

## אופטימיזציה של שאילתות

בכל בסיס נתונים הטבלאות השונות שמורות כקבצים. שאילתות, מבחינת בסיס הנתונים, הינן גישה לאותם קבצים, בחירת רשומות מסוימות, על סמך תנאים מסוימים, והצגתן, במלואן או בחלקן. בהרבה מקרים ניתן להגיע לאותה תוצאה במספר דרכים, כשכל דרך יש משמעות מבחינת זמני הריצה מול הטבלאות השונות, כמות הרשומות הנשלפות, המשפיעות על כמות הזיכרון הנדרשת וכו'. שאילתא מתבצעת בצורה יעילה אם הרשומות נשלפות במהירות, וכמות הזיכרון הנדרשת לביצוע השאילתא היא קטנה ככל האפשר. ככל שהשאילתא מורכבת יותר, מספר האפשרויות לעיבוד השאילתא רב יותר, והפער בין שאילתא יעילה לכזו שאינה יעילה רב יותר. מכיון ששאילתא יכולה לפנות לכמות רבה של נתונים, ולדרוש זמן עיבוד רב, אנו מעוניינים שהשאילתא תבצע באופן היעיל ביותר.

נניח שאנו רוצים לקבל את העמודה Y של כל הרשומות שבהן X=17. באלגברה רלציונית השאילתא נכתבת כ- $\pi_Y \sigma_{X=17} r$ . המשמעות של השאילתא היא שאנו שולפים את כל הרשומות שבהן X=17, ומהרשומות הללו אנו בוחרים את עמודה Y. אולם, אם נסתכל על השאילתא המקבילה  $\pi_Y \sigma_{X=17} \pi_{X,Y}$ , אנו נראה שהיא יעילה יותר. תחילה אנו שולפים את העמודות X ו-Y של הטבלה שלנו, והן אנו בוחרים רק את אלו שבהן X=17, ואח"כ לוקחים רק את עמודה Y. באופן הזה אנו משתמשים בפחות זיכרון.

לרוב האופן שבו מנוסחת שאילתא מציע את הדרך המיידית לקבלת התוצאה. אולם מכיון שמנסח השאילתא אינו מודע בהכרח לשיקולי יעילות, המערכת תמיר את השאילתא לשאילתא שקולה שמחיר העיבוד שלה נמוך יותר.

לצורך הדוגמאות שלנו נשתמש בסכמות הבאות:  $Customer(CName, Street, CCity)$ ,  $Deposit(Account, CName, BName, Balance)$  ו-  $Bank(BName, BCity, Assets)$ .

נתונה השאילתא הבא: הצג שם סניף, משאבי סניף, עבור סניפים להם לקוח מחיפה.

$$\pi_{BName, Assets} [\sigma_{City=Haifa} (Bank \bowtie_{BName} Deposit \bowtie_{CName} Customer)]$$

כדי לעבד את השאילתא כפי שהיא מוצגת אנו צריכים לחשב את הצירוף הטבעי של שלושת היחסים, לבחור קומץ שורות מהיחס שהתקבל ואז לבחור את העמודות הרלוונטיות לנו.

אם הטבלאות גדולות, הצירוף הטבעי של שלושת היחסים הינו גדול במיוחד, ובמקרים רבים תיווצר בעיית זיכרון, ואז יהיה צורך ליצור קבצים זמניים, דבר שמאט את זמן הביצוע של השאילתא. על מנת להוזיל את התהליך, נשאף בכל שלב להישאר עם תוצאת ביניים קטנה ככל היותר.

$$\pi_{BName, Assets} [\sigma_{City=Haifa} (Bank \bowtie_{BName} Deposit \bowtie_{CName} Customer)]$$

⇓

$$\pi_{BName, Assets} [Bank \bowtie_{BName} Deposit \bowtie_{CName} (\sigma_{City="Haifa"} Customer)]$$

□ מסקנה - יש לבצע פעולות בחירה מוקדם ככל האפשר

נניח שאנו מעוניינים בשאילתא הבאה:

$$\pi_{BName,Assets} \left[ \sigma_{City=Haifa \wedge (Balance > 0)} (Bank \bowtie_{BName} Deposit \bowtie_{CName} Customer) \right]$$

שבה קיים התנאי  $(City = Haifa) \wedge (0 < Balance)$ . השדה balance שייך לטבלה Deposit, ואילו השדה City שייך לטבלה Customer, ולכן אפשר לכתוב את השאילתא הזאת באופן הבא:

$$\pi_{BName,Assets} [Bank \bowtie_{BName} (\sigma_{0 < Balance} Deposit) \bowtie_{CName} (\sigma_{City="Haifa"} Customer)]$$

□ **מסקנה - יש להמיר ביטויים מהצורה  $\sigma_{P_1 \wedge P_2}(E)$  לביטוי  $\sigma_{P_1}(\sigma_{P_2}(E))$**

$$\left[ \sigma_{A=1 \wedge B=2} (r_1(A, B) \bowtie_B r_2(B, C)) \right]$$

⇓

$$\left[ \sigma_{A=1} r_1(A, B) \bowtie_{B=2} r_2(B, C) \right]$$

פעולת ההשלכה (בחירת עמודות) היא אמצעי נוסף להקטנת תוצאות ביניים, ולכן יש לבצע אותה מוקדם ככל האפשר.

$$\pi_{BName,Assets} [Bank \bowtie_{BName} Deposit \bowtie_{CName} (\sigma_{City="Haifa"} Customer)]$$

אנו כאמור צריכים להציג את השדות BName ו-Assets עבור סניפים להם לקוח מחיפה. מצרף  $Bank \bowtie_{BName} Deposit$  מתקבלת טבלה בעלת השדות (Account, Assets, BCity, BName), כאשר אנו נדרשים רק לשני שדות מתוך השישה. כמו כן בשביל לעשות join עם טבלת Customer אנו צריכים שדה נוסף והוא CName. ולכן אפשר להמיר את השאילתא לשאילתא הבאה:

$$\pi_{BName,Assets} \left[ \left( \pi_{BName, Assets, CName} (Bank \bowtie_{BName} Deposit) \right) \bowtie_{CName} \left( \pi_{CName} (\sigma_{City="Haifa"} Customer) \right) \right]$$

□ **מסקנה - בצע פעולת השלכה מוקדם ככל האפשר**

דרך נוספת להקטנת תוצאות הביניים היא לבחור סדר אופטימאלי לפעולות מכפלה קרטזית ו-Join. מכיון שהפעולות הללו הן אסוציאטיביות, הרי ש- $(r_1 \times r_2) \times r_3 \equiv r_1 \times (r_2 \times r_3)$ . כמו כן, הפעולות הללו הן קומוטטיביות, ולכן גם אפשר לשנות את סדר ביצוע הפעולות, כלומר  $(r_1 \times r_2) \times r_3 \equiv (r_1 \times r_3) \times r_2$ .

קימות שקלויות נוספות שאפשר להשתמש בהן על מנת לקבל תוצאות ביניים קטנות יותר, ובנהן:

$$\begin{aligned} \sigma_p(r_1 \cup r_2) &\equiv \sigma_p(r_1) \cup \sigma_p(r_2) \quad \square \\ \sigma_p(r_1 - r_2) &\equiv \sigma_p(r_1) - r_2 \equiv \sigma_p(r_1) - \sigma_p(r_2) \quad \square \\ r_1 \cup r_2 &\equiv r_2 \cup r_1 \quad \square \\ (r_1 \cup r_2) \cup r_3 &\equiv r_1 \cup (r_2 \cup r_3) \quad \square \end{aligned}$$

על מנת לתכנן את האסטרטגיה הטובה ביותר, מחזיק בסיס הנתונים סטטיסטיקות שונות, בעזרתן הוא יכול לחזות את היעילות היחסית של הפעולות השונות.

נהוג להחזיק את הנתונים הבאים: (1)  $n_r$  - מספר השורות בטבלה  $r$ . (2)  $s_r$  - הגודל (בבתים) של רשומה בטבלה  $r$ . (3)  $V(A, r)$  - מספר הערכים השונים בתכונה  $A$  בטבלה  $r$ .

בהתאם לכך, גודלה של מכפלה קרטזית,  $r_1 \times r_2$ , מוערך כ- $(s_{r_1} + s_{r_2}) \cdot (n_{r_1} \cdot n_{r_2})$ . אם אנו צריכים לעשות בחירה, כגון  $\sigma_{A_i=C} r$ , אז ההערכה של פעולה זה היא  $\frac{n_r}{V(A_i, r)}$  (ההנחה היא שיש התפלגות אחידה של הערכים). הערכת גודלו של join,  $r_1$  ו- $r_2$ , מעט יותר מורכבת. נגדיר כ- $R_1$  את השדות של  $r_1$ , וכ- $R_2$  את השדות של  $r_2$ . אם  $R_1 \cap R_2 = \emptyset$ , הרי שלא מדובר על join אלא על מכפלה קרטזית. אם  $R_1 \cap R_2$  הוא מפתח על של  $r_1$ , אז כל רשומה של  $r_2$  תצורף לרשומה אחת (לכל היותר) של  $r_1$ , ולכן  $n(r_1 \bowtie r_2) \leq n(r_2)$ . אפשרות נוספת היא ש- $R_1 \cap R_2 = \{A\}$ . במקרה הזה, כל ערך  $a_i \in A$  יופיע  $\frac{n_{r_i}}{V(A, r_1)}$  פעמים ב- $r_1$ , ולכן כל רשומה מ- $r_2$  הכוללת ערך זה תצורף ל- $\frac{n_{r_i}}{V(A, r_1)}$  שורות מ- $r_1$ . מכיון ש- $n_{r_2} = r_2$ , אז גודל התוצאה היא  $n_{r_2} \frac{n_{r_i}}{V(A, r_1)}$ . משיקולי סימטריה היינו יכולים להעריך את מספר השורות כ- $n_{r_1} \frac{n_{r_2}}{V(A, r_1)}$ . אולם, אם  $V(A, r_1) \neq V(A, r_2)$ , כלומר, אם מספר הערכים בתכונה  $A$  בשתי הטבלאות שונה, אז תתקבל הערכה שונה. במקרה כזה, לוקחים את הערך הנמוך מבין השתיים.

בנוסף, מחיר עיבוד השאילתא מושפע מהגורמים הבאים: (1) האם הרשומות מקובצות לגושים. (2) האם הרשומות ממויינות. (3) האם קיימים אינדקסים או פונקציות hash. ככלל חיפוש באינדקס דליל עולה 2 גישות לדיסק ואלו חיפוש באינדקס צפוף עולה 3 גישות לדיסק.

האסטרטגיה לפיה מחליט בסיס הנתונים להריץ את השאילתא נקראת תוכנית הגישה (access plan) והוא מפרטת את:

- א. הפעולות האלגבריות שיש לבצע
- ב. סדר הפעולות לביצוע
- ג. האינדקסים בהם יעשה שימוש
- ד. סדר המעבר על הרשומות

**דוגמא:**

אם שם משפחה הוא מפתח ראשי אזי הקובץ ממוין על-פיו ולכן נבנה לשם המשפחה אינדקס דליל.

נניח שגורם הגושיות  $7 =$

נניח שיש לנו 50 תלמידים ששם משפחתם כהן, לכמה גישות לדיסק נזדקק כדי להביאם?

$$8 = \left\lceil \frac{50}{7} \right\rceil \quad (\text{לא כולל החיפוש באינדקס})$$

לעומת זאת אם המפתח הראשי הוא מ.ז אזי לתכונת שם המשפחה נבנה אינדקס צפוף. כדי לשלוף את 50 התלמידים ששם משפחתם הוא כהן ושנתוניהם מופיעים בקובץ מפוזרים נצטרך (לכל היותר) 50 גישות לדיסק (לא כולל עלות החיפוש באינדקס).

**דוגמא:** הצג את מספר החשבון, עבור החשבונות של יוסי כהן, המתנהלים בסניף המסניף

ויותרם  $< 0.0$ . הפקדה  $\pi$

לקוח=יוסי כהן  
חשניף="המסניף"  
יתרה<0

נניח כי יחס יחיד כולל את הנתונים וכי מתקיים:

- גורם הגושיות  $BF = 20 =$

- מספר הסניפים  $= 50 =$  (הפקדה, שם סניף)  $V$

- מספר הלקוחות  $= 200 =$  (הפקדה, שם לקוח)  $V$

- (הפקדה, יתרה)  $= 5000$   $V$

- הפקדה  $= 10^4$   $n$

- ליחס אינדקס דליל הבנוי כעץ  $B^+$  עבור התכונה שם סניף.

- ליחס אינדקס צפוף – עץ  $B^+$  עבור שם-לקוח.

- התפלגות הערכים בכל תכונה אחידה.

**אפשרויות:**

א.

- רשומות כוללות נתונים אודות סניף המסניף. (כדי לאתרן נשלם מחיר  $200 = \frac{10^4}{50}$ )

של שתי גישות עבור החיפוש באינדקס דליל).

- רשומות אלה מצויות בקובץ ברצף, ע"כ קריאתם תחייב  $\frac{200}{20} = 10$  פעולות קריאה,

וע"י כך נוכל לבחון אילו מהרשומות מספקות את שני התנאים האחרים, ואותן להציג.

ב. לחילופין:  $50 = \frac{10^4}{200}$  רשומות ביחס מתיחסות ליוסי כהן. היחס אינו ממויין על פי תכונה זו ועל-כן קריאתן יחייב 50 גישות לדיסק. (+ 3 גישות לחיפוש באינדקס).

**מסקנה:** זאת לא אסטרטגיה מוצלחת.

**הערה:** לו הקובץ היה לא ממויין על פי שם סניף, ולא ממויין על פי שם לקוח, אזי אסטרטגיה זאת הייתה עדיפה.

ג. לחילופין:

א. השתמש באינדקס על שם לקוח כדי לשלוף את המצביעים לרשומות של יוסי כהן (מחיר 3 גישות).

ב. השתמש באינדקס על שם – סניף כדי לשלוף את המצביעים לגושים הכוללים רשומות של סניף המסניף (2 גישות).

ג. פנה לרשומות המופיעות בשתי הקבוצות גם יחד והצג את אלה מביניהן בהן מצב חשבון  $< 0$ .

לכמה רשומות בקובץ נאלץ לפנות:

ב -  $1/50$  מהרשומות הסניף הוא סניף המסניף.  
 ב-  $1/200$  מהרשומות הלקוח הוא יוסי כהן.  
 ולכן ב-  $\frac{1}{50} \cdot \frac{1}{200} = \frac{1}{10^4}$  מהרשומות הסניף הוא המסניף והלקוח הוא יוסי כהן.

(תחת הנחת התפלגות אחידה, ואי-תלות הערכים בתכונות השונות).  
 מכיוון שבקובץ יש  $10^4$  רשומות אנו צפויים לבדוק רשומה יחידה.

הערה : החלופות שבדקנו לא שלפו רשומות על פי התנאי (מצב חשבון  $< 0$ ) והסיבה :

- אין אינדקס על תכונה זו.  
 - הפרדיקט  $<$  הינו בדר"כ פחות סלקטיבי מעמיתו = ועל כן צפוי לחייב שליפה של יותר רשומות.

ראינו בדיקה והשוואה של שלוש אסטרטגיות גישה שונות. כפי שהזכרנו קודם, לעתים המערכת תוכל לבחון מספר ביטויים שונים המייצגים את השאילתה שהוגשה.  
 במקרים מסויימים, ביטוי אלגברי שנראה על פניו פחות יעיל, עשוי בשל קיומם של אינדקסים להתגלות כיותר חסכוני בפעולות גישה לזכרון.

## Join Strategies - 9.5

מכיוון שפעולת הצירוף הינה ראשית פעולה שכיחה ושנית פעולה יקרה יש לנו מוטיבציה גבוהה להשתדל לבצע ביעילות.

מחיר הצירוף תלוי במספר גורמים :

א. אופן הארגון הפיזי של היחסים. בפרט סדר הרשומות (ממויינות או לא) והאם מקובצות לגושים.

ב. קיומם של אינדקסים ופונקציות ערבול.

נבחן כדוגמא את עלות (לקוח  $<$  הפקדה) תחת ההנחות :

$$n_{\text{הפקדה}} = 10^4$$

$$n_{\text{לקוח}} = 200$$

גודל גוש = 20 רשומות בכל קובץ

ותחת הנחות נוספות שנציג בהמשך.

### Simple Iteration – 9.5.1

נניח בתחילה שהיחסים **אינם ממויינים** (ע"פ התכונה שעל פיה נערך הצירוף = שם לקוח) וכי **אין אינדקסים** על פי תכונה זו.

לכן כדי לצרף את היחסים יש לבדוק כל זוג אפשרי של רשומות, אחת מכל יחס, כדי לראות האם יש בהן אותו ערך בתכונה שם לקוח.

לפיכך יש לבחון :  $10^4 \cdot 200$  זוגות.

כדי לבצע בחינה זו יש לבצע לולאה כפולה בה נחזיק רשומה מאחד הקבצים כקבועה, ונבחן עברה את כל רשומות היחס השני.

נניח שהלולאה החיצונית עוברת על יחס ההפקדה והפנימית על יחס הלקוחות :

מקרה #1 (המקרה הגרוע ביותר)

רשומות היחסים אינן מקובצות לגושים ולכן כדי לבדוק כל רשומה יש לקרוא לזיכרון באופן מיוחד.

מספר פעולות הקריאה יהיה ע"כ שווה למספר הזוגות הנבדקים כלומר :  $2 \cdot 10^6$

**מקרה #2**

כמו קודם אך הרשומות מקובצות לגושים . באופן מחדלי נניח כי לרשות כל קובץ עומד חוצץ יחיד.

$$\text{המחיר: } \frac{10^4}{20} = 500 \text{ פניות שלם קריאת יחס ההפקדה (לולאה חיצונית)}$$

$$\text{עבור כל רשומה } r \text{ ביחס ההפקדה: קריאת יחס הלקוחות בשלמותו בעלות } \frac{200}{20} = 10 \text{ פעולות}$$

קריאה. כדי לאתר את השורה/ות המצטרפות ל-  $r$

$$\text{העלות הכוללת: } 500 + 10^4 \cdot 10$$

**מקרה #3**

כמו מקרה 2 אך הפעם נהיה חכמים יותר בעיבוד :

עת גוש מקובץ ההפקדה, הכולל את הרשומות  $r_1, r_2, \dots, r_{20}$  נמצאות בזיכרון וקראנו גוש מקובץ הלקוחות הכולל רשומות  $s_1, s_2, \dots, s_{20}$  נבדוק התאמה של כל  $r_i$  לכל  $s_j$  (ולא רק התאמה של  $r_i$  יחידה אל מול כל ה-  $s_j$  כפי שנהגנו במקרה #2). ולכן רק עבור כל גוש של יחס ההפקדה יהיה עלינו לקרוא את כל יחס הלקוחות (ולא עבור כל רשומה).

$$\text{המחיר: } \frac{10^4}{20} + \frac{10^4}{20} \cdot \frac{200}{20}$$

↑ לשלף את כל הגושים של יחס ההפקדה
↑ מספר הגושים של יחס ההפקדה
↑ לקרוא את יחס הלקוחות בשלמותו

## 9.5.2 - צירוף מיזוג

### מקרה #5

שני היחסים ממויינים על פי התכונה על פיה נערך הצירוף (שם הלקוח בדוגמא שלנו). לדוגמא :

יוסי כהן ....		יוסי כהן ....
דנה לוי ....		יוסי כהן ....
דן לוינגר .....		דנה לוי .....
		דן לוינגר ....

- נעים במקביל על שני היחסים.

במקרה זה נצרף את היחסים על ידי מיזוגם. נחזיק מצביע לכל קובץ.

המצביע יאותחל לרשומה הראשונה בכל קובץ, ויקודם במקביל על שני הקבצים. כל פעם נקרא לזיכרון את כל הרשומות בעלות ערך זהה בתכונה על פיה אנו מצרפים, לחילופין נתקדם על פני אחד הקבצים על כל הרשומות שערך המפתח שלהן קטן מערך המפתח עליה מצביע המצביע בקובץ השני.

המחיר : אם ניתן להחזיק בזיכרון את כל הרשומות משני הקבצים החולקות אותו ערך בתכונה על פיה נערך הצירוף, אזי כל רשומה תיקרא פעם יחידה. אחרת יתכן שאותה רשומה תיקרא מספר פעמים.

### לדוגמא :

נניח כי נתונים הקבצים הבאים ורוצים לצרף על ידי מיזוג :

		2 ב3	1 א3
		↔	↔
		ג3 7	ד3 5
	א3 א3 ד'3 ד'3	ב3 ג3 ג'3 ב'3	

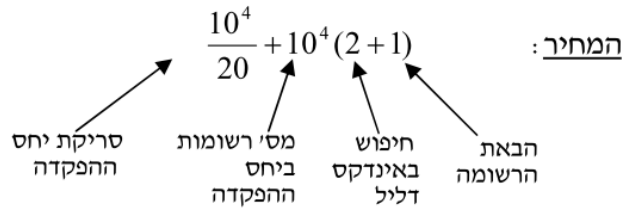
(יש לטעון כאן שוב את הגוש הראשון בקובץ השמאלי)

במקרים קיצוניים יתכן שנצטרך לטעון חלק מהגושים יותר מפעם אחת.

**9.5.3 - שימוש באינדקסים****מקרה #6**

ענה נניח כל ליחס הלקוחות (ממוין) קיים אינדקס דליל ע"פ שם לקוח (המהווה את מפתח היחס).

כדי לצרף את היחסים נסרוק סדרתית את יחס ההפקדה. עבור כל רשומה  $r$  מיחס זה נשתמש באינדקס כדי לשלוף את הרשומה המצטרפת ליחס הלקוחות.

**שאלות :**

- כמה מועיל לנו האינדקס ? (אפשר לבצע חיפוש בינארי כאשר  $n =$  מספר הגושים בקובץ).
- כמה הי עולה הצירוף לולא היה אינדקס.
- כמה היה עולה הצירוף לו היינו סורקים סדרתית את יחס הלקוחות ומשתמשים באינדקס (צפוף) על שם-לקוח ביחס ההפקדה כדי לאתר את הרשומות המצטרפות ?

**הערה :** החיסרון (לעומת אין אינדקס) מצדיק בנייה מיוחדת של אינדקס עבור השאילתא.

**Hash Join – 9.5.4****מקרה #7**

במקום באינדקס נבנה/נשתמש בפונקציה עירבול.

א. נעבור סידרתית על שני הקבצים.

עבור כל קובץ נבנה טבלת דליים נפרדת בשיטת העירבול הדינמי. כל דלי יחזיק זוגות של ערך מפתח+מצביע לקובץ הנתונים לרשומה המתאימה.

נניח שנקבל משהו כגון :



- ב. נסרוק במקביל את הטבלאות ונפנה לדליים התואמים ( #1 עם #3 ו-#4, #2 עם #5). במידה ואנו מוצאים בהם רשומות עם ערכי מפתח זהים נקרא את הרשומות ונצרפן.

**המחיר :**

א. כדי לבנות את טבלת הדליים יש לקרוא כל קובץ פעם יחידה. עלות הקריאה תלויה בשאלה האם הקובץ מקובץ / איננו מקובץ לגושים.

לשם הפשטות נניח כי את הטבלה אנו שומרים בזיכרון (וגם אם לא, סביר להניח שהיא יחסית קטנה...).

נניח כי את הטבלה אנו שומרים בזיכרון.

- ב. בעת סריקת הדליים וביצוע הצירוף תיקרא כל רשומה פעם יחידה לכל היותר (שימו לב אגב, כי על מנת שעירבול דינמי יצלח, יש לדרוש שגודל דלי  $\leq$  מספר הרשומות שעשויות לחלוק ערך מפתח זהה בקובץ).

סה"כ : אנו קוראים כל קובץ לכל היותר פעמיים.



### Three Way Join - 9.5.5

פעולת הצירוף הינה פעולה בינארית ביסודה, אולם נוכל להתעניין גם בצירוף של שלושה יחסים :

לקוח < > הפקדה < > סניף

נניח :  $n_{לקוח} = 200$

$n_{הפקדה} = 10^4$

$n_{סניף} = 50$

נבחן אפשרויות שונות לביצוע הצירוף הכפול :

א. על ידי לולאה משולשת שתרוץ על שלושת היחסים ותבחן כל פעם צירוף של רשומה מכל יחס.

נבחן לשם כך :  $10^4 \cdot 200 \cdot 50 = 10^8$  אפשרויות.

- ב. 1. צירוף לקוח < > הפקדה (איך ? – נעבור על אינדקס ביחס ההפקדה ) מכיוון ששם לקוח מהווה מפתח יחס הלקוחות, יהיה גודל היחס הנוצר  $\geq$  גודל יחס ההפקדה. (כל שורה ביחס ההפקדה תצטרף לכל היותר לשורה אחת ביחס הלקוחות ולכל הפחות לאף שורה).  
 2. בניית אינדקס ליחס הסניפים (למה צריך אינדקס ? (יותר יעיל) איזה מן אינדקס ? סביר להניח שיהיה זה אינדקס דליל על פי שם הסניף).  
 3. סריקת היחס שנוצר בסעיף 1, ועבור כל רשומה בו, שימוש באינדקס שנבנה בשלב 2 כדי לשלוף את הרשומה/ות הדרושות מיחס הסניפים (כמה ?).  
 יתבצע במחיר קבוע של שתי גישות לזיכרון לחיפוש באינדקס, ועוד אחת לשליפת הרשומות הרצויות לקובץ הסניפים. (רק אחת כי זה שם הסניף זהו המפתח, ולא יתכן שהשם יחזור פעמיים במקומות שונים).

ג. ביצוע שני הצירופים בו זמנית :

1. נבנה אינדקס ליחס הלקוחות על פי שם-לקוח.  
 נבנה אינדקס ליחס הסניפים על פי שם סניף.  
 2. נסרוק סדרתית את יחס ההפקדות. עבור כל רשומה ביחס זה נאתר (תוך שימוש באינדקסים) את הרשומות המתאימות משני היחסים האחרים.

