תרגיל DB 4

מגישים: דן קורנפלד
רותם אהרוני

חלק א:

שאלה א:

סעיף 1:

- כאמור, אין לנו Index, לכן נצטרך לספור את כמות קריאות הI\O שלנו בסריקת טבלה מלאה, היות ולא ידוע לנו איך ה-duration מפוזר כאשר הטבלה לא ממויינת.
- נרצה לחשב: $\left\lceil \frac{numberRowInMovieTable}{numberBlocks} \right\rceil$, כך נוכל לחשב את מספר הבלוקים שאנו צריכים לקרוא מהזיכרון.
- למדנו ש: $numberBlocks = \left\lfloor \frac{blockSize}{sizeOfRow} \right\rfloor = \left\lfloor \frac{1000}{52} \right\rfloor = 19$ ולכן סריקת הטבלה תעלה לנו:
- $\left\lceil \frac{10,000}{19} \right\rceil = \lceil 526.31.. \rceil = 527\ I\backslash O$
- חישוב פעולת ה-Avg לא עולה לנו עוד פעולות IO, היות ואנחנו קוראים את כל הטבלה בכל מקרה, ולכן החישוב כלול במעבר יחיד על כל הטבלה.

סעיף 2:

- כדי לחשב את דרגת הפיצול האופטימלי d נחשב לפי:
$\underbrace{(size\ of\ pointer)\cdot d}_{\text{ילדים } d} + \underbrace{(search\ key\ size)\cdot(d-1)}_{\text{ערכי חיפוש } d-1} \leqslant block\ size$ לכן נציב את הפרמטרים שלנו:
- $$\begin{aligned} block\ size &= 1000 \\ size\ of\ pointer &= 8 \\ search\ key\ size &= size(duration) = 8 \end{aligned}$$
- $$\begin{aligned} 8d + 8(d-1) &\leqslant 1000 \\ 16d - 8 &\leqslant 1000 \\ 16d &\leqslant 1008 \\ d &\leqslant 63 \end{aligned}$$
ולכן $d = 63$ יהיה דרגת הפיצול האופטימלי (הגדולה ביותר)

סעיף 3:

- ראשית נחשב את גובה העץ עם N שורות ודרגת פיצול d הוא לכל היותר $\lceil log_{\lceil \frac{d}{2} \rceil}(N) \rceil$
- עבור מעבר על העלים עם הערכים המתאימים ל-m שורות רלוונטיות ודרגת פיצול d: $\left\lceil \frac{m}{\lceil \frac{d}{2} \rceil - 1} \right\rceil$.
- נתון: הערכים ב-duration מתפלגים אחים בטווח $[1, 200]$, נרצה לחשב את היחס שרלוונטי לשאלה

מתוך סה"כ השורות: $\frac{200-100}{200-1} = 0.50 \pm$

- ולכן השורות הרלוונטיות יהיו $10,000 * 0.50 \pm .. = 5000$ שורות

- החישוב כדי להגיע לשורש העץ: $\lceil log_{\lceil \frac{63}{2} \rceil}(10,000) \rceil = \lceil log_{32}(10,000) \rceil = 3$

- מעבר על העלים: $\left\lceil \frac{5000}{\lceil \frac{63}{2} \rceil - 1} \right\rceil = 162$

- ולכן החישוב הכולל הינו $162 + 3 = 165\ I\backslash O$

שאלה ב:

סעיף 1:

- כדי לחשב את דרגת הפיצול האופטימלי d נחשב לפי:
$\underbrace{(size\ of\ pointer)\cdot d}_{\text{ילדים } d} + \underbrace{(search\ key\ size)\cdot(d-1)}_{\text{ערכי חיפוש } d-1} \leqslant block\ size$ לכן נציב את הפרמטרים

שלנו:

- $$\begin{aligned} block\ size &= 1000 \\ size\ of\ pointer &= 8 \\ search\ key\ size &= size(genre) = 10 \end{aligned}$$
- $$\begin{aligned} 8d + 10(d-1) &\leqslant 1000 \\ 18d - 10 &\leqslant 1000 \\ 18d &\leqslant 1010 \\ d &\leqslant 56.11 \end{aligned}$$ ולכן $d = 56$ יהיה דרגת הפיצול האופטימלי (הגדולה ביותר)

סעיף 2:

- ראשית נחשב את גובה העץ עם N שורות ודרגת פיצול d הוא לכל היותר $\lceil log_{\lceil \frac{d}{2} \rceil}(N) \rceil$
- עבור מעבר על העלים עם הערכים המתאימים ל-m שורות רלוונטיות ודרגת פיצול d: $\left\lceil \frac{m}{\lceil \frac{d}{2} \rceil - 1} \right\rceil$.
- נתון: הערכים ב-genre מוחלקים ל-4 קטגוריות באופן אחיד, לכן ניתן להניח שתהיה חלוקה אחידה גם בטבלת הסרטים. אם יש 10,000 שורות בטבלה המקורית, נחלק ב-4 עבור הסוג הספציפי של הז'נר שאנו מחפשים, ונקבל 2,500 שורות רלוונטיות
- יתר על כן, לא מספיק שנעבור על ה-index, נצטרך לגשת לטבלה עצמה בשורות הרלוונטיות כדי לקחת את הערכים של ה-duration לכן, מלבד המעבד עד לגובה העלים והמעבר על העלים, נצטרך לגשת לכל הבלוקים שיכולים להכליל את הערכים הרלוונטים, ניגש למינימום מבין: 2,500 בלוקים (כל שורה רלוונטית בבלוק נפרד), למס הבלוקים הכולל בטבלה (סריקת כל הטבלה). בדקנו בסעיף הראשון, ומס הבלוקים הכולל של הטבלה הינו 527 בלוקים, ולכן נוסיף את הערך הזה לחישוב הכולל בסריקת הטבלה.

- $$\lceil log_{\lceil \frac{56}{2} \rceil}(10,000) \rceil + \left\lceil \frac{2,500}{\lceil \frac{56}{2} \rceil - 1} \right\rceil + numberOfBlocks = 3 + 93 + 527 = 623\ I\backslash O$$

שאלה ג:

סעיף 1:

- כדי לחשב את דרגת הפיצול האופטימלי d נחשב לפי:

$$\underbrace{(size\ of\ pointer)\cdot d}_{\text{ילדים } d} + \underbrace{(search\ key\ size)\cdot(d-1)}_{\text{ערכי חיפוש } d-1} \leqslant block\ size$$

לכן נציב את הפרמטרים שלנו:

- $$\begin{aligned} block\ size &= 1000 \\ size\ of\ pointer &= 8 \\ search\ key\ size &= size(genre + duration) = 18 \end{aligned}$$

- $$\begin{aligned} 8d + 18(d-1) &\leqslant 1000 \\ 26d - 18 &\leqslant 1000 \\ 26d &\leqslant 1018 \\ d &\leqslant 39.15 \end{aligned}$$

ולכן $d = 39$ יהיה דרגת הפיצול האופטימלי (הגדולה ביותר)

- נתון: הערכים ב-genre מוחלקים ל-4 קטגוריות באופן אחיד, לכן ניתן להניח שתהיה חלוקה אחידה גם בטבלת הסרטים. אם יש 10,000 שורות בטבלה המקורית, נחלק ב-4 עבור הסוג הספציפי של הז'נר שאנו מחפשים, ונקבל 2,500 שורות רלוונטיות.
- יתר על כן, היות וה-index כולל את המידע על duration אז לא צריך לסרוק את הטבלה אלא מספיק לסרוק את האינדקס בחלקים הרלוונטים:
- $$\lceil log_{\lceil \frac{39}{2} \rceil}(10,000) \rceil + \left\lceil \frac{2,500}{\lceil \frac{39}{2} \rceil - 1} \right\rceil = 4 + 132 = 136\ I\backslash O$$

חלק ב:

שאלה 1:

סעיף א:

- כדי לחשב את העלות של הצירוף $Movies \bowtie PlaysIn$ נפעל לפי הנוסחה הבאה:

$B(R) + B(S) \cdot \left\lceil \frac{B(R)}{M-2} \right\rceil$ כאשר R- היחס החיצוני צריך להיות קטן מ-S היחס הפנימי

ולכן: $numberOfBlocksMovies = 527$ (חושב בשאלה הקודמת)

$$numberOfBlocksPlaysIn = \left\lceil \frac{numberRowInPlaysIn}{numberRowInBlocks} \right\rceil = 2632$$

$$numberRowInBlocks = \left\lfloor \frac{blockSize}{sizeOfRow} \right\rfloor = \left\lfloor \frac{1000}{26} \right\rfloor = 38$$

- נראה כי מס הבלוקים של הטבלה $movie$ קטן יותר, לכן הוא יהיה היחס החיצוני:

$$527 + 2632 \cdot \left\lceil \frac{527}{52-2} \right\rceil = 29479\ IO$$

סעיף ב:

- אם לוקחים $M = 50$ זה הערך הנמוך ביותר כדי ש:$\left\lceil \frac{527}{M-2} \right\rceil = 11$ ולכן זהו גודל החוצץ המינימלי הנדרש כדי לקבל עלות חישוב זהה.

שאלה 2:

סעיף א:

- כדי לחשב את פעולת מספר הפעולות בתוצאה, נחשב קודם כל את מס השורות בטבלאות המקוריות:
  - $B(S) = 1,200\ (Blocks\ in\ S)$
    $B(R) = 4,000\ (Blocks\ in\ R)$ נחשב לפי הנוסחה הבאה:

$$\left\lfloor \frac{blockSize}{sizeOfRow} \right\rfloor \Longrightarrow B(R) = \left\lfloor \frac{2000}{20} \right\rfloor = 100\ linesPerBlock \Longrightarrow \mathbf{T(R) = 400,000}$$

$$B(S) = \left\lfloor \frac{2000}{30} \right\rfloor = 66\ linesPerBlock \Longrightarrow \mathbf{T(S) = 79,200}$$

- $$\frac{T(R) \times T(S)}{max\left\{V(R,A), V(S,A)\right\}} = \frac{400,000 \cdot 79,200}{max\left\{\underbrace{V(R,A)}_{\substack{400,000 \\ A\ is\ key\ in\ R}}, \underbrace{V(S,A)}_{\substack{1000 \\ from\ the\ question}}\right\}} = \underbrace{79,200}_{lines\ after\ join}$$

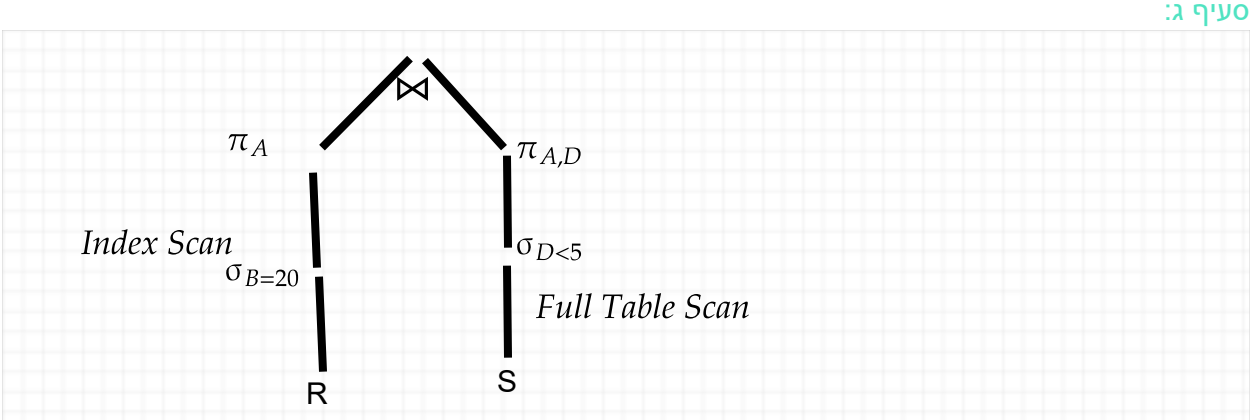
- עתה, היות ויש לנו את תנאי הבחירה נחשב לפי החוקים שלמדנו:
  - במקרים בהם אנו לא יודעים מה ההתפלגות כמו ב-$D < 5$ נחלק את התוצאה ב-3
  - בעבור $B = 20$, היות ואנו יודעים שקיימים 200 ערכים שונים לפי הנתון $V(R,B) = 200$

$$\frac{79,200}{200*3} = 132\ lines$$

סעיף ב:

- $$\left\lfloor \frac{2000}{20} \right\rfloor = 100\ Lines\ Per\ Block \Longrightarrow \left\lceil \frac{132}{100} \right\rceil = 2\ Blocks$$

סעיף ג:



סעיף ד:

- עבור (נתון) $Read(S) = B(S) := 1200$ (ראינו שאם היינו מחשבים את העלות גישה רק לשליש מהשורות לפי הנוסחאות היינו צריכים לגשת ליותר מ-1200 בלוקים, לכן בחרנו את העלות המינימלית בניהם)
- עבור R נחשב מעבר על ה-index לא נדרש היות והעלות זניחה, לאחר המעבר יהיו לנו $\frac{400,000}{200} = 2000\ lines\ in\ R\ to\ Read$. היות ובעלים יש רפרסים לטבלאות עצמן, לכל היותר נצטרך לגשת ל-2000 בלוקים שונים כדי לקבל את כל המידע הנדרש מ-R, ולכן $Read(R) = 2,000$
- $B(E_R) = \frac{B(R)}{V(R,B)} = \frac{4,000}{200} = 20$
- לכן נוכל לגשת לחישוב לפי הנוסחה:

$$Read(R) + Read(S) \times \left\lceil \frac{B(E_R)}{M-2} \right\rceil = 2,000 + 1,200 \cdot \left\lceil \frac{20}{68} \right\rceil = 3,200\ IO$$

שאלה 3:

סעיף א:

1. כתיבת שאילתה: בוצע
2. זמן ביצוע 131ms, זמן תכנון 0.197ms ולכן זמן ריצה: 131.217ms

```
                                                    QUERY PLAN
-----------------------------------------------------------------------------------------------------------------------------------
 Unique  (cost=2457.68..2459.37 rows=169 width=26) (actual time=130.664..130.897 rows=116 loops=1)
   →  Sort  (cost=2457.68..2458.11 rows=169 width=26) (actual time=130.662..130.746 rows=116 loops=1)
         Sort Key: m1.movieid, m1.title, m1.duration
         Sort Method: quicksort  Memory: 34kB
         →  Hash Join  (cost=1220.90..2451.43 rows=169 width=26) (actual time=67.971..130.555 rows=116 loops=1)
               Hash Cond: ((m1.year = movies.year) AND (m1.duration = (min(movies.duration))))
               →  Seq Scan on movies m1  (cost=0.00..968.00 rows=50000 width=30) (actual time=0.010..30.334 rows=50000 loops=1)
               →  Hash  (cost=1219.66..1219.66 rows=83 width=8) (actual time=67.950..67.951 rows=88 loops=1)
                     Buckets: 1024  Batches: 1  Memory Usage: 12kB
                     →  HashAggregate  (cost=1218.00..1218.83 rows=83 width=8) (actual time=67.820..67.877 rows=90 loops=1)
                           Group Key: movies.year
                           →  Seq Scan on movies  (cost=0.00..968.00 rows=50000 width=8) (actual time=0.004..31.512 rows=50000 loops=1)
 Planning Time: 0.197 ms
 Execution Time: 131.020 ms
(14 rows)
```

3. אנו טוענים שהשאילתה שלנו יעילה יותר היות ו:
אנו מבצעים עבור כל שנה את מציאת ה-duration המינימלי פעם אחת במקום לבצע את השאילתה הפנימית עבור כל שורה, אנו מבצעים את השאילתה הפנימית פעם אחת, דבר זה חוסך זמן יקר.
השתמשנו ב-with כדי לצור בצורה חד פעמית טבלה שתחשב פעם אחת את השאילתה הפנימית עבור כל פעם שאנו מריצים את השאילתה.

סעיף ב:

1. create index movies_index_duration on movies(duration,year);
2. זמנים: זמן תכנון 0.17ms זמן ביצוע 910.609ms זמן ריצה: 910.779ms שזה קטן (משמעותית) מ30 שניות, כמבוקש.

```
public=> explain analyze select distinct movieId, title, duration from Movies M1 where duration = (select min(duration) from Movies M2 where M2.year = M1.year) order by movieId, title, duration;
                                                    QUERY PLAN
-----------------------------------------------------------------------------------------------------------------------------------
 Unique  (cost=215969.06..215971.56 rows=250 width=26) (actual time=909.099..909.327 rows=116 loops=1)
   →  Sort  (cost=215969.06..215969.69 rows=250 width=26) (actual time=909.096..909.165 rows=116 loops=1)
         Sort Key: m1.movieid, m1.title, m1.duration
         Sort Method: quicksort  Memory: 34kB
         →  Seq Scan on movies m1  (cost=0.00..215959.10 rows=250 width=26) (actual time=6.524..908.917 rows=116 loops=1)
               Filter: (duration = (SubPlan 2))
               Rows Removed by Filter: 49884
               SubPlan 2
                 →  Result  (cost=4.29..4.30 rows=1 width=4) (actual time=0.016..0.017 rows=1 loops=50000)
                       InitPlan 1 (returns $1)
                         →  Limit  (cost=0.29..4.29 rows=1 width=4) (actual time=0.014..0.014 rows=1 loops=50000)
                               →  Index Only Scan using movies_index_duration on movies m2  (cost=0.29..2030.93 rows=508 width=4) (actual time=0.012..0.012 rows=1 loops=50000)
                                     Index Cond: ((duration IS NOT NULL) AND (year = m1.year))
                                     Heap Fetches: 49998
 Planning Time: 0.170 ms
 JIT:
   Functions: 12
   Options: Inlining false, Optimization false, Expressions true, Deforming true
   Timing: Generation 1.132 ms, Inlining 0.000 ms, Optimization 0.285 ms, Emission 6.038 ms, Total 7.455 ms
 Execution Time: 910.609 ms
(20 rows)
```

1. ראינו שיש הרבה סינונים לפי duration ובפרט עבור המינימום, למדנו שחיפוש באינדקס עבור גדלים ספציפיים זהו פתרון יעיל מאוד, יתר על כן הוספנו גם את ה-year כדי שלא נצטרך לגשת לטבלה הראשית אלא מספיק להסתכל על הרפרנס בעלה.