תרגיל 3

מגיש:
אורן מוטיעי, ת.ז 321174591

חלק א' - שאילתות SQL

שאלה 2
א. תחילה השאילתה מבצעת מכפלה קרטזית בין שתי טבלאות זהות של donors,
לאחר מכן היא משאירה רק שורות שבהן השם של תורם $n1$ מופיע לפני השם של תורם $n2$
לפי סדר אלפביתי $(d1.name < d2.name)$, וגם העמותות זהות $(d1.cause = d2.cause)$.
כעת היא מקבצת את כל השורות עם אותו $n1, n2$ לקבוצה נפרדת, ולכאורה משאירה רק קבוצות שבהן
כמות העמותות השונות זהה עבור שני התורמים, אבל השורה של ה־HAVING תמיד מחזירה
אמת כי בשורה של ה־WHERE השארנו רק שורות בהן העמותות זהות ולכן הכמות תהיה זהה.
לבסוף השאילתה מחזירה זוגות של תורמים $(n1, n2)$, ללא כפילויות, עם מיון בסדר עולה
לפי $n1$ ואחר כך לפי $n2$.

במילים אחרות, השאילתה מחזירה זוגות של תורמים $(n1, n2)$ כך שקיימת עמותה ששניהם תרמו לה,
וייתכן ש־$n2$ תרם לעמותות נוספות ש־$n1$ לא תרם להן ולהיפך, והשאילתה לא תטפל בכך.

חלק ב' - אינדקסים

1. א. נחשב כמה שורות כל בלוק יכול להכיל:

$$\left\lfloor \frac{1024}{64} \right\rfloor = 16$$

נחשב כמה בלוקים הטבלה יכולה להכיל:

$$\left\lceil \frac{1000}{16} \right\rceil = 63$$

לכן עלות חישוב השאילתה בהנחה שאין אינדקסים היא **63 פעולות I/O**.

ב. נתון כי התכונה $birthYear$ תופסת 4 bytes, מצביע תופס 4 bytes וגודל בלוק הוא 1024 bytes.
נסמן את דרגת הפיצול האופטימלית ב־d. לכל קודקוד שאינו עלה יש לכל היותר d מצביעים
ו־ $d-1$ ערכי חיפוש לכן:

$$\begin{aligned} 4d + 4(d-1) &\leq 1024 \\ 8d &\leq 1028 \\ d &\leq 128.5 \end{aligned}$$

באופן דומה אפשר להשתמש בנוסחה שראינו בשיעור:

$$d = \left\lfloor \frac{1024+4}{4+4} \right\rfloor = \lfloor 128.5 \rfloor = 128$$

לכן דרגת הפיצול האופטימלית של האינדקס היא $\boldsymbol{d = 128}$.

ג. נתון כי לטבלה 1000 שורות ובסעיף הקודם מצאנו $d = 128$,
לכן נחשב את גובה העץ באמצעות הנוסחה שראינו בשיעור:

$$\left\lceil log_{\lceil \frac{128}{2} \rceil} 1000 \right\rceil = \lceil 1.66 \rceil = 2$$

מאחר ובשורה הראשונה של השאילתה מופיע "DISTINCT "exists מספיק לעבור
על עלה אחד בעץ, כי אם קיים birthYear עם ערך גדול מ־1990 נחזיר "exists" ונסיים.
אין צורך לעבור בעלים על כל ערכי ה־birthYear שגדולים מ־1990,
כלומר מספיק לבצע INDEX UNIQUE SCAN.

לסיכום, עלות חישוב השאילתה באמצעות האינדקס היא **3 פעולות I/O**, כי עוברים על 2 בלוקים
במורד העץ ולאחר מכן עוברים על עוד בלוק אחד בעלה.

2. תחילה נעבור על 2 בלוקים במורד העץ.
כעת נחשב על כמה עלים אנחנו צריכים לעבור:
בכל עלה ניתן להכניס לפחות $\lceil 128/2 \rceil - 1 = 63$ ערכי birthYear.
נתון כי ערכי birthYear מתפלגים אחיד בטווח $[1900, 2000]$, השאילתה מבקשת למצוא
$birthYear > 1990$ לכן צריך לעבור על v ערכים בעלים כאשר:

$$v = 1000 \cdot \frac{2000-1990}{2000-1900} = 1000 \cdot \frac{10}{100} = 100$$

לפיכך, צריך לעבור על $\lceil \frac{v}{63} \rceil = \lceil 1.59 \rceil = 2$ עלים.
לסיכום, עלות חישוב השאילתה באמצעות האינדקס היא **4 פעולות I/O**
כאשר ביצענו INDEX RANGE SCAN.

3. א. נתון כי התכונה uid תופסת 4 bytes, מצביע תופס 4 bytes וגודל בלוק הוא 1024 bytes.
לכן החישוב זהה לשאלה 1 סעיף ב':

$$d = \left\lfloor \frac{1024+4}{4+4} \right\rfloor = \lfloor 128.5 \rfloor = \mathbf{128}$$

ב. מאחר שדרגת הפיצול ומספר השורות בטבלה לא שונו, גובה העץ נשאר זהה לחישוב שביצענו
בשאלה 1 סעיף ג', כלומר הוא שווה ל־2.
נשים לב ש־uid הוא primary key לכן הוא unique ולכן מספיק לעבור רק על עלה אחד.
השאילתה צריכה להחזיר name ולכן נצטרך לגשת לבלוק בטבלה עם השורה הרלוונטית.
למעשה אנחנו מבצעים INDEX UNIQUE SCAN ו־ TABLE ACCESS BYROWID.

לסיכום, עלות חישוב השאילתה באמצעות האינדקס הוא **4 פעולות I/O** - צריך לעבור על 2 בלוקים
במורד העץ, בלוק אחד בעלה ובלוק נוסף כדי להחזיר את name.

4. א. נתון כי התכונה language תופסת 10 bytes, מצביע תופס 4 bytes וגודל בלוק הוא 1024 bytes.
נשתמש שוב בנוסחה ונקבל:

$$d = \left\lfloor \frac{1024+10}{4+10} \right\rfloor = \lfloor 73.86 \rfloor = 73$$

לכן דרגת הפיצול האופטימלית של האינדקס היא $\boldsymbol{d = 73}$.

ב. נחשב את גובה העץ:

$$\left\lceil log_{\lceil \frac{73}{2} \rceil} 1000 \right\rceil = \lceil 1.91 \rceil = 2$$

בכל עלה ניתן להכניס לפחות $\lceil \frac{73}{2} \rceil - 1 = 36$ ערכי language.
נתון כי הערכים ב־language בטבלה מחולקים ל־5 קטגוריות באופן אחיד והשאילתה
צריכה למצוא $language =' Hebrew'$ לכן צריך לעבור על $1000 \cdot \frac{1}{5} = 200$ ערכים בעלים.
לפיכך, צריך לעבור על $\lceil \frac{200}{36} \rceil = \lceil 5.56 \rceil = 6$ עלים.

עבור כל ערך בעלה נרצה לגשת לבלוק בטבלה כדי לקבל את הערך birthYear, אך ייתכן
שיותר משתלם לעבור פעם אחת על כל הבלוקים בטבלה, לכן ניקח את המינימום מביניהם:

$$2 + 6 + min\,(200, 63) = 71$$

מכאן, עלות חישוב השאילתה באמצעות האינדקס הוא **71 פעולות I/O**.

ג. נתון כי התכונה language תופסת 10 bytes, התכונה birthYear תופסת 4 bytes,
מצביע תופס 4 bytes וגודל בלוק הוא 1024 bytes. נשתמש בנוסחה שראינו בשיעור:

$$d = \left\lfloor \frac{1024+14}{4+14} \right\rfloor = \lfloor 57.67 \rfloor = 57$$

לכן דרגת הפיצול האופטימלית של האינדקס היא $\boldsymbol{d = 57}$.

ד. נחשב את גובה העץ:

$$\left\lceil log_{\lceil \frac{57}{2} \rceil} 1000 \right\rceil = \lceil 2.05 \rceil = 3$$

בכל עלה ניתן להכניס לפחות $\lceil \frac{57}{2} \rceil - 1 = 28$ ערכי language.
ראינו בסעיף ב' שצריך לעבור על 200 ערכים בעלים, לכן צריך לעבור על $\lceil \frac{200}{28} \rceil = \lceil 7.14 \rceil = 8$ עלים.
מאחר והאינדקס הוא גם על birthYear אין צורך לגשת לטבלה, אלא אפשר לחשב
את avg(birthYear) באמצעות מעבר על העלים.

מכאן, עלות חישוב השאילתה באמצעות האינדקס הוא 11 **פעולות I/O** - צריך לעבור על 3 בלוקים במורד העץ ועל 8 בלוקים בעלים.