mangoDB:

א. use students**;**

```
var students_details =
[
        {"id":0, "Dep":"Industrial engineering", "age":50,
"Courses":{"math":95,"database":7,"algebra":14}},
        {"id":1, "Dep":"CS", "age":5, "Courses":{"math":46}},
        {"id":2, "Dep":"CS", "age":29, "Courses":{"math":91,"database":21,"algebra":60}},
        {"id":3, "Dep":"Electrical Engineer", "age":8,
"Courses":{"math":88,"database":10,"algebra":68,"logic":33}},
        {"id":4, "Dep":"Constructor", "age":26,
"Courses":{"math":86,"database":37,"algebra":26,"logic":95,"history":32}},
        {"id":5, "Dep":"Industrial engineering", "age":10,
"Courses":{"math":87,"database":11}},
        {"id":6, "Dep":"Electrical Engineer", "age":4,
"Courses":{"math":46,"database":86,"algebra":95}},
        {"id":7, "Dep":"Industrial engineering", "age":52,
"Courses":{"math":82,"database":48,"algebra":68}},
        {"id":8, "Dep":"Constructor", "age":53,
"Courses":{"math":23,"database":47,"algebra":93,"logic":48,"history":67,"Chemistry":48}},
        {"id":9, "Dep":"Industrial engineering", "age":21,
"Courses":{"math":48,"database":53,"algebra":100,"logic":22}},
        {"id":10, "Dep":"Industrial engineering", "age":39,
"Courses":{"math":96,"database":93,"algebra":62}},
        {"id":11, "Dep":"Constructor", "age":46,
"Courses":{"math":5,"database":0,"algebra":24,"logic":63}},
        {"id":12, "Dep":"CS", "age":15, "Courses":{"math":22,"database":54}},
        {"id":13, "Dep":"Constructor", "age":13, "Courses":{"math":82,"database":67}},
        {"id":14, "Dep":"Constructor", "age":21,
 "Courses":{"math":14,"database":13,"algebra":2}},
        {"id":15, "Dep":"Electrical Engineer", "age":35,
 "Courses":{"math":66,"database":41,"algebra":64,"logic":89}},
        {"id":16, "Dep":"Electrical Engineer", "age":25,
 "Courses":{"math":18,"database":77,"algebra":44,"logic":4}},
        {"id":17, "Dep":"Electrical Engineer", "age":38,
 "Courses":{"math":67,"database":26,"algebra":86,"logic":43}},
        {"id":18, "Dep":"CS", "age":49, "Courses":{"math":53,"database":48}},
        {"id":19, "Dep":"Industrial engineering", "age":59,
 "Courses":{"math":4,"database":76,"algebra":0}}

]


db.students.insert(students_details);
```

ב.
```
db.students.mapReduce(
        mapFunction,
        reduceFunction,
        {
                out: {}
                query: {$or: [{"Dep":"CS"}, {"Dep":"Electrical Engineer"}]}
                finalize: avgGrades
        }
)


var mapFunction = function () {

        for (var idx = 0; idx < this.items.length; idx++){

                for (var idx2 = 0; idx < this.items[idx].courses.length; idx2++){

                        var key = this.items[idx].courses[idx2].first;

                        var value = { grade: this.items[idx].courses[idx2].second, count:1 };

                        emit (key, value);

                }

        }

};


var reduceFunction = function (key, values) {

        var reduceVal = { sumGrades:0, count:0 };

        for (var idx = 0; idx < values.length; idx++){

                reduceVal.sumGrades += values[idx].grade;

                reduceVal.count += values[idx].count;

        }

        return reduceVal;

};


var finalize = function (key, value) {

        value.avg = value. sumGrades / value.count;

        return value;

};
```

Neo4j:

MATCH (frOfDany) – [:friends*1..2] -> (d:{name: Dani}) – [:watch] -> (movieW) AND (d:{name: Dani}) – [:like] -> (movieL) WITH collect(frOfDany) AS Dany's_friends, collect(movieW) AS watch_movie, collect(movieL) AS like_movie WHERE ALL (f in Dany's_friends, w in watch_movie, l in like_movie (f) – [:watch] -> (w) OR (f) – [:watch] -> (l)) return count(Dany's_friends)

elasticSearch:

1. curl – XPOST http://localhost:9200/books -H "Content-Type: application/json" -d "{"\bookName\" : "\b\", "\authors\" : "\a\", "\genre\" : "\g\", "\bookPublishing\" : "\p\", "\yearPublish\" : "\y\", "\summary\" : "\s\"}"

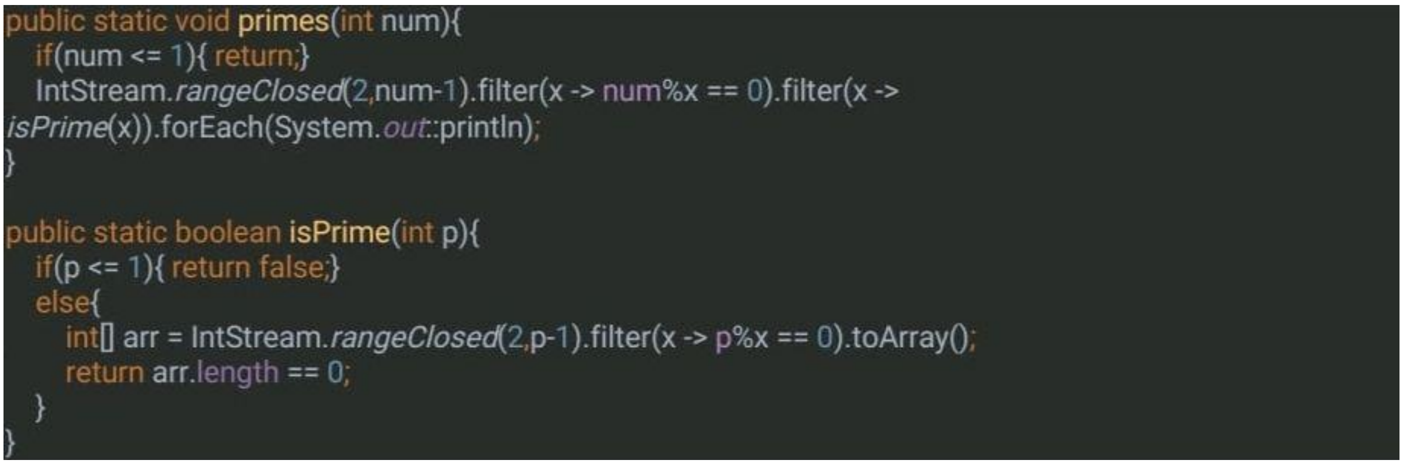
2. curl -XGET "http://localhost:9200/books/book/_search?q=genre:science fiction" -d"

```
{\"query\" :
  {\"bool\" :
    {\"filter\" :
      {\"range\" :
        {\"year\" :
          {\"gte\" : 2000} } } },
    {\"filter\" :
      {\"match\" :
        {\"summary\" : \"Science Fiction\"} } },
    {\"filter\" :
      {\"match\" :
        {\"summary\" : \"reality\"} } }
  }
}
```

X-path:

```
For $x in city
        let $pop := sum(city/institute/num)
        where $pop > 1000000
        return $x/name
```

Stream4:



```
public static void primes(int num){
   if(num <= 1){ return;}
   IntStream.rangeClosed(2,num-1).filter(x -> num%x == 0).filter(x ->
isPrime(x)).forEach(System.out::println);
}

public static boolean isPrime(int p){
   if(p <= 1){ return false;}
   else{
      int[] arr = IntStream.rangeClosed(2,p-1).filter(x -> p%x == 0).toArray();
      return arr.length == 0;
   }
}
```

RDFו:SPARQL:

א.

23	name	DANI
23	age	70
23	Father_ID	80
12	Name	MICHAEL
12	Age	23
12	Father_ID	23
45	Name	YARON
45	Age	49
45	Father_ID	67

ב.

```
select ?person where{
        ?person name:person ?fid.
        ?fid id:fid name: Dani.

}
```

TF-IDF:

A – (1/9)*log(5/2) + (1/9)*log(5/4) = 0.054

B – (1/9)log(5/4) = 0.010

C – (1/5)*log(5/2) + (1/5)*log(5/3) = 0.123

D – (1/8)*log(5/4) + (1/8)*log(5/3) = 0.039

E – (1/9)*log(5/4) + (1/9)*log(5/3) = 0.035

הדירוג:

C

A

D

E

B