Assignment II

1. Create an index having fields of the following types. Disable dynamic mapping for this index.

*Hint: This means to create an index with mapping having following types in properties.*

* Text
* Keyword
* Long
* Double
* Boolean
* geo\_point
* Date (allowing two different formats yyyy/MM/dd and MM-dd-yyyy, or any other formats that you prefer)
* Integer Range
* Array with text type values
* Inner Object (needs to have at least two properties)
* Nested object (need to have at least two properties)

PUT company

{

"mappings":{

"\_doc":{

"dynamic": false,

"properties":{

"name":{

"type":"text"

},

"Contact":{

"properties":{

"phone\_num":{

"type":"keyword"

}

}

},

"area":{

"type":"double"

},

"salary":{

"type":"long"

},

"employee\_requirements":{

"type":"boolean"

},

"total\_employees":{

"type":"integer"

},

"established\_date":{

"type":"date",

"format":"MM-dd-yyyy || yyyy/mm/dd"

},

"stake-holders":{

"type":"nested",

"properties":{

"name":{

"type":"text"

},

"position":{

"type":"text"

}

}

},

"room\_area":{

"type":"integer\_range"

}

}

}

}

}

2. Insert two valid documents, that is, with fields which match the types mentioned in (1)

PUT company/\_doc/3

{

"name":"Deerwalk Services Pvt Ltd",

"Location":{

"lat":40.2,

"lon":45.5

},

"Contact":{

"phone\_num":"4422215"

},

"area":512.56,

"salary":60000,

"total\_employee":300,

"stake-holders":{

"name":"Rudra Raj Pandey",

"position":"Ceo"

},

"class\_requirements\_fulfilled":true,

"established\_date":"01-02-2010",

"room\_area":{

"gt":70,

"lt":80

}

}

PUT company/\_doc/4

{

"name":"Leapfrog Enterprises",

"Location":{

"lat":40.2,

"lon":45.5

},

"Contact":{

"phone\_num":"4423245"

},

"area":512.56,

"salary":70000,

"total\_employee":250,

"stake-holders":{

"name":"John Barbour",

"position":"Ceo"

},

"class\_requirements\_fulfilled":true,

"established\_date":"03-12-2011",

"room\_area":{

"gt":70,

"lt":80

}

}

3. Try inserting an invalid document to see the exception thrown.

PUT company/\_doc/5

{

"name":"Leapfrog Enterprises",

"Location":{

"lat":40.2,

"lon":45.5

},

"Contact":{

"phone\_num":"4423245"

},

"area":512.56,

"salary":gfgh,

"total\_employee":250,

"stake-holders":{

"name":"John Barbour",

"position":"Ceo"

},

"class\_requirements\_fulfilled":abc,

"established\_date":"03-12-2011",

"room\_area":{

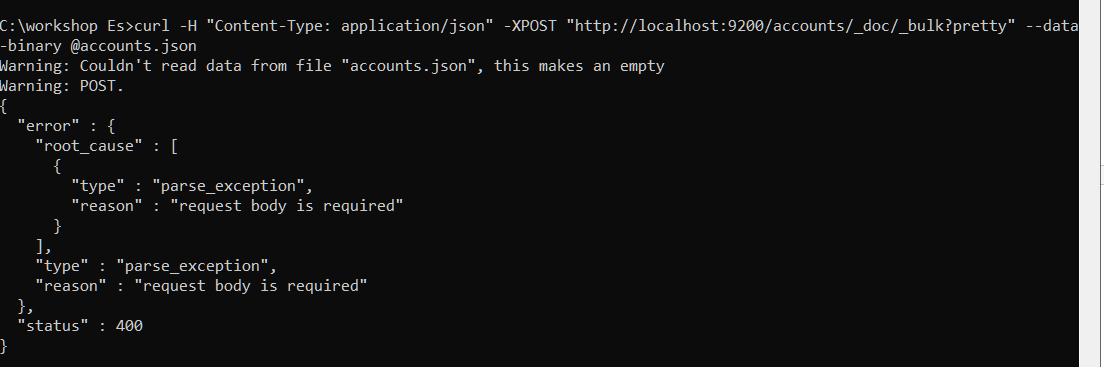
"gt":70,

"lt":80

}

}

4. Use **curl** command along with **\_bulk** API to insert the documents available in the file provided in mail (name: accounts.json) into **accounts** index.



5. Perform queries using Request URI to find the following:

* all documents

<http://localhost:9200/accounts/_doc/_search?q>=\*

GET /accounts/\_doc/\_search?q=\*

* age greater than equal to 30 and less than equal to 70

[http://localhost:9200/accounts/\_doc/\_search?q=age:[30%20TO%2070](http://localhost:9200/accounts/_doc/_search?q=age:%5b30%20TO%2070)]

GET /accounts/\_doc/\_search?q=age:[30 TO 70]

* females with age less than equals 25

[http://localhost:9200/accounts/\_doc/\_search?q=gender:F%20AND%20age:[\*%20TO%2025](http://localhost:9200/accounts/_doc/_search?q=gender:F%20AND%20age:%5b*%20TO%2025)]

GET /accounts/\_doc/\_search?q=gender:F AND age:[\* TO 25]

* males belonging to **ME** state

<http://localhost:9200/accounts/_doc/_search?q=gender:M%20and%20state=ME>

GET /accounts/\_doc/\_search?q=gender:M AND state:ME

6. Perform following \_update\_by\_query operations on accounts:

1. Add a new field **expense\_list** whose value is empty array [ ] for all documents.

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

{

"query":{

"match\_all":{}

},

"script": {

"source": "ctx.\_source.expense\_list = []",

"lang": "painless"

}

}

1. Add a value ‘student\_loan’ into the expense\_list array for members having age greater than equals 10 and less than equals 25

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

{

"query":{

"match\_all":{}

},

"script":{

"lang":"painless",

"source":"""

String val= "student\_loan";

if(ctx.\_source.age>=10 && ctx.\_source.age<=25){

ctx.\_source.expense\_list.add(val)

}

"""

}

}

1. Add two values ‘car\_loan’ and ‘house\_loan’ into expense\_list array for members having age greater than 25 and less than equals 50

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

{

"query":{

"match\_all":{}

},

"script":{

"lang":"painless",

"source":"""

String val1="car\_loan";

String val2="house\_loan";

if(ctx.\_source.age>=25 && ctx.\_source.age<=50){

ctx.\_source.expense\_list.add(val1);

ctx.\_source.expense\_list.add(val2)

}

"""

}

}

1. Add a value ‘recreation’ for members having balance greater than equals 40000.

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

{

"query":{

"match\_all":{}

},

"script":{

"lang":"painless",

"source":"""

String val="recreation";

if(ctx.\_source.balance>=40000){

ctx.\_source.expense\_list.add(val)

}

"""

}

}

1. Decrease the balance by 2000 of members of state **PA.**

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

{

"query":{

"match\_all":{}

},

"script":{

"lang":"painless",

"source":"""

if(ctx.\_source.state=="PA"){

ctx.\_source.balance-=params.val

}

""",

"params":{

"val":2000

}

}

}

8. Perform following queries using Request Body with any values you want to:

* Term query
* Range query
* Prefix query
* Wildcard Query

GET accounts/\_search

{

"query":{

"term":{

"country":{

"value":"Nepal"

}

}

}

}

GET accounts/\_search

{

"query":{

"range": {

"class": {

"gt": 1,

"lte": 10

}

}

}

}

GET accounts/\_search

{"query":{

"prefix":{

"firstname":"Ju"

}

}

}

GET product/\_search

{"query":{

"wildcard":{

"firstname":"s?b"

}

}

}

9. Refer to **Terms** query in [**Week II Notes**](https://docs.google.com/document/d/10lLTy0O-IZoMlVUJ76HvgKXC8xg6Es48tqqMlBw0nrc/edit?usp=sharing) to do the following question:

1. Create an index **college** having following fields:

* batch (integer type): example values, 2017, 2018
* students (nested type, i.e. array of inner objects): each inner object can have two properties **id** and **name**.

PUT college1

{

"mappings": {

"\_doc":{

"properties":{

"batch":{

"type":"integer"

},

"students":{

"type":"nested",

"properties":{

"id":{

"type":"integer"

},

"name":{

"type":"text"

}

}

}

}

}

}

}

1. Insert a document with certain id (example, 1), your batch (example, 2017), and an array of 3 students in index **college**.

PUT college/\_doc/1

{

"batch":2019,

"students":{

"student\_id":[547, 502, 521],

"name":["Shreeya Pandey","Alisha Pathak","Rachana Banjade"]

}

}

C. Create an index **workshop** having following fields

* students\_id
* workshop\_about
* enrolled\_year

PUT workshop

{

"mappings":{

"\_doc":{

"properties":{

"student\_id":{

"type":"integer"

},

"workshop\_about":{

"type":"text"

},

"enrolled\_year":{

"type":"date",

"format":"yyyy"

}

}

}

}

}

D. Bulk insert 5 documents in index **workshop**.

POST college2/\_doc/\_bulk

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

{"student\_id":502,"enrolled\_year":"2019","workshop\_about":"python" }

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

{"student\_id":503,"enrolled\_year":"2019","workshop\_about":"elasticsearch" }

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

{"student\_id":504,"enrolled\_year":"2017","workshop\_about":"c programming"}

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

{"student\_id":521,"enrolled\_year":"2018","workshop\_about":"video editing"}

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

{"student\_id":522,"enrolled\_year":"2019","workshop\_about":"java programming" }

E. Using **terms** query, find the students of your batch enrolled in any workshop.

GET workshop/\_search

{

"query": {

"terms": {

"workshop\_about": {

"index": "college2",

"type": "\_doc",

"id": 2,

"path": "batch"

}

}

}

}