
Hanbit eBook
Realtime 59
실무 예제로 배우는
Elasticsearch
검색엔진
기본편
정호욱 지음
한빛미디어
Hanbit Media, Inc.

실무 예제로 배우는

Elasticsearch 검색엔진

기본편

실무 예제로 배우는 Elasticsearch 검색엔진 기본편

초판발행 2014년 3월 25일

지은이 정호욱 / **펴낸이** 김태헌
펴낸곳 한빛미디어(주) / **주소** 서울시 마포구 양화로 7길 83 한빛미디어(주) IT출판부
전화 02-325-5544 / **팩스** 02-336-7124
등록 1999년 6월 24일 제10-1779호
ISBN 978-89-6848-691-3 15000 / **정가** 11,000원

책임편집 배용석 / **기획·편집** 정지연
디자인 표지 여동일, 내지 스튜디오 [밈], 조판 최송실
영업 김형진, 김진불, 조유미 / **마케팅** 박상용, 서은옥, 김옥현

이 책에 대한 의견이나 오탈자 및 잘못된 내용에 대한 수정 정보는 한빛미디어(주)의 홈페이지나 아래 이메일로 알려주십시오.
한빛미디어 홈페이지 www.hanbit.co.kr / **이메일** ask@hanbit.co.kr

Published by HANBIT Media, Inc. Printed in Korea

지금 하지 않으면 할 수 없는 일이 있습니다.
책으로 펴내고 싶은 아이디어나 원고를 메일(ebookwriter@hanbit.co.kr)로 보내주세요.
한빛미디어(주)는 여러분의 소중한 경험과 지식을 기다리고 있습니다.

저자 소개

지은이_**정호욱**

지난 13년 동안 야후코리아, NHN Technology, 삼성전자에서 커뮤니티, 소셜 검색, 광고 검색 관련 서비스를 개발해 오면서 검색엔진을 활용한 다양한 프로젝트를 수행하였다. 현재 빅 데이터 전문 기업인 그루터Gruter에서 오픈 소스 기반 검색엔진 개발자로 근무하고 있다. elasticsearch 기술에 대한 정보와 경험을 현재 개인 블로그(http://jjeong.tistory.com)를 통해 공유하고 있다.

저자 서문

검색엔진은 모든 서비스의 기본이 되는 핵심 요소입니다. 우리가 사용하는 모든 서비스에는 검색 기능이 포함되어 있습니다. 하지만 검색엔진 관련 기술은 일반 사용자가 접근하기에는 너무 어려운 기술로 남아 있습니다. 루씬Lucene이라는 오픈 소스 검색라이브러리가 진입 장벽을 많이 낮추기는 했지만, 서비스에 적용하기에는 개발자가 직접 구현해야 하는 기능이 너무 많고 관리와 유지보수가 어렵다는 문제가 있었습니다.

하지만 이런 문제점은 elasticsearch라는 오픈 소스 검색엔진이 나오면서 사라졌고 전문적인 검색엔진 및 서비스 개발자가 아니더라도 누구나 쉽게 검색 서비스를 만들 수 있게 되었습니다.

비싼 라이선스 비용을 내고 검색 품질과 기능을 커스터마이징하기 어려운 벤더 중심의 검색엔진을 사용하고 있다면 elasticsearch로 꼭 바꾸길 추천합니다. 아직 국내에는 elasticsearch 사용자층이 높지 않습니다. 이 책은 elasticsearch에 관심은 있으나 어디서부터 시작해야 할지 모르는 사용자와 검색을 모르는 사용자가 쉽게 서비스를 만들 수 있도록 도움을 주고자 집필하였습니다.

끝으로 이 책을 집필하는 데 많은 도움을 주신 그루터 권영길 대표님 그리고 이 책이 세상에 빛을 볼 수 있도록 많은 도움을 주신 한빛미디어 김창수 님, 정지연 님, 이중민 님께 감사의 말을 전합니다.

집필을 마치며

저자 정호욱

대상 독자 및 참고사항

초급 | 초중급 | 중급 | 중고급 | 고급

이 책은 elasticsearch 1.0.0을 이용해서 검색엔진을 구축하고 색인, 검색 기능을 구현하는 방법을 소개하는 책입니다. 기본 예제는 일반 쇼핑몰 상품 검색 기능을 구현할 수 있는 수준으로 작성되어 있습니다.

Eclipse로 Maven 프로젝트를 생성하여 JUnit 기반의 테스트 코드를 작성해 본 개발자라면 누구나 쉽게 읽을 수 있습니다. 또한, 실무 중심으로 예제를 구성하였으므로 이를 검색 서비스 개발에 응용할 수 있습니다.

이 책의 검색엔진 설치와 예제 코드 실행을 위해서는 linux 계열의 OS와 Java 6 이상이 설치된 개발 환경이 갖춰져 있어야 합니다.

이 책의 예제 코드와 프로젝트는 아래 링크에서 받을 수 있습니다.

- https://github.com/HowookJeong/hanb-elasticsearch-beginner

한빛 eBook 리얼타임

한빛 eBook 리얼타임은 IT 개발자를 위한 eBook입니다.

요즘 IT 업계에는 하루가 멀다 하고 수많은 기술이 나타나고 사라져 갑니다. 인터넷을 아무리 뒤져도 조금이나마 정리된 정보를 찾는 것도 쉽지 않습니다. 또한 잘 정리되어 책으로 나오기까지는 오랜 시간이 걸립니다. 어떻게 하면 조금이라도 더 유용한 정보를 빠르게 얻을 수 있을까요? 어떻게 하면 남보다 조금 더 빨리 경험하고 습득한 지식을 공유하고 발전시켜 나갈 수 있을까요? 세상에는 수많은 종이책이 있습니다. 그리고 그 종이책을 그대로 옮긴 전자책도 많습니다. 전자책에는 전자책에 적합한 콘텐츠와 전자책의 특성을 살린 형식이 있다고 생각합니다.

한빛이 지금 생각하고 추구하는, 개발자를 위한 리얼타임 전자책은 이렇습니다.

1. **eBook Only - 빠르게 변화하는 IT 기술에 대해 핵심적인 정보를 신속하게 제공합니다.**

   500페이지 가까운 분량의 잘 정리된 도서(종이책)가 아니라, 핵심적인 내용을 빠르게 전달하기 위해 조금은 거칠지만 100페이지 내외의 전자책 전용으로 개발한 서비스입니다. 독자에게는 새로운 정보를 빨리 얻을 수 있는 기회가 되고, 자신이 먼저 경험한 지식과 정보를 책으로 펴내고 싶지만 너무 바빠서 엄두를 못 내는 선배, 전문가, 고수 분에게는 보다 쉽게 집필할 수 있는 기회가 될 수 있으리라 생각합니다. 또한 새로운 정보와 지식을 빠르게 전달하기 위해 O'Reilly의 전자책 번역 서비스도 하고 있습니다.

2. **무료로 업데이트되는, 전자책 전용 서비스입니다.**

   종이책으로는 기술의 변화 속도를 따라잡기가 쉽지 않습니다. 책이 일정 분량 이상으로 집필되고 정리되어 나오는 동안 기술은 이미 변해 있습니다. 전자책으로 출간된 이후에도 버전 업을 통해 중요한 기술적 변화가 있거나 저자(역자)와 독자가 소통하면서 보완하여 발전된 노하우가 정리되면 구매하신 분께 무료로 업데이트해 드립니다.

3. 독자의 편의를 위해 DRM-Free로 제공합니다.

구매한 전자책을 다양한 IT 기기에서 자유롭게 활용할 수 있도록 DRM-Free PDF 포맷으로 제공합니다. 이는 독자 여러분과 한빛이 생각하고 추구하는 전자책을 만들어 나가기 위해 독자 여러분이 언제 어디서 어떤 기기를 사용하더라도 편리하게 전자책을 볼 수 있도록 하기 위함입니다.

4. 전자책 환경을 고려한 최적의 형태와 디자인에 담고자 노력했습니다.

종이책을 그대로 옮겨 놓아 가독성이 떨어지고 읽기 힘든 전자책이 아니라, 전자책의 환경에 가능한 한 최적화하여 쾌적한 경험을 드리고자 합니다. 링크 등의 기능을 적극적으로 이용할 수 있음은 물론이고 글자 크기나 행간, 여백 등을 전자책에 가장 최적화된 형태로 새롭게 디자인하였습니다.

앞으로도 독자 여러분의 충고에 귀 기울이며 지속해서 발전시켜 나가도록 하겠습니다.

차례

1 | Elasticsearch 시작하기

지금까지 검색엔진 개발은 매우 어려운 기술 분야로 인식되어 일부 국한된 개발자의 영역으로 자리하였다. 국내는 대형 포털 등을 제외하고는 일부 벤더 중심의 검색 솔루션이 활성화되어 있지만, 외국은 오픈 소스 기반의 검색엔진을 많이 사용하는 추세다.

최근 국내에서도 이런 오픈 소스 검색엔진을 기반으로 검색 서비스를 전환하거나 구축하려는 곳이 많아지고 있다. 이것은 오픈 소스 검색엔진인 elasticsearch와 solr가 검색 서비스 시장의 많은 부분에 사용되고 있기 때문이다. 특히 elasticsearch는 쉬운 설치와 우수한 성능 그리고 빅 데이터에 대한 실시간 검색이 가능하다는 점에서 주목받고 있다.

오픈 소스 검색엔진 덕분에 이제는 검색엔진 또는 서비스 개발이 누구나 가능하게 되었다. 검색 서비스를 사용해 본 사람이라면 elasticsearch나 solr를 이용해 쉽고 빠르게 서비스를 구축할 수 있다.

이 책에서는 elasticsearch를 이용한 검색엔진 구성과 설정, 색인 그리고 검색까지 모든 기초 과정이 포함되어 있다. 책에 나온 기본 쇼핑몰 예제를 따라 해보면서 검색의 기본 기능을 이해하는 데 도움이 되길 바란다.

1.1 Elasticsearch란?

셰이 배논Shay Bannon이 시작한 오픈 소스 검색 서버 프로젝트로, JSON 기반의 비정형 데이터 분산 검색과 분석을 지원한다. 이 검색엔진은 실시간 검색 서비스 지원과 분산 및 병렬 처리 그리고 멀티테넌시Multitenancy 기능을 제공하며, 다양한 기능을 플러그인Plugin 형태로 구현하여 적용할 수 있는 것이 큰 특징이다.

또한, 아마존 웹 서비스AWS, Amazon Web Services의 클라우드 서비스와 빅 데이터 처리를 위한 하둡Hadoop 연동도 지원하고 있다. elasticsearch는 현재 웹 문서 검색, 소셜 데이터 분석, 쇼핑몰 검색 등 다양한 서비스에서 사용되고 있으며, 앞으로도 중·소규모의 데이터부터 빅 데이터까지 광범위한 검색과 분석 서비스에 활용될 것이다.

1.2 Elasticsearch의 특징

1.2.1 실시간 검색 및 분석 서비스 지원

실시간으로 발생하는 데이터를 기반으로 검색 질의 시 결과에 반영하거나 분석을 통한 결과를 실시간으로 제공할 수 있다.

1.2.2 분산 및 병렬 처리

데이터의 분산과 병렬 처리가 되므로 실시간 검색 및 분석을 할 수 있고, SPOFSingle Point of Failure 대응을 위한 높은 가용성을 제공한다.

1.2.3 멀티테넌시

하나의 클러스터 내에서 indice와 도큐먼트 타입Document Type을 활용하여 멀티 클라이언트 구성 및 서비스를 할 수 있다. 예를 들어 shopping_mall이라는 indice에 인터넷 쇼핑몰들을 도큐먼트 타입으로 분리하여 생성하거나 인터넷 쇼핑몰들을 indice 별로 분리하여 shopping_mall이라는 별칭Alias을 생성할 수 있다.

1.2.4 플러그인 형태 구현

검색엔진을 직접 수정하지 않고 필요한 기능에 대한 플러그인을 적용하여 기능을 확장할 수 있다. 예를 들어 외부에서 제공하는 형태소 분석기나 추가적인 REST API를 구현하여 적용할 수 있다.

1.2.5 기타

그 밖의 특징으로는 NoSQL과 같은 스키마Schema free, JSON 기반의 문서 구조, 버전 관리를 통한 충돌Conflict 관리가 가능하며, 전문검색Full Text Search도 지원한다.

2 | Elasticsearch 설치 및 구성하기

이 장에서는 elasticsearch에서 자주 사용되는 용어들을 확인하고, elasticsearch를 설치한 후 단일 노드와 클러스터 환경으로 설정하는 과정을 살펴본다.

2.1 Elasticsearch 주요 용어

elasticsearch에서 자주 사용되거나 언급되는 용어들로, 검색엔진을 이해하는 데 기초가 되는 표현들이므로 확인하고 넘어가자.

2.1.1 Index

인덱스는 elasticsearch에서 데이터를 저장하기 위한 장소로, RDBMS의 데이터베이스와 유사하다. index는 하나 또는 여러 개의 도큐먼트 타입을 가질 수 있다. 실제 소스 코드나 참조 문서에는 indice라는 용어가 사용되는데 index는 검색에서 포괄적인 의미의 색인 또는 색인 파일이고, indice는 elasticsearch 내에서 물리적으로 사용되는 색인 또는 색인 파일이라고 보면 된다. 기존 검색엔진의 collection과 같은 의미가 indice다.

2.1.2 Shard

샤드는 루씬Lucene을 기준으로 검색의 기본 데이터베이스가 되는 인덱스이며, 대량의 데이터를 분산 처리하기 위한 개념으로 큰 크기의 인덱스를 여러 개의 작은 인덱스로 나누어 저장하는 것을 말한다. 샤드는 대량의 데이터를 단일 노드에 저장 시 저장소 및 성능에 대한 한계를 해결하고, 대량의 데이터를 분산 처리하여 빠르게 결과를 만들 수 있게 한다.

- Primary Shard: 색인 시 가장 먼저 생성되는 인덱스로, 복제의 기본 소스가

된다.

- Replica Shard: 레플리카 설정에 따라 primary shard를 복제하여 생성된 샤드를 말한다.

2.1.3 Replica

레플리카는 서비스 장애 발생 시 서비스의 지속성 보장과 검색 처리량을 높이는 데 유용한 방법이다. 레플리카는 분산된 다른 노드에 샤드와 같은 데이터를 복제하여 서비스의 안정성 및 유연성을 제공한다.

기본적으로 primary shard에 색인이 완료되면 이를 바탕으로 각 노드에 샤드 복제가 async하게 이루어진다. async 방식으로 복제가 이루어지기 때문에 서비스 진행 중 색인 작업이 이루어지더라도 검색 성능 저하를 최소화한다.

2.1.4 Document type

도큐먼트 타입은 물리적인 인덱스나 저장소를 가지고 있지 않다. 다만 논리적으로 단일 인덱스에 대한 서로 다른 목적의 데이터를 구분하여 저장하는 방법으로 사용된다. 데이터베이스 관점에서 보면 테이블과 유사하며, 내장 필드인 _type에 따라 저장된다.

2.1.5 Document

검색에서 가장 기본이 되는 데이터 단위로, elasticsearch에 저장되는 하나의 item 또는 article을 말한다. 도큐먼트는 RDBMS에서 테이블 내 하나의 row에 해당한다.

- 도큐먼트의 필드Field는 RDBMS에서 테이블의 column에 해당한다.

2.1.6 Node

노드는 elasticsearch를 구성하는 하나의 서버 또는 데몬으로, 독립적으로 동작 가능한 서버를 말한다.

2.1.7 Cluster

클러스터는 standalone으로 동작하는 여러 노드를 하나의 그룹으로 묶어서 데이터의 분산과 공유를 할 수 있도록 서비스를 구성하는 것을 말한다.

2.1.8 Elasticsearch와 RDBMS 용어 비교

다음 표는 elasticsearch의 주요 용어를 이해하기 쉽도록 RDBMS 용어와 비교하여 보여준다.

[표 2-1] Elasticsearch와 RDBMS 용어 비교

Elasticsearch	RDBMS
index	database
document type	table
document	row
field	column

2.2 Elasticsearch 설치하기

elasticsearch는 standalone과 클러스터 구성이 모두 가능하므로 두 가지 방법으로 설치 과정을 다루어 본다.

2.2.1 Download

오픈 소스 특성상 버전 업그레이드가 잦으므로 elasticsearch 사이트[01]를 방문하여 최신 버전과 릴리스 노트Release Note를 꼭 확인한다.

01 http://www.elasticsearch.org/download/

```
$ wget https://download.elasticsearch.org/elasticsearch/elasticsearch/
elasticsearch-1.0.0.tar.gz
```

2.2.2 디렉터리 구조

elasticsearch 설치 파일의 압축을 풀면 처음에는 bin, config, lib의 세 개 디렉터리만 존재하고, 나머지 디렉터리는 실행할 때 생성된다.

[표 2-2] Elasticsearch 디렉터리 구조

디렉터리	설명
bin	elasticsearch 실행에 필요한 스크립트와 플러그인 설치 스크립트가 있다.
config	elasticsearch.yml과 logger.yml 파일이 있다.
lib	검색엔진에서 사용하는 라이브러리가 있다.
data	별도 path를 지정하지 않으면 기본 index store의 위치가 된다.
logs	검색엔진에서 기록하는 로그파일의 위치다.
plugins	검색엔진에서 사용하는 모든 플러그인이 설치되는 위치다.
work	임시 파일 경로다.

2.2.3 실행

elasticsearch는 두 가지 방법으로 실행할 수 있다. 하나는 foreground 방법으로 이 방법을 실행하면 실행 로그가 화면에 찍히면서 올라간다. 다른 하나는 background 방법으로 실행 시 아무런 변화가 없다.

```
$ tar  - xvzf elasticsearch-1.0.0.tar.gz
$ cd elasticsearch-1.0.0
$ bin/elasticsearch -f
```

데몬 실행을 관리하기 위해 다음과 같은 옵션을 제공한다.

[표 2-3] 추가 실행 옵션

옵션	설명
-Des.config	elasticsearch.yml 파일을 지정한다.
-Des.pidfile	실행된 pid를 저장할 파일을 지정한다.
-Des.foreground	foreground로 실행할지 지정한다.
-Des.path.home	elasticsearch의 home 디렉터리를 지정한다.

2.2.4 실행 후 log 확인

백그라운드로 실행한 후 정상적으로 실행되었는지 로그 내용을 확인한다. 다음은 정상적으로 실행되었을 때 로그 메시지다.

```
$ tail -f logs/elasticsearch.log
[2014-02-14 12:37:27,074][INFO ][node                ] [standalone]
version[1.0.0], pid[2289], build[a46900e/2014-02-12T16:18:34Z]
[2014-02-14 12:37:27,075][INFO ][node                ] [standalone]
initializing ...
[2014-02-14 12:37:27,082][INFO ][plugins             ] [standalone] loaded [],
sites [bigdesk, browser, head, HQ, inquisitor]
[2014-02-14 12:37:29,563][INFO ][node                ] [standalone]
initialized
[2014-02-14 12:37:29,563][INFO ][node                ] [standalone] starting
...
[2014-02-14 12:37:29,661][INFO ][transport           ] [standalone]
bound_address {inet[/127.0.0.1:9300]}, publish_address {inet[localho
st/127.0.0.1:9300]}
[2014-02-14 12:37:32,720][INFO ][cluster.service     ] [standalone] new_master
[standalone][XeH7K7YTQBWLUfHvc1ZJ-A][jeong-ui-MacBook-Pro.local][inet[localho
st/127.0.0.1:9300]]{master=true}, reason: zen-disco-join (elected_as_master)
[2014-02-14 12:37:32,743][INFO ][discovery           ] [standalone]
elasticsearch/XeH7K7YTQBWLUfHvc1ZJ-A
[2014-02-14 12:37:32,762][INFO ][http                ] [standalone]
```

```
bound_address {inet[/127.0.0.1:9200]}, publish_address {inet[localho
st/127.0.0.1:9200]}
[2014-02-14 12:37:33,367][INFO ][gateway              ] [standalone] recovered
[2] indices into cluster_state
[2014-02-14 12:37:33,368][INFO ][node                 ] [standalone] started
```

2.2.5 실행 스크립트 구성하기

클러스터 구성으로 여러 대의 노드를 관리할 때 노드별로 실행하거나 중지하는 것은 매우 비효율적인 방법이다. 노드를 효율적으로 관리하려면 다음과 같은 실행 스크립트를 작성하여 사용한다.

[start.sh 작성]

```
#!/bin/bash

export ES_HEAP_SIZE=256m
export ES_HEAP_NEWSIZE=128m
export JAVA_OPT="-server -XX:+AggressiveOpts -XX:UseCompressedOops
-XX:MaxDirectMemorySize -XX:+UseParNewGC -XX:+UseConcMarkSweepGC
-XX:+CMSParallelRemarkEnabled -XX:CMSInitiatingOccupancyFraction=75 -XX:+UseC
MSInitiatingOccupancyOnly"

ES=/home/es/app/elasticsearch
$ES/bin/elasticsearch -Des.pidfile=$ES/bin/es.pid
-Des.config=$ES_NODE/config/elasticsearch.yml -Djava.net.preferIPv4Stack=true
-Des.max-open-files=true > /dev/null 2>&1 &
```

- $ES 변수는 실제 설치된 경로로 수정해야 한다.

[stop.sh 작성]

```
#!/bin/bash

ES=/home/es/app/elasticsearch
/bin/kill `cat < $ES/bin/es.pid`
```

- $ES 변수는 실제 설치된 경로로 수정해야 한다.

2.3 Elasticsearch Standalone 구성하기

이 절에서는 elasticsearch 데몬을 하나 구성하여 설정하는 과정을 살펴본다. 이를 바탕으로 클러스터 구성까지 확장하므로 이번 절의 기본 설정을 잘 이해하도록 한다.

2.3.1 Cluster 명 설정

이 설정은 standalone 구성 시 필요하지 않으나 관리적인 면에서 설정하는 것을 추천한다. 각 노드를 클러스터 명 기준으로 그루핑하여 서비스하므로, 이름이 다르거나 틀릴 경우 정상적으로 그루핑되지 않는다.

cluster.name은 elasticsearch 실행 시 data 디렉터리 밑에 같은 이름의 디렉터리를 생성한다. 이를 바탕으로 같은 클러스터의 노드들은 data 디렉터리의 클러스터 명 아래로 색인 데이터를 저장한다.

- cluster.name: cluster_standalone을 지정하지 않으면 elasticsearch로 자동 생성된다.

2.3.2 Node 명 설정

노드 명은 클러스터 구성 시 노드 구분을 위해 설정하며, 관리의 편리성과 개발의 직관성을 높여준다. 별도로 설정하지 않으면 내부적으로 자동 생성되지만 추천하지 않는다.

```
node.name: standalone
```

2.3.3 Node 역할 설정

각 노드는 마스터 노드Master Node와 데이터 노드Data Node, 로드 밸런서 노드Load Balancer Node, 클라이언트 노드Client Node의 네 가지 역할을 한다. 노드 구성에 대한 자세한 내용은 뒤에서 다루고[02], 여기에서는 각 노드의 역할 설정만 설명한다.

싱글 노드 구성을 위해 마스터 노드와 데이터 노드 설정을 모두 true로 한다.

```
node.master: true
node.data: true
```

- 마스터로 설정되면 클러스터와 노드에 대한 정보와 상태를 관리하고, 인덱스와 샤드에 대한 coordination을 수행한다. 데이터로 설정되면 자신의 노드에 색인 데이터를 저장할 수 있다.

2.3.4 Index Shard와 Index Replica 설정

샤드와 레플리카의 기본값은 각각 5와 1로 구성된다. 하지만 단일 구성에서는 복제 설정을 하더라도 할당할 노드가 없으므로 레플리카 설정은 0으로 한다.

```
index.number_of_shards: 5
index.number_of_replicas: 0
```

- 인덱스 샤드는 색인 저장소로 사용되는 인덱스를 물리적으로 작은 단위의 샤드에 분산해서 저장하기 위해 설정하며, 인덱스 레플리카는 장애가 발생하거나 샤드가 깨졌을 때 복구와 대응을 하기 위해 설정한다.

02 2.5 Elasticsearch node 구성의 이해 참고

2.3.5 Index 기타 설정

elasticsearch에는 기본 설정 외에도 다양한 설정 값이 있다. 기본 설정에 대한 정보를 제외하고, 대부분의 설정 정보는 자세히 나와 있지 않다. 자세한 내용을 알려면 소스코드를 보거나 elasticsearch에서 제공하는 웹사이트의 guide 문서[03]를 참고한다. 하지만 이런 상세한 설정을 하지 않더라도 기본 설정만으로 중소 규모 사이트는 구성할 수 있다.

[표 2-4] 인덱스 기타 설정

기본 설정 값	설명
index.mapper.dynamic: true	자동으로 필드 매핑을 설정한다.
index.refresh_interval: "1s"	색인 operation에 대한 실시간 검색 반영을 위한 주기 설정이다.
action.auto_create_index: true	자동 인덱스 생성에 대한 설정으로 패턴을 지원한다.
action.disable_shutdown: true	REST API를 이용하여 노드 shutdown 기능을 활성화한다.

2.3.6 Network 설정

네트워크 설정은 클라이언트와 서버 간 통신을 위해 프로토콜Protocol과 포트Port 정보를 구성한다. elasticsearch에서는 클라이언트 개발을 위한 접근이 쉽도록 대부분의 API를 REST 방식으로 지원하며, JSON 기반의 데이터형을 사용한다.

[표 2-4] 네트워크 기본 설정

기본 설정 값	설명
network.host: localhost	IP 정보로 설정한다.
transport.tcp.port: 9300	TCP 기본 포트로, 변경할 수 있다.
transport.tcp.compress: true	TCP 통신 시 데이터 압축을 설정한다.
http.port: 9200	http(REST API) 기본 포트로, 변경할 수 있다
http.enabled: true	http(REST API) 사용을 활성화한다.

03 http://www.elasticsearch.org/guide/

2.3.7 Gateway 설정

게이트웨이 모듈 설정에서는 클러스터의 메타 정보와 인덱스 설정, 매핑 정보 등을 어떻게 저장하고 운영할지 구성하게 된다. 이를 바탕으로 하나 또는 전체 노드를 재시작할 때 저장된 정보를 이용하여 서비스가 안전하게 운영되도록 해준다.

```
gateway.type: local
```

- 게이트웨이 타입Gateway Type은 색인 저장소 유형으로 사용하는 스토어 타입Store Type과는 다른 설정으로 local, shared fs, hadoop 그리고 s3의 4가지 유형을 제공한다. 현재 elasticsearch에서는 local gateway를 추천하고, 이외 타입들은 삭제될 예정이나 레거시 코드Legacy Code를 지원하기 위해 아직 남아 있다.

2.3.8 Discovery 설정

클러스터 내에서 노드 간 통신과 마스터 노드 관리를 설정하는 역할을 담당한다. standalone 구성에서는 별로 필요하지 않으나 클러스터 구성 때 매우 중요한 역할을 담당한다. 특히 네트워크 통신을 요구하는 클러스터 구성에서는 데이터 손실이 발생할 수 있기 때문에 이 설정이 더욱 중요하다.

```
discovery.zen.ping.multicast.enabled: false #같은 네트워크 구간에 있는 모든 노드와 discovery 통신을 한다.
discovery.zen.minimum_master_nodes: 2 #master 역할을 수행할 노드의 최소 단위를 지정한다.
discovery.zen.ping.timeout: 3s
discovery.zen.ping.unicast.hosts: ["localhost:9300","localhost:9301","localhost:9302"]
```

- multicast.enabled 설정은 같은 네트워크 구간에 있는 모든 노드와 불필요한 트래픽을 유발하므로 이를 방지하기 위해 true로 설정하는 것을 추천한다.
- minimum_master_nodes 설정은 master 역할을 수행할 최소 단위의 노드 크기를 지정하게 된다. 데이터 손실 및 서비스 안정성을 확보하기 위해 최소 2개 이상 설정하는 것을 추천한다.
- 그 외에는 서버 구성과 특성에 맞춰서 설정한다.

2.3.9 elasticsearch.yml 작성

이제 본격적인 설정 작업을 해 보자. attribute와 value 작성 시 콜론(:)과 띄어쓰기에 주의해야 한다.

- 속성 명과 콜론은 붙이고, 값은 한 칸 띄어서 쓴다.
  형식) ATTRIBUTE.NAME: VALUE
- 콜론 뒤에 띄어쓰기 없이 값을 입력하면 elasticsearch를 실행하여 설정 값을 로딩할 때 오류가 발생할 수 있다.

```
$ vi config/elasticsearch.yml
```

[elasticsearch.yml]

```
node.name: standalone
node.master: true
node.data: true

index.number_of_shards: 5
index.number_of_replicas: 0
index.mapper.dynamic: true
index.refresh_interval: "1s"
```

```
action.auto_create_index: true #패턴 형식의 인덱스 명 생성 판단 설정이다.
                               settings, mappings 정보 없이 바로 문서를 등록할
                               경우 자동으로 인덱스 생성 여부 결정한다.
action.disable_shutdown: true

network.host: localhost
transport.tcp.port: 9300
transport.tcp.compress: true
http.port: 9200
http.enabled: true

gateway.type: local
```

2.3.10 실행 및 확인

터미널을 빠져나온 후에도 elasticsearch 데몬을 계속 실행시키기 위해 백그라운드로 실행한다.

```
$ bin/elasticsearch -Des.pidfile=es.pid > /dev/null 2>&1 &
```

실행 후 정상적으로 elasticsearch 데몬이 동작하는지 REST API를 이용하여 노드 정보를 확인한다.

```
$ curl -XGET http://localhost:9200/_nodes?pretty=true
```

다음은 데몬이 정상적으로 실행되었을 때 결과값이다.

[실행 결과]

```
{
  "cluster_name" : "elasticsearch",
  "nodes" : {
    "eL_3STjkRvC0mzvmCyd4Kg" : {
    "name" : "Zzzax",
    "transport_address" : "inet[/192.168.0.117:9300]",
    "host" : "jeong-ui-MacBook-Pro.local",
    "ip" : "192.168.0.117",
    "version" : "1.0.0",
    "build" : "a46900e",
    "http_address" : "inet[/192.168.0.117:9200]",
    "settings" : {
      "path" : {
        "logs" : "/Users/hwjeong/server/app/elasticsearch/elasticsearch-
1.0.0/logs",
        "home" : "/Users/hwjeong/server/app/elasticsearch/
elasticsearch-1.0.0"
      },
      "cluster" : {
        "name" : "elasticsearch"
      },
      "foreground" : "yes",
      "name" : "Zzzax"
    },
    "os" : {
      "refresh_interval" : 1000,
      "available_processors" : 8,
      "cpu" : {
        "vendor" : "Intel",
        "model" : "MacBookPro10,1",
```

```
      "mhz" : 2400,
      "total_cores" : 8,
      "total_sockets" : 8,
      "cores_per_socket" : 16,
      "cache_size_in_bytes" : 256
    },
    "mem" : {
      "total_in_bytes" : 8589934592
    },
    "swap" : {
      "total_in_bytes" : 5368709120
    }
  },
  "process" : {
    "refresh_interval" : 1000,
    "id" : 2383,
    "max_file_descriptors" : 10240,
    "mlockall" : false
  },
  "jvm" : {
    "pid" : 2383,
    "version" : "1.6.0_65",
    "vm_name" : "Java HotSpot(TM) 64-Bit Server VM",
    "vm_version" : "20.65-b04-462",
    "vm_vendor" : "Apple Inc.",
    "start_time" : 1392354224036,
    "mem" : {
      "heap_init_in_bytes" : 268435456,
      "heap_max_in_bytes" : 1060372480,
      "non_heap_init_in_bytes" : 24317952,
      "non_heap_max_in_bytes" : 136314880,
      "direct_max_in_bytes" : 1060372480
```

```
        },
        "gc_collectors" : [ "ParNew", "ConcurrentMarkSweep" ],
        "memory_pools" : [ "Code Cache", "Par Eden Space", "Par Survivor
Space", "CMS Old Gen", "CMS Perm Gen" ]
      },
      "thread_pool" : {
         "generic" : {
            "type" : "cached",
            "keep_alive" : "30s"
         },
         "index" : {
            "type" : "fixed",
            "min" : 8,
            "max" : 8,
            "queue_size" : "200"
         },
         "get" : {
            "type" : "fixed",
            "min" : 8,
            "max" : 8,
            "queue_size" : "1k"
         },
         "snapshot" : {
            "type" : "scaling",
            "min" : 1,
            "max" : 4,
            "keep_alive" : "5m"
         },
         "merge" : {
            "type" : "scaling",
            "min" : 1,
            "max" : 4,
```

```
    "keep_alive" : "5m"
  },
  "suggest" : {
    "type" : "fixed",
    "min" : 8,
    "max" : 8,
    "queue_size" : "1k"
  },
  "bulk" : {
    "type" : "fixed",
    "min" : 8,
    "max" : 8,
    "queue_size" : "50"
  },
  "optimize" : {
    "type" : "fixed",
    "min" : 1,
    "max" : 1
  },
  "warmer" : {
    "type" : "scaling",
    "min" : 1,
    "max" : 4,
    "keep_alive" : "5m"
  },
  "flush" : {
    "type" : "scaling",
    "min" : 1,
    "max" : 4,
    "keep_alive" : "5m"
  },
  "search" : {
```

```
      "type" : "fixed",
      "min" : 24,
      "max" : 24,
      "queue_size" : "1k"
    },
    "percolate" : {
      "type" : "fixed",
      "min" : 8,
      "max" : 8,
      "queue_size" : "1k"
    },
      "management" : {
      "type" : "scaling",
      "min" : 1,
      "max" : 5,
      "keep_alive" : "5m"
    },
    "refresh" : {
      "type" : "scaling",
      "min" : 1,
      "max" : 4,
      "keep_alive" : "5m"
    }
  },
  "network" : {
    "refresh_interval" : 5000,
    "primary_interface" : {
      "address" : "192.168.0.117",
      "name" : "en0",
      "mac_address" : "28:CF:E9:14:C5:C9"
    }
  },
```

```
      "transport" : {
         "bound_address" : "inet[/0:0:0:0:0:0:0:0%0:9300]",
         "publish_address" : "inet[/192.168.0.117:9300]"
      },
      "http" : {
         "bound_address" : "inet[/0:0:0:0:0:0:0:0%0:9200]",
         "publish_address" : "inet[/192.168.0.117:9200]",
         "max_content_length_in_bytes" : 104857600
      },
      "plugins" : [ ]
   }
}
```

2.4 Elasticsearch Cluster 구성하기

이 절에서는 앞 절에서 구성한 데몬을 바탕으로 클러스터를 구성해 본다.

2.4.1 Prerequisite

단일 서버에 세 개의 노드를 하나의 클러스터로 구성한다. 같은 버전의 elasticsearch 폴더를 node1과 node2, node3으로 복사해서 준비한다.

[노드 생성]

```
$ cp -rf elasticsearch-1.0.0 node1
$ cp -rf elasticsearch-1.0.0 node2
$ cp -rf elasticsearch-1.0.0 node3
```

2.4.2 Configure

서버 1대에 elasticsearch instance를 세 개나 구성하므로 각 노드 설정 시 node.name과 포트 설정에 주의한다.

[node1에 대한 elasticsearch.yml 설정]

```
cluster.name: cluster_node

node.name: node1
node.master: true
node.data: true

index.number_of_shards: 5
index.number_of_replicas: 0
index.mapper.dynamic: true
index.refresh_interval: "1s"
action.auto_create_index: true
action.disable_shutdown: true

network.host: localhost
transport.tcp.port: 9300
transport.tcp.compress: true
http.port: 9200
http.enabled: true

gateway.type: local
discovery.zen.ping.multicast.enabled: false
discovery.zen.minimum_master_nodes: 2
discovery.zen.ping.unicast.hosts: ["localhost:9300","localhost:9301","localho
st:9302"]
```

[node2에 대한 elasticsearch.yml 설정]

```
cluster.name: cluster_node

node.name: node2
```

```
node.master: true
node.data: true

index.number_of_shards: 5
index.number_of_replicas: 0
index.mapper.dynamic: true
index.refresh_interval: "1s"
action.auto_create_index: true
action.disable_shutdown: true

network.host: localhost
transport.tcp.port: 9301
transport.tcp.compress: true
http.port: 9201
http.enabled: true

gateway.type: local
discovery.zen.ping.multicast.enabled: false
discovery.zen.minimum_master_nodes: 2
discovery.zen.ping.unicast.hosts: ["localhost:9300","localhost:9301","localho
st:9302"]
```

[node3에 대한 elasticsearch.yml 설정]

```
cluster.name: cluster_node

node.name: node3
node.master: true
node.data: true

index.number_of_shards: 5
index.number_of_replicas: 0
```

```
index.mapper.dynamic: true
index.refresh_interval: "1s"
action.auto_create_index: true
action.disable_shutdown: true

network.host: localhost
transport.tcp.port: 9302
transport.tcp.compress: true
http.port: 9202
http.enabled: true

gateway.type: local
discovery.zen.ping.multicast.enabled: false
discovery.zen.minimum_master_nodes: 2
discovery.zen.ping.unicast.hosts: ["localhost:9300","localhost:9301","localho
st:9302"]
```

2.4.3 실행 및 확인

node1부터 node3까지 차례대로 실행한다.

```
$ node1/bin/elasticsearch -Des.pidfile=es.pid > /dev/null 2>&1 &
$ node2/bin/elasticsearch -Des.pidfile=es.pid > /dev/null 2>&1 &
$ node3/bin/elasticsearch -Des.pidfile=es.pid > /dev/null 2>&1 &
```

모든 노드를 실행한 후 클러스터 구성이 정상적으로 되었는지 확인하기 위해 cluster health REST API 실행한다.

```
$ curl -XGET http://localhost:9200/_cluster/health?pretty=true
```

[실행 결과]

```
{
  "cluster_name" : "cluster_node",
  "status" : "green",
  "timed_out" : false,
  "number_of_nodes" : 3,
  "number_of_data_nodes" : 3,
  "active_primary_shards" : 0,
  "active_shards" : 0,
  "relocating_shards" : 0,
  "initializing_shards" : 0,
  "unassigned_shards" : 0
}
```

정상적으로 클러스터링 되었다면 작성한 설정과 동일하게 노드들이 등록되었는지 cluster nodes info REST API를 실행하여 확인한다. 기본 정보가 너무 많이 나오기 때문에 설정에 해당하는 settings 정보만 요청한다.

```
$ curl -XGET http://localhost:9200/_nodes?pretty=true #전체 정보를 가져온다.
$ curl -XGET http://localhost:9200/_nodes/settings?pretty=true #settings 정보만 가져온다.
```

다음은 정상적으로 클러스터링 된 노드 목록과 정보의 결과값이다.

[실행 결과]

```
{
  "cluster_name" : "cluster_node",
  "nodes" : {
```

```
"-Rr1fhFpTy2tYjQk9qWSQw" : {
   "name" : "node2",
   "transport_address" : "inet[/127.0.0.1:9301]",
   "host" : "jeong-ui-MacBook-Pro.local",
   "ip" : "192.168.0.117",
   "version" : "1.0.0",
   "build" : "a46900e",
   "http_address" : "inet[localhost/127.0.0.1:9201]",
   "attributes" : {
      "master" : "true"
   },
   "settings" : {
      "index" : {
         "mapper" : {
            "dynamic" : "true"
         },
      "number_of_replicas" : "0",
      "number_of_shards" : "5",
      "refresh_interval" : "1s"
   },
   "gateway" : {
      "type" : "local"
   },
   "pidfile" : "es.pid",
   "network" : {
      "host" : "localhost"
   },
   "node" : {
      "data" : "true",
      "master" : "true",
      "name" : "node2"
   },
```

```
"http" : {
   "port" : "9201",
   "enabled" : "true"
},
"transport" : {
   "tcp" : {
      "compress" : "true",
      "port" : "9301"
   }
},
"name" : "node2",
"action" : {
   "disable_shutdown" : "true",
   "auto_create_index" : "true"
},
"path" : {
   "logs" : "/Users/hwjeong/server/app/elasticsearch/node2/logs",
   "home" : "/Users/hwjeong/server/app/elasticsearch/node2"
},
"cluster" : {
   "name" : "cluster_node"
},
"discovery" : {
   "zen" : {
      "minimum_master_nodes" : "2",
      "ping" : {
         "unicast" : {
           "hosts" : [ "localhost:9300", "localhost:9301", "localhost:9302" ]
         },
         "multicast" : {
            "enabled" : "false"
         }
```

```
            }
          }
        },
        "foreground" : "yes"
      }
    },
    "CJF0dGOYRiq7izHmq2y0wg" : {
        "name" : "node3",
        "transport_address" : "inet[/127.0.0.1:9302]",
        "host" : "jeong-ui-MacBook-Pro.local",
        "ip" : "192.168.0.117",
        "version" : "1.0.0",
        "build" : "a46900e",
        "http_address" : "inet[localhost/127.0.0.1:9202]",
        "attributes" : {
          "master" : "true"
        },
        "settings" : {
          "index" : {
            "mapper" : {
              "dynamic" : "true"
            },
            "number_of_replicas" : "0",
            "number_of_shards" : "5",
            "refresh_interval" : "1s"
          },
          "gateway" : {
            "type" : "local"
          },
          "pidfile" : "es.pid",
          "network" : {
            "host" : "localhost"
```

```
},
"node" : {
   "data" : "true",
   "master" : "true",
   "name" : "node3"
},
"http" : {
   "port" : "9202",
   "enabled" : "true"
},
"transport" : {
   "tcp" : {
      "compress" : "true",
      "port" : "9302"
   }
},
"name" : "node3",
"action" : {
   "disable_shutdown" : "true",
   "auto_create_index" : "true"
},
"path" : {
   "logs" : "/Users/hwjeong/server/app/elasticsearch/node3/logs",
   "home" : "/Users/hwjeong/server/app/elasticsearch/node3"
},
"cluster" : {
   "name" : "cluster_node"
},
"discovery" : {
   "zen" : {
      "minimum_master_nodes" : "2",
      "ping" : {
```

```
            "unicast" : {
              "hosts" : [ "localhost:9300", "localhost:9301", "localhost:9302" ]
            },
            "multicast" : {
               "enabled" : "false"
            }
          }
        }
      },
      "foreground" : "yes"
   }
},
"imJTqSzfS6CSb_Hx1AN0Dg" : {
   "name" : "node1",
   "transport_address" : "inet[localhost/127.0.0.1:9300]",
   "host" : "jeong-ui-MacBook-Pro.local",
   "ip" : "192.168.0.117",
   "version" : "1.0.0",
   "build" : "a46900e",
   "http_address" : "inet[localhost/127.0.0.1:9200]",
   "attributes" : {
      "master" : "true"
   },
   "settings" : {
      "index" : {
         "mapper" : {
            "dynamic" : "true"
         },
         "number_of_replicas" : "0",
         "number_of_shards" : "5",
         "refresh_interval" : "1s"
      },
```

```
"gateway" : {
   "type" : "local"
},
"pidfile" : "es.pid",
"network" : {
   "host" : "localhost"
},
"node" : {
   "data" : "true",
   "master" : "true",
   "name" : "node1"
},
"http" : {
   "port" : "9200",
   "enabled" : "true"
},
"transport" : {
   "tcp" : {
      "compress" : "true",
      "port" : "9300"
   }
},
"name" : "node1",
"action" : {
   "disable_shutdown" : "true",
   "auto_create_index" : "true"
},
"path" : {
   "logs" : "/Users/hwjeong/server/app/elasticsearch/node1/logs",
   "home" : "/Users/hwjeong/server/app/elasticsearch/node1"
},
"cluster" : {
```

```
            "name" : "cluster_node"
         },
         "discovery" : {
            "zen" : {
               "minimum_master_nodes" : "2",
               "ping" : {
                  "unicast" : {
                    "hosts" : [ "localhost:9300", "localhost:9301", "localhost:9302" ]
                  },
                  "multicast" : {
                     "enabled" : "false"
                  }
               }
            }
         },
         "foreground" : "yes"
      }
   }
}
```

2.5 Elasticsearch Node 구성의 이해

elasticsearch의 모든 노드는 노드 설정을 통해 각자의 역할을 부여받는다.

2.5.1 Master Node

마스터 노드는 클러스터 구성에서 전체 클러스터와 노드, 샤드 등에 대한 조정자 Coordinator 역할을 한다. 한 클러스터 내에서 마스터 노드는 최소 2개 이상으로 구성하는 것이 좋다. 마스터로 지정된 노드들은 처음 선출된 마스터 노드에 장애가 발생할 경우 자동으로 다음 마스터 노드를 선출하여 장애 대응이 가능하다.

```
node.master: true
```

2.5.2 Data Node

데이터 노드로 지정된 노드는 색인 데이터를 저장하고, 검색 요청 시 실행되며, 실제 모든 작업을 수행한다.

```
node.data: true
```

2.5.3 Search Loadbalancer Node

검색 요청에 대한 트래픽 분산 및 그 결과를 통합하여 리턴하는 역할을 한다.

```
node.master: false
node.data: false
```

2.5.4 Client Node

클라이언트 노드로 지정하면 node.master 설정이 default false로 처리되므로 마스터 노드로 사용하지 않을 경우 지정한다.

```
node.client: true
```

2.6 Elasticsearch Route 기능의 이해

2.6.1 Route 기능 활용

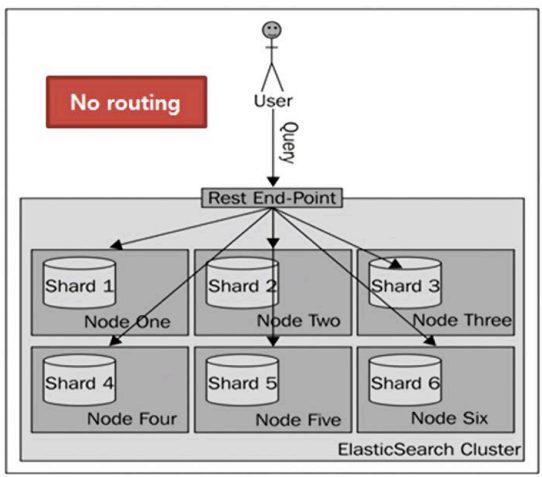
라우트는 특정 path를 갖는 데이터를 물리적으로 같은 공간에 분류하는 기능으로, 라우트를 설정하면 검색 성능을 향상할 수 있다. 예를 들어 route path 값으로 문서

의 카테고리 정보를 이용하면, 같은 카테고리의 문서들은 같은 샤드로 지정하여 색인 및 검색할 수 있다.

■ No Routing

라우트를 설정하지 않으면 모든 샤드를 대상으로 색인과 검색을 수행한다. 따라서 검색 처리에 대한 성능은 라우트를 적용했을 때보다 좋지 않다.

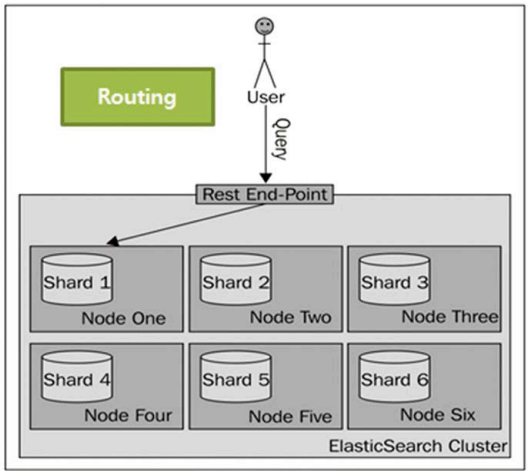
[그림 2-1] No Routing



■ Routing

라우트를 설정하면 색인과 검색을 수행해야 하는 대상 샤드를 바로 지정하고 찾을 수 있다. 라우트를 설정하지 않았을 때보다 검색 처리 성능이 매우 좋다.

[그림 2-2] Routing



2.6.2 Route 기능

분산된 샤드에 지정한 카테고리 정보를 이용하여 카테고리별 도큐먼트를 지정된 샤드로 저장할 수 있으며, 카테고리 정보를 이용하여 지정한 샤드의 문서를 검색할 수 있도록 지원한다.

색인 필드 중 unique key에 해당하는 값을 routing path로 지정한다. 검색 시 지정한 path(카테고리)를 쿼리Query에 줘서 분산된 인덱스의 샤드를 모두 검색하지 않고 지정된 인덱스의 샤드만 검색한다.

- 라우팅 필드Routing Field는 스토어 옵션인 yes와 index not_analyzed로 설정되어야 한다.

```
"routing" : {
  "required" : true,
  "path" : "market.cat_first_id"
}
```