מערכות קבצים
234322

החוק היבש...

מרצה: גלה ידגר

שעת קבלה: ימי ב' 12:30-13:30

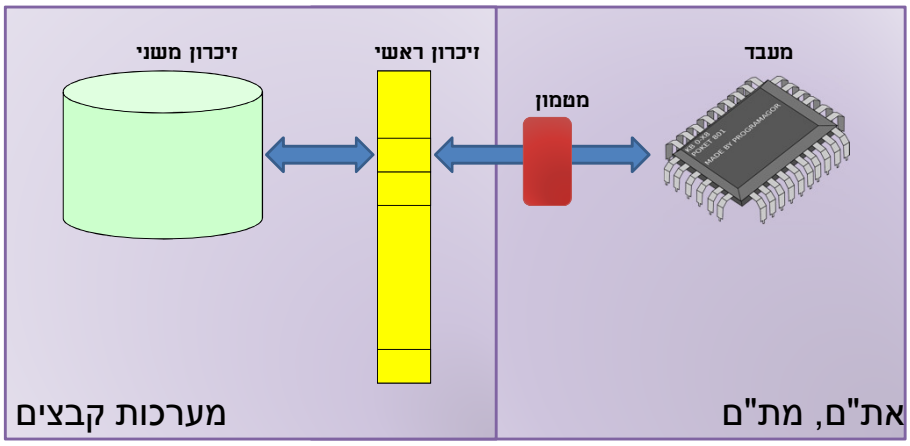
חישוב הציון:

- הציון התקף מורכב מ: 3% * 4 תרגילים יבשים - מגן
  88% בחינה סופית
- אם הציון בבחינה הסופית נמוך מ 45, הציון בקורס זהה לציון בבחינה הסופית

דרישות קדם: מבני נתונים 1, הסתברות, רצוי מערכות הפעלה

מבוא

ארגון המחשב



זיכרון ראשי לעומת זיכרון משני

סוג זיכרון	נפח	מחיר	מחיר ל-1GB	מהירות גישה/עדכון	זמן גישה אקראית
ראשי (SRAM)	4 GB	30$	7.5 $	1333 MHz	9 nsec
משני (hard disk)	1 TB	80$	8 cents	7200סל"ד	4.16 msec

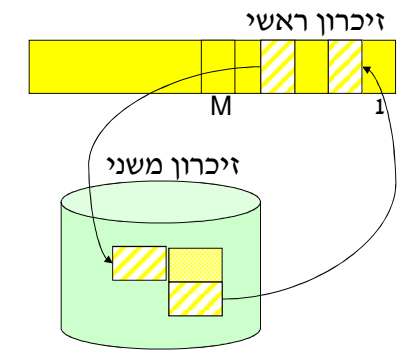
יחס בין זמני גישה אקראית:
ראשי : משני ≈ 1 :460,000

זיכרון משני:
- שמירת כמויות גדולות של מידע, במחיר זול יחסית
- שמירת מידע באופן אמין וממושך (גם כאשר המערכת כבויה)

זיכרון משני
✓ מחיר
✓ נפח
× מהירות

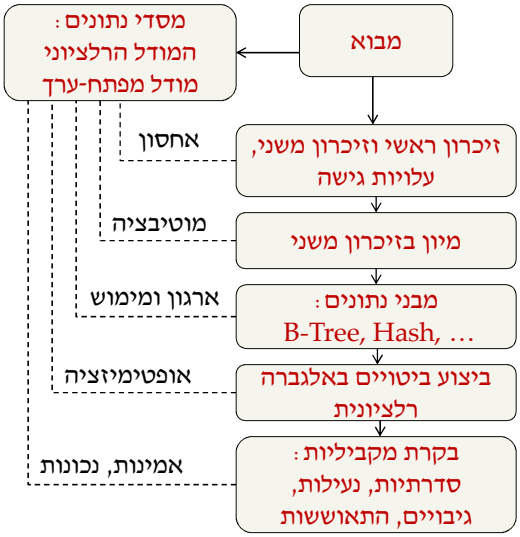
ניהול הזיכרון המשני

- הנתונים נמצאים בזיכרון המשני, מחולקים לבלוקים
- לא מבצעים חישובים ישירות בזיכרון המשני, אלא מביאים בלוק של נתונים לזיכרון הראשי ושם מעבדים אותם
- הזיכרון הראשי מחולק למסגרות, כ"א יכולה להכיל כל בלוק מהזיכרון המשני
  M מתוכן מוקצות לביצוע שאילתות
- הזיכרון הראשי קטן יחסית לזיכרון המשני, ולפעמים צריך לפנות בלוקים
  (בלוקים מעודכנים נכתבים לזיכרון המשני)



כלל 1: קריאה וכתיבה לזיכרון המשני מתבצעות רק בבלוקים

שאלות ונושאים בהם עוסק הקורס



- ✓מהו המבנה של מערכות אחסון מידע ומה השפעתו על הביצועים שלהן?
- ✓איך לתכנן מבני נתוני גדולים מאוד שלא נכנסים לזיכרון הראשי?
- ✓איך מחשבים את העלות (המקורבת) של ביצוע פעולות?
- ✓איך לאפשר גישה במקביל והתאוששות?

מערכת קבצים

מערכת תוכנה המטפלת בניהול, ארגון, שליטה ומתן גישה ושירותים למשתמשים בקבצים.

מערכת ניהול מסדי נתונים

מערכת תוכנה המאפשרת יצירת וניהול מסדי נתונים מאפשרת ארגון, שליטה ומתן גישה ושירותים למאגרי מידע מובנים

סוגי קבצים

לא מובנים (unstructured): המבנה אינו ידוע למערכת הקבצים
- ניתן לגשת רק לקובץ השלם (לקרוא, לכתוב או למחוק)
- למשל: קבצי אודיו, וידאו

מובנים (structured): סדרה ארוכה של רשומות בעלות מבנה זהה או דומה.
– בדרך כלל, מאורגנות על-פי שדה מפתח (אחד או יותר).

במיוחד: קבצים הנוצרים על ידי מערכת לניהול מסדי נתונים

- ניתן לגשת לכל רשומה (ולרוב גם לעדכנה) בנפרד
- פעולות:
  – חיפוש רשומה, עדכון רשומה, הכנסת רשומה, מחיקת רשומה
  – מעבר סדרתי על כל הרשומות
  – חיפוש תת-תחום – חיפוש רשומות הממלאות תנאי לוגי נתון
  – פעולות הקבצה (aggregation) : חישוב סכום, ממוצע וכדומה

מסדי נתונים רלציונים (טבלאיים)

דרך מקובלת לארגון מאגרי נתונים גדולים
- המידע נמצא בטבלאות (structured)
- המערכת שומרת את המידע בקבצים מתאימים ומנהלת את הקבצים
  - יצירת אינדקס לטבלה
  - מתן הרשאות גישה למשתמשים שונים לפרטי מידע שונים

רשומות ושדות

רשומה (record) - יחידת מידע בעלת מבנה זהה לאחרות (או דומה)
- קבצים בעלי רשומות שוות גודל (fixed size records)
- קבצים בעלי רשומות עם גודל משתנה (variable size records)
- בקורס זה נניח שהרשומות הן שוות גודל

שדה (field) - כל רשומה מתחלקת לשדות

שדה מפתח (key field) - מזהה את הרשומה לצרכי מיון או חיפוש

שדה מפתח ראשי (primary key field) - משמש לארגון הפנימי של הקובץ

שם השדה	מספר סטודנט	שם סטודנט	ציון בת"ב 1	ציון בת"ב 2	ציון בת"ב 3	ציון במבחן מועד א'	ציון במבחן מועד ב'
גודל השדה בבתים	9	20	1	1	1	1	1

יחס (רלציה, טבלה)

שורה בטבלה היא רשומה

תחום (domain): קבוצה של ערכים אפשריים עבור שדה ברשומה
- כל המחרוזות באורך 20
- מיקוד (כל המספרים בין 10000 ל-99999)
- תאריך (31.12.2999 -1.1.1900)
- מספר שלם

יחס (relation): רב-קבוצה (multiset) של איברים מהמכפלה הקרטזית של התחומים

$Customer \subseteq string \times no \times city$

עיר מגורים	מספר חשבון	שם פרטי
ת"א	23412	משה
חיפה	56234	שרה
גדרה	90567	דוד
עפולה	4591	אליהו
חיפה	62902	מינה

סכמת היחס

relation scheme: שם היחס והתכונות שלו, ותחומיהם.

שם היחס: בדוגמא הקודמת - Customer

התכונות (attributes): בדוגמא הקודמת - {שם פרטי, מספר חשבון, עיר}

דוגמאות (קצת שונה):

Branch(b_name, assets, b_city)
Customer(c_name, street, c_city)
Deposit(b_name, account_number, customer_name, balance)
Borrow(b_name, loan_number, c_name, amount)

לפעמים נצרף את שם היחס S לשם התכונה a ונכתוב S.a,
למשל Branch.b_name או Deposit.b_name

סדר וכפילויות

בפועל, סדר שמירת הרשומות משפיע על המימוש של חלק מהפעולות.

- לעיתים, משתמשים מעוניינים בתוצאה ממוינת או רק ברשומות הראשונות של יחס (על פי סדר כלשהו)

בפועל, במערכות לניהול מסדי נתונים טבלאיים, יחס הוא רב קבוצה (רשומה יכולה להופיע יותר מפעם אחת).

- אנחנו נניח כי כל הפעולות הן פעולות על רב-קבוצות.

שאילתות במסד נתונים רלציוני

פעולות אבסטרקטיות על הנתונים באופן שאינו תלוי במימוש.

שאילתות נכתבות בשפה שמיועדת לבני-אדם, על-מנת לפשט אותן.
למשל, הצג את שמות כל הסטודנטים בני פחות מ-20, ב-SQL:

```
SELECT name
FROM   Students
WHERE  birth_year > 1992;
```

השאילתות מתורגמות לפעולות על יחסים (כמו תרגום Java לשפת מכונה),
אשר מגדירות אלגברה של יחסים (רלציונית)

בהמשך, נראה איך מבצעים את הפעולות על היחסים (בנפרד, או כחלק מביטוי מורכב)

פעולות על יחסים

<u>פעולות אונריות שמחזירות יחס חדש</u>

- π – הטלה (projection): בוחר חלק מהעמודות
- σ בחירה (selection): בוחר חלק מהרשומות
- δ: הפוך את היחס לקבוצה (השאר עותק אחד מכל רשומה)
- $\rho_{A \to B}$: החלפת שם של שדה
- Sort: מארגן מחדש את היחס על ידי מיון על פי תכונה או מספר תכונות

<u>פעולות בינריות שמחזירות יחס חדש</u>

- $\cup \cap \setminus \times$: איחוד, חיתוך, הפרש, מכפלה
- Join: צירוף

<u>פונקציות הקבצה על עמודה המכילה מספרים:</u> max, min, average

<u>פונקציית הקבצה על טבלה:</u> count

בחירה σ select

בוחר את הרשומות המקיימות תנאי נתון C

Customer

עיר מגורים	מספר חשבון	שם פרטי
ת"א	23412	אלון
חיפה	56234	אלה
גדרה	90567	אורן
חיפה	62902	הדס

$\sigma_{city = \text{חיפה}}(Customer)$

עיר מגורים	מספר חשבון	שם פרטי
חיפה	56234	אלה
חיפה	62902	הדס

$\sigma_C(R) = \{r \in R : r \text{ satisfies } C\}$

שאילתות טווח $\sigma_{a \le A_2 \le b}(R)$ מוגדרות באמצעות [a,b]

הטלה π projection

בוחרת חלק מעמודות היחס

Customer

עיר מגורים	מספר חשבון	שם פרטי
ת"א	23412	אלון
חיפה	56234	אלה
גדרה	90567	אורן
חיפה	62902	הדס

$\pi_{\text{שם, מספר}}(Customer)$

מספר חשבון	שם פרטי
23412	אלון
56234	אלה
90567	אורן
62902	הדס

$\pi_{\text{עיר}}(Customer)$

עיר מגורים
ת"א
חיפה
גדרה
חיפה

עשוי ליצור רשומות כפולות

$\pi_{\ell 1, \ell 2, \ldots, \ell n}(R)=\{(r_{\ell 1}, r_{\ell 2}, \ldots r_{\ell n}) : (r_1, \ldots, r_m) \in R\}$

איחוד ∪

R1 ∪ R2

עיר מגורים	רחוב מגורים	שם פרטי
ת"א	הרצל	אלון
חיפה	חושי	אלה
גדרה	הרצל	אורן
ת"א	הרצל	אלון
ת"א	הרצל	אלון
חיפה	חושי	כרמל

מספר המופעים של רשומה הוא סכום מספר המופעים בשני היחסים

$$R \cup S = \{r: r \in R \text{ or } r \in S\}$$

R1

עיר מגורים	רחוב מגורים	שם פרטי
ת"א	הרצל	אלון
חיפה	חושי	אלה
גדרה	הרצל	אורן
ת"א	הרצל	אלון

R2

עיר מגורים	רחוב מגורים	שם פרטי
ת"א	הרצל	אלון
חיפה	חושי	כרמל

דורש סכמות זהות

חיתוך ∩

R1 ∩ R2

עיר מגורים	רחוב מגורים	שם פרטי
ת"א	הרצל	אלון

מספר המופעים של רשומה הוא המינימום בין מספר המופעים בכל אחד מהיחסים

$$R \cap S = \{r: r \in R \text{ and } r \in S\}$$

R1

עיר מגורים	רחוב מגורים	שם פרטי
ת"א	הרצל	אלון
חיפה	חושי	אלה
גדרה	הרצל	אורן
ת"א	הרצל	אלון

R2

עיר מגורים	רחוב מגורים	שם פרטי
ת"א	הרצל	אלון
חיפה	חושי	כרמל

דורש סכמות זהות

הפרש \

R1

עיר מגורים	רחוב מגורים	שם פרטי
ת"א	הרצל	אלון
חיפה	חושי	אלה
גדרה	הרצל	אורן
ת"א	הרצל	אלון

R2

עיר מגורים	רחוב מגורים	שם פרטי
ת"א	הרצל	אלון
חיפה	חושי	כרמל

R1 \ R2

עיר מגורים	רחוב מגורים	שם פרטי
ת"א	הרצל	אלון
חיפה	חושי	אלה
גדרה	הרצל	אורן

מספר המופעים של רשומה הוא ההפרש בין מספר המופעים ביחס הראשון ומספר המופעים ביחס השני (אבל לא פחות מאפס)

$$R \setminus S = \{r: r \in R \text{ and } r \notin S\}$$

דורש סכמות זהות

מכפלה קרטזית ×

R

A1	A2	A3
a	b	c
a	b	c
c	b	d

S

B1	B2
a	b
c	b

R × S

A1	A2	A3	B1	B2
a	b	c	a	b
a	b	c	a	b
c	b	d	a	b
a	b	c	c	b
a	b	c	c	b
c	b	d	c	b

מספר הרשומות בתוצאה: (מספר רשומות S) × (מספר רשומות R)

$$R \times S = \{(r_1,r_2,\ldots,r_n,s_1,\ldots,s_m): r \in R \text{ and } s \in S\}$$

צירוף join: ⋈

יחס חדש שמתקבל משני יחסים שיש להם תכונות עם שמות זהים.

היחס מתקבל מצירוף הרשומות שמסכימות על התכונות עם השמות הזהים.

למשל R(name,address) ⋈ S(name,tel#) = Q(name, address,tel#)

name	address
Nick	2nd st
Jane	Maine
Jane	NYC

⋈

name	tel#
Nick	234
Jane	892

=

name	address	tel#
Nick	2nd st	234
Jane	Maine	892
Jane	NTC	892

עוד דוגמה ל join

customers מכיל לקוחות: מס' לקוח, שם וכתובת.

bills מכיל חיובים: מס' לקוח וסכום שיש לחייב.

customers

Name	Add	No
Jim	1st st	1
Mary	Main	4
Joe	5th Ave.	17
Alice	Main	23

⋈

bills

No	Sum
17	23.1
1	12.0
1	100.5

=

Name	Add	No	Sum
Jim	1st st	1	12.0
Jim	1st st	1	100.5
Joe	5th Ave.	17	23.1

ב- SQL

```
SELECT Name, Add, No, Sum
FROM customers, bills
WHERE customers.No = bills.No
```

באופן יותר פורמלי

קבוצת התכונות המשותפות ליחסים R ו-S היא A ($A = attr(R) \cap attr(S)$)

נגדיר join בין שורות:

- יהיו $r \in R, s \in S$ כך ש- $r.A = s.A$
- $r \bowtie s$ היא השורה שמתקבלת משרשור r ו-s
- תוך שמירת עותק אחד בלבד של תכונות A.

s = (Homer Simpson, 555-2121)
r = (Homer Simpson, 123 Fake St. Springfield)
r ⋈ s = (Homer Simpson, 123 Fake St. Springfield, 555-2121)

עבור היחסים

$R \bowtie S = \{ r \bowtie s : r \in R, s \in S, \pi_A(r) = \pi_A(s) \}$

ביצוע פעולות במעבר יחיד

קל לבצע פעולה בחירה σ במעבר יחיד

- נקרא את רשומות R בבלוקים
- לכל רשומה r ב R: אם r מקיימת את התנאי, נוציא אותה

באופן דומה, אפשר לבצע הטלה π או פעולת הקבצה, למשל (min, max, sum, average)
מספר גישות לדיסק: מספר הבלוקים ב R

גם איחוד אפשר לבצע במעבר יחיד
הקלט והפלט הן רב-קבוצות לא ממוינות.
מספר גישות לדיסק: מספר הבלוקים ב S + מספר הבלוקים ב R

מה עם פעולות בינריות אחרות?

ניתן לבצע במעבר יחיד אם היחס הקטן מבין השניים נכנס כולו לזיכרון

<u>דוגמה: R \ S, כאשר S נכנס לזיכרון הראשי</u>

- נשמור עותק מכל רשומה של S עם מונה של מספר המופעים שלה.
- נעבור על כל רשומה r ב R,
  אם היא מופיעה ב-S, נקטין את המונה ב 1.
- בסוף נחזיר כל איבר של S שהמונה שלו נותר חיובי,
  בריבוי המתאים.

דוגמה לביצוע R \ S

S

A	B
20	c1
11	c2
30	c3
20	c1
11	c2
11	c2

(בדיסק)

R

A	B
11	c2
40	c2
20	c1
11	c2
20	c1
20	c1
50	c5
40	c2

(בדיסק)

דוגמה לביצוע S \ R

S

A	B	#
20	c1	0
11	c2	1
30	c3	1

(בזיכרון הראשי)

R

A	B
11	c2
40	c2
20	c1
11	c2
20	c1
20	c1
50	c5
40	c2

(בדיסק)

פלט

(11,c2)
(30,c3)

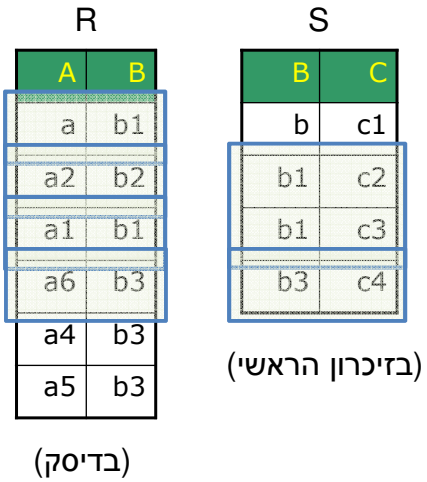
ביצוע צירוף במעבר יחיד

אפשרי אם היחס S נכנס לזיכרון

- נכניס את כל רשומות S לזיכרון הראשי (בבלוקים)
- נקרא את רשומות R (בבלוקים)
- לכל רשומה r ב R (כמידת ריבויה):
  - נחפש בין רשומות S שנמצאות בזיכרון הראשי
  - לכל רשומה s עבורה r.A = s.A, נוציא r ⋉ s לפלט

מספר גישות לדיסק: מספר הבלוקים ב S + מספר הבלוקים ב R

דוגמה להרצה



פלט

(a,b1,c2)
(a,b1,c3)
(a1,b1,c2)
(a1,b1,c3)
(a6,b3,c4)
.
.
.

ואם שני היחסים גדולים?

לולאה כפולה

נקרא בלוק של S ולכל רשומה s

נעבור על כל הבלוקים של R, ולכל רשומה r

נבדוק אם אפשר לצרף את r עם s

גישות לדיסק: מספר הרשומות ב S × מספר הבלוקים ב R

שיפור: ממלאים את הזיכרון בבלוקים של S (M בלוקים)

קוראים את R בבלוקים ומנסים לצרף כל רשומה של R עם כל הרשומות של S שנמצאות בזיכרון

גישות לדיסק (בערך): M/(מספר הבלוקים ב S × מספר הבלוקים ב R)

☝ כלל 2: גודל הזיכרון הראשי קובע את מספר הגישות לדיסק

שיפור אם היחסים ממוינים

נניח שהצירוף לפי תכונה אחת A, שהיא **מפתח** של S
וש R ו-S ממוינים לפי A

עוברים על R ו-S בבלוקים,
וכל פעם בוחנים רשומה s של S ורשומה r של R
אם r.A > s.A, עבור לרשומה הבאה ב-S
אם r.A < s.A, עבור לרשומה הבאה ב-R
אחרת, r.A = s.A, הוצא לפלט את הרשומה $r \bowtie s$
ועבור לרשומה הבאה ב-R

גישות לדיסק: מספר הבלוקים ב S + מספר הבלוקים ב R

דוגמה להרצה

S

A	C
b0	c1
b1	c2
b2	c3
b3	c4

R

B	A
a	b1
a1	b1
a2	b2
a6	b3
a4	b3

פלט

(a,b1,c2)
(a1,b1,c2)
(a2,b2,c3)
(a6,b3,c4)
.
.
.

☝ כלל 3: כדאי לגשת למידע באופן סידרתי

מסקנות

1: חייבים לגשת ליחסים (קבצים) בבלוקים

2: הגודל של הזיכרון הראשי קובע

3: מאוד כדאי לגשת לנתונים (קבצים / יחסים) באופן סידרתי
(ולכן טוב אם הם ממוינים)

מאגר מפתח-ערך (Key-Value Store)

- לכל אובייקט נתונים (ערך) מוצמד מפתח חד-ערכי לזיהוי האובייקט
- מבנה הנתונים ומשמעותם לא ידועים למערכת (unstructured)
- מפתחות: מחרוזת טקסט, מספר סידורי, תוצאת ערבול ועוד
- ערכים: קבצים מכל הסוגים, מצביעים וכתובות, URL, מחרוזת טקסט ועוד

מפתח	ערך
"תמונת_שער"	Front.jpg
"p_song"	0x112b210
"1.1"	www.cs.technion.ac.il/courses
"סיסמא"	“123abc??”

מאגרי מפתח-ערך

פעולות

- הכנסה: של ערך חדש על פי מפתח שלא נמצא עדיין במאגר
- הוצאה: של ערך קיים על פי מפתח שנמצא במאגר
- עדכון: של ערך קיים על פי מפתח שנמצא במאגר (בפועל החלפה של האובייקט)
- חיפוש: של ערך על פי מפתח שנמצא/לא נמצא במאגר
- פעולות מורכבות: שאילתות במודל MapReduce

דוגמאות

- Apache Cassandra (Some of Facebook's features)
- Amazon Simple Storage Service (Dropbox)
- Memcached

MapReduce

- ביצוע שאילתות מורכבות במאגר מפתח-ערך
- נתון מאגר התחלתי (בדרך כלל מבוזר)
- השאילתא מוגדרת בשני שלבים:
- 1. map (k1,v1) → list (k2,v2)
- 2. reduce (k2, list (v2)) → list(v3)

- המתכנת מגדיר את הסמנטיקה של הפונקציות map, reduce
- המערכת אחראית על
  - חלוקת תפקידים בין שרתים
  - Shuffle - העברת תוצאות שלב ה-map לשרת המתאים
  - פלט של התוצאות ממויינות על פי k2

דוגמא - MapReduce

- לחברה להגנת הטבע מספר רשימות של לקוחות ושותפים:
  - משתתפי חוגים (שם, דוא"ל, טלפון, גיל, כתובת, חוג)
  - תורמים (שם, דוא"ל, כתובת, סכום תרומה, תאריך תרומה אחרונה)
  - חברים (שם, כתובת, פרטי הוראת קבע, דוא"ל)

- סניף חיפה רוצה ליצור רשימת דיוור להפצת עלון אלקטרוני
- Map(name, v1)→ (city,(name,email))
- Reduce(city, list(name,email))→ list(name,email)

- על המתכנת לזהות את שדה הכתובת ולחלץ מתוכו את העיר
- ניתן "לזרוק" את תוצאות כל הערים שאינן חיפה (בכל אחד מן השלבים או בסוף)

דוגמא - MapReduce

Input

שם	כתובת	ה. קבע	דוא"ל
אלכס	הזמיר, חיפה	ויזה לאומי	alex@
אייל	השלום, ר"ג	בנק הפועלים	eyal@

שם	דוא"ל	כתובת	סכום	תאריך
אלה	ela@	ביאליק, ר"ג	300	13.8.13
אורלי	orly@	מוריה, חיפה	180	25.9.13

שם	דוא"ל	טלפון	גיל	כתובת	חוג
חנן	han@	12345	7	הרצל, חיפה	טבע
נועה	noa@	23456	13	אלנבי, ת"א	סיור

Map()

city	name	email
חיפה	אורלי	orly@
חיפה	חנן	han@
חיפה	אלכס	alex@

city	name	email
ר"ג	אייל	eyal@
ר"ג	אלה	ela@

city	name	email
ת"א	נועה	noa@

MapReduce - דוגמא

city	name	email
חיפה	אורלי	orly@
חיפה	חנן	han@
חיפה	אלכס	alex@

city	name	email
ר"ג	אייל	eyal@
ר"ג	אלה	ela@

city	name	email
ת"א	נועה	noa@

Reduce()

Output

name	email
אורלי	orly@
אלכס	alex@
חנן	han@

השוואה בין המודלים

יתרונות המודל הרלציוני

- ביטוי קשרים מורכבים בין נתונים
- מימוש יעיל (על ידי המערכת) של שפת שאילתות עשירה

יתרונות מאגר מפתח-ערך

- פשטות: יתרון במימוש ובהגנה מפני התקפות חיצוניות
- יעילות בפעולות פשוטות: כתיבה וחיפוש על פי מפתח
- קל למיקבול: מיעוט תלויות
- קל לביזור: על פני שרתי אחסון מרובים, בענן
- מקרה פרטי של NoSQL database