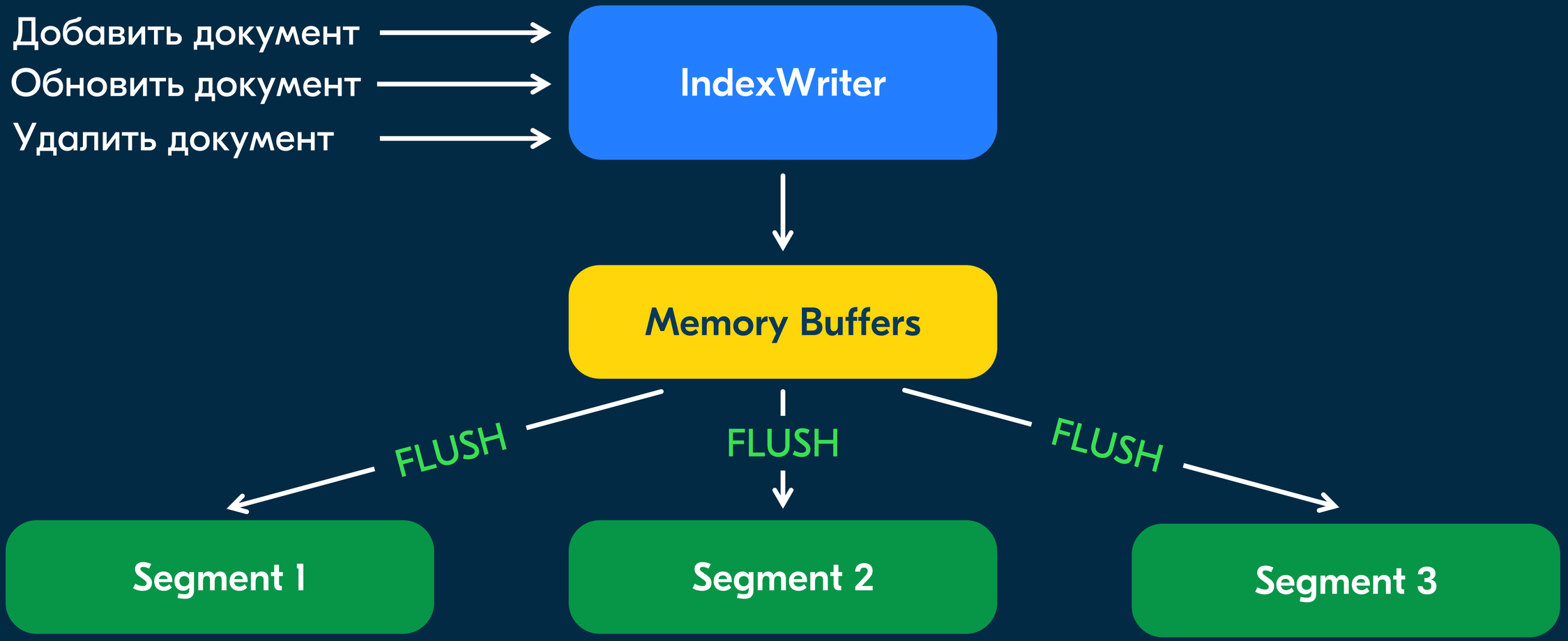
Индексация

Ресурсы

- CPU-bound:
  - Продвинутый анализ текста
  - Сложные структуры данных (FST, kd-trees, knn-vectors)
  - Много техник компрессии данных (doc values): delta encoding, gcd, vints, deduplication
  - Массивно параллельная обработка в памяти

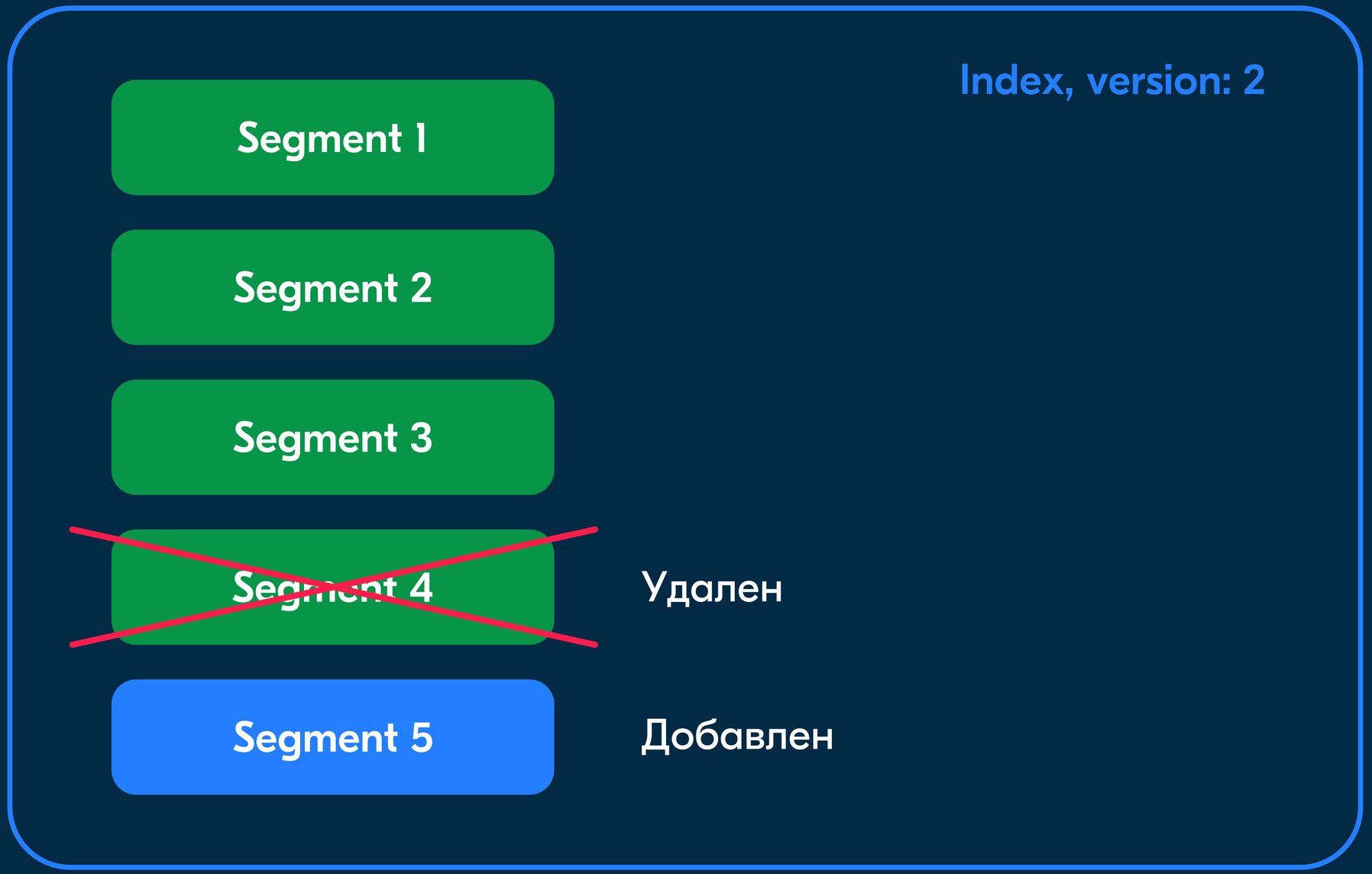
Архитектура индексации в Apache Lucene

ozon tech



Поисковый индекс

Поисковый индекс



А если сегментов
будет **слишком много**?

ozon{ech

Слишком много сегментов

- Проблемы с местом на диске
- Не можем искать за приемлемое время

Архитектура индексации в Apache Lucene

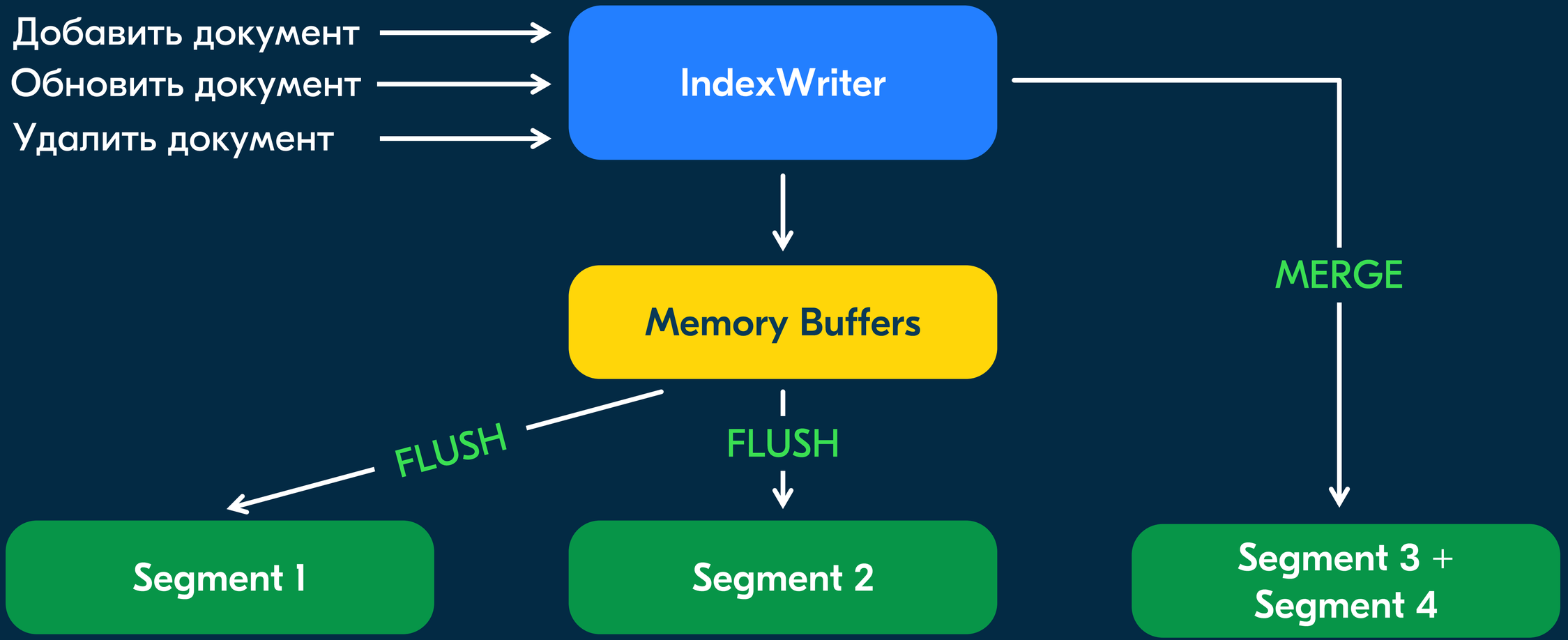
ozon tech

Слишком много сегментов

- Нужны операции merge и compaction как в LSM-tree

Архитектура индексации в Apache Lucene

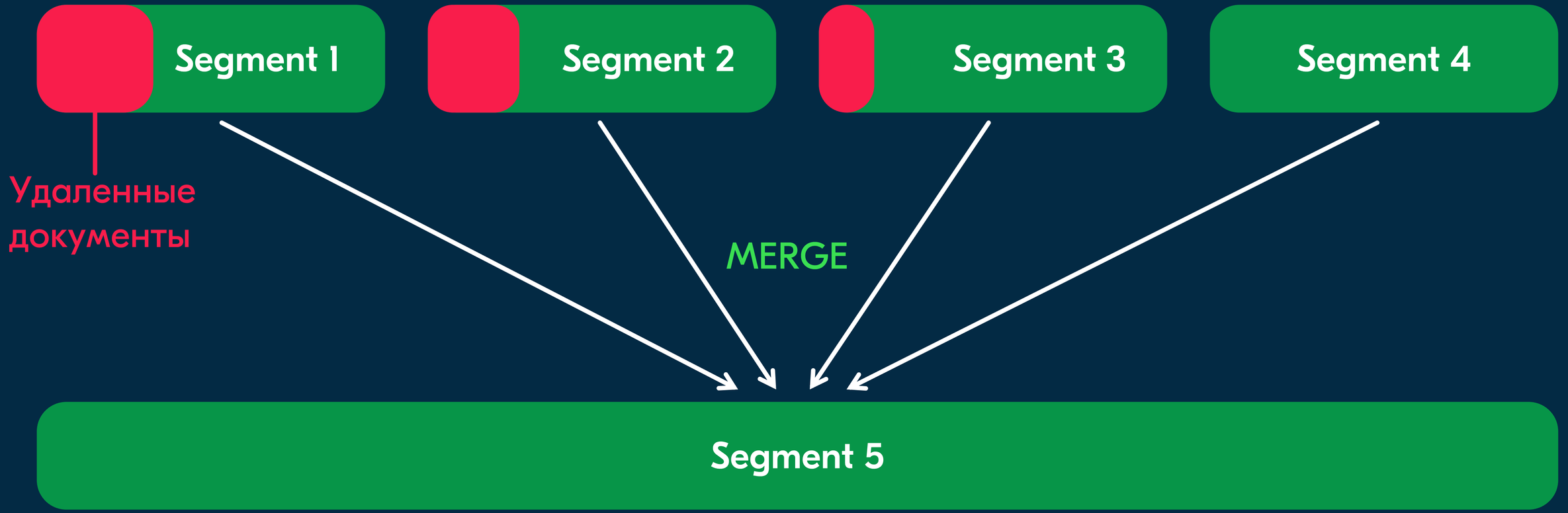
ozon{ech



Архитектура индексации в Apache Lucene

ozon{ech

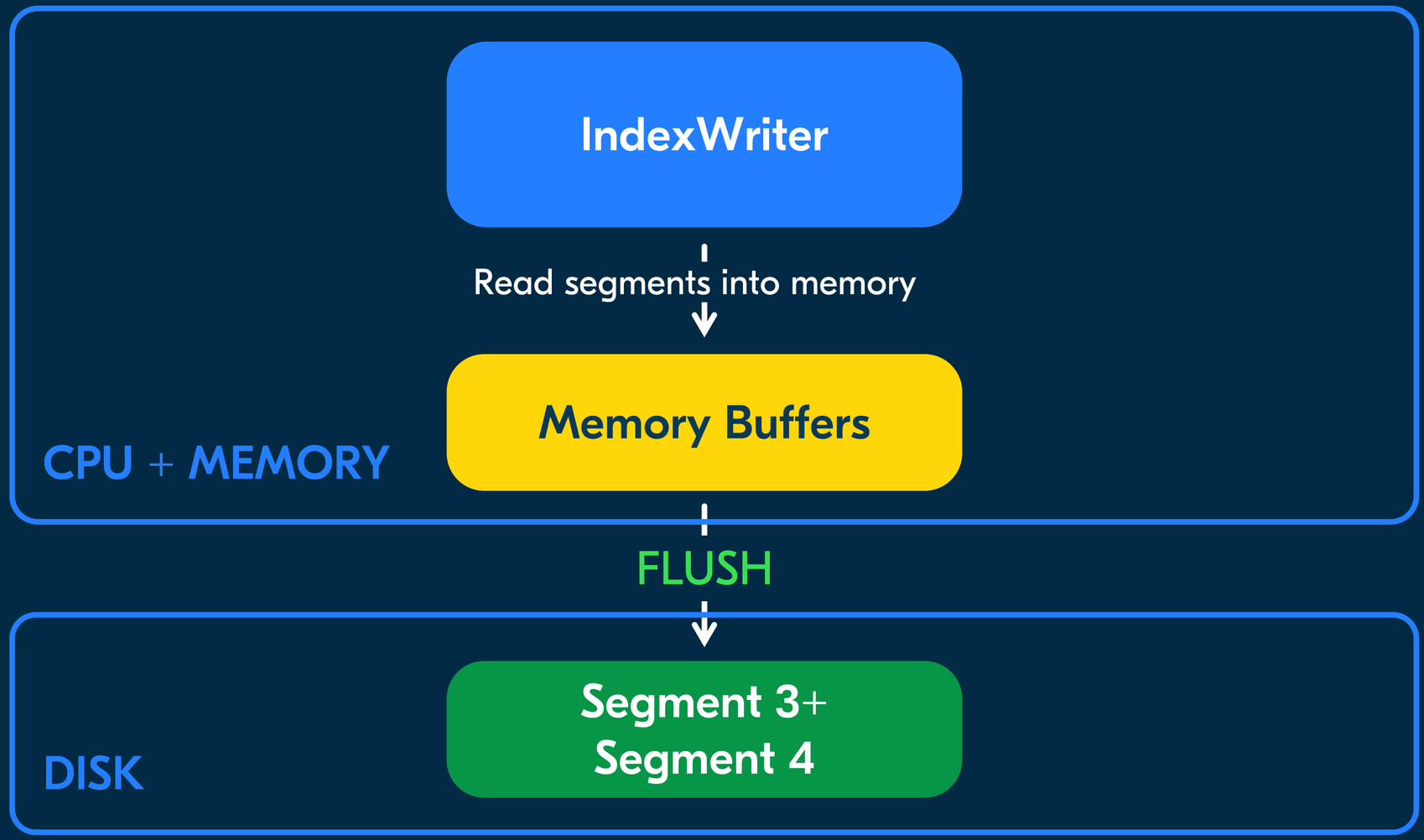
Операция merge



Индексация

Операция merge

Merge использует CPU, диск и память



Индексация

Ресурсы

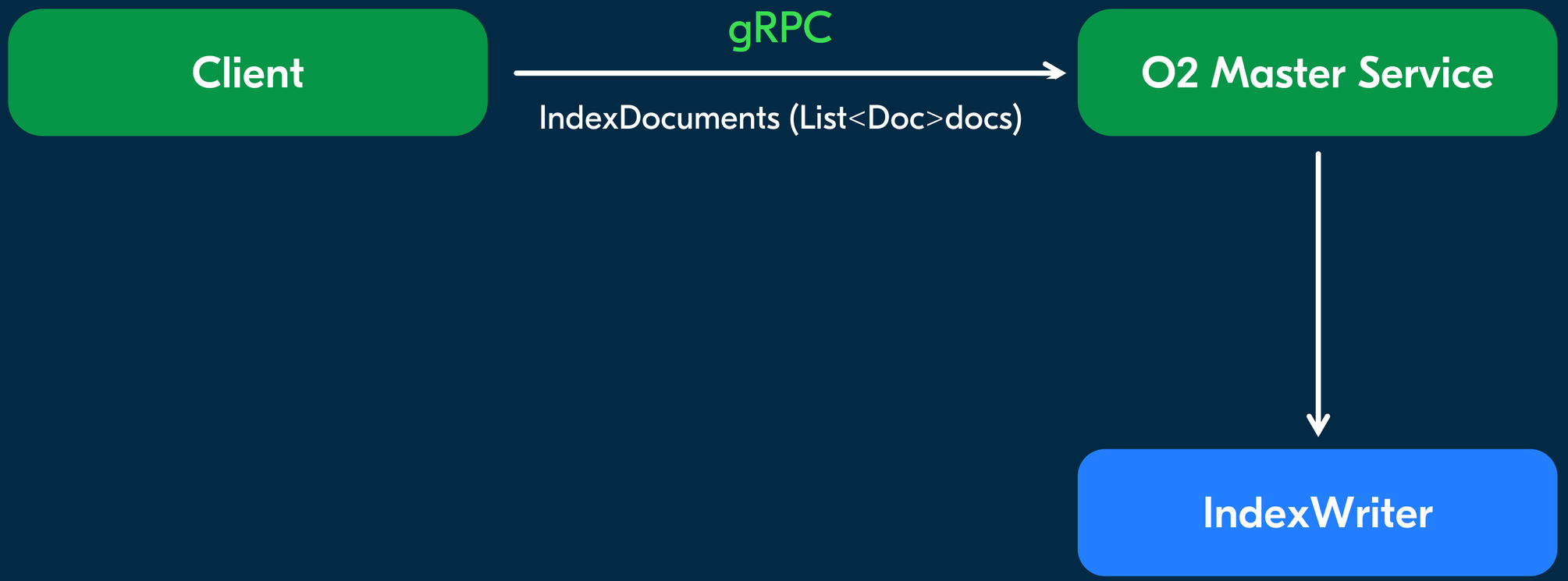
- CPU-bound-задача (INDEX, MERGE)
- Вторичные ресурсы: память и диск (MERGE, FLUSH)

Вертикальное масштабирование индексации

Архитектура кластера O2 Master (2021)

ozon{ech

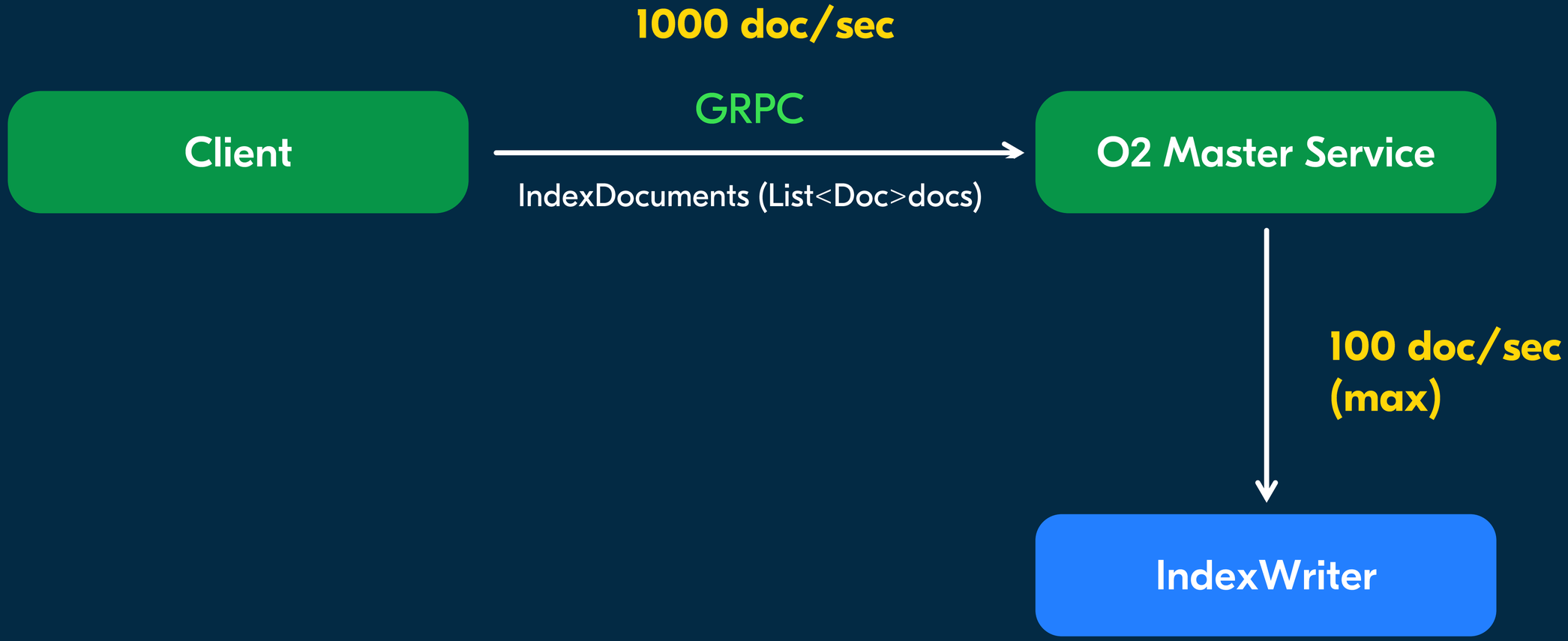
Первая версия архитектуры



Архитектура кластера O2 Master (2021)

ozon{ech

Первая версия архитектуры

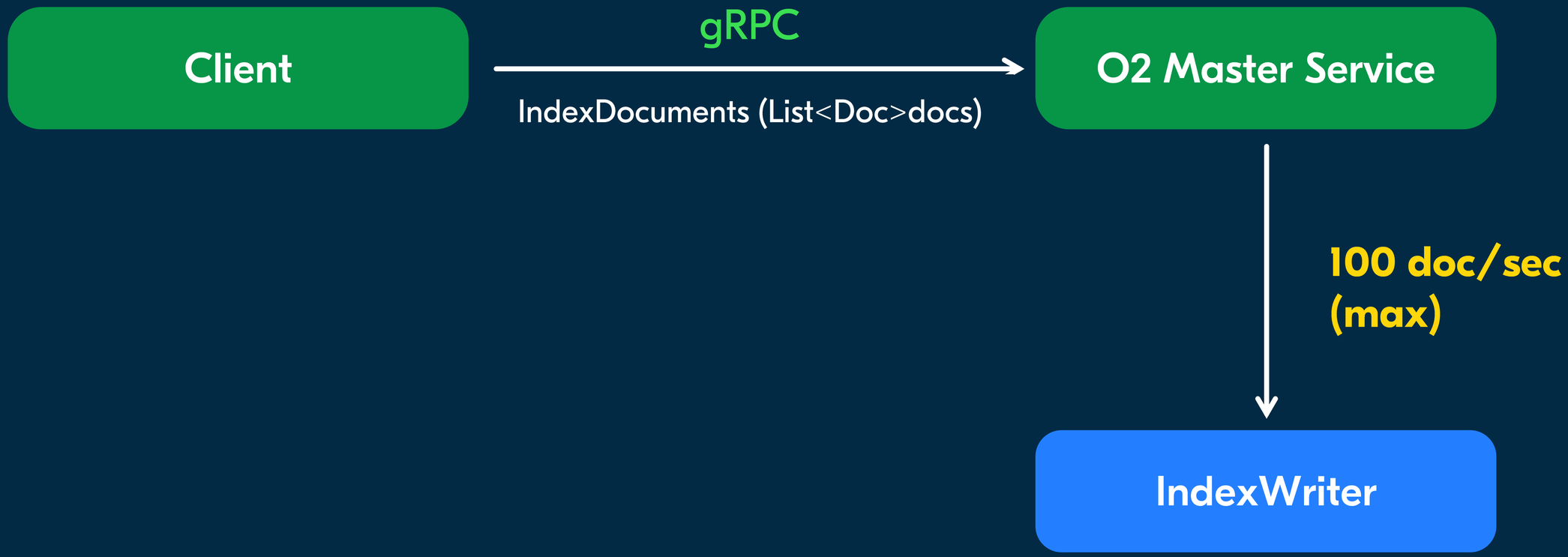


Архитектура кластера O2 Master (2021)

ozon{tech

Первая версия архитектуры

* Память может закончиться
* Сервер перегружен



Архитектура кластера O2 Master (2021)

ozon tech

Первая версия архитектуры

Скорость индексации
не превышала **1000 doc/sec**

Архитектура кластера O2 Master (2021)

ozon tech

Вторая версия архитектуры

Архитектура кластера O2 Master (2021)

ozon{ech

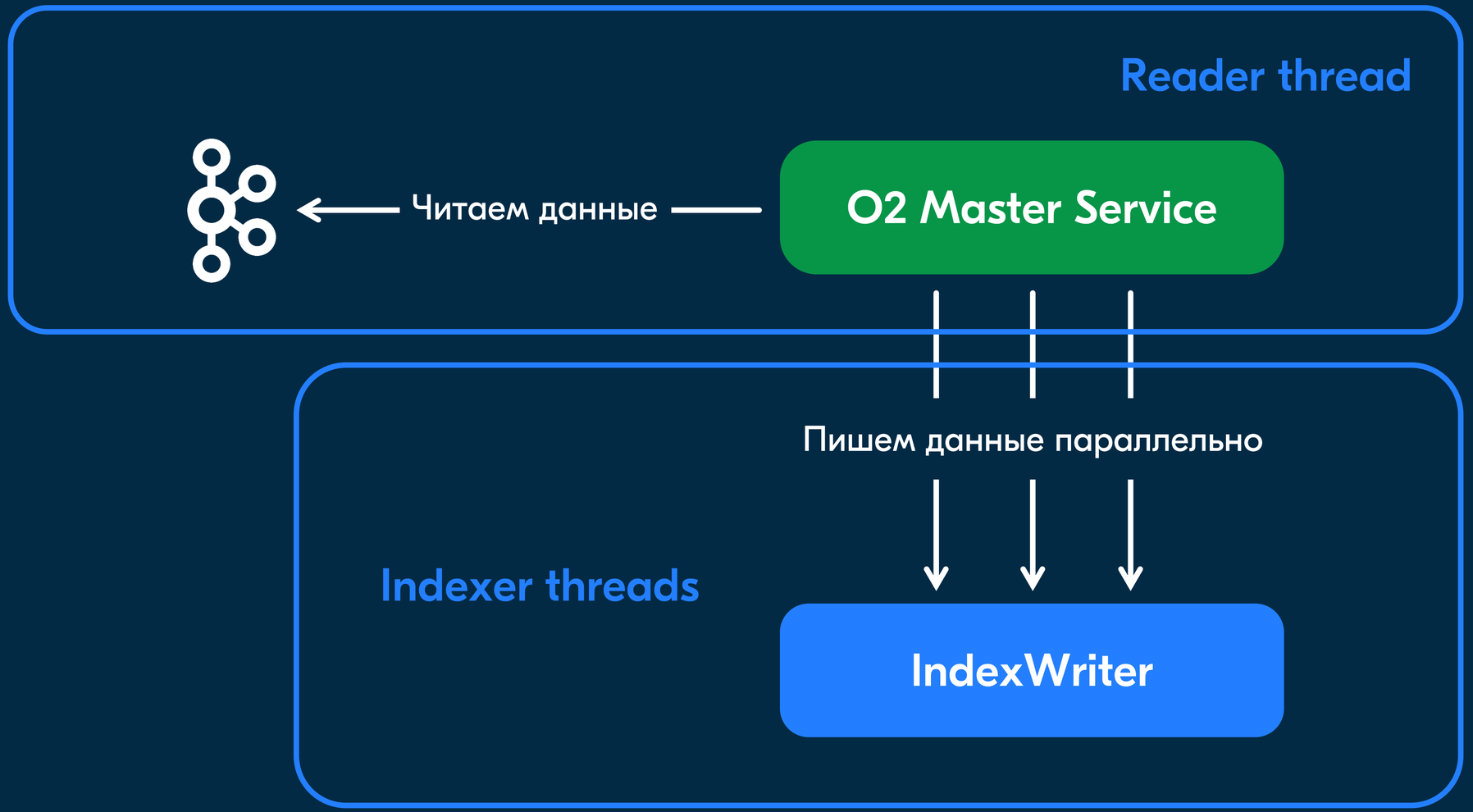
Вторая версия архитектуры

- Во второй версии заменили API-вставки на очередь (Kafka)
- Стало лучше:
  - Клиент и сервер больше не связаны напрямую
  - Скорость индексации выросла до 1500 doc/sec, и мы видели дальнейший потенциал

Архитектура кластера O2 Master (2021)

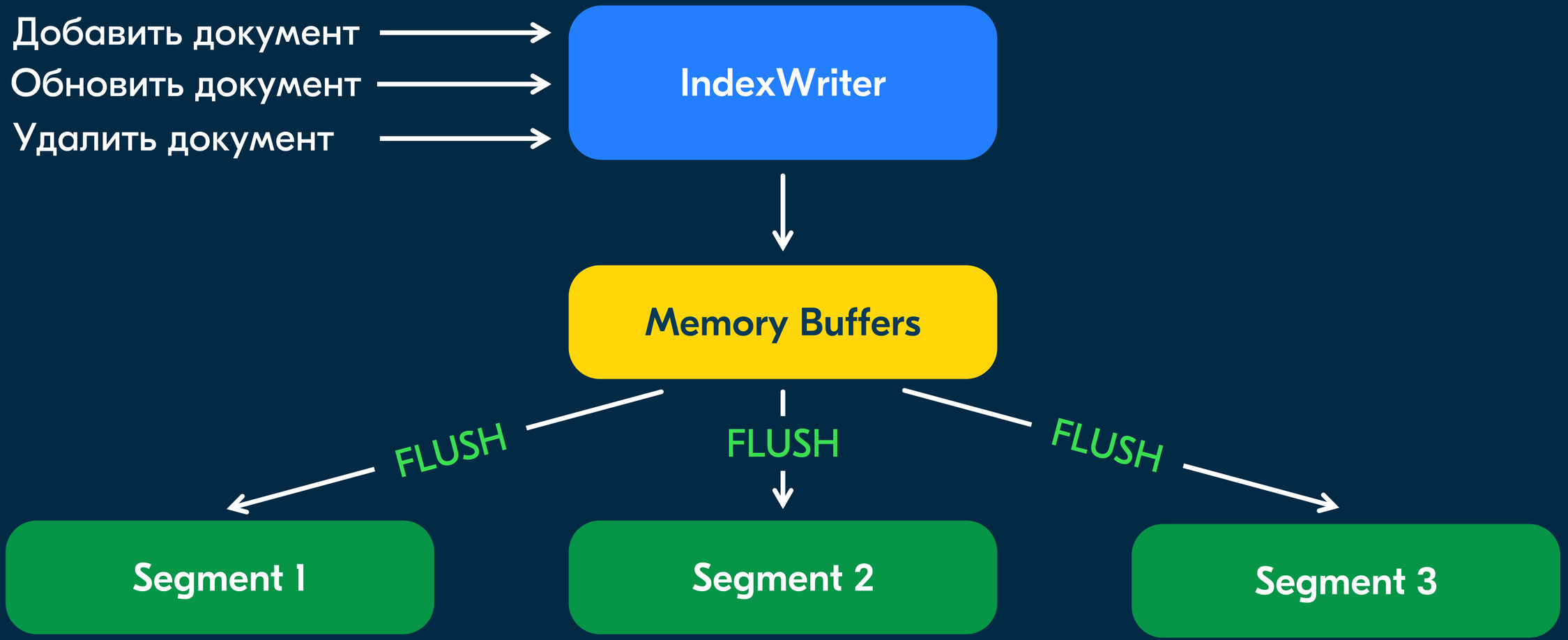
ozon tech

Вертикальное масштабирование индексации



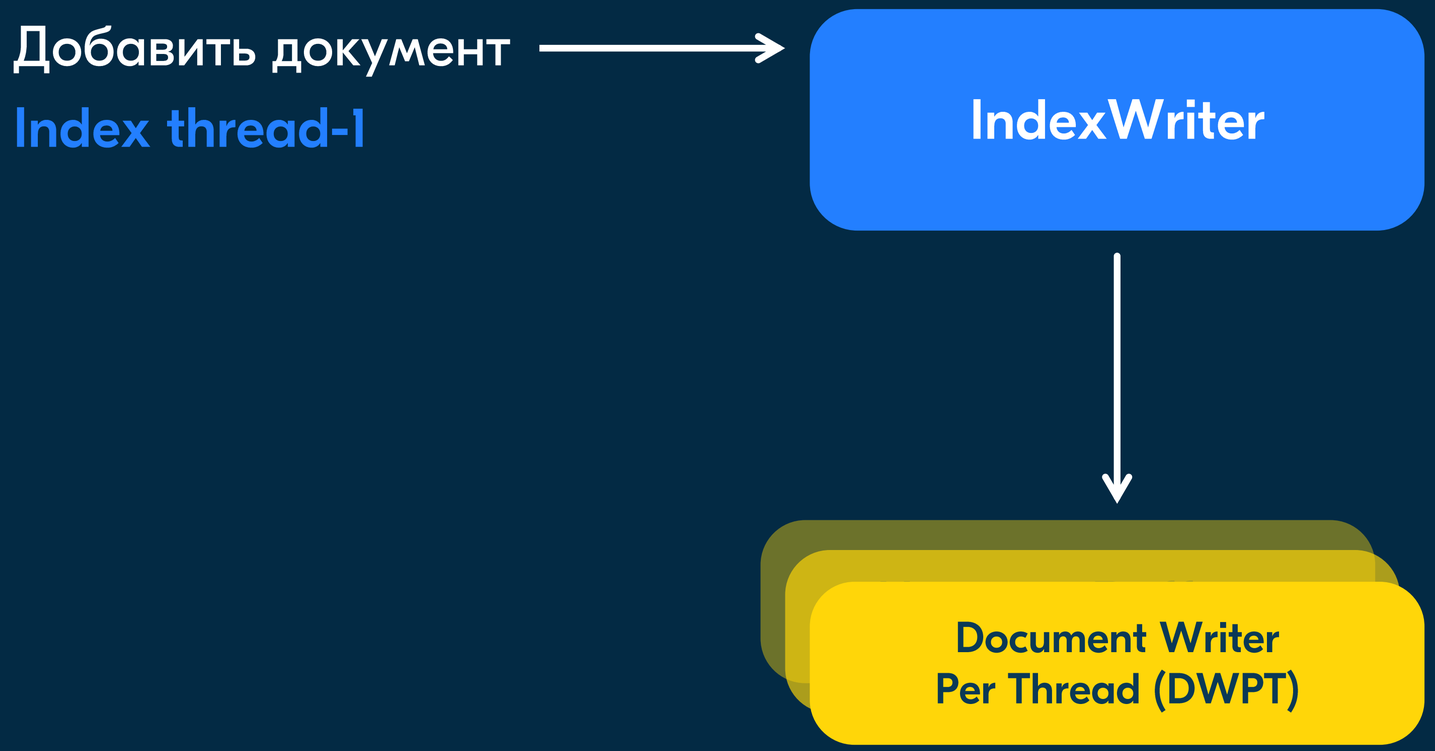
Архитектура индексации в Apache Lucene

ozon tech



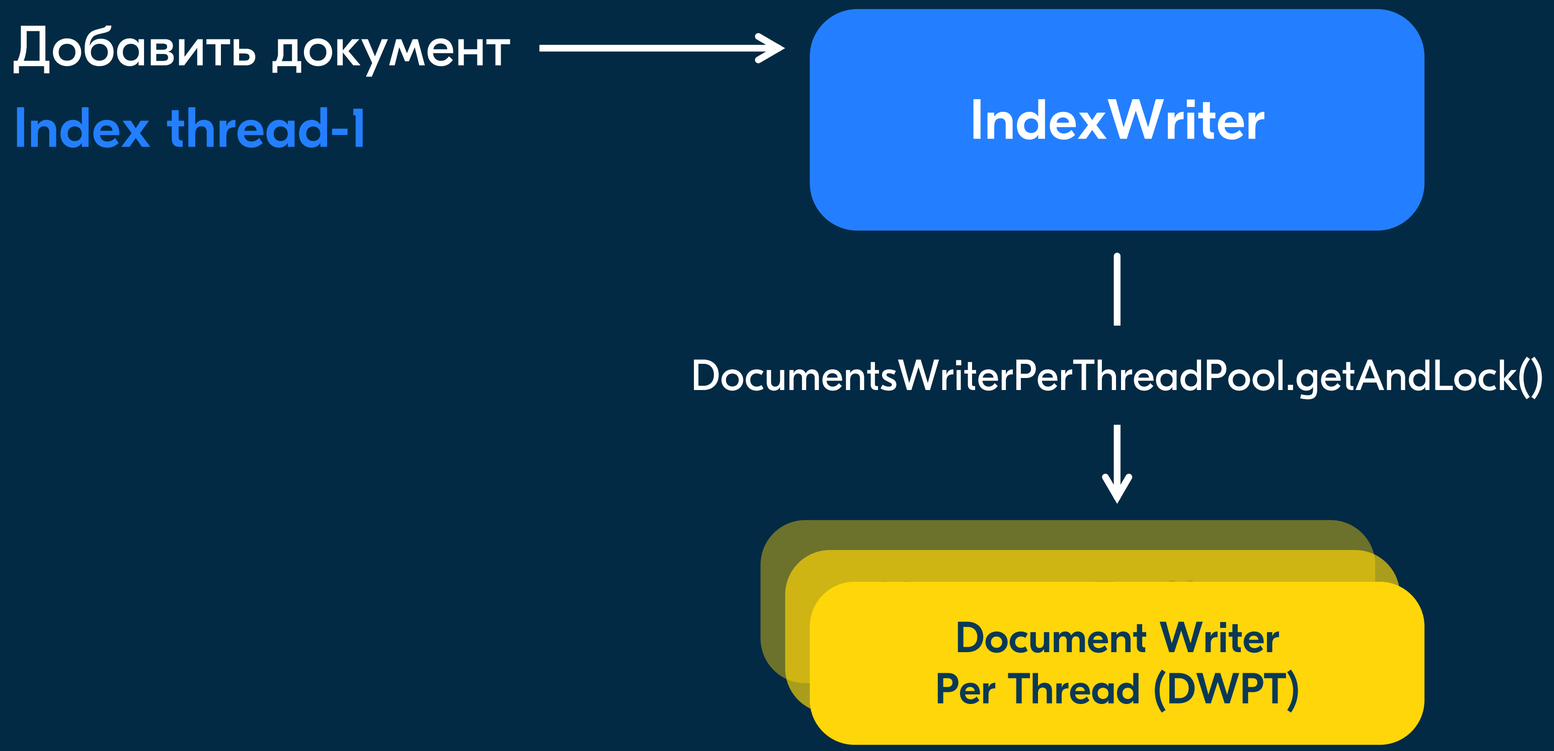
Архитектура индексации в Apache Lucene

ozon{ech



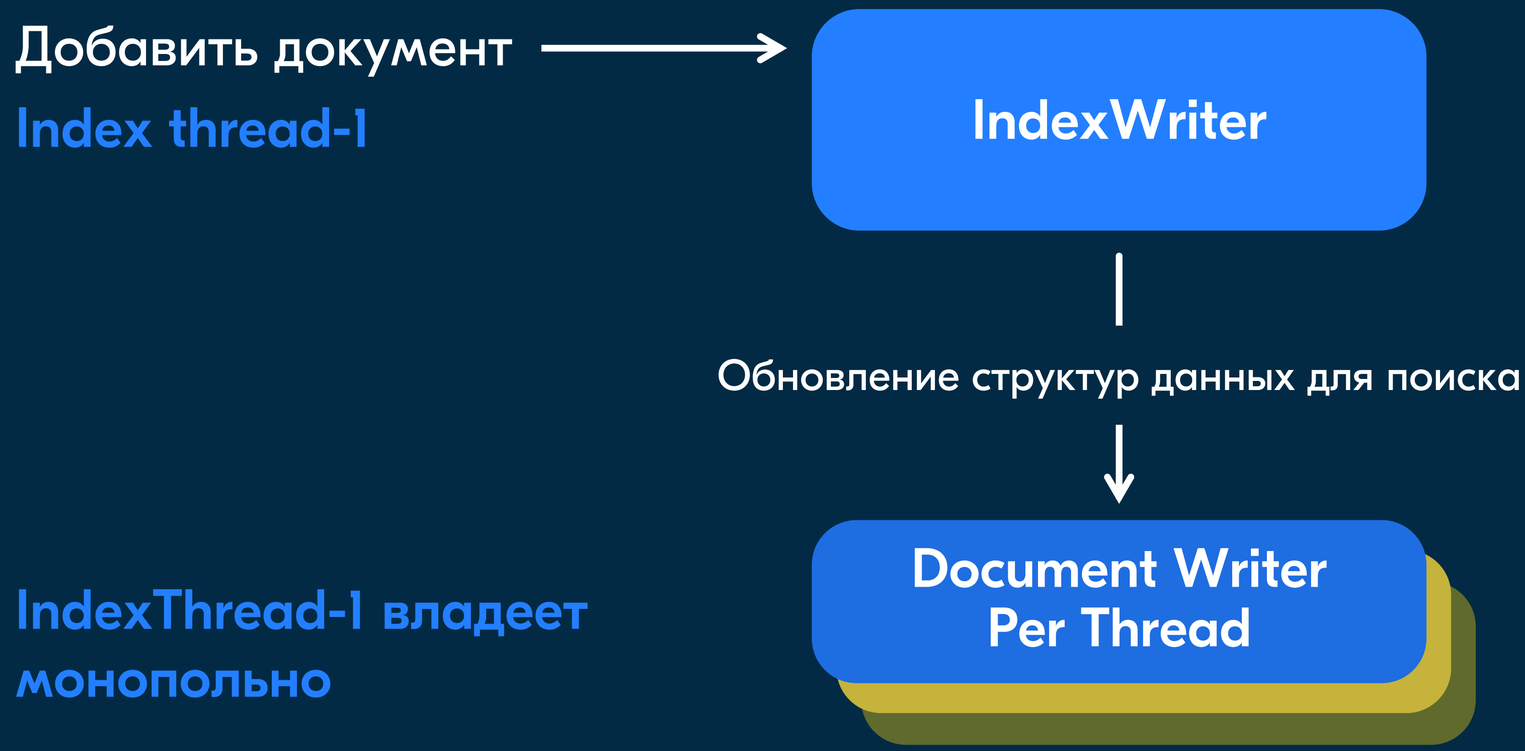
Архитектура индексации в Apache Lucene

ozon{ech



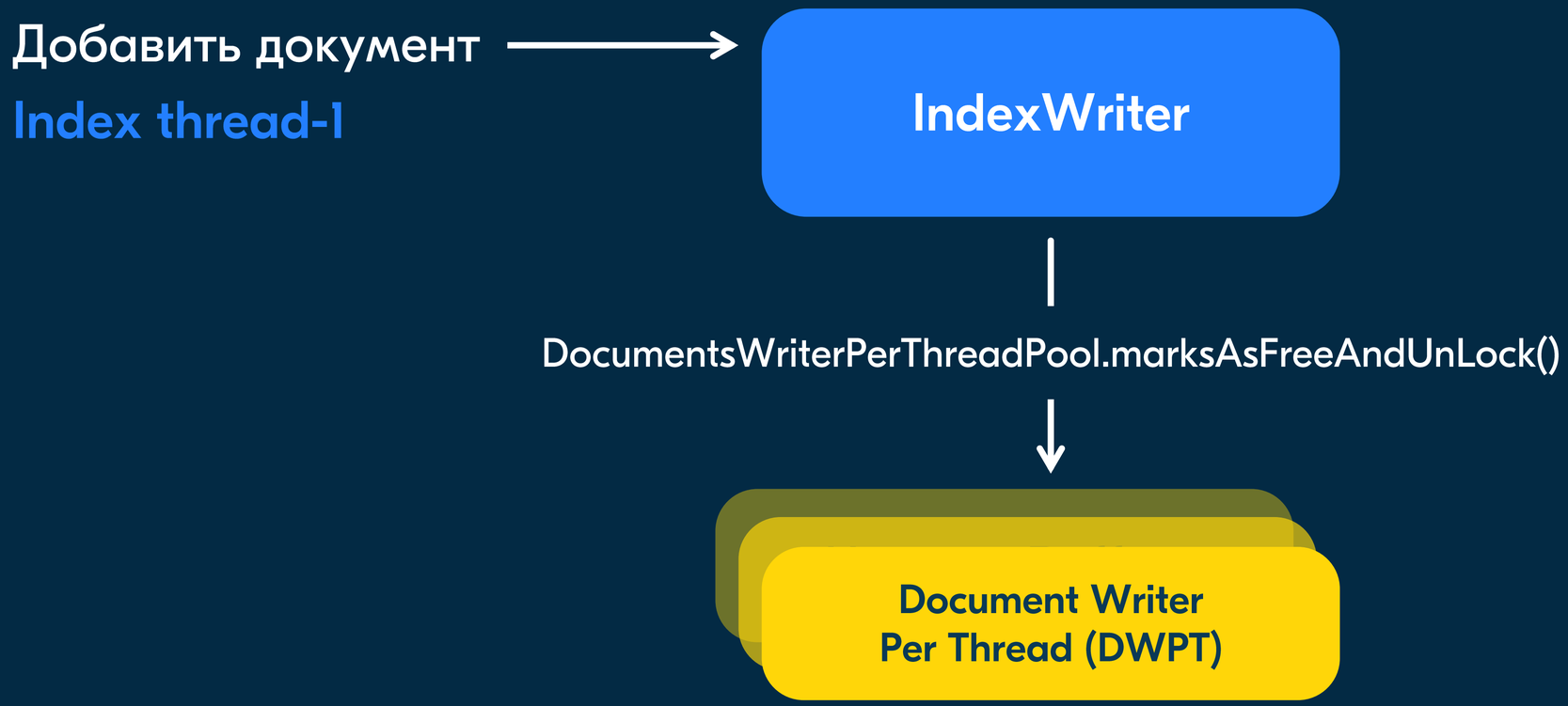
Архитектура индексации в Apache Lucene

ozon{ech



Архитектура индексации в Apache Lucene

ozon tech



Архитектура кластера O2 Master (2021)

ozon tech

Вертикальное масштабирование индексации

- Итоги:
  - Создаем пул потоков, равный количеству ядер
  - Получаем линейный рост целевой метрики?

Не все так просто

Реализация O2 Master

- Добавили ядер/потоков, а **throughput не растет** 😕

Реализация O2 Master

ozon tech

Давайте посмотрим в профайлер...

[6] java.util.concurrent.locks.ReentrantLock.lock
[7] org.apache.lucene.index.DocumentsWriterFlushControl.markForFullFlush
[8] org.apache.lucene.index.DocumentsWriter.flushAllThreads
[9] org.apache.lucene.index.IndexWriter.getReader
[10] org.apache.lucene.index.StandardDirectoryReader.doOpenFromWriter
[11] org.apache.lucene.index.StandardDirectoryReader.doOpenIfChanged
[12] org.apache.lucene.index.StandardDirectoryReader.doOpenIfChanged
[13] org.apache.lucene.index.DirectoryReader.openIfChanged
[14] org.apache.lucene.search.SearcherManager.refreshIfNeeded
[15] org.apache.lucene.search.SearcherManager.refreshIfNeeded
[16] org.apache.lucene.search.ReferenceManager.doMaybeRefresh
[17] org.apache.lucene.search.ReferenceManager.maybeRefreshBlocking
[18] ru.ozon.search.o2.index.controller.O2IndexWriter.refreshAndAcquireSearcher
[19] ru.ozon.search.o2.index.controller.O2IndexWriter.search
[20] ru.ozon.search.o2.index.controller.O2IndexWriter.documentsCount

Реализация O2 Master

ozon tech

Понятнее не стало 😕

[6] java.util.concurrent.locks.ReentrantLock.lock
[7] org.apache.lucene.index.DocumentsWriterFlushControl.markForFullFlush
[8] org.apache.lucene.index.DocumentsWriter.flushAllThreads
[9] org.apache.lucene.index.IndexWriter.getReader
[10] org.apache.lucene.index.StandardDirectoryReader.doOpenFromWriter
[11] org.apache.lucene.index.StandardDirectoryReader.doOpenIfChanged
[12] org.apache.lucene.index.StandardDirectoryReader.doOpenIfChanged
[13] org.apache.lucene.index.DirectoryReader.openIfChanged
[14] org.apache.lucene.search.SearcherManager.refreshIfNeeded
[15] org.apache.lucene.search.SearcherManager.refreshIfNeeded
[16] org.apache.lucene.search.ReferenceManager.doMaybeRefresh
[17] org.apache.lucene.search.ReferenceManager.maybeRefreshBlocking
[18] ru.ozon.search.o2.index.controller.O2IndexWriter.refreshAndAcquireSearcher
[19] ru.ozon.search.o2.index.controller.O2IndexWriter.search
[20] ru.ozon.search.o2.index.controller.O2IndexWriter.documentsCount

Реализация O2 Master

ozon tech

Что странно?

[6] java.util.concurrent.locks.ReentrantLock.lock
[7] org.apache.lucene.index.DocumentsWriterFlushControl.markForFullFlush
[8] org.apache.lucene.index.DocumentsWriter.flushAllThreads
[9] org.apache.lucene.index.IndexWriter.getReader
[10] org.apache.lucene.index.StandardDirectoryReader.doOpenFromWriter
[11] org.apache.lucene.index.StandardDirectoryReader.doOpenIfChanged
[12] org.apache.lucene.index.StandardDirectoryReader.doOpenIfChanged
[13] org.apache.lucene.index.DirectoryReader.openIfChanged
[14] org.apache.lucene.search.SearcherManager.refreshIfNeeded
[15] org.apache.lucene.search.SearcherManager.refreshIfNeeded
[16] org.apache.lucene.search.ReferenceManager.doMaybeRefresh
[17] org.apache.lucene.search.ReferenceManager.maybeRefreshBlocking
[18] ru.ozon.search.o2.index.controller.O2IndexWriter.refreshAndAcquireSearcher
[19] ru.ozon.search.o2.index.controller.O2IndexWriter.search
[20] ru.ozon.search.o2.index.controller.O2IndexWriter.documentsCount

Реализация O2 Master

ozon tech

Мы тут говорим про запись документов.
А в профайлере блокировка торчит из поиска?

[6] java.util.concurrent.locks.ReentrantLock.lock
[7] org.apache.lucene.index.DocumentsWriterFlushControl.markForFullFlush
[8] org.apache.lucene.index.DocumentsWriter.flushAllThreads
[9] org.apache.lucene.index.IndexWriter.getReader
[10] org.apache.lucene.index.StandardDirectoryReader.doOpenFromWriter
[11] org.apache.lucene.index.StandardDirectoryReader.doOpenIfChanged
[12] org.apache.lucene.index.StandardDirectoryReader.doOpenIfChanged
[13] org.apache.lucene.index.DirectoryReader.openIfChanged
[14] org.apache.lucene.search.SearcherManager.refreshIfNeeded
[15] org.apache.lucene.search.SearcherManager.refreshIfNeeded
[16] org.apache.lucene.search.ReferenceManager.doMaybeRefresh
[17] org.apache.lucene.search.ReferenceManager.maybeRefreshBlocking
[18] ru.ozon.search.o2.index.controller.O2IndexWriter.refreshAndAcquireSearcher
[19] ru.ozon.search.o2.index.controller.O2IndexWriter.search
[20] ru.ozon.search.o2.index.controller.O2IndexWriter.documentsCount

Реализация O2 Master

ozon{ech

Поиск мешает индексации

- Проблема в методе **IndexWriter.getReader**
- Метод может вызывать операцию flush all, которая сбрасывает все данные из памяти на диск
- Захватывает ту же блокировку, что и индексация (flush control)

Реализация O2 Master

Поиск мешает индексации

- Убрали все поисковые вызовы
- Скорость индексации достигла **~2500-3000 doc/sec**

Выбор GC

Выбор GC

- Какой GC выбрать?
- Может, ZGC?

Выбор GC

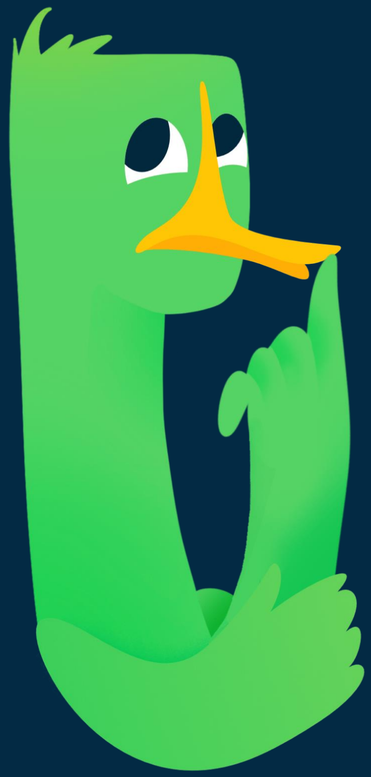
ZGC (JDK 14)

- -XX:+UnlockExperimentalVMOptions -XX:+UseZGC

Выбор GC

ZGC (JDK 14)

- И однажды...



Выбор GC

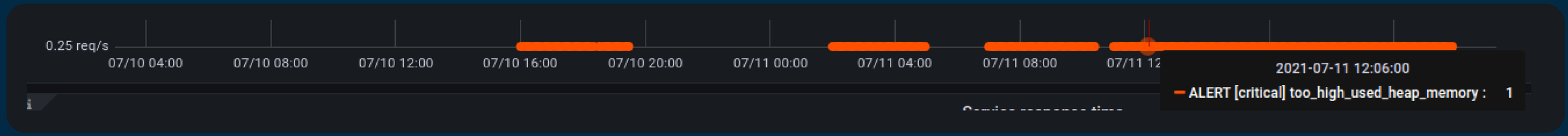
ZGC (JDK 14)

- Сервису стало совсем плохо 😕
- RT вырос **до 5 секунд** (max query timeout) в p90

Реализация O2 Master

ZGC (JDK 14)

- Сработал алерт на размер занятого хипа



Реализация O2 Master

ozon{ech

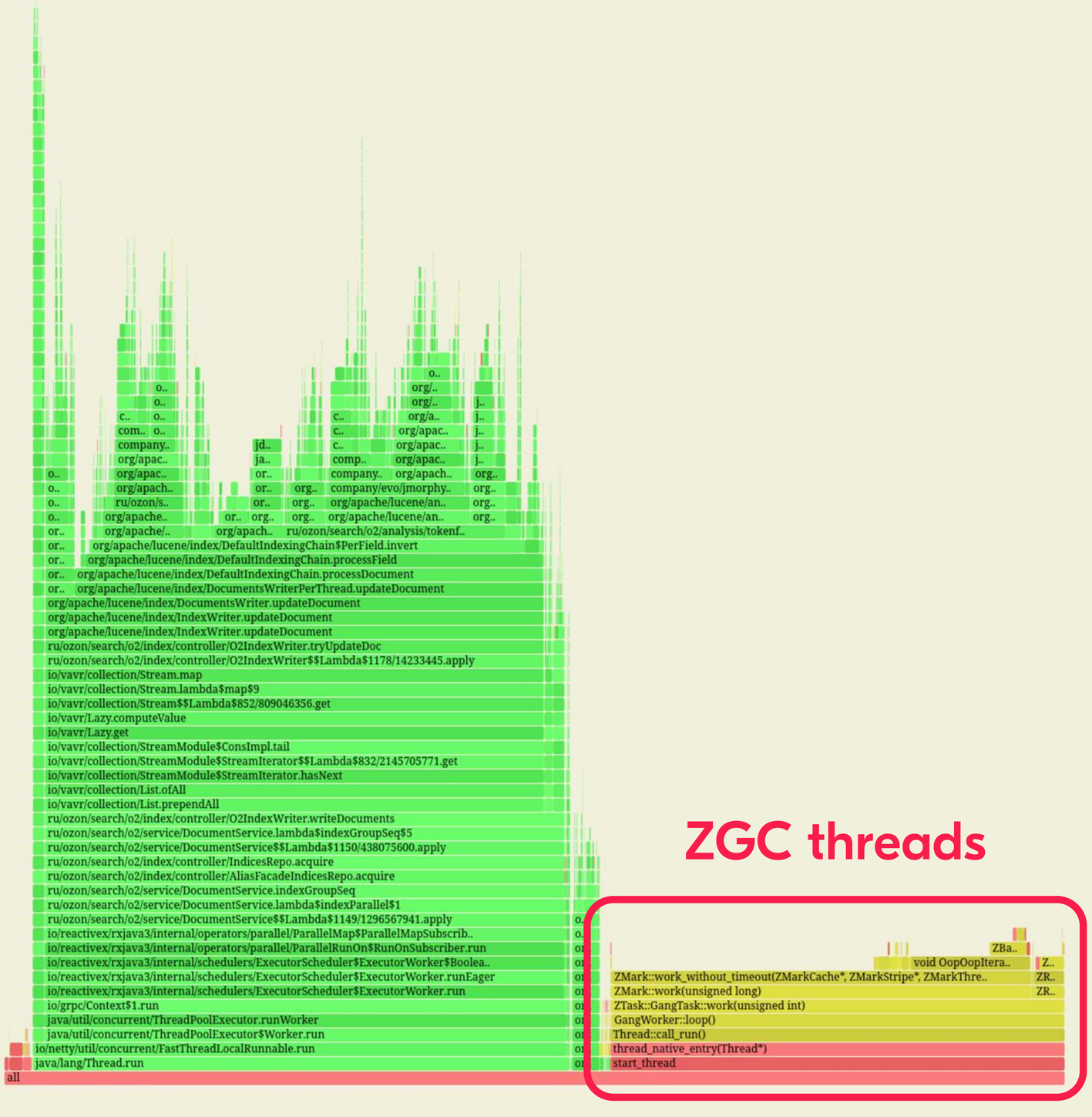
ZGC (JDK 14)

- Давайте посмотрим **в профайлер...**

Реализация O2 Master

ZGC (JDK 14)

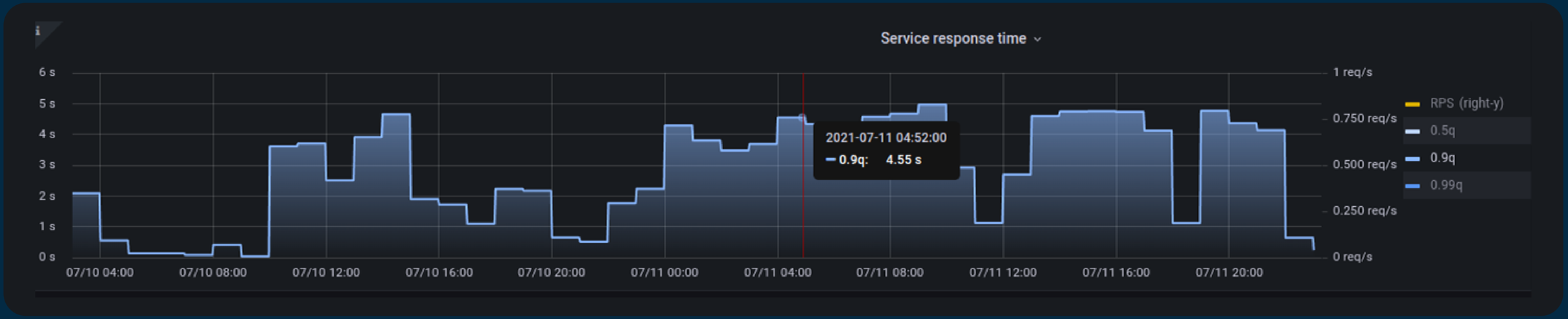
- Половину ядер занял ZGC (а мы CPU bound)



Выбор GC

ZGC (JDK 14)

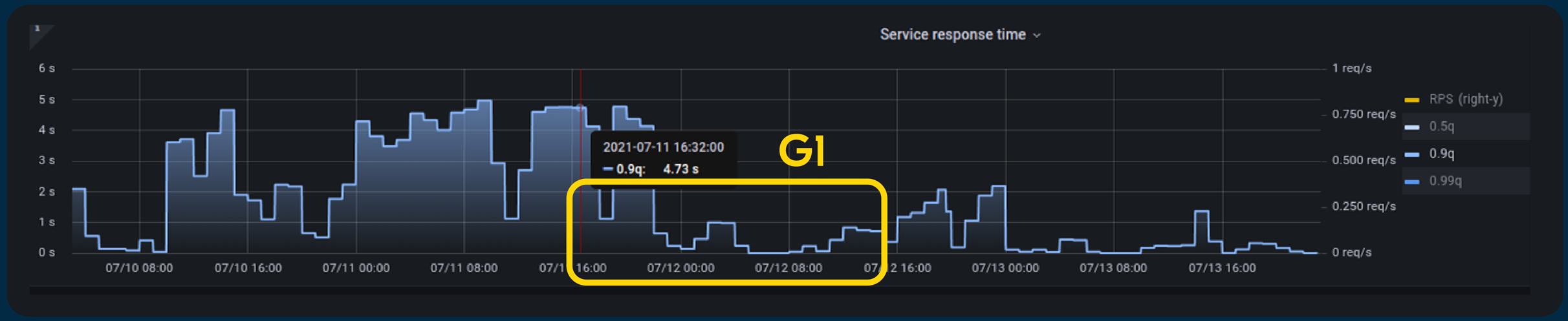
- В нашем случае ZGC занимал до 15-30% CPU 😕
- В какой-то момент GC перестал успевать очищать хип



Выбор GC

G1

- Переключились на G1



OZON tech

А вы пробовали **обновить java**?

Выбор GC

Ситуация на сегодня

Прошло два года,
а что сегодня?

Выбор GC

ozon{ech

G1 vs ZGC (JDK 17)

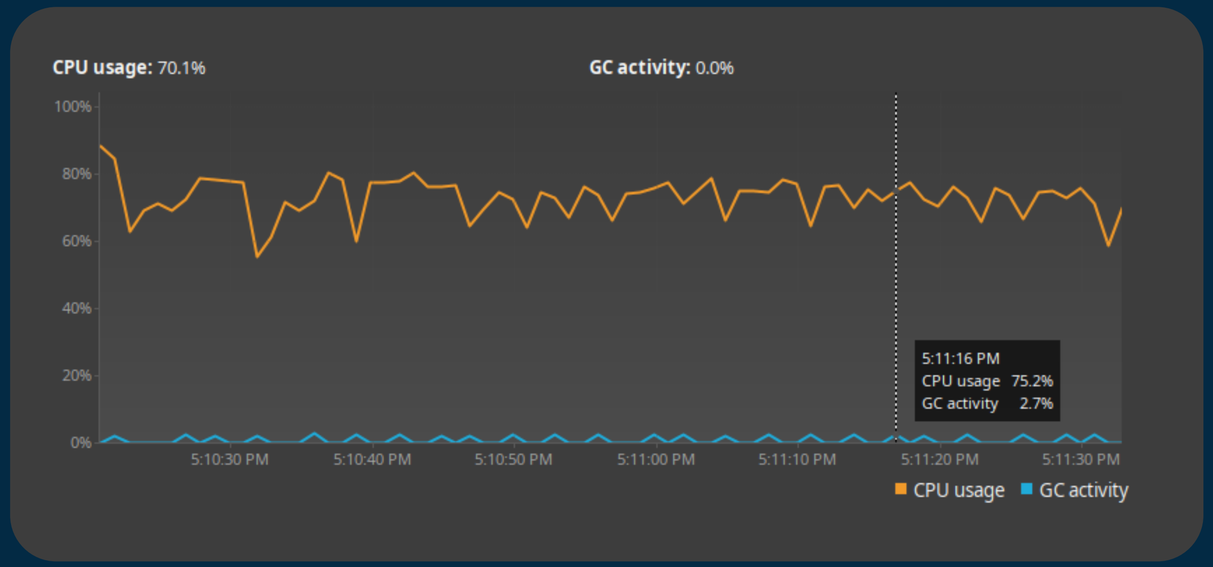
- Тест: приложение индексации документов
- Реальные данные
- 42 cores, 800 gb RAM
- Настройки GC дефолтные, -Xmx80g

Выбор GC

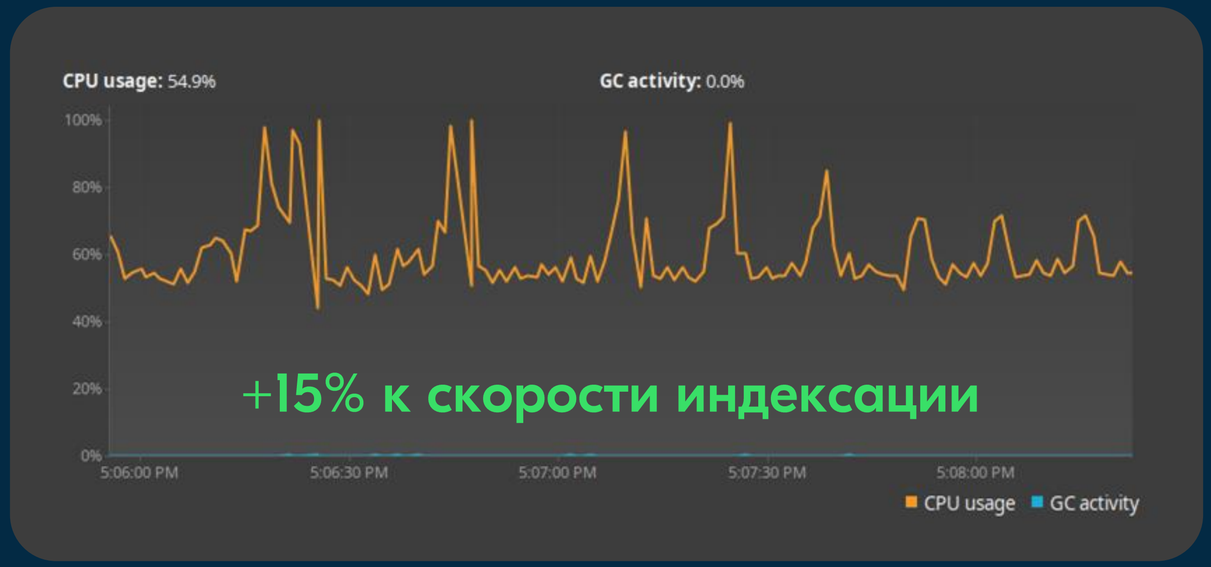
ozon tech

G1 vs ZGC (JDK 17)

- Нагрузка и данные одинаковы
- У ZGC размер пауз меньше на порядок за счет более частых сборок
- ZGC использует больше CPU



ZGC



G1

Выбор GC

ozon tech

Выводы

- В каждом приложении необходимо выбирать GC через анализ метрик

- **ZGC** — хороший выбор для приложений, где важен latency

- **G1** все еще неплох для batch-нагрузки (CPU-bound)

Потребление памяти

Реализация O2 Master

Память как ресурс

Сколько памяти нужно, чтобы загрузить все потоки индексации?

(много)

Реализация O2 Master

ozon tech

Память, как ресурс

- На каждый документ надо иметь несколько мегабайт памяти
- Memory buffers для индексации
- Память под мержи
- Другие накладные расходы

Реализация O2 Master

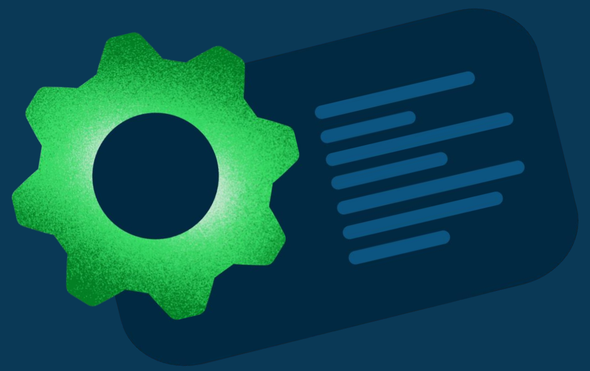
ozon tech

Представление документа внутри

- bytes
- Protobuf
- org.apache.lucene.document.Document (объектное представление) 😕
- Конечные структуры данных внутри codec

Реализация O2 Master

Представление документа внутри



В некоторых индексах у нас **500-1000 полей** в документе в среднем (!)

Реализация O2 Master

ozon{ech

Как уменьшить потребление памяти java-приложения?

- Использовать примитивные коллекции (например, **fastutil**)
- Переиспользовать объекты
- Профилировать аллокации
- Использовать off-heap

Реализация O2 Master

ozon tech

Что используем мы?

JDK/guava

(больше в индексации)

fastutil

(больше в поиске)

Реализация O2 Master

Memory-mapped I/O

- Memory-mapped I/O (mmap) для неизменяемых данных

Реализация O2 Master

ozon tech

Memory-mapped I/O

- org.apache.lucene.store.BufferedIndexInput
- one.nio.mem.MappedFile + Unsafe [1]
- Можно работать с файлами > **2gb**

[1] https://github.com/odnoklassniki/one-nio

Реализация O2 Master

ozon{ech

Project Panama

- Project Panama [1][2] (строим HNSW [3] на C++)

[1] https://openjdk.org/projects/panama/
[2] https://denisgabaydulin.medium.com/first-look-into-panama-b1e016aa1a15
[3] https://arxiv.org/abs/1603.09320

Реализация O2 Master

ozon tech

Project Panama

- Строим HNSW граф в hnswlib (C++)
- Копируем данные целиком и пишем в формате Lucene vector codec
- До 3x быстрее, чем делает сам Lucene 9.2.0

Реализация O2 Master

ozon tech

Итоги

- Выбрали GC/heap size
- Уменьшили потребление памяти
- Смогли утилизировать **около 40 ядер**
- Скорость индексации достигла **~6000 doc/sec**

Индексация на максималках

Реализация O2 Master

ozon{ech

Включили индексацию на максималках

~x2 ядер

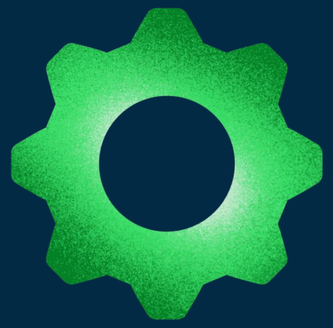
А throughput вырос ничтожно 😕

Прирост меньше 15%

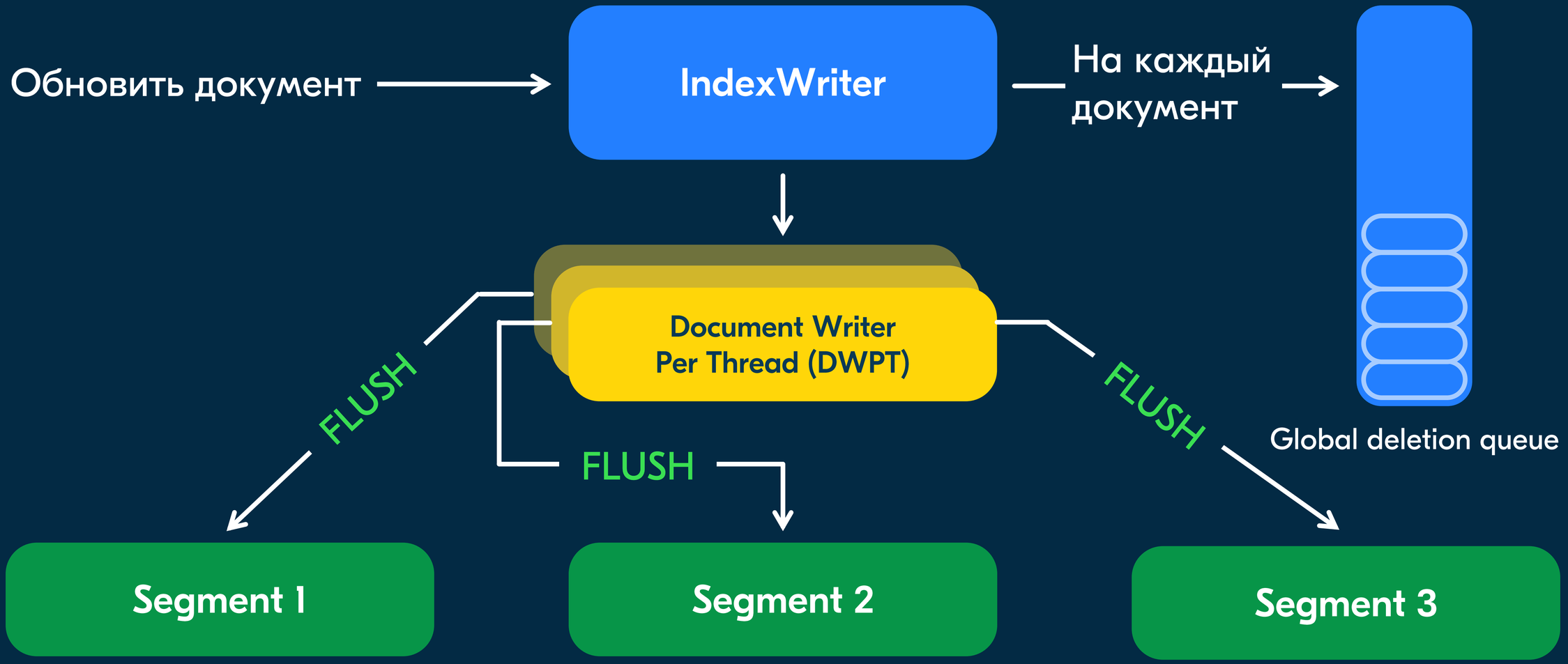
Реализация O2 Master

ozon{ech

- Исторически у нас все вставки документов были, как апдейты
- Сигнатура метода: **updateDocument(Term id, Document doc)**
- Update в Lucene: **insert** + **delete**, только один документ



Архитектура индексации в Apache Lucene

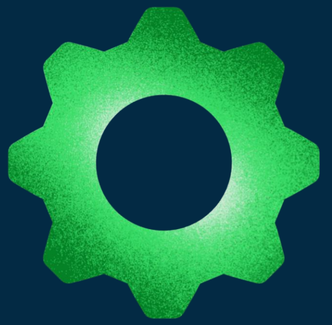


Реализация O2 Master

ozon{ech

Проблема апдейтов

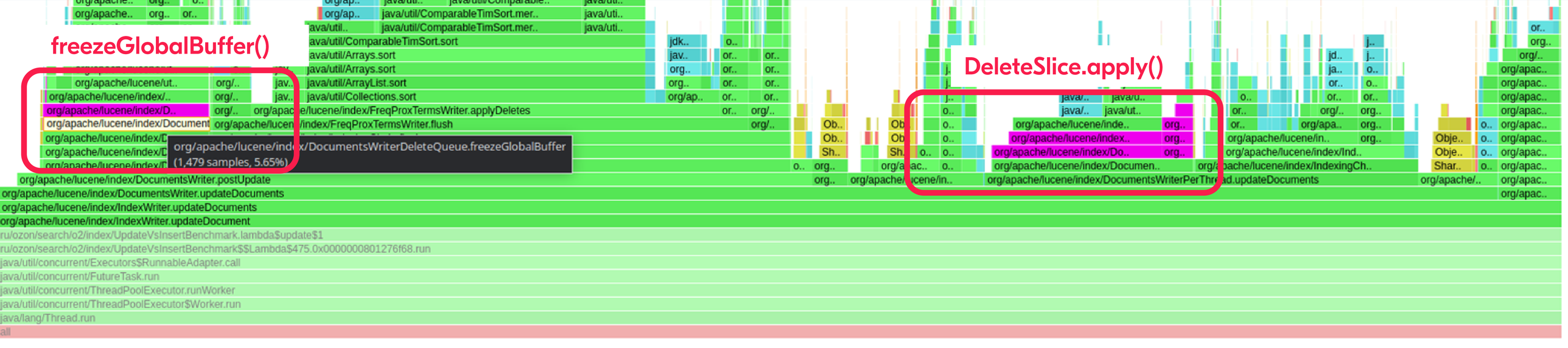
- Сделали микробенчмарк
- **addDocument(Document doc)** работает на **20-30%** быстрее даже в один поток
- profiler.sh -d 30 -e **lock** -o flamegraph=total
- profiler.sh -d 30 -e **cpu** -o flamegraph



Реализация O2 Master

Проблема апдейтов

- В случае update тратим существенную часть времени на операциях связанных с удалением документов



Реализация O2 Master

Проблема апдейтов #1

- Update имеет оверхед на каждую операцию
- Нет bulk-версии

Реализация O2 Master

ozon{ech

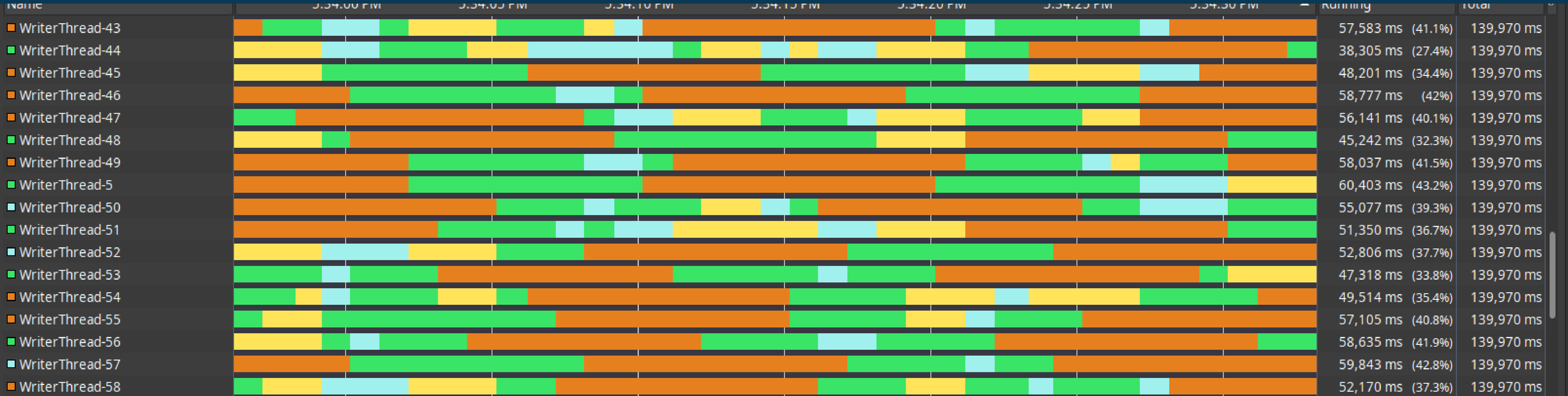
Проблема апдейтов #2

- При увеличении количества потоков flush становится узким местом в updateDocument()

Реализация O2 Master

Проблема апдейтов #2

- Блокировки внутри потоков индексации на flush, выделены голубым



Реализация O2 Master

ozon tech

Проблема апдейтов #2

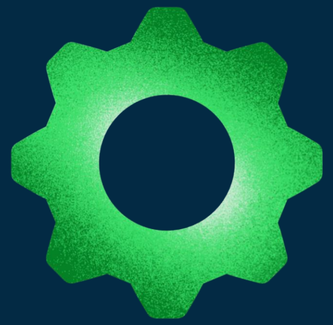
- Подробнее в отдельной статье [1]

[1] https://t.ly/EajqB

Реализация O2 Master

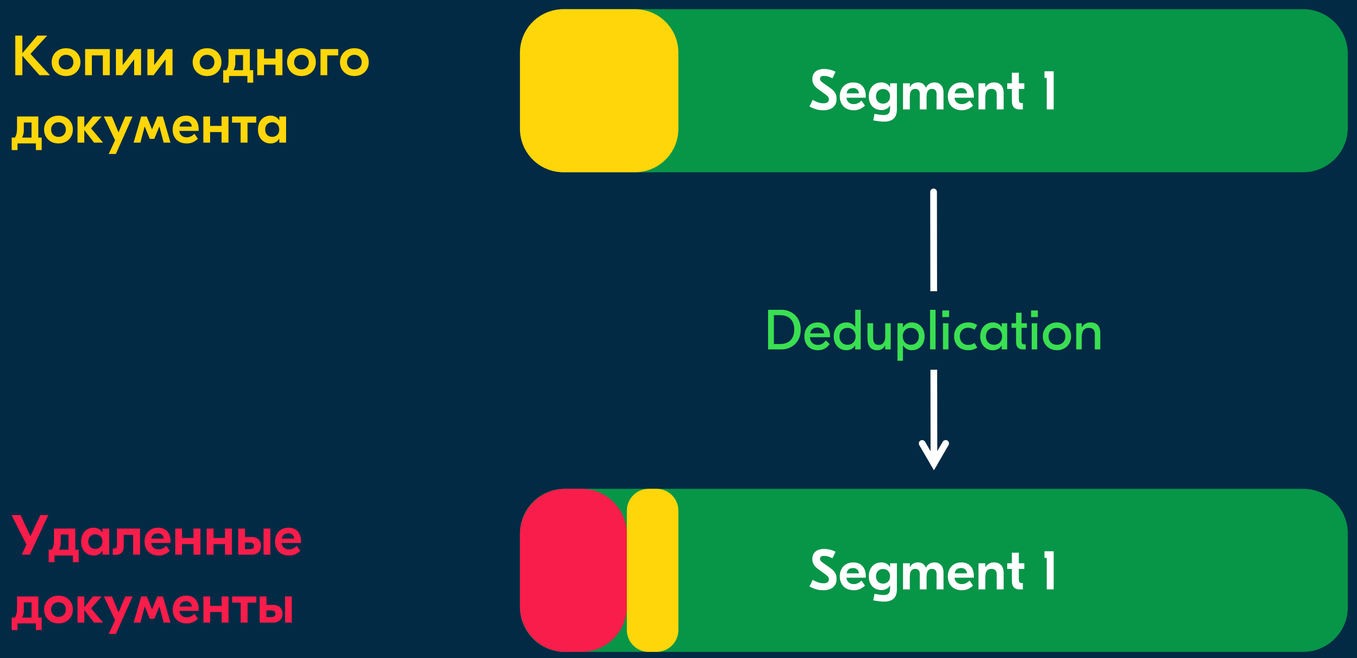
Вставка документов вместо апдейтов

- Можно вставлять не по одному документу, а сразу пачкой, так еще быстрее
- Метод **addDocuments(Iterable<Document> docs)**
- Потребовалось сделать ручную дедупликацию документов



Реализация O2 Master

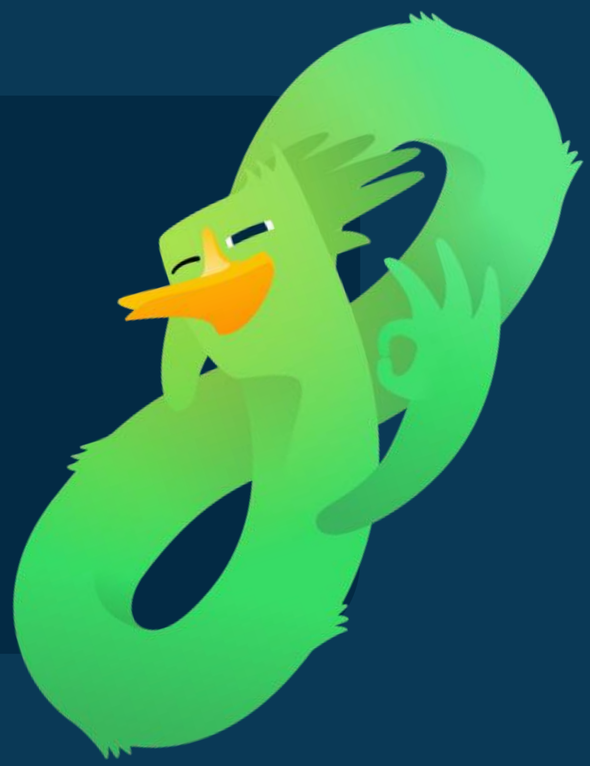
Дедупликация документов



Реализация O2 Master

Итоги

Скорость индексации
достигла **~9000 doc/sec**



Реализация O2 Master

ozon tech

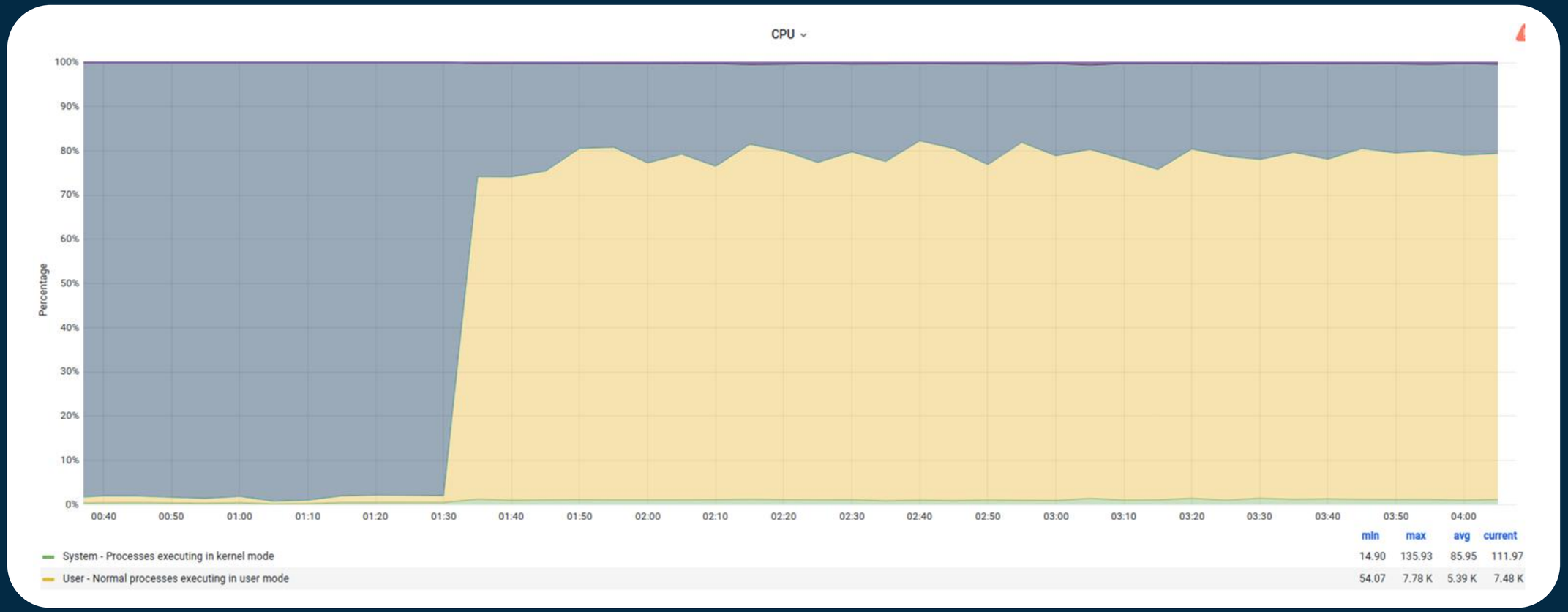
Результат замены апдейтов вставками

Архитектура кластера O2 Master

ozon tech

Результаты

- Утилизация ядер во время индексации
- Используется 70-80 ядер из 96



Реализация O2 Master

ozon tech

Вертикальное масштабирование индексации

- Спроектировали оптимальную архитектуру
- Разделили поиск и индексацию (**3000 doc/sec**)
- Снизили потребление памяти
- Подобрали GC/размер хипа (**6000 doc/sec**)
- Использовали bulk-операции, где это возможно (**9000 doc/sec**)

Выводы

1. Вертикальное масштабирование дается **непросто**

2. Реализация так же важна, как архитектура

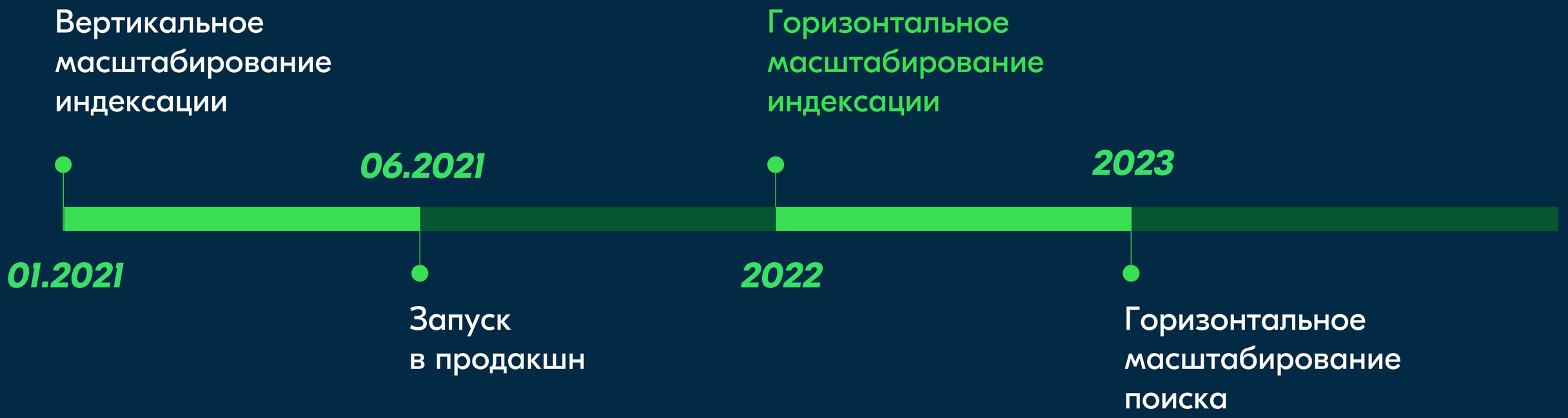
Инструменты

ozon tech

- async-profiler
- visual vm
- Java Microbenchmark Harness (JMH)
- Исходный код Lucene

Поисковая платформа Ozon

Таймлайн разработки движка O2

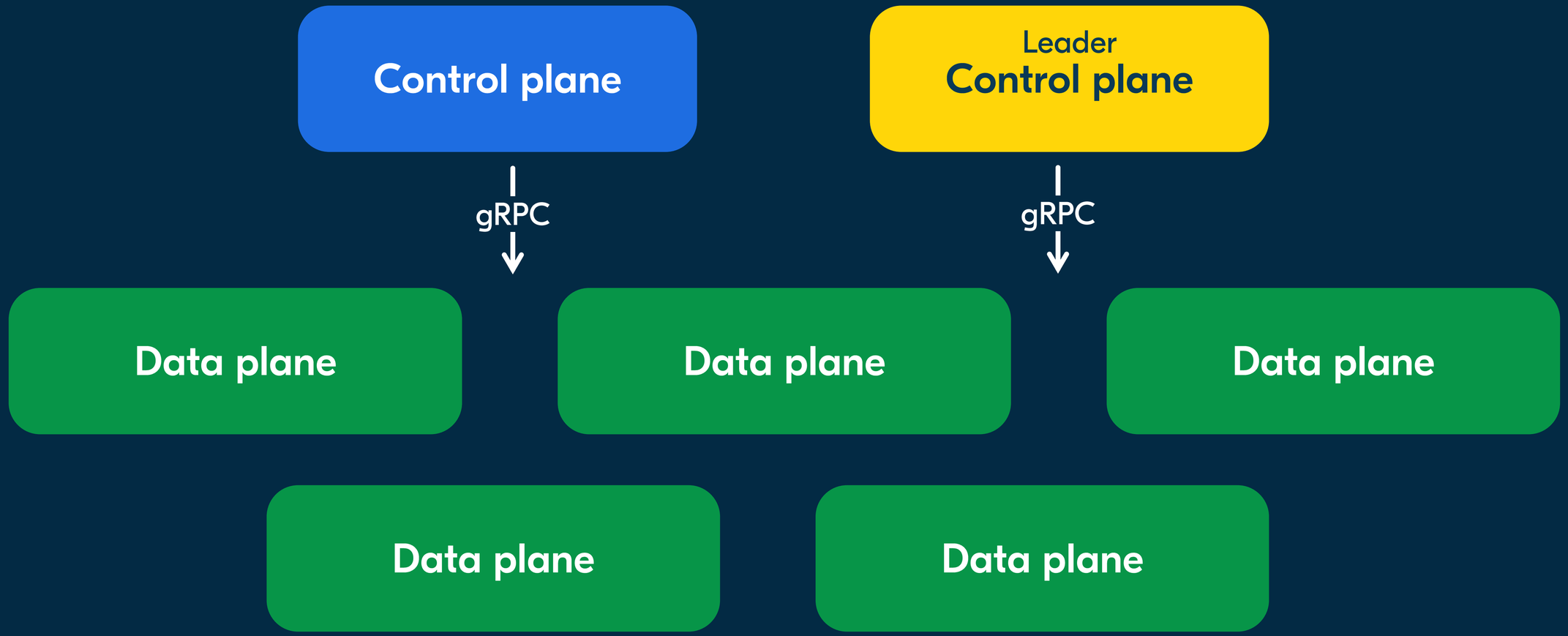


Горизонтальное масштабирование индексации (2022)

Архитектура кластера O2 Master (2022)

ozon tech

Горизонтальное масштабирование индексации



Архитектура кластера O2 Master (2022)

ozon{ech

Горизонтальное масштабирование индексации

- Отдельный кластер в каждом ЦОД
- Два типа нод: control plane и data plane

Control plane — это API для клиента (достаточно двух-трех нод на кластер)

Data plane — это сервера индексации и публикации индекса (может быть несколько сотен нод в одном кластере)

Control plane

ozon{tech

Горизонтальное масштабирование индексации

- Управляет метаданными индексов
- Предоставляет API для работы с индексами
- Выбирает лидера индекса
- Делает балансировку индексов по data plane-нодам
- Задает мэппинг индексов на реплики базового поиска
- Использует мало ресурсов (2-4 ядра, 8 GB RAM)

Data plane

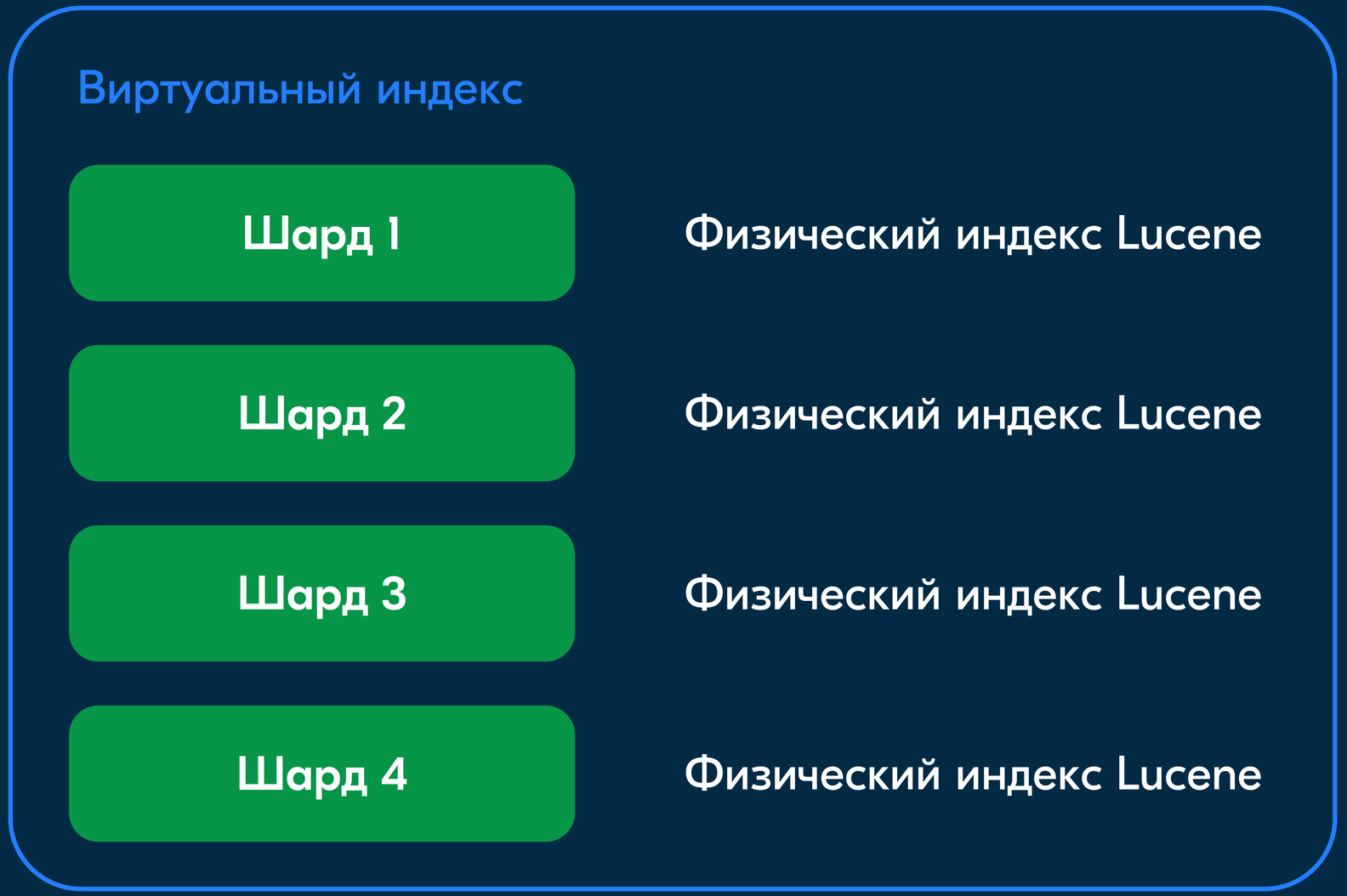
Горизонтальное масштабирование индексации

- Создает поисковые индексы
- Использует много ресурсов (80+ ядер, 700+ GB RAM)

Архитектура кластера O2 Master (2022)

ozon tech

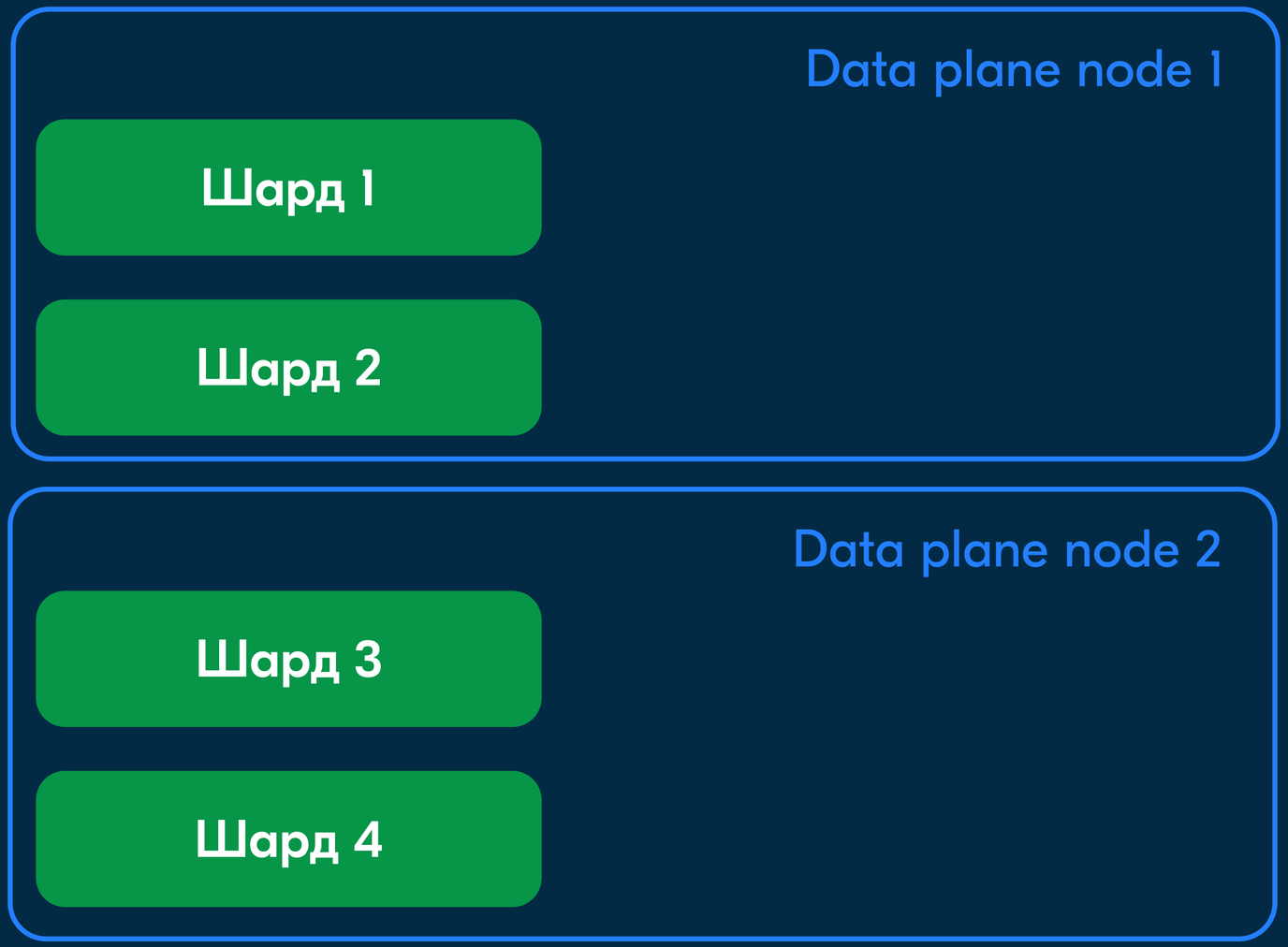
Горизонтальное масштабирование индексации



Архитектура кластера O2 Master (2022)

ozon{ech

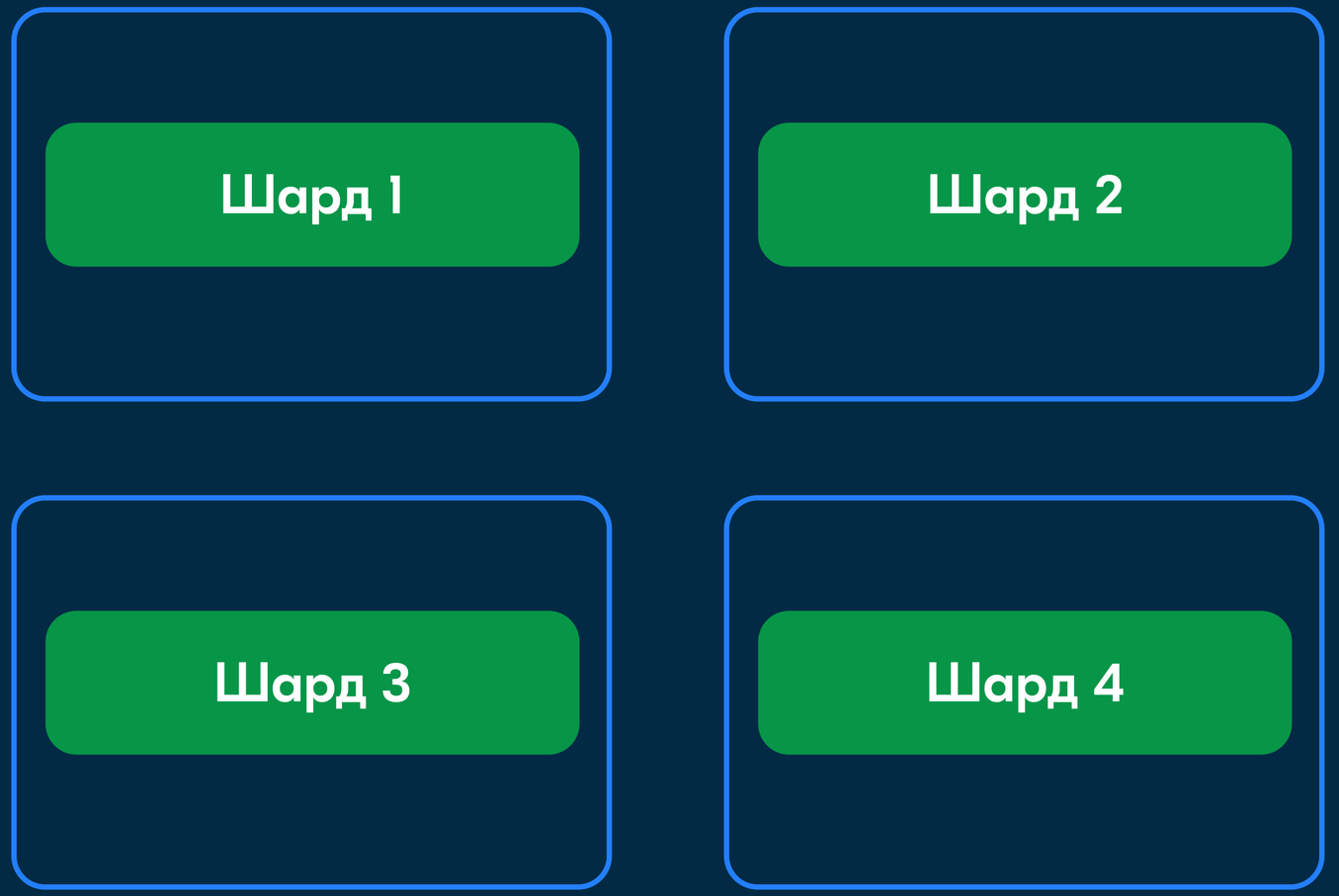
Горизонтальное масштабирование индексации



Архитектура кластера O2 Master (2022)

ozon tech

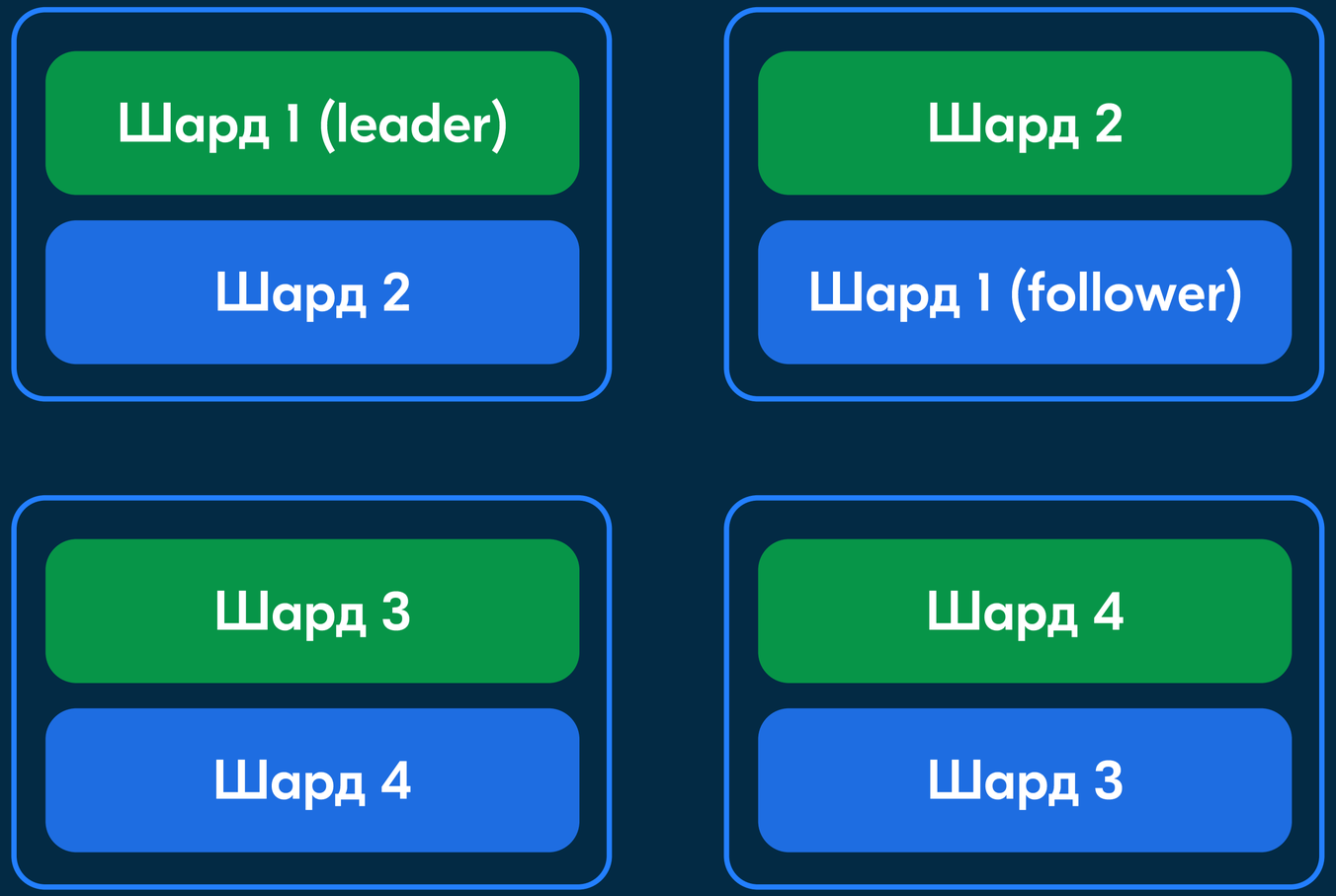
Динамическая балансировка



Архитектура кластера O2 Master (2022)

ozon tech

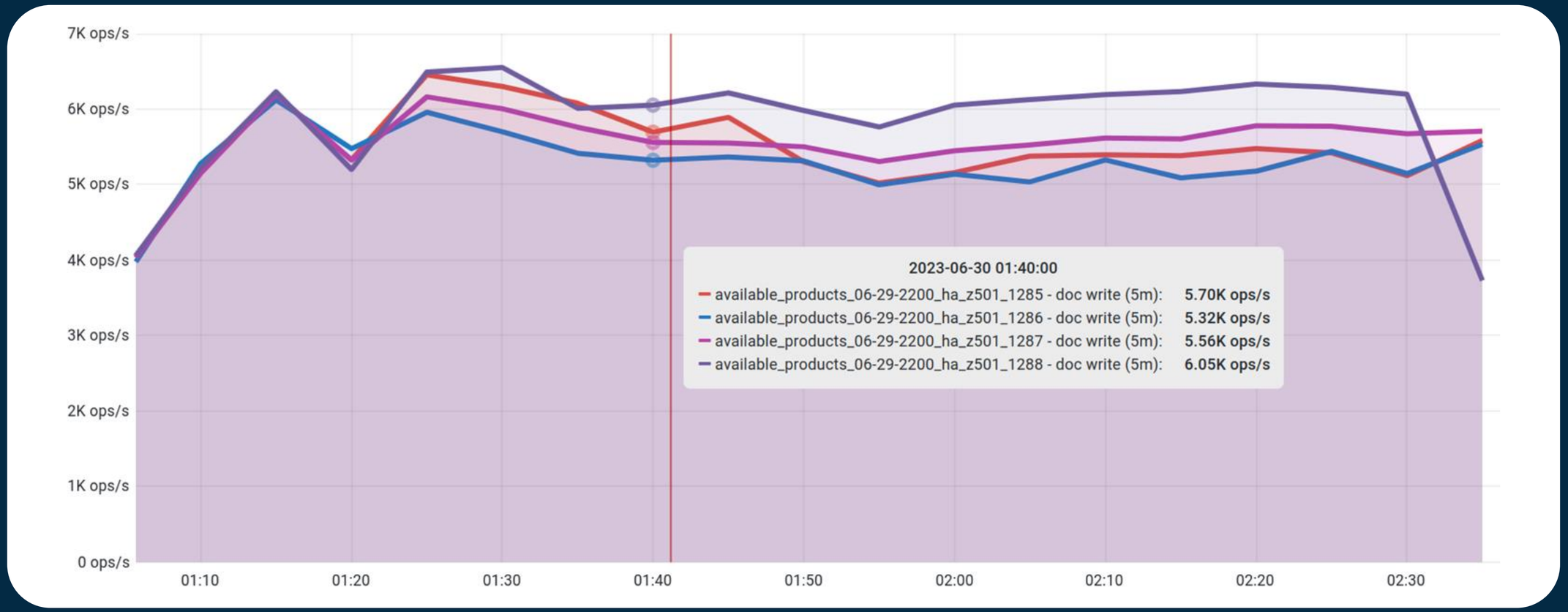
Hot standby



Архитектура кластера O2 Master (2022)

Результаты

- Сборка шардированного индекса



Отказоустойчивость

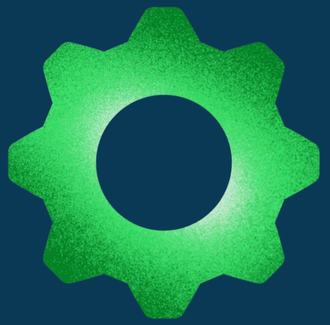
Архитектура кластера O2 Master

ozon{ech

Отказоустойчивость

- Собираем несколько одинаковых индексов в каждом ЦОД

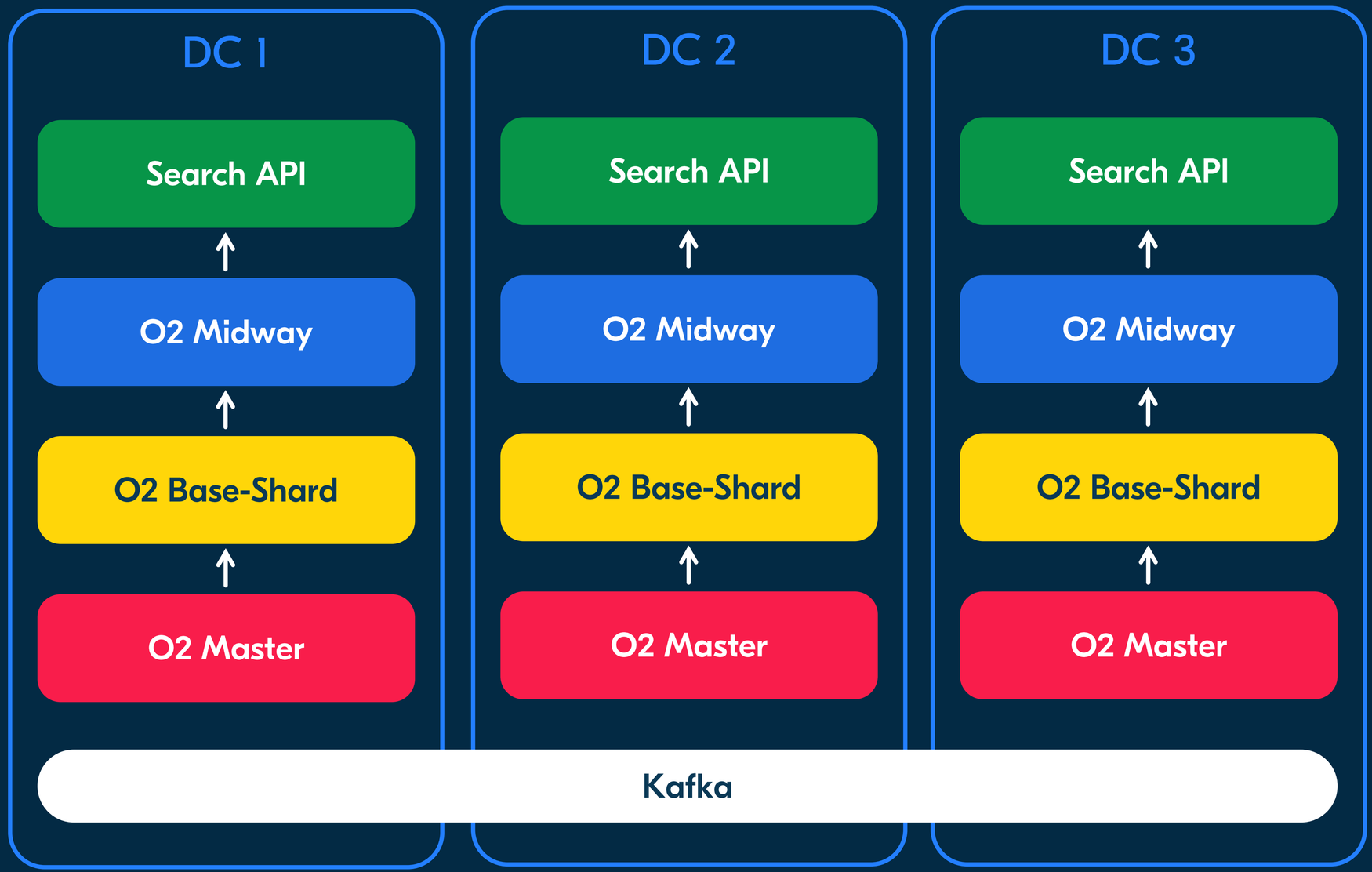
- Защищены от случайной порчи индекса (corrupted)



Поисковая платформа Ozon

ozon tech

Отказоустойчивость

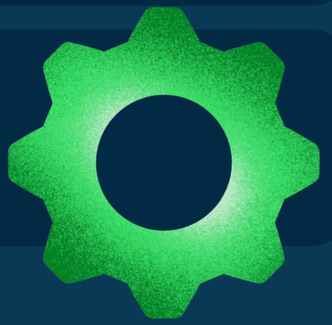


Архитектура кластера O2 Master

ozon tech

Отказоустойчивость

- Все типы нод поиска имеют более одного экземпляра
- Метаданные хранятся в PG (3 реплики)
- ETCD-кластер (5 реплик в трех ЦОД)
- Кафка в трех ЦОД



Архитектура кластера O2 Master

ozon tech

Результаты

1. Переживаем отказ **любой ноды**
2. Переживаем **отказ стойки**
3. Переживаем **отказ ЦОД**

В заключение

ozon tech

Цели достигнуты

- Индексация масштабируется вертикально и горизонтально
- Сервис отказоустойчив

В заключение

ozon tech

Как достичь производительности

- Знание стека и инструментов
- Изучение исходного кода открытых продуктов
- Реальное и виртуальное **общение** с коллегами в вашей области

Приглашение к дискуссии

ozon{ech

Что не вошло в доклад

- Как работает наш поисковый кластер?
- Собственная реализация поисковых запросов
- Низкоуровневые оптимизации в Apache Lucene
- Кастомная политика мержа сегментов
- Распределенные алгоритмы коммита, ребалансировки

OZON tech

Спасибо за внимание

Денис Габайдулин, Search Runtime

E-mail: gabaden@gmail.com Telegram: sherman81