Ай-ТИ Фреш

ТЕХНИЧЕСКИЙ РАЗБОР

Полнотекстовый поиск на 100 млн документов: наша методология

Elasticsearch 9.x: маппинг, русская морфология, ILM-кластер и гибридный kNN-поиск под 200 RPS

Июль 2026

itfresh.ru · ИТ-аутсорсинг для юридических лиц

О ЧЁМ ЭТОТ РАЗБОР

Суть проблемы

Поиск по корпоративному архиву — СЭД, файловые шары, договорная база — на десятках миллионов документов деградирует: узел Elasticsearch «из коробки» отвечает 5–8 секунд, а на пике падает целиком. Мы перестраиваем поиск в кластер с явным маппингом, русской морфологией, ILM и гибридным kNN: p95 ≤ 250 мс на 100 млн документов.

Почему это важно бизнесу

- Сотрудник ищет документ 20–30 минут вместо секунд — на 50 РМ это часы оплаченного времени каждый день
- Юрист или бухгалтер не находит договор/акт к дедлайну — срыв сделки, штраф по требованию ФНС
- Перегруженный поиск падает и валит зависимые сервисы: СЭД, портал и интеграции слепнут разом
- Права доступа мимо поискового слоя — конфиденциальные документы всплывают в чужой выдаче
- Рост индекса без ILM — компания оплачивает SSD под архив, который никто не открывает

ЧТО МЫ ЗАКЛАДЫВАЕМ

Ключевые параметры реализации

9.4.x

Актуальная ветка Elasticsearch, которую мы ставим в новые проекты; 8.19 (финальный минор 8.x)…

Elastic Stack 9.4

≤31 ГБ

Потолок JVM heap на data-ноде: выше теряются compressed oops; heap = 50% RAM, остальное — кешу…

по докам Elastic, JVM sizing

10–50 ГБ

Целевой размер шарда: держим rollover по max_primary_shard_size=50gb вместо фиксированного чис…

по докам Elastic, Size your shards

2–15

min_gram/max_gram edge_ngram-фильтра автокомплита — подсказки за 10–20 мс без wildcard-запросов

наш стандарт

x5–7

Прирост скорости поиска внутри отдела при custom routing: запрос уходит в один шард вместо все…

наш стандарт

200 RPS

Расчётная нагрузка, под которую держим p95: простой поиск ≤100 мс, фасетный ≤250 мс, гибридный…

наш стандарт

РЕАЛИЗАЦИЯ 1 ИЗ 3

Маппинг и русская морфология индекса documents

Что настраиваем

Индекс СЭД заказчика на 100 млн записей: поля title/body + служебные атрибуты, hot-ярус data-нод

Как мы это делаем

1. Фиксируем схему: dynamic: strict и index.mapping.total_fields.limit=200 — Elasticsearch больше не плодит поля из случайного JSON
2. Ставим community-плагин analysis-morphology сборкой под нашу версию ES (elasticsearch-plugin install <URL zip-архива>): словарная русская морфология вместо коллизий...
3. Собираем synonym_graph из analysis/synonyms_ru.txt с доменной терминологией заказчика: «допсоглашение => дополнительное соглашение», «инвойс => счёт»
4. Multi-field для title: text с морфологией + keyword для точного совпадения + edge_ngram 2–15 с отдельным search_analyzer без n-грамм на стороне запроса
5. Фильтруемые атрибуты (document_type, department, access_level) — строго keyword: фасеты считаются на doc_values, без fielddata и разогрева heap

РЕЗУЛЬТАТ

Число полей под контролем, выдача корректна по русским словоформам, автокомплит отвечает за 10–20 мс. Фасеты по типу/отделу/году считаются в том же запросе через aggs + post_filter: счётчики категорий видны даже при включённом фильтре.

КЛЮЧЕВОЙ НЮАНС

search_analyzer автокомплит-поля обязан быть без edge_ngram: иначе сам запрос режется на n-граммы и матчится на всё подряд. В документации это одна строка, на проде — главный источник «мусорных» подсказок.

РЕАЛИЗАЦИЯ 2 ИЗ 3

Кластер из 13 узлов и жизненный цикл индексов (ILM)

Что настраиваем

3 master-eligible + 6 data (16 CPU, 64 ГБ RAM, SSD) + 2 coordinating + 2 ingest; оркестрация в Kubernetes

Как мы это делаем

1. Разносим node.roles: выделенные master ради стабильного кворума, coordinating на приёме запросов, ingest под attachment-pipeline для парсинга PDF/DOCX
2. JVM heap = 50% RAM, но не выше 31 ГБ; bootstrap.memory_lock: true, swap отключён — вторая половина памяти достаётся filesystem-кешу Lucene
3. ILM-политика: hot — rollover max_primary_shard_size=50gb / max_age=30d на SSD; warm — shrink до 2 шардов + forcemerge max_num_segments=1; cold — replicas 0
4. SLM-снапшоты в S3-совместимый репозиторий каждую ночь; cold-ярус без реплик допускаем только после проверенного тестового restore
5. Bulk-переиндексация: на время загрузки refresh_interval=30s и number_of_replicas=0, после — возврат штатных значений; индексация ускоряется в разы

РЕЗУЛЬТАТ

Кластер переживает отказ любой data-ноды без потери данных, свежие документы ищутся с SSD, архив старше года не занимает дорогие диски. Деградация из «поиск падает на 100 млн» превращается в плановое масштабирование добавлением data-нод.

КЛЮЧЕВОЙ НЮАНС

forcemerge запускаем только на read-only индексах warm-яруса: на живом hot-индексе он порождает гигантские сегменты, которые больше никогда не сольются автоматически, и съедает I/O в часы нагрузки.

РЕАЛИЗАЦИЯ 3 ИЗ 3

Релевантность: BM25, function_score и гибрид с kNN

Что настраиваем

Поисковый слой поверх кластера: query-шаблоны + GPU-узел эмбеддингов intfloat/multilingual-e5-base (768 dims)

Как мы это делаем

1. multi_match type=best_fields с весами title^3 / tags^2 и fuzziness=AUTO; права доступа и даты — только в filter-контексте: кешируется и не искажает score
2. function_score: gauss-decay по created_at (scale=90d, decay=0.5) поднимает свежее + field_value_factor по view_count с modifier=log1p учитывает популярность
3. dense_vector 768 dims, similarity=cosine, квантование int8_hnsw — векторный индекс в 4 раза компактнее без заметной потери recall
4. Гибрид: bool should из BM25 (boost 0.7) и knn с num_candidates=100 (boost 0.3); слияние через retriever rrf — только при подписке Enterprise, на Basic остаёмся на в…
5. Второй этап — re-ranking LightGBM по кликовым сигналам (открыл документ и провёл >30 с = позитив); переобучение еженедельно на свежих логах

РЕЗУЛЬТАТ

Запрос «компенсация ущерба при просрочке» находит документ со «штрафными санкциями за нарушение сроков» — семантика закрывает то, где BM25 пасует. Коротким релевантным письмам больше не проигрывают 200-страничные контракты с одним вхождением.

КЛЮЧЕВОЙ НЮАНС

Вектор не заменяет BM25: точные номера договоров и ФИО семантика теряет, а синонимичные формулировки теряет полнотекст. Работает только гибрид, и num_candidates подбираем замером: выше — точнее, но дороже по С…

ЧТО МЫ ВЫУЧИЛИ НА ПРАКТИКЕ

Подводные камни

- ✕ **Dynamic mapping раздувает схему**
  Поля плодятся из каждого нового JSON до тысяч, кластер деградирует. Ставим dynamic: strict + total_fields.limit=200 с первого дня.

- ✕ **Шарды-гиганты и шарды-пыль**
  Ручное число шардов «на глаз» даёт либо 200-гигабайтные шарды, либо тысячи мелких. Лечим rollover по max_primary_shard_size=50gb через ILM.

- ✕ **JVM heap больше 31 ГБ**
  Теряются compressed oops, GC-паузы растут, кешу Lucene не остаётся памяти. Правило: heap = 50% RAM и не выше 31 ГБ, memory_lock включён.

- ✕ **Штатный русский стеммер**
  Обрубание окончаний даёт ложные склейки и разрывы основ. Подключаем словарный analysis-morphology или hunspell ru_RU с dedup.

- ✕ **Глубокая пагинация from+size**
  Страница 1000 заставляет каждый шард держать 10 000+ хитов в памяти. Всё глубже 10k результатов — через search_after с PIT.

- ✕ **Wildcard в начале строки**
  Запрос *слово сканирует весь термсловарь индекса. Автокомплит строим на edge_ngram или search_as_you_type, wildcard-поиск запрещаем на API.

- ✕ **replicas=0 на проде ради скорости**
  Потеря одной ноды = потеря данных и красный кластер. Ноль реплик допустим только на cold-ярусе при проверенных SLM-снапшотах.

- ✕ **Права доступа после выдачи**
  Фильтрация ACL в приложении ломает пагинацию и светит чужие документы в счётчиках. access_level режем filter-контекстом в самом запросе.

УРОВНИ РЕАЛИЗАЦИИ

Как правильно

МИНИМУМ

- Явный маппинг dynamic: strict; keyword для фильтров, text — для контента
- Минимум 3 ноды и 1 реплика на каждый шард — одиночный узел не для прода
- SLM-снапшоты в S3/FS-репозиторий ежесуточно + тестовый restore
- Русский анализатор со стоп-словами, потолок полей 200

НОРМАЛЬНО

- 3 выделенные master-ноды + разнос ролей data/coordinating/ingest
- ILM hot/warm/cold с rollover 50gb/30d вместо ручного управления индексами
- Словарная морфология analysis-morphology + доменный файл синонимов
- Мониторинг heap, p95 латентности и статуса шардов с алертами в Telegram

ХОРОШО

- Custom routing по естественному ключу данных (отдел/тенант) — x5–7 к скорости
- Гибридный поиск: dense_vector int8_hnsw + BM25; retriever rrf при Enterprise, иначе…
- ML re-ranking на кликовых сигналах (LightGBM, еженедельное переобучение)
- Нагрузочная регрессия Rally на каждое изменение маппинга и версии

ПРОВЕРЬТЕ У СЕБЯ

Чек-лист самопроверки

- [ ] Маппинг задан явно (dynamic: strict) и проходит ревью при каждом изменении схемы документов?
- [ ] Размер шардов держится в коридоре 10–50 ГБ автоматическим rollover, а не ручным пересозданием?
- [ ] JVM heap ≤31 ГБ и ≤50% RAM ноды; bootstrap.memory_lock включён, swap отключён?
- [ ] Есть ILM-политика hot/warm/cold и данные старше года не занимают дорогие SSD?
- [ ] Снапшоты снимаются по SLM-расписанию и восстановление реально проверялось на стенде?
- [ ] Русская морфология словарная (analysis-morphology/hunspell), а не только штатный стеммер?
- [ ] Права доступа режутся filter-контекстом в самом запросе, а не в приложении после выдачи?
- [ ] Замеряется p95 латентности под боевой нагрузкой, а не средняя в тихие часы?
- [ ] Пагинация глубже 10 000 результатов переведена на search_after/PIT?
- [ ] Кластер закрыт: TLS между нодами, аутентификация включена, порт 9200 не торчит наружу?

Если хотя бы на два вопроса ответ «нет» или «не знаю» — тема требует внимания.

ЧТО МЫ ДЕЛАЕМ

Как поможет ITFresh

ITFresh — ИТ-аутсорсинг для юридических лиц до 50 рабочих мест в Москве и области. 15+ лет практики, собственная инфраструктура в дата-центре МТС (8 серверов Dell Xeon Platinum).

- Аудит действующего кластера Elasticsearch/OpenSearch: маппинг, шарды, JVM, ILM — с планом фиксов
- Проектирование и развёртывание кластера под ключ: роли нод, ILM, SLM-снапшоты, мониторинг
- Русскоязычная релевантность: морфология, синонимы, function_score под домен заказчика
- Гибридный поиск: эмбеддинги, kNN, ML re-ranking — от пилота до продакшена
- Миграции и апгрейды 7.x→8.x→9.x и ES↔OpenSearch без остановки поиска

15+
лет в ИТ-поддержке

50
рабочих мест — наш профиль

МТС
дата-центр, Москва

КОНТАКТЫ

Обсудить вашу задачу

Сайт	**itfresh.ru**
Телефон	**+7 903 729-62-41**
Telegram	**@ITfresh_Boss**

Бесплатно посмотрим вашу инфраструктуру по этому чек-листу и скажем, где тонко — без обязательств.

itfresh.ru

ДОКУМЕНТАЦИЯ И НАШ СТЕК

Техническая база

01 Elasticsearch Guide — Size your shards (elastic.co — 9.4)

02 Elasticsearch Guide — Index Lifecycle Management (elastic.co — 9.4)

03 Text analysis: hunspell filter, synonym_graph, edge_ngram (elastic.co — 9.4)

04 Dense vector field type, kNN search и RRF retriever (elastic.co — 9.4)

05 Advanced configuration — heap size, memory lock (elastic.co — 9.4)

06 Upgrade to 9.x — prepare on 8.19, Upgrade Assistant (elastic.co — 9.4)

07 OpenSearch Documentation — version history (opensearch.org — 3.7)

08 Наш шаблон индекса documents + ILM-политика ITfresh (itfresh.ru — 2026)

Основано на официальной документации продуктов и нашей практике внедрения.