Глава 1
Начало работы с Elasticsearch

ВВЕДЕНИЕ

В этой главе мы поближе познакомимся с Elasticsearch. Начнем с обсуждения преимуществ Elasticsearch и того, как эта система может помочь компаниям достичь целей в области управления данными. Затем рассмотрим, что представляет собой Elasticsearch и как она использует мощный поисковый движок Lucene для быстрого и масштабируемого поиска. Чтобы как следует разобраться в основах Elasticsearch, мы рассмотрим некоторые базовые понятия, такие как узлы, кластеры, документы, индексы и шарды. Это необходимо для понимания того, как Elasticsearch хранит и организует данные в целях эффективного поиска и извлечения информации.

Затем мы рассмотрим некоторые основные сценарии использования Elasticsearch, включая поиск данных, ведение логов и анализ, мониторинг производительности приложений и систем, а также визуализацию данных. Эти примеры подчеркивают универсальность Elasticsearch и демонстрируют его аналитический потенциал в широком спектре отраслей и приложений. Кроме того, мы обсудим клиенты Elasticsearch, доступные для разработчиков, такие как Java, PHP, Perl, Python, .NET и JavaScript. Эти клиенты позволяют разработчикам применять поисковые возможности Elasticsearch в языке программирования, который они используют, и его экосистеме.

Наконец, мы обсудим, как использовать Elasticsearch в качестве первичного источника данных, вторичного источника данных или отдельной системы. Мы дадим рекомендации по принятию обоснованных решений о включении Elasticsearch в архитектуру данных на основе конкретных бизнес-потребностей и технических требований.

СТРУКТУРА ГЛАВЫ

В этой главе:

- Введение в поиск данных
- Что такое Elasticsearch и почему он важен для поиска и аналитики
- Обзор архитектуры и компонентов Elasticsearch
- Области применения и сценарии использования Elasticsearch
- Клиенты Elasticsearch и сценарии их использования

ЦЕЛИ

В этой главе представлен обзор движка Elasticsearch и его возможностей. В начале главы вводятся понятия поиска и аналитики и объясняется, почему они важны в современном мире, основанном на данных. Далее рассказывается о том, как эффективно использовать клиенты Elasticsearch.

ВВЕДЕНИЕ В ПОИСК ДАННЫХ

В современном мире экспоненциальный рост объема оцифрованных данных из различных источников, таких как смарт-устройства, датчики IoT и онлайн-транзакции, представляет собой серьезную проблему. Один из основных вызовов, стоящих перед отраслью, — преобразование неструктурированных данных в структурированную форму для оптимизации процесса хранения данных. Однако настоящая проблема кроется в поиске нужной информации в хранящихся данных. Традиционные системы хранения данных, такие как РСУБД, не подходят для текстового поиска из-за сложности написания SQL-запросов и неэффективности поиска даже после применения всех необходимых индексов. В отличие от них Elasticsearch, поисковая система на базе Lucene, предлагает сложный механизм поиска с выдачей релевантных результатов, агрегацией данных и многими другими преимуществами, недоступными РСУБД. Поэтому понимание важности поиска и того, как Elasticsearch способна помочь оптимизировать хранение и поиск данных, крайне важно для любой организации, работающей с большими объемами данных.

Поиск — важнейший компонент современных приложений, поскольку именно благодаря ему пользователи могут быстро и точно находить нужную информацию. Для всех приложений, работающих с большими объемами данных, будь то блог, интернет-магазин или любое другое приложение, механизм поиска необходим, чтобы предоставлять пользователям релевантные результаты.

Важность быстрых и точных результатов поиска трудно переоценить, поскольку пользователи, скорее всего, откажутся от приложения, которое не соответствует их требованиям к поиску. Поэтому оптимизация работы поиска чрезвычайно важна для эффективного взаимодействия с пользователями и сохранения их вовлеченности.

Помимо предоставления быстрых результатов, необходимо учитывать и другие важные параметры, такие как релевантность, агрегация и анализ данных. Это требование успешно обеспечивает Elasticsearch — мощная и масштабируемая поисковая система, способная работать с разными типами и источниками данных. Используя возможности Elasticsearch, приложения предоставляют пользователям быстрые, точные и релевантные результаты; это делает ее важнейшим компонентом современных приложений, ориентированных на поиск.

Важность функциональности поиска трудно переоценить. Помимо обеспечения быстрого отклика на запрос и получения релевантных результатов, существуют и другие критерии.

- **Предложение поиска.** Эффективная система поиска должна предлагать потенциальные поисковые запросы, как только пользователь начинает вводить текст.
- **Нечеткий поиск.** Система должна предлагать релевантные результаты, даже если пользователь допустил орфографическую ошибку в поисковом термине или использовал синоним.
- **Поиск по производным.** Качественная поисковая система должна распознавать производные поисковых терминов, например слова во множественном или единственном числе, чтобы предоставлять наиболее полные результаты.
- **Агрегация данных.** Система должна поддерживать агрегацию данных, чтобы предлагать пользователю дополнительные опции и фильтры, например ценовой диапазон, рейтинги, бренды и другую соответствующую информацию.
- **Релевантные результаты.** Результаты поиска должны выводиться в порядке соответствия запросу, с учетом таких факторов, как частота поискового термина, периодичность и поведение пользователей.
- **Расширенные фильтры.** Пользователи должны иметь возможность применять расширенные фильтры к результатам поиска, например по разрешению экрана, объему оперативной памяти, цвету и другим важным критериям.
- **Быстрое время отклика.** Поисковая система должна предоставлять результаты поиска быстро, в течение нескольких секунд, чтобы избежать

промедлений в работе пользователей и не вызывать у них негативных эмоций.

Учитывая эти критерии, разработчики могут создавать эффективные поисковые системы, выдающие быстрые и точные результаты, что в конечном итоге обеспечивает повышение вовлеченности и удовлетворенности пользователей.

ЧТО ТАКОЕ ELASTICSEARCH И ПОЧЕМУ ОН ВАЖЕН ДЛЯ ПОИСКА И АНАЛИТИКИ

Поисковый движок Elasticsearch был создан *Шаем Баноном (Shay Banon)*, основателем компании Elastic, которая занимается дальнейшей разработкой и поддержкой этого продукта. Elasticsearch — это ПО с открытым исходным кодом, которое может быть запущено на одном сервере или на сотнях серверов, чтобы обрабатывать петабайты данных. Elasticsearch — это мощная система, которая используется для поиска нужных данных в хранилищах большого объема.

В наш информационный век объем данных растет в геометрической прогрессии благодаря оцифровке и появлению новых источников данных, таких как смарт-устройства, датчики IoT (интернета вещей) и онлайн-транзакции. Эти данные могут быть структурированными или неструктурированными, относящимися к конкретному устройству или временным рядам и поступать из разных источников, что затрудняет их поиск вручную. Для решения этих проблем Elasticsearch предоставляет распределенную, масштабируемую и документоориентированную поисковую систему на базе библиотеки Lucene. Lucene — это высокопроизводительная библиотека, которая обеспечивает быстрые и эффективные результаты поиска. Однако для ее использования требуется сложный код Java, и ее нелегко распределить по нескольким узлам.

Elasticsearch инкапсулирует все сложности Lucene и предоставляет REST API, которые облегчают пользовательское взаимодействие с Elasticsearch. Он также обеспечивает поддержку нескольких языков программирования с помощью языковых клиентов, поэтому можно писать код на желаемом языке и при этом взаимодействовать с Elasticsearch. Кроме того, с Elasticsearch можно взаимодействовать с помощью инструмента командной строки cURL.

Итак, Elasticsearch — это мощный поисковый и аналитический движок, обеспечивающий быстрый и эффективный поиск в больших объемах данных, что делает его жизненно важным инструментом для организаций, стремящихся извлечь из своих данных максимальную ценность.

ОБЗОР АРХИТЕКТУРЫ И КОМПОНЕНТОВ ELASTICSEARCH

Elasticsearch имеет распределенную архитектуру, которая позволяет обрабатывать большие объемы данных на нескольких узлах. Она включает несколько компонентов, которые работают вместе и обеспечивают масштабируемую и высокодоступную платформу для поиска и аналитики.

Узел

В Elasticsearch под узлом (node) понимается отдельный рабочий экземпляр поисковой системы. Elasticsearch состоит из одного или нескольких узлов, которые являются экземплярами сервера Elasticsearch. Например, в кластере из десяти серверов, на которых работает Elasticsearch, каждый сервер будет считаться узлом. В некоторых случаях для нерабочих сред может быть достаточно кластера с одним узлом. Однако по мере увеличения объема данных возникает необходимость в дополнительных узлах для горизонтального масштабирования кластера, что также обеспечивает отказоустойчивость. Зная о других узлах кластера, узел может передавать запросы клиентов соответствующему узлу. Стоит отметить, что узлы могут играть разные роли: узлы данных хранят и выполняют запросы; главные узлы управляют операциями всего кластера; узлы-координаторы пересылают запросы на соответствующие узлы. Каждый узел работает независимо и взаимодействует с другими, образуя кластер. Узлы можно добавлять или удалять из кластера динамически, не влияя на работу всей системы. Узлы могут быть разных типов.

Узел, допустимый для выбора в качестве главного (master-eligible node)

В Elasticsearch 8 узел, допустимый для выбора в качестве главного узла, или мастера, отвечает за управление состоянием кластера, включая добавление или удаление узлов, распределение шардов между узлами и поддержание работоспособности кластера. Рекомендуется иметь в кластере не менее трех таких узлов-кандидатов, чтобы обеспечить высокую доступность и избежать ситуаций разделения мозга (split-brain) (см. врезку ниже).

Выделенный узел, допустимый для выбора в качестве главного (dedicated master-eligible node)

Выделенный узел, который является кандидатом в мастер-узел, — это узел в кластере Elasticsearch, который сконфигурирован как главный и не имеет никаких других обязанностей, таких как хранение данных или обработка поисковых

запросов. Задача выделенного узла — повысить стабильность и надежность кластера, в котором задачи главного узла выполняет выделенный узел.

Чтобы сконфигурировать узел как главный (мастер-узел) в Elasticsearch 8, необходимо задать следующие параметры в файле конфигурации `elasticsearch.yml`:

```
node.roles: [ master ]
```

Параметр `node.roles` должен иметь значение `master`, указывая, что этот узел может быть использован в качестве главного.

Ситуация разделения мозга (split-brain)

В Elasticsearch ситуация разделения мозга возникает, когда кластер разделяется на несколько подкластеров, каждый из которых считает, что только он контролирует ситуацию. Это может произойти из-за нарушения связи между узлами в кластере вследствие сетевых проблем или других сбоев. В таких случаях каждый подкластер продолжает работать независимо, самостоятельно обновляя одни и те же данные, что может привести к несогласованности данных.

Ситуации split-brain можно предотвратить, создав систему на основе кворума, когда большинство узлов должно согласиться с принятым решением. В Elasticsearch это достигается путем наделения определенных узлов правами главных и требованием, чтобы для нормального функционирования кластера было доступно большинство таких узлов. Когда узел с правами главного обнаруживает, что больше не имеет связи с большинством этих узлов, он отключается и инициирует процесс выбора нового главного узла.

Важно правильно настроить Elasticsearch для обработки ситуаций разделения мозга, чтобы обеспечить согласованность данных и избежать их потери. Рекомендуется использовать выделенный главный узел и следить, чтобы в кластере всегда было нечетное количество узлов-кандидатов в главный узел (для гарантии большинства). Правильная настройка сетевых параметров и мониторинг потенциальных проблем также помогут предотвратить возникновение ситуаций split-brain.

ПРИМЕЧАНИЕ Важно отметить, что главный узел не должен быть перегружен другими задачами, поскольку ему требуется достаточно ресурсов для управления состоянием кластера. Если один главный узел выходит из строя или становится недоступным, оставшиеся узлы-кандидаты выбирают новый главный узел для управления состоянием кластера.

Голосующий узел (voting-only master-eligible node)

В Elasticsearch голосующий узел — это узел, который только участвует в процессе выбора главного узла, но сам не может им стать. Когда главный узел выходит из строя или становится недоступным, оставшиеся узлы должны выбрать новый главный узел для поддержания стабильности кластера. В это время узлы-кандидаты участвуют в выборе нового главного узла. Чтобы настроить в Elasticsearch 8 голосующий узел, необходимо задать следующие параметры в файле конфигурации `elasticsearch.yml`:

```
node.roles: [ data, master, voting_only ]
```

В примере выше мы задаем голосующий узел данных. Также можно задать выделенный голосующий узел, используя опцию в файле конфигурации `elasticsearch.yml`:

```
node.roles: [ master, voting_only ]
```

В примере выше мы устанавливаем голосующий узел, не имеющий обязанностей узла данных.

Узел данных (data node)

Узлы данных отвечают за хранение и управление данными, а также за выполнение CRUD-операций, поиска и агрегации данных. Можно настроить узел как узел данных, установив для параметра `node.data` значение `true` в файле конфигурации Elasticsearch. Если необходим отдельный узел данных, можно установить для других типов значение `false`, как показано в следующем фрагменте кода:

```
node.roles: [ data ]
```

В примере выше мы устанавливаем для параметра `"node.role"` значение `data`, а для всех остальных параметров — `false`, что делает узел выделенным узлом данных. Добавив в кластер больше узлов данных, можно горизонтально масштабировать кластер и обрабатывать большие объемы данных. Узлы данных также могут выполнять распределение и ребалансировку шардов, что помогает равномерно распределять данные по кластеру для повышения производительности и отказоустойчивости.

Узел поглощения данных (ingest node)

Узел поглощения данных — это специализированный тип узла в Elasticsearch, который позволяет выполнять предварительную обработку документов перед их индексацией. Он отвечает за обработку данных по мере их прохождения через пайплайн Elasticsearch, например, добавление в документы

дополнительных данных, обработку значений полей и выполнение преобразований данных.

Узлы поглощения имеют собственный специальный пайплайн, который можно настраивать для извлечения и преобразования данных с помощью различных процессоров, таких как grok, dissect и geoip. Это особенно полезно в случаях, когда необходимо извлечь релевантную информацию из неструктурированных данных, таких как файлы логов или потоки социальных сетей.

Чтобы сконфигурировать узел поглощения данных, установим для параметра `node.roles` в файле конфигурации Elasticsearch значение `ingest`. Вот пример:

```
node.roles: [ ingest ]
```

Узел поглощения позволяет выполнять предварительную обработку данных без необходимости писать собственный код или использовать внешние инструменты. Это упрощает пайплайн обработки данных и делает его более эффективным, особенно когда требуется обрабатывать большие объемы данных в реальном времени.

Узел машинного обучения (machine learning node)

Узел машинного обучения — это узел, способный выполнять задания машинного обучения на данных, хранящихся в кластере Elasticsearch. Узлы машинного обучения имеют специализированные аппаратные конфигурации и оптимизированы для обработки больших объемов данных в реальном времени.

Чтобы настроить узел машинного обучения, необходимо активировать функцию машинного обучения в файле конфигурации Elasticsearch и назначить узлу роль узла машинного обучения. Это можно сделать, добавив следующую строку в файл конфигурации `elasticsearch.yml`:

```
xpack.ml.enabled: true
```

Параметр `xpack.ml.enabled` включает функцию машинного обучения, а параметр `node.ml` назначает узлу роль узла машинного обучения.

После того как узел настроен, можно создавать задания машинного обучения с помощью API машинного обучения Elasticsearch. Эти задания могут анализировать данные в реальном времени и выявлять паттерны, аномалии и тенденции.

Чтобы создать выделенный узел машинного обучения, отредактируйте файл конфигурации Elasticsearch (`elasticsearch.yml`) и добавьте следующую строку:

```
node.roles: [ ml, remote_cluster_client]
```

Эта строка указывает Elasticsearch настроить узел в качестве узла машинного обучения и узла удаленного клиента кластера. Настройка удаленного клиента кластера необходима, поскольку она позволяет узлу машинного обучения получать доступ к данным из других кластеров, что может понадобиться для анализа. Узел машинного обучения — платная функция Elasticsearch, которая позволяет запускать модели машинного обучения на данных Elasticsearch.

Предположим, что у вас есть узел машинного обучения в одном кластере Elasticsearch, но данные, которые вы хотите использовать для машинного обучения, хранятся в другом кластере. Настроив узел машинного обучения как узел удаленного клиента кластера, вы можете получить доступ к данным, хранящимся в другом кластере, не перенося их в кластер узла машинного обучения. Это позволяет сэкономить время и ресурсы, а также избежать ненужного дублирования данных.

Узел горячих данных (hot data node)

Узлы горячих данных — это особый тип узлов данных в Elasticsearch, которые оптимизированы для работы с высокопроизводительными рабочими нагрузками и высоким трафиком. Они предназначены для хранения наиболее часто используемых и запрашиваемых данных, также известных как *горячие данные*, и обычно развертываются на высокопроизводительном оборудовании для обеспечения быстрого времени отклика.

Узлы горячих данных характеризуются способностью обрабатывать большой объем запросов на чтение и запись в реальном времени. Они оптимизированы для эффективного индексирования и поиска данных, что делает их идеальными для таких сценариев использования, в которых требуется быстрый и частый доступ к данным, например для интернет-магазинов, социальных сетей и приложений финансовых услуг.

Чтобы настроить узел горячих данных, укажите следующие параметры в файле конфигурации Elasticsearch:

```
node.roles: [ data-hot ]
```

Используя настройки выше, можно определить узел горячих данных в Elasticsearch.

Узел теплых данных (warm data node)

Узел теплых данных в Elasticsearch — это тип узла, оптимизированный для хранения и поиска больших объемов данных, к которым обращаются реже. К ним могут относиться старые или реже используемые логи, исторические данные или резервные копии. Теплый узел обычно имеет меньшую производительность

хранения и меньшие требования к памяти по сравнению с горячими узлами, но он может хранить большой объем данных при меньших затратах.

Теплые узлы также предназначены для работы с нагрузками, связанными с чтением, и могут быть не так отзывчивы на запросы записи, как горячие узлы. Чтобы настроить узел теплых данных в Elasticsearch 8, задайте следующие параметры в файле конфигурации `elasticsearch.yml`:

```
node.roles: [ data-warm ]
```

Таким образом узел будет настроен для обработки теплых данных и оптимизирован для хранения нечасто используемых данных и доступа к ним.

Узел холодных данных (cold data node)

Узел холодных данных — это тип узла данных в Elasticsearch, специально предназначенный для хранения редко используемых или архивных данных. Холодные узлы обычно создаются более медленными и имеющими меньшие вычислительные ресурсы, что делает выгодным их использование для хранения больших объемов данных с редким доступом.

Чтобы настроить узел холодных данных в Elasticsearch 8, измените параметр `node.roles` в файле конфигурации `elasticsearch.yml` следующим образом:

```
node.roles: [ data-cold ]
```

Таким образом узел будет настроен для обработки холодных данных и оптимизирован для хранения редко используемых данных и управления ими, в то время как другие узлы в кластере будут обрабатывать более активные и часто используемые данные.

Узлы холодных данных полезны для долгосрочного хранения данных, а также соблюдения нормативных и регуляторных требований. Отделение холодных данных от горячих или теплых позволит оптимизировать распределение ресурсов и производительность кластера Elasticsearch. Узлы холодных данных обычно используются для данных, которые не требуют частых запросов или анализа в реальном времени, но при этом должны храниться и быть доступными для соблюдения нормативных требований или для исторических целей.

Узел замороженных данных (frozen data node)

В Elasticsearch 8 узел замороженных данных — это специализированный тип узла, оптимизированный для хранения данных, к которым обращаются редко или которые доступны только для чтения. Узлы замороженных данных предназначены для эффективного и малозатратного хранения больших

объемов данных, которые не запрашиваются и не обновляются. Такие узлы особенно хорошо подходят для долгосрочного архивирования и обеспечения соответствия нормативным требованиям, когда данные должны храниться в течение определенного времени, но доступ к ним осуществляется редко. Отделение замороженных данных от других типов, таких как горячие или теплые данные, позволяет оптимизировать использование ресурсов и повысить общую производительность кластера. Узлы замороженных данных позволяют хранить большие объемы данных и управлять ими с малыми затратами, обеспечивая при этом доступность данных и соответствие нормативным требованиям.

Чтобы настроить узел замороженных данных в Elasticsearch, измените параметр `node.roles` в файле конфигурации `elasticsearch.yml` следующим образом:

```
node.roles: [ data-frozen ]
```

Таким образом узел будет настроен для обработки замороженных данных и оптимизирован для их хранения и управления ими.

ПРИМЕЧАНИЕ Важно отметить, что узлы замороженных данных предназначены только для чтения, то есть данные в них можно записывать лишь на этапе начального индексирования. После того как данные будут проиндексированы и сохранены, их нельзя будет изменить или обновить. Однако они останутся доступными для поиска и извлечения, что делает узлы замороженных данных полезным средством для хранения и архивирования крупных данных в Elasticsearch.

Кластер

В Elasticsearch кластер — это набор узлов, которые взаимодействуют между собой, создавая целостную и распределенную среду для хранения и обработки данных. Каждый кластер идентифицируется уникальным именем, что позволяет узлам соединяться и взаимодействовать с конкретным кластером, к которому они принадлежат. Узлы внутри кластера работают вместе, обеспечивая единое и согласованное представление данных. Они взаимодействуют друг с другом для равномерного распределения и репликации данных на нескольких узлах; это позволяет повысить доступность данных, отказоустойчивость и общую производительность системы.

Распределяя данные по узлам, кластер обеспечивает горизонтальную масштабируемость. По мере увеличения объема данных или роста рабочей нагрузки в кластер могут быть добавлены дополнительные узлы, чтобы он мог справиться с возросшими требованиями к хранению и обработке данных. Такая масштабируемая архитектура позволяет Elasticsearch обрабатывать

большие массивы данных и эффективно справляться с большими объемами запросов.

Кластеры также играют важную роль в обеспечении надежности и отказоустойчивости данных. Благодаря репликации данных на нескольких узлах кластер может выдерживать сбои и предотвращать потерю данных. Если один из узлов становится недоступным или выходит из строя, кластер автоматически перераспределяет хранящиеся на нем данные на другие доступные узлы, поддерживая необходимую избыточность данных и гарантируя их доступность даже при сбое узлов.

Помимо распределения данных и обеспечения отказоустойчивости, кластеры предоставляют централизованную точку управления для мониторинга и администрирования. Такие операции, как мониторинг состояния кластера, управление индексами и ребалансировка данных, могут выполняться на уровне кластера; таким образом осуществляется единый и систематизированный подход к управлению всем развертыванием Elasticsearch.

Индекс

В Elasticsearch индекс представляет собой логическое пространство имен, или контейнер, в котором хранится коллекция документов. Индекс можно рассматривать как традиционную базу данных, где данные организованы и хранятся в структурированном виде. Задача индекса — сгруппировать документы, которые имеют схожие характеристики или принадлежат к одной категории. Например, в приложении онлайн-магазина можно задать отдельные индексы для товаров, клиентов и заказов. Это позволит выполнять эффективный поиск и извлекать необходимую информацию по определенным предметным областям (доменам).

Elasticsearch использует структуру инвертированного индекса для быстрого полнотекстового поиска. Инвертированный индекс состоит из списка уникальных терминов, встречающихся во всех документах в индексе, а также идентификаторов документов, указывающих на вхождения каждого термина. Подобная техника индексирования значительно ускоряет операции поиска за счет предварительного вычисления связей между терминами и документами.

Для обработки больших объемов данных и распределения рабочей нагрузки индекс делится на один или несколько шардов. Каждый шард — это независимое подмножество данных индекса, которое может храниться на отдельном узле в кластере Elasticsearch. Разбивая индекс на шарды, Elasticsearch может распараллелить операции поиска и индексирования, повышая производительность и масштабируемость.