## Document Management

DELETE hiking

PUT hiking

GET hiking/\_doc/\_search

PUT hiking/\_doc/1

{

"hiked\_on": "1994/05/20",

"coordinator": "Ruby Shrestha",

"total\_students": "12",

"cost": "5600.32",

"bus\_booked": "true",

"checklist": [

"water",

"food",

"first-aid"

],

"test\_field":null

}

PUT hiking/\_doc/2

{

"hiked\_on": "2015/05/20",

"coordinator": "Sabi Shrestha",

"total\_students": "8",

"cost": "3400.32",

"bus\_booked": "true",

"test\_field":null

}

#use of routing: shard = hash(routing) % no.of primary shards

#some shards have more impact than others (example, data not uniformly indexed across shards)

#routing example

PUT hiking/\_doc/3?routing=test

{

"hiked\_on": "1994/05/20",

"coordinator": "Sharad Shrestha",

"total\_students": "12",

"cost": "5600.32",

"bus\_booked": "true",

"checklist": [

"water",

"food"

]

}

#will not available here since routing value provided during indexing is not available

GET hiking/\_doc/3

#to make it available, use routing

GET hiking/\_doc/3?routing=hike

#search can be directed to particular route

GET hiking/\_search?routing=hike

#Get All docs in an index

GET hiking/\_doc/\_search

{

"query":{

"match\_all":{}

}

}

#Get doc with single term matching

GET hiking/\_doc/\_search

{

"query":{

"term": {

"coordinator": {

"value": "ruby"

}

}

}

}

#conflicts=proceed ; do not abort even if version conflict arises(conflicts=proceed)

POST hiking/\_update\_by\_query?conflicts=proceed

{

"query": {

"term": {

"coordinator": {

"value": "shrestha"

}

}

},

"script": {

"source": """

if (ctx.\_source.containsKey('checklist')){

if (!ctx.\_source.checklist.contains(params.suggested)){

ctx.\_source.checklist.add(params.suggested);

}

}else{

ctx.\_source.checklist=[];

ctx.\_source.checklist.add(params.suggested);

}

""",

"params": {

"suggested": "first-aid"

}

}

}

#Update if doc is available, else insert

POST hiking/\_doc/3/\_update

{

"script": "ctx.\_source.bus\_booked=false",

"upsert":{

"hiked\_on":"2018/04/12",

"coordinator":"Munmun kumar Shrestha"

}

}

#delete by query examples

POST hiking/\_delete\_by\_query

{

"query":{

"term":{

"coordinator":{

"value":"munmun"

}

}

}

}

#To delete multiple docs

POST hiking/\_delete\_by\_query

{

"query": {

"terms": {

"\_id": [

4,

5

]

}

}

}

#To truncate the table

POST hiking/\_delete\_by\_query

{

"query": {

"match\_all": {}

}

}

# /mget api

GET /\_mget

{

"docs": [

{

"\_index":"hiking",

"\_type":"\_doc",

"\_id":1,

"\_source":false

},{

"\_index":"product",

"\_type":"\_doc",

"\_id":14,

"\_source":{

"includes":["name", "price"]

}

}

]

}

GET hiking/\_doc/\_mget

{

"docs": [

{

"\_id":1

},{

"\_id":14

}

]

}

GET hiking/\_doc/\_mget

{

"ids": [

1,

2

]

}

#Note: better to delete index and recreate than truncate, performance wise.

#When delete operation is triggered or update triggered, old doc is marked in .liv lucene file as delete and new created

#Bulk API: saves network overhead

POST hiking/\_doc/\_bulk

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

{"hiked\_on":"2018/09/09","coordinator":"Elnino Tamu","total\_students":10}

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

{"hiked\_on":"2018/12/09","coordinator":"Ayush Malla","total\_students":4}

{"update":{"\_id":2}}

{"doc":{"testingfield":"testingvalue"}}

#bulk api with curl: data loading

#curl -H "Content-Type: application/json" -XPOST "http://localhost:9200/product/\_doc/\_bulk?pretty" --data-binary @test-data.json

## Mapping and Data Type

#view dynamically created index

GET hiking/\_doc/\_mapping

#creating index and providing mapping during index creation

PUT hiking

{

"mappings": {

"\_doc": {

"dynamic": false

}

}

}

#if index is already available, you can edit the mapping

PUT hiking/\_mapping/\_doc

{"dynamic":false}

#now, if you index the following document, testfield will not appear in mapping since dynamic is turned off.

PUT hiking/\_doc/3

{"testfield":"testvalue"}

#About Date Detection

#turning on dynamic mapping again to understand date detection

PUT hiking/\_mapping/\_doc

{"dynamic":true}

#The following doc indexing will not cause testdate1field to be mapped as date since its format does not fall under dynamic\_date\_format

PUT hiking/\_doc/3

{"testdate1field":"05-20-1950"}

#updating mapping's dynamic\_date\_format

PUT hiking/\_mapping/\_doc

{

"dynamic": true,

"dynamic\_date\_formats": [

"MM-dd-yyyy", "yyyy/MM/dd"

]

}

#insert a doc with date in yyyy/MM/dd format

#following will also be recognized as date type

PUT hiking/\_doc/3

{"testdate4field":"1950/05/20"}

#date detection can be turned off during index creation

PUT hiking

{

"mappings": {

"\_doc": {

"date\_detection": false

}

}

}

#editing mapping to make date detection false which is true by default

PUT hiking/\_mapping/\_doc

{"date\_detection":false}

#testdate4field will not be recognized as date field even if its format falls under dynamic\_date\_formats since date\_detection is turned off.

PUT hiking/\_doc/3

{"testdate4field":"05-20-1950"}

#enable numeric detection which is by default false

#during index creation

PUT hiking

{"mappings": {"\_doc":{"numeric\_detection":true}}}

#if index already exists, update mapping

PUT hiking/\_mapping/\_doc

{"numeric\_detection":true}

#by default takes long for whole number and float for decimal values

PUT hiking/\_doc/3

{"testnumber":"5"}

PUT hiking/\_doc/3

{"testfloat":"88.99"}

#Assigning field types by ourself

DELETE hiking

#Core Data Types (ppt)

#dynamic: false vs strict

#dynamic: false ignores fields unseen in mapping, dynamic: strict throws exception when fields unseen in mapping appears

PUT hiking

{

"mappings": {

"\_doc": {

"dynamic":false,

"properties":{

"coordinator":{

"type":"text"

},

"coordinator\_phn":{

"type":"keyword"

},

"budget":{

"type":"scaled\_float",

"scaling\_factor":100

},

"total\_students":{

"type":"byte"

},

"hiked\_on":{

"type":"date",

"format":"MM-dd-yyyy"

},

"bus\_taken":{

"type":"boolean"

},

"logo":{

"type":"binary"

},

"distance\_covered":{

"type":"integer\_range"

},

"time\_taken":{

"type":"date\_range",

"format":"MM-dd-yyyy HH:mm:ss"

},

"required\_items":{

"type":"text"

}

}

}

}

}

#binary example: U29tZSBiaW5hcnkgYmxvYg==

PUT hiking/\_doc/1

{"coordinator":"Ruby Shrestha",

"coordinator\_phn":"9849770220",

"budget":"50.35",

"total\_students":8,

"hiked\_on":"05-20-2017",

"bus\_taken":true,

"logo":"U29tZSBiaW5hcnkgYmxvYg==",

"distance\_covered":{"gt":45,"lt":50},

"time\_taken":{"gte":"05-20-2017 05:00:00","lte":"05-20-2017 05:30:00"},

"required\_items":["bottle", "food"]

}

#try inserting values of invalid format and see exception

GET hiking/\_search

#types of fields: flat, nested, object

#example of inner object mapping

PUT hiking/\_mapping/\_doc

{

"properties": {

"destination": {

"properties": {

"name":{

"type":"text"

},

"address":{

"type":"text"

}

}

}

}

}

PUT hiking/\_mapping/\_doc

{

"properties": {

"destination": {

"dynamic":true,

"properties": {}

}

}

}

POST hiking/\_doc/1/\_update

{

"doc": {

"destination": {

"name": "Ghyampe Dada",

"address": "Bhaktapur"

}

}

}

#by default takes dynamic false since the field attendees is part of hiking index which has dynamic set to false

#Such being the case, if properties of nested objects are not mentioned they won't be indexed and hence won't be searchable. If you do not know properties use dynamic: true

PUT hiking/\_mapping/\_doc

{

"properties": {

"attendees": {

"type": "nested",

"properties": {

"name": {

"type": "text"

},

"batch": {

"type": "keyword"

}

}

}

}

}

POST hiking/\_doc/1/\_update

{

"doc": {

"attendees": [{

"name": "Ruby Shrestha",

"batch": "2017"

}, {

"name": "Munu Shrestha",

"batch": "2018"

}]

}

}

#geo\_point data type

PUT hiking/\_mapping/\_doc

{

"properties": {

"destination": {

"properties": {

"location": {

"type": "geo\_point"

}

}

}

}

}

#4 different ways to provide latitude and longitude

POST hiking/\_doc/1/\_update

{

"doc": {

"destination": {

"location": {

"lat": 44.5,

"lon": -44.3

}

}

}

}

POST hiking/\_doc/1/\_update

{

"doc": {

"destination": {

"location": "44.5, -55.6"

}

}

}

POST hiking/\_doc/1/\_update

{

"doc": {

"destination": {

"location": [44.5, -43.5]

}

}

}

POST hiking/\_doc/1/\_update

{

"doc": {

"destination": {

"location": "dxztjubr61px"

}

}

}

#alias datatype

POST hiking/\_mapping/\_doc

{"properties":{

"transportation":{

"type":"alias",

"path":"bus\_taken"

}

}}

GET hiking/\_search

{"query": {

"term":{

"transportation": {

"value": "true"

}

}

}}

#if we want to add aggregation capacity to text fields, like to see how many hikes a particular coordinator has taken part in, we can do so by adding keyword as sub field in the existing text field as follows

PUT hiking/\_mapping/\_doc

{

"properties": {

"coordinator": {

"type":"text",

"fields": {

"keyword": {

"type":"keyword"

}

}

}

}

}

## Introduction to Searching

GET product/\_search

{

"explain": true,

"query": {

"term": {

"name": "sea"

}

}

}

###Term Level Queries

#Term, value not analyzed

GET product/\_search

{

"query": {

"term": {

"is\_active": {

"value": "true"

}

}

}

}

#gives 0 result since name is text field which is analyzed

GET product/\_search

{

"query": {

"term": {

"name": {

"value": "Ecolab - Balanced Fusion"

}

}

}

}

#Terms

GET product/\_search

{

"query": {

"terms": {

"name": ["balanced", "silver", "solid"]

}

}

}

PUT clubs

PUT clubs/\_doc/1

{"name":"media\_club","student\_ids":["434", "238", "544", "666"],

"total\_count":3}

DELETE internship

PUT internship

{"settings": {"index.max\_terms\_count":2}}

POST internship/\_doc/\_bulk

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

{"student":"434","domain":"design","started\_year":"2015"}

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

{"student":"233","domain":"arts","started\_year":"2013"}

GET internship/\_search

{

"query": {

"terms": {

"student": {

"index": "clubs",

"type": "\_doc",

"id": 1,

"path": "student\_ids"

}

}

}

}

PUT clubs/\_doc/1

{

"name": "media\_club",

"students": [

{"id":434, "name":"Ruby Shrestha"},

{"id":238, "name":"Sabi Shrestha"},

{"id":564, "name":"Munu Shrestha"},

{"id":777, "name":"Sharad Shrestha"}

],

"total\_count": 3

}

#Terms Set

GET product/\_search

{"query":{

"terms\_set":{

"name":{

"terms":["weed", "wine", "lobster", "sea"],

"minimum\_should\_match\_script":{

"source":"params.num\_terms/2"

}

}

}

}}

POST product/\_doc/\_update\_by\_query

{

"query": {

"match\_all": {}

},

"script": "ctx.\_source.required\_matches=2"

}

GET product/\_search

{"query":{

"terms\_set":{

"name":{

"terms":["weed", "wine", "lobster", "sea"],

"minimum\_should\_match\_field":"required\_matches"

}

}

}}

#Range

GET product/\_search

{"query":{

"range": {

"in\_stock": {

"gte": 10,

"lte": 50

}

}

}}

GET product/\_search

{"query":{

"range": {

"created": {

"gt": "2006",

"lt": "2008",

"format": "yyyy"

}

}

}}

#prefix query, value not analyzed

GET product/\_search

{"query":{

"prefix":{"name":"sw"}

}}

#wildcard

GET product/\_search

{"query":{

"wildcard":{"name":"s?a"}

}}

GET product/\_search

{"query":{

"wildcard":{"name":"n\*i"}

}}

#ids, similar to \_mget with ids

GET product/\_search

{"query":{

"ids":{"values":[14,32]}

}}

GET hiking/\_doc/\_mget

{

"ids": [

1,

2

]

}