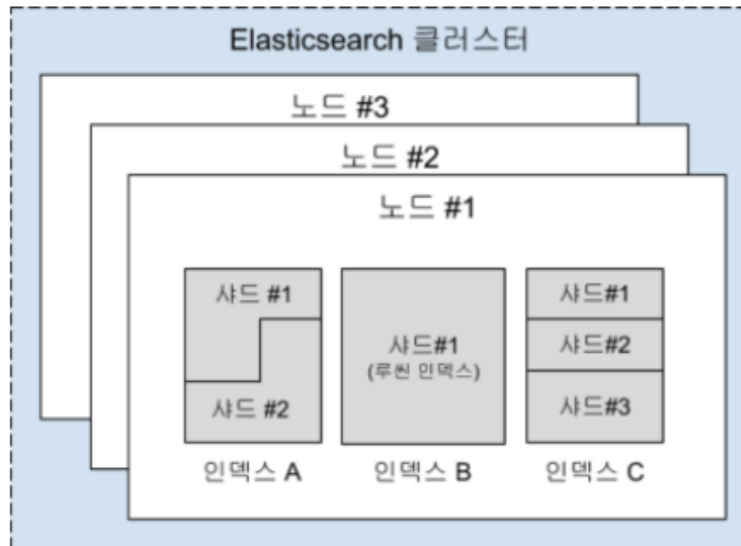


ElasticSearch 속제



- 샤드가 적을 때 발생하는 이슈

샤드 하나 당 큰 크기를 가지면(샤드 하나 당 큰 크기를 가지면 샤드가 많이 필요없어집니다. 따라서 이 경우 샤드가 적을 때로 봐도 무방합니다.) 장애시 클러스터 복구 능력에 좋지 않은 영향을 끼칩니다. 샤드 최대 크기에 대한 제한은 없습니다만, 실용사례를 보았을때, 50GB가 넘지 않는 것이 좋습니다.

- 샤드가 많을 때 발생하는 이슈

작은 샤드는(샤드 하나 당 작은 크기를 가지면 샤드가 많이 필요합니다. 따라서 이 경우 샤드가 많을 때로 봐도 무방합니다.) 작은 세그먼트를 만들며 부하를 증가시킵니다. 평균 샤드 크기를 최소한 수 GB와 수십 GB 사이를 유지하세요. 시간 기반 데이터를 사용한 과거 사례를 보면, 20GB ~ 40GB 정도의 사이즈가 적당합니다.

클러스터에 존재하는 모든 샤드는 마스터 노드에서 관리합니다. 그러므로 샤드가 많아질수록 마스터노드의 부하도 덩달아서 증가합니다. 샤드의 개수가 많아질수록 관리해야하는 정보의 양도 많아지기 때문입니다. 마스터 노드가 처리해야하는 정보가 많아지면 검색이나 색인이 덩달아 느려질 수 있습니다. 또한 마스터 노드의 메모리 사용량도 그에 비례해서 늘어나게 됩니다. 마스터 노드는 빠른 처리를 위해 샤드 정보와 같은 관리 데이터를 모두 메모리에 올려서 제공하기 때문입니다.

그러므로 너무 많은 샤드로 인해 마스터노드의 메모리가 부족해지지 않도록 주의해야하고 만약 마스터 노드에 장애가 발생한다면 클러스터 전체가 마비되는 대형사고로 번질 수 있습니다.

해결방법

샤드의 부하는 세그먼트의 개수와 크기에 따라 결정됩니다. **forcemerge** 기능을 사용하여 작은 세그먼트를 큰 세그먼트로 병합시키세요. 이 작업은 이상적으로 인덱스에 더 이상 데이터가 입력되지 않을 때 실행되어야 합니다. 그리고 무척 부하가 큰 작업이니 피크 시간을 피하여 수행해야 하는 것을 명심하세요.

- 샤드가 적은게 좋은건지 많은게 좋은건지 혹은 적은게 나쁜건지 많은게 나쁜건지

쿼리 성능 관점에서 최대 샤드 크기를 결정하는 최고의 방법은 실제 데이터와 쿼리로 벤치마크 테스트를 해보는 것입니다. 쿼리 하나로 최적화 작업을 하면 왜곡된 결과가 발생할 수 있으니, 항상 운영 환경에서 실제로 처리해야 하는 쿼리와 인덱싱 부하를 고려하여 테스트하기 바랍니다.

위 구문에서 확인하는 바와 같이 샤드는 적은게 좋은지 많은게 좋은지는 실제 운영 환경에 따라 달라집니다.

샤드가 지나치게 적으면 장애 발생시 복구 오류가 생기게 됩니다. 반면, 샤드가 지나치게 많으면 부하가 증가되어 메모리 부족 현상이 발생합니다. 만약 마스터 노드에 장애가 발생된다면 클러스터 전체가 마비되는 대형사고가 발생할 수 있습니다. 따라서 **샤드는 운영 환경에 맞춰 “적당한 개수” 그리고 “적당한 크기”를 가지는 것이 중요합니다.** 굳이 샤드가 적은 것과 샤드가 많은 것 중 더 나쁜 것 하나를 선택한다면, 대형사고가 날 수 있는 샤드가 많은 것이 더 나쁠 거 같습니다.

- 인덱스에 100기가 데이터가 들어있을때 샤드는 몇 개를 만들어야 할까?

만약 하나의 샤드가 20GB 정도의 크기를 가지는 것이 적절하다고 가정한다면 샤드의 수는 5개면 충분합니다. 하지만, 미래에 추가될 데이터까지 감안한다면 이를 포함해서 샤드의 수를 적절히 계산해서 생성해야 합니다.

- A인덱스가 10기가라서 샤드를 1개 만들었는데 그 A인덱스에 데이터가 계속 쌓여서 100기가가 되면 어떻게 해야 하는지

만약 샤드의 개수를 변경시키려면?

현재로서는 프라이머리 샤드의 개수를 변경할 방법이 따로 없기 때문에 엘라스틱서치에서는 프라이머리 샤드의 개수를 변경해야할 경우 새로운 인덱스를 생성하고 재색인하도록 안내하고 있습니다. 엘라스틱서치에는 이미 존재하는 인덱스를 새로운 인덱스로 다시 색인하는 Reindex을 사용해야 합니다.

- 인덱스 몇 개까지 만들 수 있는 지 혹은 인덱스 하나 당 허용 용량은 얼마인지?

인덱스는 그 자체로 만들 수 있는 개수의 제한이 없습니다. 하지만 샤드는 개수의 제한이 있어요.

하나의 샤드에서 색인할 수 있는 문서 수가 대략적으로 20억 개 정도이고 인덱스는 최대 1024개까지의 샤드를 가질 수 있기 때문에 이론적으로 엘라스틱서치에서 생성한 개별 인덱스가 가질 수 있는 최대 문서 수는 약 2조 개 라고 생각하면 됩니다.



(루씬에서 생성 가능한 최대 문서 수) * (인덱스 생성 시 설정 가능한 샤드 수) = 20억*1024 = 2조

하지만 이는 이론상의 숫자일 뿐이므로 데이터의 물리적인 크기와 하드웨어 리소스를 기반으로 적절한 문서 수를 계산해서 유지하는 것이 좋습니다.

- 도큐먼트 몇 개까지 만들 수 있는 지 혹은 도큐먼트 하나 당 허용 용량은 얼마인지?

각 Elasticsearch 샤드는 Lucene 색인입니다. 단일 Lucene 색인이 포함할 수 있는 문서 수의 최대 한도가 있습니다. [LUCENE-5843](#)에 따르면 2,147,483,519개(= Integer.MAX_VALUE - 128)입니다. [{ref}/cat-shards.html\[_cat/shards\]](#) API를 사용하여 샤드 크기를 모니터링할 수 있습니다. 예를 들자면, 만약 샤드가 5개라면 5*2,147,483,519개의 문서를 만들 수 있습니다.

Elasticsearch가 지원하는 기본 최대 문서 크기(문서)는 최대 100MB이지만 이 최대값을 Lucene의 제한인 2GB까지 늘릴 수 있습니다. 그러나 매우 큰 문서는 종종 문제를 유발하므로 이 점을 명심하는 것이 중요합니다.

- 샤드는 인덱스 당 몇 개까지 만들 수 있는 지 혹은 샤드의 적정 용량은 얼마인지?

개별 인덱스를 생성할 때 설정 가능한 샤드의 수는 현재 1024개로 제한되어 있습니다. 개별 인덱스를 생성할 때 1024 이상의 값을 설정할 경우 오류가 발생합니다.

샤드크기는 수GB에서 수십GB 사이가 적당하며, 경험상 시계열 데이터의 경우 20GB~40GB가 적당하다고 합니다.

- 시스템 하나당 만들 수 있는 샤드의 개수는?

하나의 노드에 저장할 수 있는 샤드의 개수는 가용한 힙의 크기와 비례하지만, Elasticsearch에서 그 크기를 제한하고 있지는 않습니다.

- 힙당 샤드의 개수는?

만약 힙메모리 30GB로 엘라스틱서치 구동시 샤드는 최대 600개 정도라고 합니다. 32GB를 힙에 할당하고 활성화된 인덱스가 50개라고 한다면, 인덱스당 600샤드 / 50인덱스로 인덱스당 최대 12개의 샤드를 설정할 수 있습니다. 하지만 이보다는 적게 유지하는 것이 더 좋습니다. 일반적으로 적게 유지하는 구성이 클러스터를 건강하게 유지하는데 도움이 됩니다.

- TF-IDF와 BM25의 차이

TF/IDF

루씬의 TF/IDF 알고리즘은 아래와 같습니다.

$$score_{q,d} = norm(q) \times \sum_{t \text{ in } q} \sqrt{tf_{t,d}} \times idf_t^2 \times norm(d,field) \times boost(t)$$

그림. 루씬 TF/IDF 수식(출처: [Real-time Analytics & Anomaly detection using Hadoop.. Elasticsearch and Storm](#))

TF(Term Frequency): 단어 빈도

텀(term)이 문서에 등장하는 횟수로, 많이 등장할수록 문서의 검색 점수는 당연히 높아집니다.

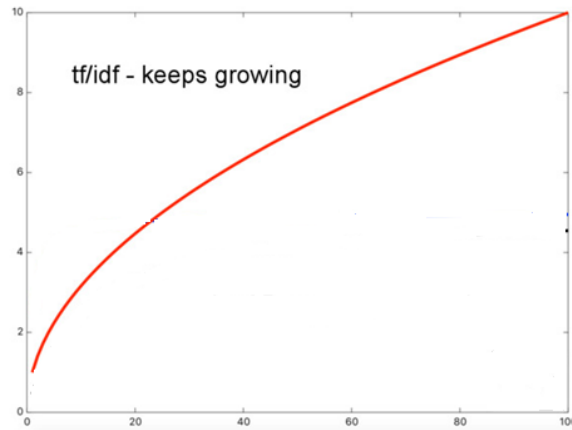


그림. TF가 높아질수록 점수가 높아진다(출처: [Elasticsearch, Improved Text Scoring with BM25](#))

IDF(Inverse Document Frequency): 문서 역빈도

문서 빈도(DF: Document Frequency)는 텀(term)이 등장하는 문서 개수로, 여러 문서에 등장할수록 문서의 검색 점수가 낮아집니다. IDF는 $1/DF$ 입니다.

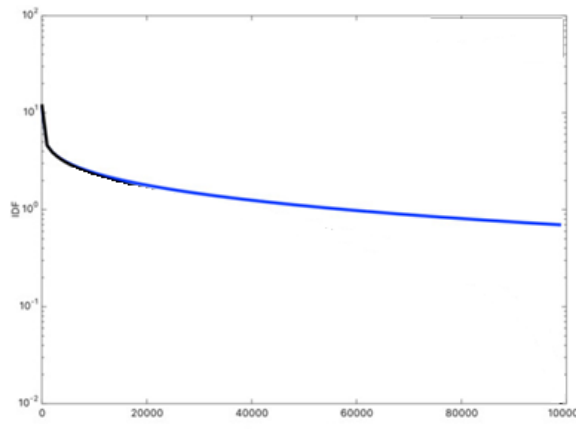


그림. DF가 높아질수록 점수가 낮아진다(출처: [Elasticsearch, Improved Text Scoring with BM25](#))

IDF는 "the", "a", "is"와 같이 문장에서 거의 항상 등장하는 텀(term)은 검색 결과에 영향을 미치지 않게 하기 위한 것으로, 이러한 단어들은 불용어(stop word)라고 부릅니다.

norm: 문서 길이 가중치

텀(term)이 등장하는 문서의 길이에 대한 가중치로, 문서 길이가 길수록 검색 점수가 낮아집니다.

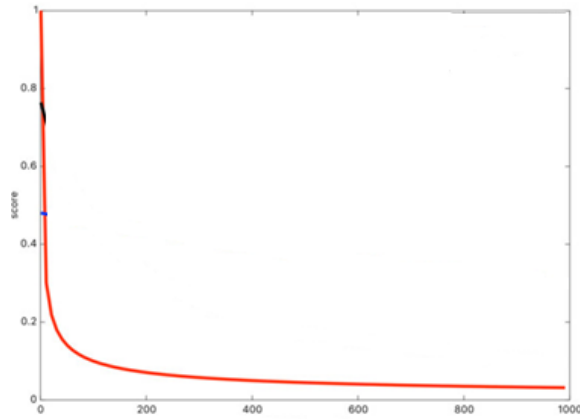


그림. 문서 길이가 길수록 점수가 낮아진다(출처: [Elasticsearch, Improved Text Scoring with BM25](#))

"철수의 취미는 축구다"와 "영희의 취미는 축구와 야구다"와 같은 문장이 있을 때 "축구"로 검색한 경우, "철수의 취미는 축구다"라는 문서의 검색 점수가 더 높습니다. 직관적으로 보더라도, 여가생활에 철수는 "축구"만 하고, 영희는 "축구"와 "야구"를 섞어서 할 것이므로, 철수는 영희보다 "축구"를 더 좋아할 가능성이 높을 것입니다.

BM25

BM25를의 검색 점수를 계산하는 수식은 다음과 같습니다.

$$bm25(d) = \sum_{t \in q, f_{t,d} > 0} \log \left(1 + \frac{N - df_t + 0.5}{df_t + 0.5} \right) \cdot \frac{f_{t,d}}{f_{t,d} + k \cdot (1 - b + b \frac{l(d)}{avgdl})}$$

그림. BM25 수식(출처: [Elasticsearch, Improved Text Scoring with BM25](#))

$$score_{q,d} = norm(q) \times \sum_{t \text{ in } q} \sqrt{tf_{t,d}} \times idf_t^2 \times norm(d, field) \times boost(t)$$

$$bm25(d) = \sum_{t \in q, f_{t,d} > 0} \log \left(1 + \frac{N - df_t + 0.5}{df_t + 0.5} \right) \cdot \frac{f_{t,d}}{f_{t,d} + k \cdot (1 - b + b \frac{l(d)}{avgdl})}$$

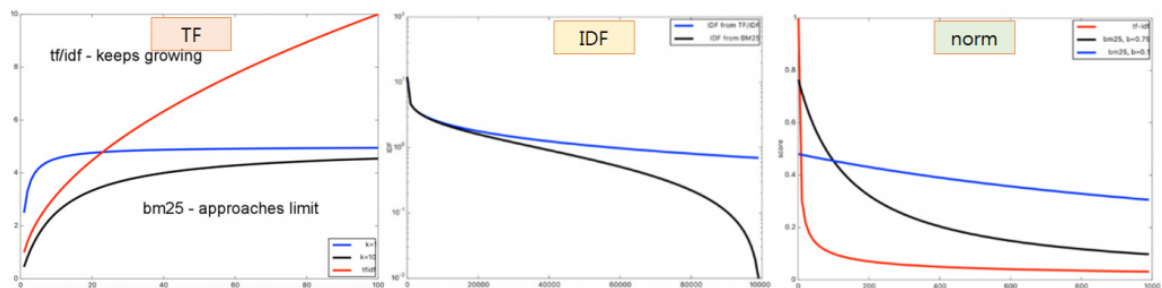


그림. 항목별 TF/IDF와 BM25 비교(출처: [Elasticsearch, Improved Text Scoring with BM25](#))

TF(Term Frequency): 단어 빈도

TF의 영향이 줄어듭니다.

TF에서는 단어 빈도가 높아질수록 검색 점수도 지속적으로 높아지는 반면, BM25에서는 특정 값으로 수렴합니다.

IDF(Inverse Document Frequency): 문서 역빈도

IDF의 영향이 커집니다.

BM25에서는 DF가 높아지면 검색 점수가 0으로 급격히 수렴하므로, 불용어가 검색 점수에 영향을 덜 미칩니다.

norm: 문서 길이 가중치

문서 길이의 영향이 줄어듭니다.

BM25에서는 문서의 평균 길이($avgdl$)를 계산에 사용하며, 문서의 길이가 검색 점수에 영향을 덜 미칩니다.

보통은 BM25가 루씬의 TF/IDF 알고리즘보다 더 좋은 성능을 보유하고 있습니다.

(근거 : [Elasticsearch, Improved Text Scoring with BM25](#))


- **n-gram, edge-gram 개념**

n-gram은 간단하게 말해서 입력한 문자열을 n개의 기준 단위로 절단하는 방법을 말합니다. 단어 개수에 따라 부르는 명칭이 다른데 한글자를 묶어서 사용하면 unigram 2개의 단어를 묶어서 사용하면 bigram 3개면 trigram이라고 부릅니다. n-gram을 토큰필터를 사용하면 저장되는 텀의 갯수도 기하급수적으로 늘어나고 검색 결과를 예상하기 어렵기 때문에 일반적인 텍스트 검색에는 사용하지 않는 것이 좋습니다. ngram을 사용하기 적합한 사례는 카테고리 목록이나 태그목록과 같이 전체 개수가 많지 않은 데이터 집단에 자동완성 같은 기능을 구현하는데에 적합합니다.

edge-gram은 텀 앞쪽의 ngram만 저장하기 위한 토큰 필터입니다. 설정방법은 "type": "edgeNGram"입니다. ngram과 마찬가지로 min_gram과 max_gram을 설정할 수 있습니다.

How many shards should I have in my Elasticsearch cluster?

Elasticsearch는 다양한 유스케이스를 지원하는 매우 다재다능한 플랫폼이며, 데이터 구성 및 복제 전략에 따라 큰 유연성을 제공합니다. 그러나 이러한 유연성은 여러분 데이터의 효율적인 인덱스와 샤드 구성을 복잡하게 만듭니다. 특히, 여러분이 Elastic


 <https://www.elastic.co/kr/blog/how-many-shards-should-i-have-in-my-elasticsearch-cluster>



<https://rudecamel.tistory.com/17>

기본 개념 | Elasticsearch 설명서 [5.4] | Elastic

Elasticsearch에서는 몇 가지 핵심 개념을 사용합니다. 처음부터 이 개념을 알아두면 훨씬 더 수월하게 학습할 수 있습니다. Elasticsearch는 NRT 검색 플랫폼입니다. 즉 문서를 색인화하는 시점부터 문서가 검색 가능해지는 시점까지 약간의 대기 시간(대개 1초)이 있습니다. 클러스터는 하나 이상의 노드(서버)가 모인 것이며, 이를 통해 전체 데이터를 저장하고 모든 노드를 포괄하는 통합 색인화 및


 <https://www.elastic.co/guide/kr/elasticsearch/reference/current/gs-basic-concepts.html>

<https://danawalab.github.io/elasticsearch/2020/07/21/Elasticsearch-Index-Shard-How.html>

<https://www.popit.kr/bm25-elasticsearch-5-0%EC%97%90%EC%84%9C-%EA%B2%80%EC%83%89%ED%95%98%EB%8A%94-%EC%83%88%EB%A1%9C%EC%9A%B4-%EB%B0%A9%EB%B2%95/>

Is there a limit on the number of indexes that can be created on Elastic Search?

Thanks for contributing an answer to Stack Overflow! Please be sure to answer the question. Provide details and share your research! Asking for help, clarification, or responding to other answers. Making statements based on opinion; back them up with

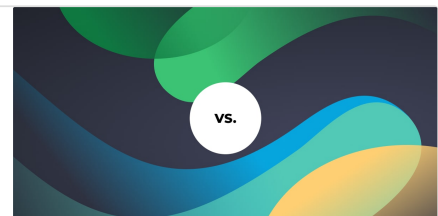
 <https://stackoverflow.com/questions/39393809/is-there-a-limit-on-the-number-of-indexes-that-can-be-created-on-elastic-search/39396505>



Open Source Comparison: Elasticsearch vs. MongoDB | Logz.io

Elasticsearch and MongoDB are the two most popular distributed datastores used to manage NoSQL data. Both of these technologies are highly scalable and have document-oriented design at the core. There are differences between

<https://logz.io/blog/elasticsearch-vs-mongodb/>



<https://brunch.co.kr/@alden/39>