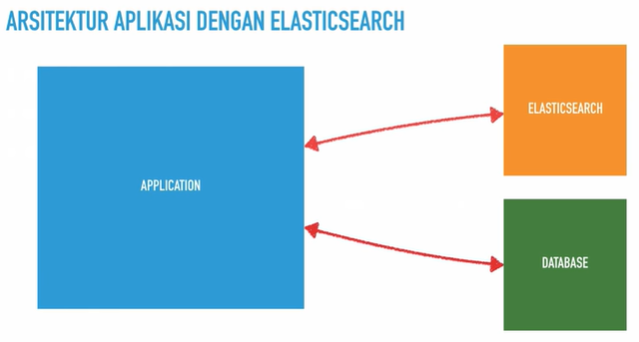
**Elasticsearch Documentation**

www.eleastic.co

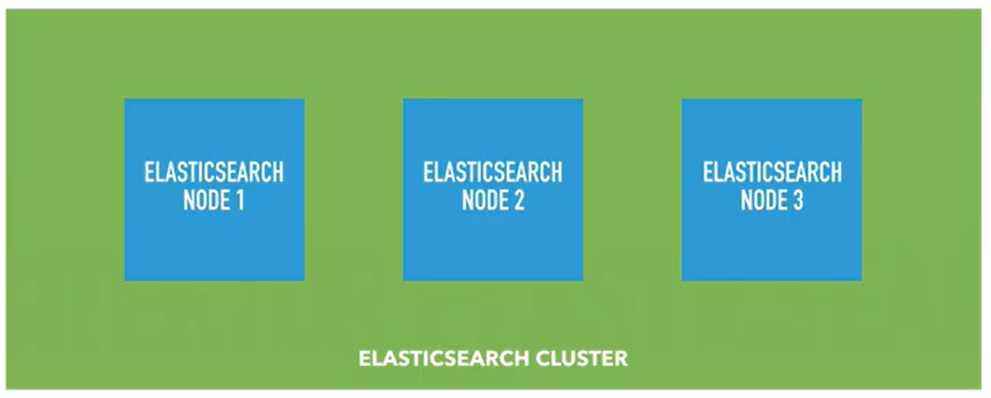
Elastic search 🡪 database untuk pencarian saat query yg kompleks, pendamping database permanen

Kompleks 🡪 aggregating/group by yg banyak, search autocomplete, ngram, sinonim, stc



Elasticsearch node biasanya minimal 3 dan meningkat ganjil, agar jika ada network branching maka yg sedikit nonaktif

Elasticsearch cluster = node yg berjalan bareng



Elasticsearch’s index is analogous with database’s table, consists of key value/column row

Index sharding = index partitioning 🡪 by default, sebuah index akan dipartisi menjadi 5 partisi

By default, sebuah index akan direplikasi dengan 1 replika

Data pada elasticsearch immutable, tidak akan diubah/didelete. Jadi jika ada perintah delete maka lokasi data yg didelete ditulis di file .del, tapi file aslinya tetap ada. Jika edit, maka lokasi file yg diedit disimpan di .del dan buat data baru dengan lokasi baru sebagai utput data edited. Data asli akan didelete pada clean up pada routine tertentu. Ini yg membuat proses search sangat cepat.

Proses searching = query (parallel pada setiap shard) + patch (menggabungkan hasil query dari setiap shard/partisi) 🡪 mirip map reduce

Problem 🡪 deep pagination, Ketika kita mengambil data page ke 100, jika tiap page ada 100 record, maka elastic search akan memproses 10000 record data. Default max record adalah 10000

Jika ingin mengambil semua record 🡪 gunakan fungsi scroll

**Running elastic search**

1. Buka cmd, ketik :

cd C:\Users\ASUS\Desktop\elasticsearch-8.0.0\bin

elasticsearch

**Membuat index**

1. Buka postman, tembak api :

PUT http://localhost:9200/namaIndex

Di body, raw, JSON:

{

“settings”: {

“index” : {

“number\_of\_shards” : 3,

“number\_of\_replicas” : 2

{

}

}

1. Untuk cek informasi index yang sudah dibuat, buka url:

Localhost:9200/-cat/indices?v

**Insert/Update data**

1. Buka postman, tembak api :

PUT http://localhost:9200/namaIndex/\_doc/1

Di body, raw, JSON:

{

“first\_name” : “Eko”,

“last\_name” : “Khannedy”,

“address” : {

“country” : “Indonesia:,

“city” : “Jakarta”

},

“hobby” : [“game”, “coding”, “travelling”]

}

1. Untuk cek informasi index yang sudah dibuat, tembak api:

GET http://localhost:9200/namaIndex/\_doc/1

**Update data**

Update tanpa harus menulis semua data JSON, hanya yg ingin diubah saja 🡪 data to be kind of Mutable, tidak disarankan

1. Buka postman, tembak api :

POST http://localhost:9200/namaIndex/\_update/1

Di body, raw, JSON:

{

“doc” : {

“first\_name” : “Eki”

}

}

1. Untuk cek informasi index yang sudah dibuat, tembak api:

GET http://localhost:9200/namaIndex/\_doc/1

**Delete data**

1. Buka postman, tembak api :

DELETE http://localhost:9200/namaIndex/\_doc/1

**Query data**

1. Untuk search di semua index, buka postman, tembak api:

GET http://localhost:9200/\_search

Di body, raw, JSON:

{

“query” : {

“match” : {

“first\_name” : “Eko”

}

}

1. Untuk search index tertentu, buka postman, tembak api:

GET http://localhost:9200/namaIndex/\_search

Di body, raw, JSON:

{

“query” : {

“match” : {

“first\_name” : “Eko”

}

}

}

1. Untuk query dengan kondisi/WHERE kombinasi, gunakan:

Bool: must (AND), should (OR), must\_not (NOT)

**Aggregating hasil query**

Grouping hasil query per kategori. Di ecmommerce, setelah kita search barang, di kiri ada groping hasil search per kategori.

1. Buka postman, tembak api :

GET http://localhost:9200/\_search

Di body, raw, JSON:

{

// search query

,

“aggs” : {

“genre\_hobi ” : {

“terms” : {

“field” : “hobi.keyword”

}

}

}

}

**Cek mapping**

1. Buka url :

http://localhost:9200/namaIndex/\_mapping

**Cek cluster health**

1. Buka url :

http://localhost:9200/\_cluster/health

1. Status:

Green: oke

Yellow: primary node masih aman

Red: tidak oke, primary node ada yg mati

Pastikan jumlah node, shard, replica sesuai kebutuhan