הטכניון – מכון טכנולוגי לישראל הפקולטה למדעי המחשב

מרצה: פרופ' בני קימלפלד סמסטר אביב תשפ"א

מתרגלים: רואי קיסוס

חמודי סיף

גיא הורוביץ

מסדי נתונים

236363

מועד א'

29 ביולי 2021

פתרון

פירוט החלקים והניקוד:

שאלה	נושא	ניקוד	הערות
1	ERD Design Theory	25	
2	RA, RC Datalog	20	
3	SQL	20	
4	Concurrency Control	11	
5	XML	12	**יש לבחור 2 שאלות מתוך 5,6,7**
6	Neo4j MongoDB	12	**יש לבחור 2 שאלות מתוך 5,6,7**
7	RDF	12	**יש לבחור 2 שאלות מתוך 5,6,7**

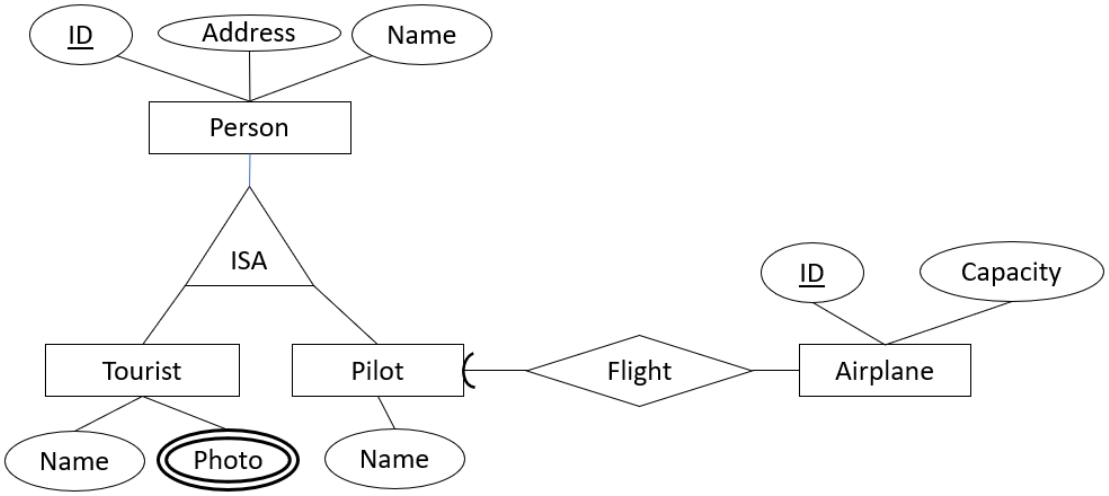
הנחיות לנבחנים:

1. כתבו את התשובות אך ורק בטופס הבחינה ובמקום המיועד להן, מחברת הטיוטה לא תיבדק.
2. ניתן להביא למבחן חומר כתוב\מודפס על גבי 6 דפי A4 (דו צדדיים).
3. אין לקבל או להעביר חומר כלשהו בזמן הבחינה.
4. יש להשתמש רק בסימנים או פונקציות שנלמדו בתרגול או בהרצאה והמופיעים בשקפים של הקורס. כל שימוש בסימון שאינו כזה מחייב הסבר מלא של משמעות הסימון.
5. משך הבחינה הינו שלוש שעות, תכננו את הזמן בהתאם.
6. אין לכתוב בעפרון.

בהצלחה!

שאלה 1 – ERD, Design Theory

התבוננו בתרשים ה-ERD שלפניכם:



א. תרגמו את תרשים ה-ERD לטבלאות המתאימות על פי הכללים שנלמדו בקורס. עבור כל טבלה, עליכם לרשום את סכמת הטבלה שתתקבל בתרגום, **כולל סימון מפתחות בקו תחתון וציון מפתחות זרים.** (6 נק')

Person: id Address Name
Tourist: id Name, id FK of Person
Tourist.photo: id photo, id FK of **Tourist**
Pilot: id Name, id FK of Person
Airplane: id capacity Pilot.id, Pilot.id FK of **Pilot**

<u>הסעיפים הבאים אינם תלויים בסעיף הקודם</u>

חלק זה מתייחס לסכמה (U,F) עבור $U=\{A,B,C,D,E\}$, $F=\{A\to B, B\to C, C\to D, D\to A\}$.
שימו לב כי התלויות ב-F מהוות מעגל המכיל את ארבעת השדות הראשונים $\{A,B,C,D\}$ ולא את החמישי $\{E\}$.

ב. הוכיחו כי הפירוק הבא <u>אינו</u> משמר מידע: $\{AC, CE, BDE\}$.
לשם כך, השתמשו באלגוריתם לבדיקת שימור מידע שנלמד בקורס. (5 נק')

A	B	C	D	E
a	x1	a	y1	z1
v2	x2	a	y2	a
v3	a	u3	a	a

C→D

A	B	C	D	E
a	x1	a	y1	z1
v2	x2	a	y1	a
v3	a	u3	a	a

D→A

A	B	C	D	E
a	x1	a	y1	z1
a	x2	a	y1	a
v3	a	u3	a	a

A→B

A	B	C	D	E
a	x1	a	y1	z1
a	x1	a	y1	a
v3	a	u3	a	a

היחס מקיים את התלויות ואין בו שורה של a, ומכאן שהפירוק אינו משמר מידע.

ג. הציעו תת-סכמה בעלת שני שדות בדיוק שהוספתה לפירוק בסעיף הקודם תהפוך את הפירוק למשמר מידע. הוכיחו שזה אכן המקרה בעזרת האלגוריתם לבדיקת שימור מידע. (7 נק')

נוסיף את הסכמה AB . נמשיך את ההרצה מקודם עם השורה הנוספת:

A	B	C	D	E
a	x1	a	y1	z1
a	x1	a	y1	a
v3	a	u3	a	a
a	a	u4	y4	z4

A→B

A	B	C	D	E
a	a	a	y1	z1
a	a	a	y1	a
v3	a	u3	a	a
a	a	u4	y4	z4

B→C

A	B	C	D	E
a	a	a	y1	z1
a	a	a	y1	a
v3	a	a	a	a
a	a	u4	y4	z4

C→D

A	B	C	D	E
a	a	a	a	z1
a	a	a	a	a
v3	a	a	a	a
a	a	a	y4	z4

כיוון שקיבלנו שורה של a, הפירוק משמר מידע.

ד. הוכיחו כי כל פירוק של הסכמה שלנו הינו מהצורה הנורמלית השלישית. (7 נק')

נסתכל על תת-סכמה עם התלויות המושרות עליה. מפתח קביל של הסכמה חייב להכיל את E כיוון ש-E לא נובע מאף שדה אחר (כי הוא כלל לא משתתף בתלויות), ועוד לפחות שדה אחד מ-A,B,C,D. אבל כל אחד מארבעת אילו מספיק כדי ליצור מפתח, כי הוא גורר את כל האחרים ברביעייה. מכאן נובע שכל שדה בתת הסכמה הינו ראשוני, ולכן, כל תלות מקיימת את התנאי לצורה הנורמלית השלישית.

שאלה 2 – RA, RC, Datalog

נתונות הסכמות הבאות:

1. Student(studentID, name) – כל הסטודנטים הרשומים במערכת.
2. Room(roomID, capacity) – כל הכיתות הרשומות במערכת והקיבולת של כל אחת (כמות הסטודנטים המקסימלית בה).
3. Course(courseID, name) – כל הקורסים הרשומים במערכת.
4. Attends(studentID, courseID) – כל מזהי הסטודנטים הלוקחים קורס מסוים.
5. In(courseID, roomID) – הכיתות בהן מתבצע המבחן של כל קורס.

מפתחות מסומנים בקו תחתון, כאשר:

1. studentID בסכמה Attends הוא מפתח זר ל-studentID ב-Student.
2. courseID בסכמה Attends הוא מפתח זר ל-courseID ב-Course.
3. courseID בסכמה In הוא מפתח זר ל-courseID ב-Course.
4. roomID בסכמה In הוא מפתח זר ל-roomID בסכמה Room.

הניחו שלכל קורס במערכת קיימת לפחות כיתה אחת זמינה (כלומר, קיימת רשומה ב-In) וכל סטודנט לוקח לפחות קורס אחד (כלומר, קיימת רשומה ב-Attends).

א. כתבו שאילתת RA המחזירה את כל מזהי הסטודנטים שלוקחים את כל הקורסים בהם לכל אחת מהכיתות יש קיבולת של יותר מ-20. (6 נק')

$$Attends \div (\pi_{courseID}(Course) \backslash (\pi_{courseID}\left(In \bowtie \left(\pi_{roomID}\left(\rho_{capacity \leq 20}(Room)\right)\right)\right)))$$

ב. נועה, סטודנטית בקורס מסדי נתונים ניסחה את שאילתת ה-RC הבאה במטרה להחזיר את כל הסטודנטים שאינם לוקחים לפחות קורס אחד:

$$\{(s,n) | Student(s,n) \wedge \exists c_1[\neg Attends(s,c_1)]\}$$

איתי טוען כי השאילתה שגויה כיוון שהיא תלויה בתחום. (6 נק')

- הסבירו על ידי דוגמא מדוע איתי צודק.
- הציעו שאילתה מתאימה שאינה תלויה בתחום.
- הסבירו האם ניתן לנסח את השאילתה שהצעתם ב-RA.

איתי צודק, עבור

Student = {(1, 1)}, Attends = {(1, 1)}

נקבל תשובה שונה עבור הדומיינים הבאים:

$D_1 = \{1\}, D_2 = \{1,2\}$

ניתן לנסח:

$$\{(s,n) | Student(s,n) \wedge \exists c_1, x: [Course(c_1,x) \wedge \neg Attends(s,c_1)]\}$$

ניתן לנסח ב-RA כל שאילתה שלא תלויה בתחום שניתן ב-RC.

הערה: התקבלו תשובות שהניחו כי הכוונה ב-"במטרה להחזיר את כל הסטודנטים שאינם לוקחים לפחות קורס אחד" היתה שאינם לוקחים אף קורס.

ג. הניחו כי הסכמות נתונות כ-EDB.

הסתכלו על הגרף בו הסטודנטים הם הצמתים ויש קשת בין סטודנטים <u>שונים</u> אם הם לוקחים את אותו הקורס.

כתבו תוכנית Datalog, במידת הצורך עם שלילה, המחזירה את כל זוגות הסטודנטים שמרחקם בגרף (כלומר, אורך המסלול הקצר ביותר ביניהם) שווה בדיוק ל-3 (המרחק בין סטודנט לעצמו הוא 0).

אם נדרש סמנו כל רובד בנפרד. (8 נק').

$$Dis0(A,A) \leftarrow Student(A,X)$$

$$Dis1(A,B) \leftarrow \neg Dis0(A,B), Attends(A,C), Attends(B,C)$$

$$Dis2(A,B) \leftarrow \neg Dis1(A,B), Dis1(A,C), Dis1(B,C)$$

$$Dis3(A,B) \leftarrow \neg Dis0(A,B), \neg Dis1(A,B), \neg Dis2(A,B), Dis1(A,C), Dis2(B,C)$$

רבודה:

$$E_0 - EDB, Dis0$$

$$E_1 - Dis1$$

$$E_2 - Dis2$$

$$E_3 - Dis3$$

שאלה 3 – SQL

נתון מסד הנתונים הבא המייצג סופרמרקטים, מוצרים והספקים שלהם:

Supplier:

pId	Name

Supermarket:

sId	District

Inventory:

Id	sId	pId	ExpDate	Quantity

הטבלה Supplier מחזיקה את מזהה ושמות הספקים:

- pId – מזהה הספק.
- Name – שם הספק.

הטבלה Supermarket מחזיקה את מזהה ומחוזות הסופרמרקטים השונים:

- sId – מזהה הסופרמרקט.
- District – מחוז הסופרמרקט.

הטבלה Inventory מחזיקה מידע עבור המוצרים השונים:

- Id – מזהה המוצר.
- sId – מזהה הסופרמרקט שבו נמצא המוצר, מפתח זר לsId בטבלה Supermarket.
- pId – מזהה הספק של המוצר, מפתח זר לpId בטבלה Supplier.
- ExpDate – מועד פקיעת התוקף של המוצר. ערך זה אינו יכול להיות NULL.
- Quantity – כמות היחידות ממוצר זה. ערך זה אינו יכול להיות שלילי.

מפתחות מסומנים בקו תחתון.

מזהים ו-Quantity הם מספרים, שמות/מחוז הם מחרוזות (TEXT), ExpDate מוגדר כתאריך (DATE).

א. כתבו פקודת SQL אשר מגדירה את הסכמה Inventory. הניחו כי שתי הטבלאות האחרות כבר מוגדרות (3 נק').

```
CREATE TABLE Inventory (

ID INTEGER,

sID INTEGER,

pID INTEGER,

ExpDate DATE NOT NULL,

Quantity INTEGER,

CHECK(Quantity>=0),

PRIMARY KEY(ID, sID, pID),

FOREIGN KEY(sID) REFERENCES supermarket(sID) ON DELETE CASCADE,

FOREIGN KEY(pID) REFERENCES Supplier(pID) ON DELETE CASCADE);
```

ב. מוצר חסר מוגדר להיות מוצר עם כמות אפס (Quantity=0) או שהוא פג תוקף (CURRENT_DATE > ExpDate). כתבו שאילתת SQL המחזירה את מזהי כל המוצרים החסרים בסופרמרקט כלשהו ללא חזרות.
הערה: ניתן להשתמש במשתנה CURRENT_DATE לצורך בדיקת התאריך הנוכחי.
(2 נק')

```
SELECT DISTINCT id

FROM inventory

WHERE quantity=0 OR CURRENT_DATE > expdate
```

ג. הסבירו מי מבין ההגדרות הבאות של אינדקסים צפויה לייעל את חישוב השאילתה בצורה הטובה ביותר. בנוסף, הסבירו למה כל אחת מהאפשרויות האחרות פחות טובה מהאינדקס שבחרתם. (3 נק')

1. $CREATE\ INDEX\ ON\ Inventory(ExpDate, Quantity);$
2. $CREATE\ INDEX\ ON\ Inventory(Quantity, ExpDate);$
3. $CREATE\ INDEX\ ON\ Inventory(ExpDate);\ CREATE\ INDEX\ ON\ Inventory(Quantity);$

בשני האינדקסים האחרים העובדה שקיים לנו מיון לפי התאריך ואז כמות או להיפך פחות עדיפה מאחר ועדיין נצטרך לעבור על כל הערכים השונים עבור האטריביוט השני ולכן עדיף שני אינדקסים

ד. סופרמרקט הינו "נחוץ" אם הוא מוכר מוצר שלא נמכר בסופרמרקט אחר באותו מחוז, כלומר, המוצר מופיע ביחד עם סופרמרקט זה בטבלה Inventory ולא עם אף אחד מהסופרמרקטים האחרים במחוז.
כתבו שאילתת SQL המחזירה את מזהי הסופרמרקטים הנחוצים במערכת. (5 נק')

```
SELECT s.sid

FROM inventory AS i JOIN supermarket AS s on i.sid=s.sid

WHERE NOT EXISTS (

    SELECT *

    FROM inventory AS i2 JOIN supermarket AS s2 on i2.sid=s2.sid

    where s2.district=s.district AND s2.sid<>s.sid AND i2.id=i.id

 )
```

ה. נרצה לדעת מי הם המוצרים שהמלאי שלהם במחוז 'North' גדול (ממש) מהמלאי שלהם במחוז 'South'.
מלאי של מוצר במחוז כלשהו מוגדר להיות סך כל המלאים בכל הסופרמרקטים במחוז זה.
כתבו שאילתת SQL אשר מחזירה זוגות של מזהה המוצר והמלאי שלו במחוז 'North' המקיימים את התנאי לעיל.
שימו לב: כי ייתכן מצב שמוצר נמצא במחוז 'North' ולא ב-'South' ובמקרה זה צריך להחזיר אותו גם כן. (8 נק')

```
SELECT ID, SUM(quantity) AS quantity
From inventory AS I JOIN supermarket AS S ON I.sid=S.sid
WHERE district='North'
GROUP BY id
HAVING SUM(quantity) > COALESCE(
    (SELECT SUM(quantity)
    FROM inventory as I2 JOIN supermarket AS S2 ON I2.sid=S2.sid
    WHERE district='South' AND I2.id=I.id
    GROUP BY ID), 0
)
```

שאלה 4 – Concurrency Control

יהי s תזמון בעל התכונה שכל פריט (משתנה) מופיע לכל היותר פעמיים. לדוגמא, x מופיע בדיוק פעמיים בפעולות $W_j(x)$ ו-$R_i(x)$, y מופיע בדיוק פעמיים בפעולות $R_i(y)$ ו-$R_i(y)$, ו-z מופיע פעם אחת תחת $W_j(z)$.

הוכח כי s בר סדרתיות מבטים (view serializable) אם ורק אם s בר סדרתיות קונפליקטים (conflict serializable). (11 נק')

אם s בר סדרתיות קונפליקטים אז s בר סדרתיות מבטים כי הראשון גורר את האחרון. אז נניח ש-s בר סדרתיות מבטים, ונוכיח שהוא בר סדרתיות קונפליקטים. יהי r תזמון סדרתי שקול מבטים ל-s. נראה כי r שקול קונפליקטים ל-s, ולכן s בר סדרתיות קונפליקטים.
נסתכל על זוג פעולות $A_i(x)$ ו-$B_j(x)$ בקונפליקט ונניח ש-$A_i(x)$ מופיע לפני $B_j(x)$ ב-s. צריך להוכיח ש- $A_i(x)$ מופיע לפני $B_j(x)$ ב-r. אם שתי הפעולות הן כתיבה אז $B_j(x)$ היא זאת שכותבת את הערך הסופי ב-s ולכן היא השנייה ב- r כיוון שגם שם היא כותבת את הערך הסופי. אם $A_i(x)$ היא קריאה ו-$B_j(x)$ היא כתיבה, אז $A_i(x)$ קוראת את הערך ההתחלתי ב-s ולכן גם ב-r, ומכאן מופיעה לפני $B_j(x)$. אם $B_j(x)$ הוא קריאה אז $A_i(x)$ היא כתיבה ואז $B_j(x)$ קוראת את הערך שנכתב ע"י $A_i(x)$ ב-s, ולכן גם ב- r, ומכאן $A_i(x)$ מופיעה לפני $B_j(x)$ ב-r.

שאלה 5 - XML

שאלה זו עוסקת במסמך ExamSystem.xml אשר מציית למסמך ה-DTD הבא:

```
<!DOCTYPE ExamSystem [
    <!ELEMENT ExamSystem ((Exam)+, (StaffMember)*, (Class)+)>
    <!ELEMENT Student (Last, First)>
    <!ATTLIST Student
        id ID #REQUIRED>
    <!ELEMENT StaffMember (Last, First)>
    <!ATTLIST StaffMember
        id ID #REQUIRED>
    <!ELEMENT Supervisor (Last, First)>
    <!ELEMENT Class ((Supervisor)+, (Student)+)>
    <!ATTLIST Class
        id ID #REQUIRED
        occupancy CDATA #REQUIRED>
    <!ELEMENT Exam (#PCDATA)>
    <!ATTLIST Exam
        id ID #REQUIRED
        name CDATA #REQUIRED
        classes IDREFS #REQUIRED
        staff IDREFS #REQUIRED>
    <!ELEMENT Last (#PCDATA)>
    <!ELEMENT First (#PCDATA)>
]>
```

א. לפניכם מסמך לדוגמה אשר ידוע שהוא לא מציית ל-DTD לעיל. ציינו את ההפרות לפי הכללים שנלמדו בקורס ונמקו על סמך ה-DTD. (6 נק')

```
<ExamSystem>
    <Exam id="DB_236363" name="Databases" classes="Class_1" staff="Staff_1 Student_1234"></Exam>
    <Class id="Class_0" occupancy="50">
        <Supervisor>
            <Last>VisorLast</Last>
            <First>VisorFirst</First>
        </Supervisor>
        <Student id="Student_1234">
            <Last>Israeli</Last>
            <First>Israel</First>
        </Student>
    </Class>
    <StaffMember id="Staff_1">
        <Last>Kisous</Last>
        <First>Roei</First>
    </StaffMember>
    <StaffMember id="Staff_1">
        <Last>Kimelfeld</Last>
        <First>Benny</First>
    </StaffMember>
</ExamSystem>
```

יש 3 בעיות:

- למבחן יש רפרנס למזהה שאינו קיים במערכת.
- שני אלימנטים עם אותו מזהה
- הסדר של האלימנטים אינו תואם את ההגדרה של הDTD

שימו לב: העובדה שבמזהה הסגל יש רפרנס לסטודנט אינה שגיאה.

ב. האם יתכן מסמך XML שמתאים ל-DTD אך אינו מכיל את התווית StaffMember? הסבירו תשובתכם. (2 נק')

ייתכן מצב כזה. StaffMember מוגדר להיות עם * מה שלא מחייב קיום אחד. בנוסף, מאחר והאטריביוט staff במבחן יכול להיות מחרוזת ריקה אין חובה של קיום מזהה או תווית StaffMember (הוא יכול להכיל גם הצבעה למזהים אחרים – אין אכיפה)

ג. נתונה השאילתה הבאה:

$$//Class[@occupancy = '50'\ and\ count(../Exam[id(@classes)[@occupancy < '50']])]$$

הסבירו במילים מהי משמעות השאילתה? (4 נק')

את הכיתות עם תפוסה 50 בתנאי שקיים לפחות מבחן אחד במערכת עם כיתה בתפוסה קטנה מ-50

שאלה 6 – mongoDB, Neo4j

לרשותכם מסד נתונים השייך לאו"ם ב-mongoDB ובו אוסף (Collection) בודד הנקרא Vaccines אשר מכיל מידע על אזרחים וחיסוני הקורונה אותם קיבלו. כל מסמך באוסף הוא מהצורה הבאה:

```
{
    _id: <ObjectId>,
    name: <string>,
    age: <int>,
    country: <string>,
    vaccines: [
        {
            vaccineName: <string>,
            cost: <int>,
            day: <int>,
            month: <int>,
            year: <int>
        }
        ,…,
        {
            vaccineName: <string>,
            cost: <int>,
            day: <int>,
            month: <int>,
            year: <int>
        }
}
```

דוגמא למסמך אפשרי:

```
{
    “_id”: ObjectId(056ab83901a09b),
    “name”: "Roei",
    "age": 25,
    “country”: "Israel",
```

```
“vaccines”: [
        {
                “vaccineName”: "Covid19-dose1",
                “cost”: 0,
                “day”: 2,
                “month”: 2,
                “year”: 2021
        }
        ,
        {
                “vaccineName”: "Covid19-dose2",
                “cost”: 0,
                “day”: 23,
                “month”: 2,
                “year”: 2021
        }
}
```


במידה ואזרח לא קיבל חיסון או בן פחות מ-12 – רשימתו תהייה ריקה.

נתונות השאילתות הבאות ב-mongoDB:

- db.Vaccines.mapReduce(
  function(){ emit(this.country, this.vaccines.length); },
  function(key, values){ return (Array.sum(values)/values.length); },
  {
  out: {"result"},
  query: { age: {$gte: 12}}
  }
)

- db.Vaccines.aggregate([
  $match : { age: { $gte: 12 } },
  $group: { _id: "$country" , value: { $avg: {$size: { "$vaccines" } } }
  ])

א. מה מחזירה כל אחת מהשאילתות הנ"ל? הסבירו כל אחת מן השורות בשאילתה הראשונה (mapReduce). בנוסף הסבירו כיצד השאילתה מגדירה את חלוקת העבודה בין השרתים. (8 נק')

שתי השאילתות מחזירות לכל מדינה את ממוצע החיסונים שעשה אדם בן 12 או יותר.

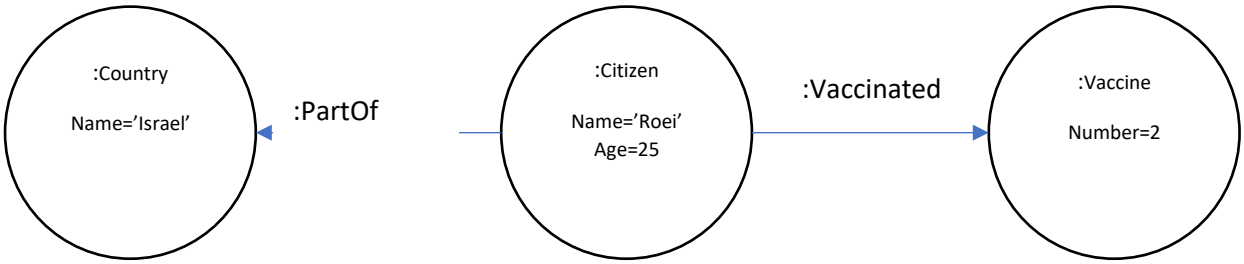
שאילתת הMapReduce תמפה כל מדינה (מפתח) לשרת אחר ותבצע את השלב הreduce בצורה מבוזרת ללא תלות בין רשומות שלא שייכות לאותו המפתח (מדינה).

באו"ם לא היו מרוצים מהשימוש ב-mongoDB ולכן החליטו להשתמש ב-Neo4j.

הגרף מומש לפי הכללים הבאים:

1. כל צומת בגרף מחזיק בתווית (label) אחת בדיוק מבין האפשרויות הבאות: **Citizen**, **Country**, **Vaccine**.
2. כל אחד מן הצמתים **Citizen**, **Country** מחזיק בתכונה (Attribute) בשם **Name**.
3. צמתים בעלי תווית **Citizen** מחזיקים בתכונה בשם **Age**.
4. צמתים בעלי תווית **Vaccine** מחזיקים בתכונה בשם **Number** (בעלי ערך 0\1\2).
5. צמתים בעלי תווית **Citizen** יכולים להיות מחוברים לצמתים בעלי תווית **Country**, ללא הגבלת כמות, ע"י קשר בעל תווית (label) **PartOf**.
6. צמתים בעלי תווית **Citizen** יכולים להיות מחוברים לצמתים בעלי תווית **Vaccine**, לכל היותר פעם אחת, ע"י קשר בעל תווית (label) **Vaccinated**.

להלן **דוגמא** לגרף המקיים את ששת הכללים:



ב. מה מחזירה השאילתה הבאה: (4 נק')

```
MATCH (C:Country {Name: "Israel"}) <- [:PartOf] – (P:Citizen) – [:Vaccinated] ->
                                        (V:Vaccine {Number: 2})
WITH count(P) as x
MATCH (C1:Country {Name: "Israel"}) <- [:PartOf] – (P1:Citizen)
RETURN  x>=count(P1)*0.5
```

i. השאילתה מחזירה האם לפחות 50% מהאוכלוסייה בישראל מחוסנת בשתי מנות.
ii. השאילתה מחזירה האם מספר החיסונים שניתנו בישראל הוא לפחות חצי מכמות האוכלוסייה.
iii. השאילתה מחזירה האם לפחות 50% מהאוכלוסייה בישראל לא מחוסנת בשתי מנות.
iv. השאילתה מחזירה האם לפחות 50% מהאוכלוסייה בישראל קיבלו חיסון כלשהו.
v. פלט השאילתה אינו מוגדר היטב.

שאלה 7 - RDF

בשאלה זו הניחו את קיום ה-namespaces הבאים:

- rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
- rdfs: <http://www.w3.org/2000/01/rdf-schema#>
- ex: <http://example.maman.cs.technion/>

התבוננו במאגר הנתונים הבא:

ex:Iza	ex:servesDrink	ex:BlueMoon
ex:Macolet	ex:servesDrink	ex:BlueMoon
ex:Macolet	ex:servesDrink	ex:Malka
ex:NolaSocks	ex:servesDrink	ex:Guinness
ex:NolaSocks	ex:servesFood	ex:chips
ex:Amsterdam	ex:servesFood	ex:chips
ex:Macolet	ex:near	ex:NolaSocks
ex:NolaSocks	ex:near	ex:Macolet
ex:Amsterdam	ex:near	ex:Iza
ex:Iza	ex:near	ex:Amsterdam

א) רשמו את תוצאת ההפעלה של כל אחת מן השאילתות הבאות על מאגר הנתונים הנתון:

1. (3 נקודות)

```
SELECT ?p1 ?b ?p2 {
        ?p1 ex:servesDrink ?b
        OPTIONAL {
                ?p1 ex:near ?p2.
                ?p2 ex:servesFood ex:chips
        } .
        ?p2 ex:servesDrink ?b
}
```

{?p1←ex:Nola, ?b←ex:Guinness, ?p2←ex:Nola }

2.(3 נקודות)

```
SELECT ?p1 ?p2 {
        ?p1 ex:near ?p2
        MINUS
        { ?p2 ex:near ?p3.
          ?p3 ex:servesDrink ?d.
          ?p3 ex:servesFood ex:chips
        }
}
```

{?p1←ex:Iza, ?p2←ex:Amsterdam},
{?p1←ex: Amsterdam, ?p2←ex:Iza},
{?p1←ex: Macolet, ?p2←ex:NolaSocks}

3.(3 נקודות)

```
SELECT ?p1 ?b ?p2 {
        { ?p1 ex:servesDrink ?b
          OPTIONAL {?p2 ex:servesDrink ?b}
        }
        MINUS
        { ?p2 ex:near ?p3.
          ?p3 ex:servesDrink ex:Guinness
        }
}
```

{?p1←ex: Iza, ?b←ex:BlueMoon, ?p2←ex:Iza},
{?p1←ex: Macolet, ?b←ex:BlueMoon, ?p2←ex:Iza},
{?p1←ex:NolaSocks, ?b←ex:Guinness, ?p2←ex: NolaSocks }

ב) (3 נקודות) הוסיפו לגרף הנתון שלישייה חדשה, כך שהשאילתה

```
SELECT ?p1 {
        ?p1 rdf:type ex:pub
}
```

כאשר תרוץ על מנוע שתומך ב-RDFS תניב את הפלט הבא:

{?p1←ex:Iza}, {?p1←ex:Macolet}, {?p1←ex:NolaSocks}

תשובה:

ex:servesDrink	rdfs:domain	ex:pub