## Assignment II

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

PUT employee2

{

"mappings": {

"\_doc": {

"dynamic": false,

"properties": {

"intro": {

"properties": {

"name": {

"type": "text"

},

"address": {

"type": "keyword"

},

"contact": {

"type": "long"

}

}

},

"salary": {

"type": "double"

},

"paid": {

"type": "boolean"

},

"location": {

"type": "geo\_point"

},

"date": {

"type": "date",

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

},

"expected\_employee": {

"type": "integer\_range"

},

"employee\_name": {

"type": "nested",

"properties": {

"fname": {

"type": "text"

},

"lname": {

"type": "text"

},

"squad": {

"type": "text"

}

}

},

"manager": {

"properties": {

"age": {

"type": "integer"

},

"about": {

"properties": {

"first": {

"type": "text"

},

"last": {

"type": "text"

}

}

}

}

}

}

}

}

}

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

PUT employee2/\_doc/1

{

"name": "Sweekriti",

"address": "Chabahil",

"contact": 981234808291,

"salary": 25000,

"paid": true,

"location": {

"lat": 41.12,

"lon": -71.34

},

"date": "12-11-2018",

"expected\_employee": {

"gte": 10,

"lte": 20

},

"employee\_name": {

"fname": "Sweekriti",

"lname": "Gautam",

"squad": "QA"

},

"manager": {

"age": 30,

"name": {

"first": "John",

"last": "Smith"

}

}

}

PUT employee2/\_doc/2

{

"name":"Kshitiz",

"address":"Chabahil",

"contact":9841228291,

"salary":5000,

"paid":false,

"location":{

"lat": 30.90,

"lon": -50.34

},

"date":"2-07-2019",

"expected\_employee":{

"gte" : 30,

"lte" : 80

},

"employee\_name":{

"fname":"Kshitiz",

"lname":"Gautam",

"squad":"Developing"

},

"manager":{

"age":48,

"name": {

"first": "Jack",

"last": "Green"

}

}

}

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

PUT employee2/\_doc/3

{

"name": "Ayusha",

"address": "Baneshwor",

"contact": 98127823822,

"salary": "75000",

"paid": true,

"location": {

"lat": 50,

"lon": -23

},

"date": "2018-12-11",

"expected\_employee": {

"gte": 10,

"lte": 20

},

"employee\_name": {

"fname": "Ayusha",

"lname": "Gautam",

"squad": "Design"

},

"manager": {

"age": "None",

"name": {

"first": "James",

"last": "Smith"

}

}

}

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.

Ans: curl -H "Content-Type: application/json" -XPOST "<http://localhost:9200/accounts/_doc/_bulk?pretty>" --data-binary @accounts.json

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]**

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"

}

}

GET /accounts/\_search

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 value= "student\_loan";

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

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

}

"""

}

}

GET /accounts/\_doc/\_search?q=age:[10 TO 25]

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 car="car\_loan";

String house="house\_loan";

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

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

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

}

"""

}

}

GET /accounts/\_doc/\_search?q=age:[25 TO 50]

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 recreation="recreation";

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

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

}

"""

}

}

GET /accounts/\_doc/\_search

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

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

{

"query": {

"match": {

"state":"PA"

}

},

"script": {

"lang": "painless",

"source":"ctx.\_source.balance-=params.value",

"params": {

"value":1000

}

}

}

GET /accounts/\_doc/\_search?q=state:"PA"

1. Perform \_delete\_by\_query to delete all records belonging to state: **KY**.

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

{

"query": {

"match": {

"state": "KY"

}

}

}

GET /accounts/\_search

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

* Term query

GET /accounts/\_search

{

"query": {

"term" : { "firstname":"Virginia" }

}

}

GET /accounts/\_search

* Range query

POST /accounts/\_search

{

"query": {

"range" : {

"age" : {

"gte" : 20,

"lte" : 30

}

}

}

}

GET /accounts/\_search

* Prefix query

GET /accounts/\_search

{"query":{

"prefix":{"employer":"Tr"}

}}

GET /accounts/\_search

* Wildcard Query

GET product/\_search

{"query":{

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

}}

GET /accounts/\_search

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:

PUT college

{

"mappings": {

"\_doc": {

"dynamic": false,

"properties": {

"batch": {

"type": "integer"

},

"student": {

"type": "nested",

"properties": {

"id": {

"type": "integer"

},

"name": {

"type": "text"

}

}

}

}

}

}

}

B. 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,

"student": [

{

"id": 544,

"name": "Sweekriti"

},

{

"id": 545,

"name": "Ayush"

},

{

"id": 547,

"name": "Shreeya"

}

]

}

GET college/\_search

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

PUT workshop

{

"mappings": {

"\_doc": {

"dynamic": false,

"properties": {

"student\_id": {

"type": "integer"

},

"workshop\_about": {

"type": "text"

},

"enrolled\_year": {

"type": "date",

"format": "yyyy"

},

"batch": {

"type": "integer"

}

}

}

}

}

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

POST workshop/\_doc/\_bulk

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

{"student\_id":"544","workshop\_about":"elasticsearch","enrolled\_year":"2019","batch":2019}

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

{"student\_id":"545","workshop\_about":"android","enrolled\_year":"2018","batch":2018}

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

{"student\_id":"547","workshop\_about":"laravel","enrolled\_year":"2017","batch":2017}

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

{"student\_id":"542","workshop\_about":"networking","enrolled\_year":"2017","batch":2019}

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

{"student\_id":"541","workshop\_about":"designing","enrolled\_year":"2016","batch":2020}

GET workshop/\_search

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

GET workshop/\_search

{

"query": {

"term" : { "batch":2019 }

}

}