מטלה 3:

שימו לב! יש להעלות כל הגשה פעם אחת בלבד!

יש להעלות קובץ word שמכיל את כל התשובות ביחד.

על מנת למנוע אי נעימויות – הקפידו לכתוב את מספרי הזהות של כל השותפים **בשם הקובץ.**

**שאלה 1:**

# mongoDB

עבור colliection של ספרים (בכל מסמך קיימים נתונים על הספר- שם, מחברים, הוצאה, שנת הוצאה וכן הספר עצמו(

1.כתוב פונקציית הכנסה של מסמך חדש ל-collection לעיל.

use mydb

function Insert\_newBook(name, author,subAuthor, publishDate)

{

mydb.books.insert({  
 name:name,  
authors:{author:author, sub:subAuthor},  
publishDate:publishDate

});

}

כעת ניתן לטעון את הפונקציה:  
load("insert\_newBook.js")

2 .כתוב פונקציית mapReduce המחזירה כמה מילים קיימים בספר לכל אורך של מילה לדוגמא- אם הספר מכיל את הטקסט הבא-

SUPPOSING that Truth is a woman--what then?

נקבל את התוצאה הבאה:

{\_id:”1”, \_value:”1”}, {\_id:”2”, \_value:”1”}, {\_id:”4”, \_value:”3”}, {\_id:”5”, \_value:”2”}, {\_id:”9”, \_value:”1”}

|  |
| --- |
| var map = function() { |
|  | var summary = this. text; |
|  | if (summary) { |
|  | // here we do lower case |
|  | summary = summary.toLowerCase().split(" "); |
|  | for (var i = summary.length - 1; i >= 0; i--) { |
|  | if (summary[i]) { // make sure there's something |
|  | emit(summary[i].length(), 1); // store a 1 for each word's length |
|  | } |
|  | } |
|  | } |
|  | }; |
|  |  |
|  | var reduce = function( key, values ) { |
|  | var count = 0; |
|  | key.forEach(function(v) { |
|  | count +=v; |
|  | }); |
|  | return count; |
|  | } |
|  |  |
|  | // full thing |
|  | // db.support\_emails.mapReduce(map, reduce, { out: "word\_count" }) |
|  |  |
|  | // subset |
|  | mydb.support\_books.mapReduce(map, reduce, {limit: 1000, out: "length\_count" }) |
|  | mydb.length\_count.find().pretty() |

**שאלה 2:**

# :Neo4j

נתון גרף המתאר מבנה של חברת הייטק

בגרף ישנם צמתים מהסוגים הבאים:

worker המכיל את שם העובד, ת"ז, כתובת ומשכורת manager המכיל את שם המנהל, ת"ז, כתובת ומשכורת.

department המכיל את שם המחלקה

בנוסף, קיימים הקשרים הבאים:

work\_In בין עובד/מנהל למחלקות שבהן הוא עובד המכיל תאריך התחלת עבודה.

friend\_Of בין 2 עובדים/מנהלים.

כתוב שאילתא ב neo4j המחזירה רשימה ממוינת של שמות כל העובדים בחברה (כולל מנהלים) מתל אביב שמשכורתם לפחות 19999 כל החברים שלהם עד רמה 3 (חברים, חברים של חברים, חברים של חברים של חברים) מקבלים גם כן משכורת שהיא לפחות 19999 ועובדים ב QA.

פתרון:

match (worker1)-[:friend\_Of\*1..3]-(worker2), (worker2)-:[work\_In]->(QA) where  
((worker2.salary>=19999) and (worker1.salary>=19999 and worker1.address="Tel Aviv"))  
return distinct worker1.name order by worker1.name

**שאלה 3:**

# :elasticSearch

עבורtype של ספרים, בכל מסמך קיימים נתונים על הספר- שם, מחברים, ז'אנר הספר, הוצאה, שנת הוצאה ותקציר הספר

1. כתוב פקודת הוספה של מסמך לindex 'ספרים'.

PUT /books/book "-d"

{

"name":"input\_name",

"authors": [{"author1": "input\_author1", "author2": "input\_author2"}],

"category": "input\_category",

"published\_by": "input\_published\_by",

"year": "input\_year",

"description": "input\_description"

}

1. כתוב שאילתא שתחזיר את הספרים שהוצאו משנת 2999 והלאה ושז'אנר הספר הוא- 'מדע בדיוני' ושבתקציר הספר יש את המילים"Fiction Science" ו-"reality" -

GET /books/book/\_search?"-d"

**{** \"query\" :

**{** \"bool\" :

**{** \"filter\" :

**{** \"match\" :

**{** \"category" : \"Science fiction\" **}** **}**,

\"filter\" :

**{** \"range\" :

**{** \"year\" :

**{** \"gte\" : "2999/d" **}** **}** **}**

"wildcard": **{**

"description": **{**

"value": "\*Fiction Science\*"

**}**

**}**

"wildcard": **{**

"description": **{**

"value": "\*reality\*"

**}**

**}** **}** **}** **}**

Wildcard

סוג מיוחד של שאילתה שמאפשר לנו לחפש תת מחרוזת

**שאלה 4:**

# :X-pathk

נניח שקיים לנו קובץ XML המכיל מדינות. כל מדינה (country) מכילה attribute שם וקודקודים בנים המייצגים את כל הערים שבה. כל עיר (city) מכילה בתור attribute את מספר התושבים שגרים בעיר זאת(num).

כתוב שאילתה X-path שתחזיר את שמות כל המדינות שסך כל התושבים בהם גדול ממליון.

פתרון:

doc("countries.xml")//country [city/@num>1000000]/data(@countryName)

**שאלה 5:**

# Stream:

# .בהינתן מודל של אוניברסיטה (קורסים, סטודנטים וציוני קורסים – ראו קוד מוכן לקימפול- קובץ המצורף)

# יש לבצע את המשימות הבאות באמצעות שימוש ב- API Stream 8 Java בלבד:

# להדפיס למסך שם של תלמיד והממוצע שלו בכל הקורסים שהוא לוקח. אם הנתונים הם:

# Yossi: Literature=30, Algebra=70

# Ayellet: English=70, Geometry=80

# הפלט צריך להיות:

# Yossi: 50 Ayellet: 75

# להדפיס למסך שם של קורס והממוצע באותו קורס של כל התלמידים המשתתפים:

# הפלט יהיה מהצורה

# Astronomy: 60 Infi: 79

# את פתרון השאלה הזו יש לרשום בקובץ ה MAIN המצורף ולרשום שם בהערות את מספרי תז של המגיש. הפתרון נמצא כאן:

**שאלה 6:**

# :SPARQLו RDF

המר את הטבלה הבאה לטבלת RDF ורשום שאילתא בSPARQL שמחזירה את שמות כל הנכדים של ראובן.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | name | age | Father\_ID |
| 111 | ישראל ישראלי | 15 | 444 |
| 222 | פלוני אלמוני | 2 | 333 |
| 333 | ג'ון סמית | 30 | 444 |
| 444 | ראובן אריאל | 81 | 555 |

הסבר: ג'ון סמית וישראל ישראלי הם הבנים של ראובן אריאל. פלוני אלמוני הוא הבן של ג'ון סמית ולכן הפלט יהיה "פלוני אלמוני".

|  |  |  |
| --- | --- | --- |
| S | P | O |
| Ttr:111 | name | "ישראל ישראלי" |
| Ttr:111 | age | "15" |
| Ttr:111 | Father\_ID | "444" |
| Ttr:222 | name | "פלוני אלמוני" |
| Ttr:222 | age | "2" |
| Ttr:222 | Father\_ID | "333" |
| Ttr:333 | name | "ג'ון סמית" |
| Ttr:333 | age | "30" |
| Ttr:333 | Father\_ID | "444" |
| Ttr:444 | name | "ראובן אריאל" |
| Ttr:444 | age | "81" |
| Ttr:444 | Father\_ID | "555" |

SELECT ?name WHERE

{

?id name "ראובן אריאל"

?id name ?name

}

**שאלה 7:**

# :TF-IDF

דרג את המשפטים הבאים לפי מדד TF-IDF:

Q: איזה יום היום

1: היום שימשי מאוד בחוץ.

2: היום יום חמישי.

3: איזה יום נעים היום!

4: היום יום הולדת ליונתן.

פתרון:

Q: איזה יום היום

D1: היום שימשי מאוד בחוץ.

D2: היום יום חמישי.

D3: איזה יום נעים היום!

D4: היום יום הולדת ליונתן.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | משפטים | איזה | יום | היום | מילים במשפט |
|  | **D1** | 0 | 0 | 1 | 4 |
|  | **D2** | 0 | 1 | 1 | 3 |
|  | **D3** | 1 | 1 | 0 | 4 |
|  | **D4** | 0 | 1 | 1 | 4 |
| סה"כ |  | 1 | 3 | 3 |  |

(לוג בסיס 10)

*D1:*

*D2:*

*D3:*

*D4:*

*מכאן שהדירוג הוא:*

*#1: D3*

*#2: D2*

*#3: D4*

*#4: D1*