Table of Contents

[Global workaround 2](#__RefHeading___Toc131_455276633)

[Add a mapping or update existing mappings 3](#__RefHeading___Toc137_455276633)

[Create Index with Settings 4](#__RefHeading___Toc139_455276633)

[Create Index with settings – config analyzers 5](#__RefHeading___Toc155_455276633)

[Index Mapping Example 6](#__RefHeading___Toc114_455276633)

[Add index with settings and mappings 7](#__RefHeading___Toc159_455276633)

[Multi-Get 8](#__RefHeading___Toc1535_162537273)

[bulk Insert products and services 9](#__RefHeading___Toc1537_162537273)

[Partial update 10](#__RefHeading___Toc1539_162537273)

[Ip Data type 11](#__RefHeading___Toc1541_162537273)

[Geo-point example 12](#__RefHeading___Toc1543_162537273)

[Search in URI 14](#__RefHeading___Toc1545_162537273)

[Pagination 15](#__RefHeading___Toc1547_162537273)

[Search DSL 16](#__RefHeading___Toc1549_162537273)

[Query with range 17](#__RefHeading___Toc1578_162537273)

[Filtering Fields 18](#__RefHeading___Toc1551_162537273)

[index mapping template 19](#__RefHeading___Toc1553_162537273)

[My\_Store Structured Query – constant\_score 20](#__RefHeading___Toc1580_162537273)

[Full text search – Single Term 21](#__RefHeading___Toc1582_162537273)

[Full text - Single/multiple word match query 22](#__RefHeading___Toc1584_162537273)

[Full text – multiple boolean combination 23](#__RefHeading___Toc1586_162537273)

[get by productID with term (which returns nothing) 24](#__RefHeading___Toc1588_162537273)

[get by productID with full text query (which returns sth) 25](#__RefHeading___Toc1590_162537273)

[Analyze a field 26](#__RefHeading___Toc1592_162537273)

[Change productID to a keyword 27](#__RefHeading___Toc1594_162537273)

[Sort by productID (working version) 28](#__RefHeading___Toc1603_162537273)

[Python connect to ES 29](#__RefHeading___Toc2819_2226756342)

[Exercise 1 Answer 30](#__RefHeading___Toc2582_2169869404)

[1.0 30](#__RefHeading___Toc2584_2169869404)

[1.1 30](#__RefHeading___Toc2586_2169869404)

[1.2 31](#__RefHeading___Toc2588_2169869404)

[1.3 31](#__RefHeading___Toc2590_2169869404)

[1.4 32](#__RefHeading___Toc2592_2169869404)

[Exercise 2 33](#__RefHeading___Toc2594_2169869404)

[2.1 33](#__RefHeading___Toc2596_2169869404)

[Exercise 3 35](#__RefHeading___Toc2598_2169869404)

[3.1 35](#__RefHeading___Toc2600_2169869404)

# Global workaround

PUT /\_all/\_settings

{"index.blocks.read\_only\_allow\_delete": null}

# Add a mapping or update existing mappings

PUT my\_index1/\_mapping/\_doc

{

"properties": {

"email": {

"type": "keyword"

}

}

}

# Create Index with Settings

PUT /my\_index

{

"settings":{

"number\_of\_shards":3,

"number\_of\_replicas":0

}

}

# Create Index with settings – config analyzers

PUT /spanish\_docs

{

"settings": {

"analysis": {

"analyzer": {

"es\_std": {

"type": "standard",

"stopwords": "\_spanish\_"

}

}

}

}

}

# Index Mapping Example

PUT /bankaccountdetails

{

"mappings":{

"report":{

"\_all":{

"enabled":true

},

"properties":{

"name":{ "type":"string"}, "date":{ "type":"date"},

"balance":{ "type":"double"}, "liability":{ "type":"double"}

}

}

}

}

# Add index with settings and mappings

put my\_index4

{

"settings":{

"number\_of\_shards":3,

"number\_of\_replicas":0

},

"mappings": {

"doc": {

"properties": {

"title": { "type": "text" },

"name": { "type": "text" },

"age": { "type": "integer" },

"created": {

"type": "date",

"format": "strict\_date\_optional\_time||epoch\_millis"

}

}

}

}

}

# Multi-Get

GET /my\_index4/doc/\_mget

{

"docs":[

{

"\_id": 2

},

{

"\_id": 4

}

]

}

# bulk Insert products and services

POST /my\_store/products/\_bulk

{ "index": { "\_id": 1 }}

{ "price" : 10, "productID" : "XHDK-A-1293-#fJ3", "productName" : "Wrench" }

{ "index": { "\_id": 2 }}

{ "price" : 20, "productID" : "KDKE-B-9947-#kL5", "productName" : "Saw" }

{ "index": { "\_id": 3 }}

{ "price" : 30, "productID" : "JODL-X-1937-#pV7" , "productName" : "Knife" }

{ "index": { "\_id": 4 }}

{ "price" : 30, "productID" : "QQPX-R-3956-#aD8", "productName" : "Saw" }

POST /your\_store/services/\_bulk

{ "index": { "\_id": 1 }}

{ "price" : 100, "serviceName" : "Log Collection" }

{ "index": { "\_id": 2 }}

{ "price" : 200, "serviceName" : "Log Collection and Analysis" }

{ "index": { "\_id": 3 }}

{ "price" : 300, "serviceName" : "Cluster Analysis" }

# Partial update

POST /your\_store/services/1

{

"price" : 200

}

# Ip Data type

PUT my\_index

{

"mappings": {

"\_doc": {

"properties": {

"ip\_addr": {

"type": "ip"

}

}

}

}

}

PUT my\_index5/\_doc/1

{

"ip\_addr": "192.168.1.1"

}

GET my\_index5/\_search

{

"query": {

"term": {

"ip\_addr": "192.168.0.0/16"

}

}

}

# Geo-point example

PUT my\_index6

{

"mappings": {

"\_doc": {

"properties": {

"location": {

"type": "geo\_point"

}

}

}

}

}

PUT my\_index6/\_doc/2

{

"text": "Geo-point as a string",

"location": "41.12,-71.34" 

}

PUT my\_index6/\_doc/3

{

"text": "Geo-point as a geohash",

"location": "drm3btev3e86" 

}

PUT my\_index6/\_doc/4

{

"text": "Geo-point as an array",

"location": [ -71.34, 41.12 ] 

}

GET my\_index6/\_search

{

"query": {

"geo\_bounding\_box": { 

"location": {

"top\_left": {

"lat": 42,

"lon": -72

},

"bottom\_right": {

"lat": 40,

"lon": -74

}

}

}

}

}

# Search in URI

GET my\_store/\_search?q=productName=Saw

GET my\_store/\_search?q=%2BproductName%3ASaw+%2Bprice%3A20

# Pagination

GET /\_search?size=5

GET /\_search?size=5&from=5

GET /\_search?size=5&from=10

GET bank/account/\_search?size=1&from=0

# Search DSL

GET /my\_store/products/\_search

{

"query" : {

"constant\_score" : {

"filter" : {

"term" : {

"price" : 20

}

}

}

}

}

# Query with range

GET /my\_store/products/\_search

{

"query" : {

"constant\_score" : {

"filter" : {

"range" : {

"price" : {

"gte" : 20,

"lt" : 40

}

}

}

}

}

}

# Filtering Fields

get nyc\_visionzero/\_search

{

"\_source": {

"includes": [ "number\_of\_motorist\_injured"],

"excludes": [ ]

},

"query" : {

"match\_all": {}

}

}

GET my\_store/\_search

{

"query" : {

"match\_all": {}

},

"script\_fields" : {

"name" : {

"script" : "params['\_source']['productName']"

}

}

}

# index mapping template

PUT \_template/template\_1

{

"index\_patterns": ["logstash-2015\*"],

"settings": {

"number\_of\_shards": 1

},

"mappings": {

"log": {

"properties": {

"geo": {

"properties": {

"coordinates": {

"type": "geo\_point"

}

}

}

}

}

}

}

# My\_Store Structured Query – constant\_score

GET /my\_store/products/\_search

{

"query" : {

"constant\_score" : {

"filter" : {

"bool" : {

"should" : [

{ "term" : {"price" : 20}},

{ "term" : {"productID" : "XHDK-A-1293-#fJ3"}}

],

"must\_not" : {

"term" : {"price" : 30}

}

}

}

}

}

}

# Full text search – Single Term

GET shakespeare/doc/\_search

{

"query": {

"constant\_score": {

"filter": {

"term": { "text\_entry": "late" }

}

}

}

}

# Full text - Single/multiple word match query

GET /shakespeare/doc/\_search

{

"query": {

"match": {

"text\_entry": "ventidius"

}

}

}

GET /shakespeare/doc/\_search

{

"query": {

"match": {

"text\_entry": "Ventidius Go"

}

}

}

# Full text – multiple boolean combination

GET /shakespeare/doc/\_search

{

"query": {

"bool": {

"must": [

{"match": {

"text\_entry": "Ventidius Go"

}}

],

"must\_not": [

{"match":

{"play\_name": "Timon of Athens"}}

]

}

}

}

# get by productID with term (which returns nothing)

GET /my\_store/products/\_search

{

"query" : {

"constant\_score" : {

"filter" : {

"term" : {

"productID" : "XHDK-A-1293-#fJ3"

}

}

}

}

}

# get by productID with full text query (which returns sth)

GET /my\_store/products/\_search

{

"query": {

"match": {

"productID": "XHDK-A-1293-#fJ3"

}

}

}

# Analyze a field

GET /my\_store/\_analyze

{

"field": "productID",

"text": "XHDK-A-1293-#fJ3"

}

# Change productID to a keyword

PUT /my\_store

{

"mappings" : {

"products" : {

"properties" : {

"productID" : {

"type" : "keyword",

"index" : true

}

}

}

}

}

# Sort by productID (working version)

GET /my\_store/products/\_search

{

"sort" : ["productID"],

"query": {

"match": {

"productID": "XHDK-A-1293-#fJ3"

}

}

}

# Python connect to ES

$pip install elasticsearch

$python

>>>from datetime import datetime

>>> from elasticsearch import Elasticsearch

>>> es = Elasticsearch()

>>> es.index(index="my-index", doc\_type="test-type", id=42, body={"any": "data", "timestamp": datetime.now()})

>>> es.get(index="my-index", doc\_type="test-type", id=42)['\_source']

# Exercise 1 Answer

## 1.0

PUT students/doc/\_bulk

{"index" : {"\_id":1}}

{"id": 1, "studentNo": "TH-CHEM-2016-C001", "name": "Jonh Smith", "major":"Chemistry", "gpa": 4.8, "yearOfBorn": 2000, "classOf": 2016, "interest": "soccer, basketball, badminton, chess"}

{"index" : {"\_id":2}}

{"id": 2, "studentNo": "TH-PHY-2018-C001", "name": "Isaac Newton", "major":"Physics", "gpa": 3.6, "yearOfBorn": 2001, "classOf": 2018, "interest": "novel, soccer, cooking"}

{"index" : {"\_id":3}}

{"id": 3, "studentNo": "BU-POLI-2016-C001", "name": "John Kennedy", "major":"Politics", "gpa": 4.2, "yearOfBorn": 2000, "classOf": 2016, "interest": "talking, dating, boxing, shooting, chess"}

{"index" : {"\_id":4}}

{"id": 4, "studentNo": "BU-POLI-2015-C001", "name": "John Kerry", "major":"Politics", "gpa": 4.1, "yearOfBorn": 1999, "classOf": 2015, "interest": "money, basketball"}

{"index" : {"\_id":5}}

{"id": 5, "studentNo": "BU-ARTS-2016-C002", "name": "Da Vinci", "major":"Arts", "gpa": 4.8, "yearOfBorn": 1995, "classOf": 2016, "interest": "drawing, music, wine"}

## 1.1

GET /students/doc/\_mget

{

"docs":[

{

"\_id": 1

},

{

"\_id": 3

},

{

"\_id": 5

}

]

}

## 1.2

GET /students/doc/\_search

{

"query" : {

"constant\_score" : {

"filter" : {

"term" : {

"name" : "john"

}

}

}

}

}

## 1.3

GET /students/doc/\_search

{

"query" : {

"constant\_score" : {

"filter" : {

"range" : {

"classOf" : {

"gte" : 2016

}

}

}

}

}

}

## 1.4

POST /students/doc/4/\_update

{

"script" : "ctx.\_source.interest+=',poker'"

}

# Exercise 2

## 2.1

GET /students/\_doc/\_search

{

"query": {

"bool": {

"must": [

{

"match":

{

"name": "John"

}

},

{

"range": {

"classOf": {

"gte": 2016

}

}

},

{

"match" : {

"interest": "chess"

}

}

]

}

}

}

# Exercise 3

## 3.1

GET /students/doc/\_search

{

"sort" : [{"gpa" : "asc"} ],

"query": {

"bool": {

"must": [

{

"range": {

"classOf": {

"gte": 2016

}

}

},

{

"match" : {

"interest": "soccer"

}

}

]

}

}

}

PUT students/\_mappings/\_doc

{

"properties": {

"classOf": {

"type": "long"

},

"gpa": {

"type": "float"

},

"id": {

"type": "long"

},

"interest": {

"type": "text",

"fields": {

"keyword": {

"type": "keyword",

"ignore\_above": 256

}

}

},

"major": {

"type": "text",

"fields": {

"keyword": {

"type": "keyword",

"ignore\_above": 256

}

}

},

"name": {

"type": "text",

"fields": {

"keyword": {

"type": "keyword",

"ignore\_above": 256

}

}

},

"studentNo": {

"type": "keyword"

},

"yearOfBorn": {

"type": "long"

}

}

}

GET /students/doc/\_search

{

"sort" : [{"studentNo.keyword" : "asc"} ],

"query": {

"bool": {

"must": [

{

"range": {

"classOf": {

"gte": 2016

}

}

},

{

"match" : {

"interest": "soccer"

}

}

]

}

}

}