Contents

1 Basic Test Results

```
Extracting Archive:
Archive:  /tmp/bodek.GbL7tu/db/ex4/ofer.feinstein/presubmission/submission
  inflating: ex4.pdf
 extracting: README

************************************************************
** Testing that all necessary files were submitted:
README:
    SUBMITTED
ex4.pdf:
    SUBMITTED

************************************************************
** Checking for correct README format:
```

2 README

```
ofer.feinstein, noabarlia
```

מסדי נתונים (67506) | תרגיל 4

עופר פיינשטיין, 316413434 | נועה ברליאה, 318813789

14 בדצמבר 2020

שאלה 1

סעיף 1

א. בטבלה $Movies$ יש 4 אטריביוטים בגודל 4 בתים ו־2 אטריביוטים בגודל 10 בתים, לכן כל שורה בטבלה היא בגודל $4 \cdot 4 + 10 \cdot 2 = 36$ בתים.

נחשב את מספר השורות בבלוק: $\lfloor \frac{8,192}{36} \rfloor = 227$.

בטבלה $Movies$ יש 10,000 שורות, לכן הטבלה $Movies$ בגודל $\lceil \frac{10,000}{227} \rceil = 45$ בלוקים.

בטבלה $PlaysIn$ יש 2 אטריביוטים בגודל 4 בתים ואטריביוט אחד בגודל 10 בתים, לכן כל שורה בטבלה היא בגודל $4 \cdot 2 + 10 = 18$ בתים.

נחשב את מספר השורות בבלוק: $\lfloor \frac{8,192}{18} \rfloor = 455$.

בטבלה $PlaysIn$ יש 100,000 שורות, לכן $B(PlaysIn) = \lceil \frac{100,000}{455} \rceil = 220$, כלומר הטבלה בגודל 220 בלוקים.

ראינו כי העלות של אלגוריתם BNL עבור יחסים S, R היא $B(S) + B(R) \lceil \frac{B(S)}{M-2} \rceil$, כאשר S הוא היחס הקטן. נתון ש־$M = 15$ ולכן עלות צירוף הטבלאות $Movies, PlaysIn$ היא:

$$B(Movies) + B(PlaysIn) \lceil \frac{B(Movies)}{M-2} \rceil = 45 + 200 \cdot \lceil \frac{45}{15-2} \rceil = 925$$

כלומר **925** פעולות I/O.

ב. ראשית נבדוק האם ניתן להפעיל את אלגוריתם $Hash\,Join$. על מנת שנוכל להשתמש ב־HJ צריך להתקיים:

$$\lceil \frac{B(Movies)}{M-1} \rceil < M-1 \; or \; \lceil \frac{B(PlaysIn)}{M-1} \rceil < M-1$$

נשים לב כי מתקיים $\lceil \frac{B(Movies)}{M-1} \rceil = \lceil \frac{45}{14} \rceil = 4 < 14$, לכן ניתן לבצע HJ.

ראינו כי העלות של אלגוריתם HJ עבור יחסים S, R היא $3B(S) + 3B(R)$. לכן עלות צירוף הטבלאות $Movies, PlaysIn$ היא:

$$3B(Movies) + 3B(PlaysIn) = 3 \cdot 45 + 3 \cdot 220 = 795$$

כלומר **795** פעולות I/O.

ג. ראשית נבדוק האם ניתן להפעיל את אלגוריתם $Sort\,Merge\,Join$. על מנת שנוכל להשתמש ב־SMJ נצטרך למיין כל אחת מהטבלאות, לשם כך צריך להתקיים:

$$\lceil \frac{B(Movies)}{M} \rceil < M \ and \ \lceil \frac{B(PlaysIn)}{M} \rceil < M$$

נתון כי $M = 15$. נשים לב כי $\lceil \frac{B(PlaysIn)}{M} \rceil = \lceil \frac{220}{15} \rceil = 15$, ולכן התנאי לא מתקיים עבור $PlaysIn$. כלומר **לא נוכל לבצע SMJ.**

סעיף 2

א. כעת נתון ש־$M = 16$ ולכן עלות צירוף הטבלאות $Movies, PlaysIn$ היא:

$$B(Movies) + B(PlaysIn) \lceil \frac{B(Movies)}{M-2} \rceil = 45 + 200 \cdot \lceil \frac{45}{16-2} \rceil = 925$$

כלומר **925** פעולות I/O.

ב. כעת $M = 16$, הגדלנו את החוצץ ולכן עדיין ניתן להפעיל את אלגוריתם $Hash\,Join$. עלות צירוף הטבלאות $Movies, PlaysIn$ לא משתנה.

כלומר **795** פעולות I/O.

ג. ראשית נבדוק האם ניתן להפעיל את אלגוריתם $Sort\,Merge\,Join$. על מנת שנוכל להשתמש ב־SMJ נצטרך למיין כל אחת מהטבלאות, לשם כך צריך להתקיים:

$$\lceil \frac{B(Movies)}{M} \rceil < M \ and \ \lceil \frac{B(PlaysIn)}{M} \rceil < M$$

מתקיים: $\lceil \frac{B(Movies)}{M} \rceil = \lceil \frac{45}{16} \rceil = 3 < 16 \ and \ \lceil \frac{B(PlaysIn)}{M} \rceil = \lceil \frac{220}{16} \rceil = 14 < 16$

כלומר נוכל למיין גם את $PlaysIn$ וגם את $Movies$, ולכן נוכל לבצע SMJ.

כדי שנוכל לבצע את האלגוריתם בעלות אופטימלית (המנעות ממיון מלא של היחסים) צריך להתקיים:

$$\lceil \frac{B\left(Movies\right)}{M} \rceil + \lceil \frac{B\left(PlaysIn\right)}{M} \rceil < M$$

זה לא מתקיים כי $3 + 14 > 16$, ולכן עלות צירוף הטבלאות $Movies, PlaysIn$ בעזרת SMJ היא:

$$5B\left(Movies\right) + 5B\left(PlaysIn\right) = 5 \cdot 45 + 5 \cdot 220 = 1,325$$

כלומר **1,325** פעולות I/O.

סעיף 3

א. נצטרך בלוק אחד עבור $Movies$, בלוק אחד עבור $PlaysIn$ ובלוק אחד עבור ה־$Output$, כלומר מינימום **3** בלוקים.

ב. נחשב את מספר הבלוקים המינימלי:

$$\begin{aligned} \lceil \frac{B\left(Movies\right)}{M-1} \rceil &< M-1 \\ 45 &< \left(M-1\right)^2 \\ 45 &< M^2 + 2M + 1 \\ 8 &< M \end{aligned}$$

כלומר מינימום **9** בלוקים.

ג. ראינו בסעיף 1ג' כי 15 בלוקים אינם מספיקים לשימוש ב־SMJ, ובסעיף 2ג' ראינו כי 16 בלוקים כן מספיקים, כלומר מינימום **16** בלוקים.

ד. כדי שנוכל לבצע את האלגוריתם בעלות אופטימלית (המנעות ממיון מלא של היחסים) צריך להתקיים:

$$\begin{aligned} \lceil \frac{B\left(Movies\right)}{M} \rceil + \lceil \frac{B\left(PlaysIn\right)}{M} \rceil &< M \\ \frac{45}{M} + \frac{220}{M} &< M \\ \frac{265}{M} &< M \\ 265 &< M^2 \\ 16 &< M \end{aligned}$$

כלומר מינימום **17** בלוקים.

שאלה 2

א. כדי להעריך את גודל התוצאה בבלוקים, עבור בחירת שורות מטבלה S המקיימות כי ערך אטריביוט C שלהן הוא 8, ראשית נחשב את מספר השורות בטבלה. נתון כי $B(S) = 1000$ ובכל בלוק של S יש 50 שורות, לכן סך מספר השורות הוא $T(S) = 50 \cdot 1000 = 50,000$. נתון כי $V(S,C) = 200$, ותחת הנחת התפלגות אחידה, מספר השורות המתאימות הוא $\frac{50,000}{200} = 250$. נתון כי בכל בלוק יש 50 שורות, לכן מספר הבלוקים הוא $\frac{250}{50} = 5$, כלומר **5** בלוקים בתוצאה.

ב. כדי להעריך את גודל התוצאה בבלוקים, עבור בחירת שורות מטבלה R המקיימות כי ערך אטריביוט A שלהן קטן מ־10, ראשית נחשב את מספר השורות בטבלה. נתון כי $B(R) = 30$ ובכל בלוק של S יש 100 שורות, לכן סך מספר השורות הוא $T(R) = 30 \cdot 100 = 30,000$. לא נתונה התפלגות אודות מספר הערכים ב־A שקטנים מ־10, לכן בהתאם לכלל האצבע נקבל כי מספר השורות המתאימות הוא $\frac{30,000}{3} = 10,000$, כלומר $10,000$ שורות. נתון כי בכל בלוק יש 300 שורות, לכן נקבל כי מספר הבלוקים הוא $\frac{30,000}{300} = 100$, כלומר **100** בלוקים בתוצאה.

ג. ראינו בהרצאה שמספר השורות בתוצאת צירוף נתון על ידי הנוסחה הבאה: $\frac{T(R)\cdot T(S)}{max\{V(R,B),V(S,B)\}}$. נרצה לכפול את התוצאה ב־$\frac{1}{200} \cdot \frac{1}{3}$ עבור הבחירות על R ו־S כמו שתיארנו בסעיפים הקודמים. בנוסף, נתון כי $V(R,B) = 100$ וכי B הוא מפתח בטבלה S, ולכן $V(S,B) = T(S) = 50,000$. נציב את הנתונים בנוסחה ונקבל:

$$\frac{T(R)\cdot T(S)}{max\{V(R,B),\, V(S,B)\}} \cdot \frac{1}{V(S,C)} \cdot \frac{1}{3} = \frac{30,000 \cdot 50,000}{max\{100, 50,000\}} \cdot \frac{1}{200} \cdot \frac{1}{3} = 50$$

כלומר בתוצאת הביטוי יש **50** שורות .

ד. נרצה לנבצע את פעולות הבחירה לפני ביצוע הצירוף היכן שניתן. נגדיר:

$$E_S = \sigma_{C=8} S(B,C)$$
$$E_R = \sigma_{A<10} R(A,B)$$

ראינו בסעיפים א' וב' שמתקיים: $B(E_S) = 5$ ו־$B(E_R) = 100$.

נחשב את עלות הבחירה מ־S בעזרת האינדקס על האטריביוט C:

נתון כי עלות הגישה היא זניחה, ולכן במקרה הגרוע ביותר כל שורה תמצא בבלוק אחר, ולכן $Read(E_S) = T(E_S) = 250$. נשים לב כי $Read(E_S) < B(S) = 1000$, לכן נעדיף לקרוא באמצעות האינדקס על C, ולא לבצע $Full\,Table\,Scan$.

לא נתון אינדקס על R, ולכן נשתמש תמיד ב־$Full\,Table\,Scan$ בפעולת הבחירה, לכן $Read(E_R) = B(R) = 300$.

נחשב עבור כל אלגוריתם את עלות פעולת הצירוף ונבחר באלגוריתם בעל העלות הנמוכה ביותר, כאשר נתון כי גודל החוצץ הוא 10.

- BNL - נבצע את הבחירה משני היחסים לפני פעולת הצירוף. נבחר את היחס S כיחס החיצוני מכיוון שביחס זה מספר הבלוקים הוא הקטן יותר לאחר הבחירה, נציב בנוסחה לחישוב עלות צירוף BNL ונקבל:

$$Read(E_S) + Read(E_R) \cdot \left\lceil \frac{B(E_S)}{M-2} \right\rceil = 250 + 300 \cdot \left\lceil \frac{5}{10-2} \right\rceil = 550$$

כלומר עלות אלגוריתם זה היא **550** פעולות I/O.

- INL - נתון כי קיים ליחס S אינדקס על שדה הצירוף B, לכן נוכל להשתמש באלגוריתם זה. נשים לב שלא נוכל לבצע את פעולת הבחירה מהטבלה S לפני פעולת הצירוף, לכן נבצע בחירה רק על הטבלה R. נתון ש־B הוא מפתח ביחס S, ולכן יש רק שורה שאחת מתאימה לכל ערך, כלומר $cost\,of\,select$ הוא 1. עלות השימוש הנה:

$$Read\,(E_R) + T(E_R) \cdot cost\,of\,select = 300 + 10,000 \cdot 1 = 10,300$$

כלומר עלות אלגוריתם זה היא **10,300** פעולות I/O.

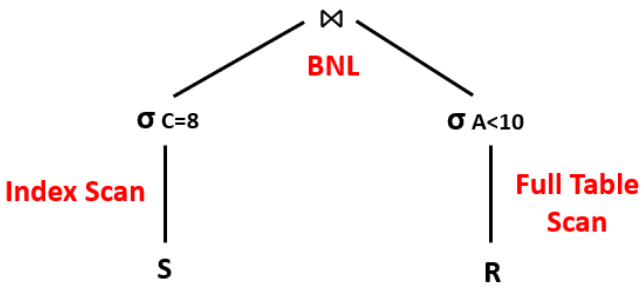
- HJ - נבדוק האם ניתן להשתמש באלגוריתם זה, כלומר אם מתקיים: $\left\lceil \frac{B(E_S)}{M-1} \right\rceil < M-1$ או $\left\lceil \frac{B(E_R)}{M-1} \right\rceil < M-1$. נציב ונקבל כי $\left\lceil \frac{5}{10-1} \right\rceil < 10-1$, אי שוויון זה אכן מתקיים וזהו תנאי מספיק לשימוש באלגוריתם זה. כעת, נחשב את עלות השימוש באלגוריתם:

$$Read\,(E_S) + Read\,(E_R) + 2B\,(E_S) + 2B\,(E_R) = 250 + 300 + 2 \cdot 5 + 2 \cdot 100 = 760$$

כלומר עלות אלגוריתם זה היא **760** פעולות I/O.

- SM - נבדוק האם ניתן להשתמש באלגוריתם זה, כלומר אם מתקיים $\left\lceil \frac{B(E_S)}{M} \right\rceil < M$ וגם $\left\lceil \frac{B(E_R)}{M} \right\rceil < M$. נציב ונקבל כי אכן מתקיים $\left\lceil \frac{5}{10} \right\rceil < 10$, אולם עבור ההצבה $\left\lceil \frac{100}{10} \right\rceil < 10$ אי השוויון לא מתקיים, לכן לא ניתן להשתמש באלגוריתם זה עבור פעולת הצירוף.

סך הכל, קיבלנו כי **אלגוריתם** BNL הוא בעל העלות הנמוכה ביותר, ולכן יהיה היעיל לחישוב התוצאה.
עץ ה־$query\ plan$:



ה. כפי שחושב בסעיף הקודם, עלות החישוב הנה **550** פעולות I/O.

שאלה 3

א. כדי לחשב את מספר השורות בתוצאה, ראשית נחשב את מספר הבלוקים בכל טבלה. נתון כי בכל בלוק מספר הבייטים הוא $2,000$.

- עבור הטבלה R: מספר האטריביוטים ב־R הוא 2, גודל כל אחד מהאטריביוטים הוא 10 בייטים, לכן סך הבייטים בשורה הוא 20 ומספר השורות בבלוק הוא $\left\lfloor \frac{2,000}{20} \right\rfloor = 100$. בטבלה R יש $4,000$ בלוקים, לכן מספר השורות בטבלה הוא $4,000 \cdot 100 = 400,000$. נחשב את מספר השורות בטבלה לאחר הבחירה על אטריביוט B. נתון כי $V(R,B) = 10$, ומהנחת התפלגות אחידה נקבל כי מספר השורות המתאימות יהיו $\left\lceil \frac{400,000}{10} \right\rceil = 40,000$.

- עבור הטבלה S: מספר האטריביוטים ב־S הוא 3, גודל כל אחד מהאטריביוטים הוא 10 בייטים, לכן סך הבייטים בשורה הוא 30 ומספר השורות בבלוק הוא $\left\lfloor \frac{2,000}{30} \right\rfloor = 66$. מספר הבלוקים ב־$S$ הוא $1,200$,לכן מספר השורות בטבלה הוא $1,200 \cdot 66 = 79,200$. לא ידועה התפלגות הערכים המקיימים כי אטריביוט D שלהם קטן מ־5, לכן ככלל אצבע נקבל כי מספר השורות המתאימות הוא $\frac{79,200}{3} = 26,400$.

מספר השורות בתוצאה נתון על ידי החישוב הבא $\frac{T(R) \cdot T(S)}{max\{V(R,A), V(S,A)\}} = \frac{40,000 \cdot 79,200}{max\{40,000, 1,000\}} \cdot \frac{1}{10} \cdot \frac{1}{3} = 2,640$, כלומר מספר השורות בתוצאה הוא **2,640**.

ב. נשים לב כי בביטוי אנחנו מטילים על האטריביוטים A, D, כלומר גודל כל שורה בתוצאת הצירוף היא 20 בייטים. מספר השורות שנכנסות בבלוק הוא $\left\lceil \frac{2000}{20} \right\rceil = 100$. כפי שחישבנו בסעיף הקודם, מספר השורות בתוצאה הוא 2,640 ולכן גודל התוצאה בבלוקים הוא $\left\lceil \frac{2,640}{100} \right\rceil = 27$, כלומר **27** בלוקים.

ג. בכל אחד מהאלגוריתמים נבצע את פעולות הבחירה וההטלה לפני ביצוע הצירוף. מכיוון שאין לנו אינדסקים באף אחת מן הטבלאות נשתמש ב־$Full\,Table\,Scan$ כדי לבצע את הבחירה של השורות הרלוונטיות מ־S ומ־R. נגדיר:

$$E_R = \pi_A \sigma_{B=20} R\,(A, B)$$
$$E_S = \pi_{A,D} \sigma_{D<5} S\,(A, C, D)$$

נשים לב כי לאחר ההטלה של S על האטריביוטים A, D כל שורה בטבלה היא בגודל 20 בתים, ולכן יש $\left\lfloor \frac{2,000}{20} \right\rfloor = 100$ שורות בבלוק. כלומר מתקיים:

$$B\,(E_S) = \left\lceil \frac{T\,(E_S)}{100} \right\rceil = \left\lceil \frac{26,400}{100} \right\rceil = 264$$

לאחר ההטלה של R על האטריביוט A כל שורה בטבלה היא בגודל 10 בתים, ולכן יש $\left\lfloor \frac{2,000}{10} \right\rfloor = 200$ שורות בבלוק. כלומר מתקיים:

$$B\,(E_R) = \left\lceil \frac{T\,(E_R)}{200} \right\rceil = \left\lceil \frac{40,000}{200} \right\rceil = 200$$

מכיוון שאחנו משתמשים ב־$Full\,Table\,Scan$ מתקיים $Read\,(E_S) = B\,(S) = 1,200$, ו־ $Read\,(E_R) = B\,(R) = 4,000$.

נחשב עבור כל אלגוריתם את עלות פעולת הצירוף ונבחר באלגוריתם בעל העלות הנמוכה ביותר, כאשר נתון כי גודל החוצץ הוא 70.

- BNL - נבחר את היחס R כיחס החיצוני מכיוון שביחס זה מספר הבלוקים הוא הקטן יותר לאחר ההטלה, נציב בנוסחה לחישוב עלות צירוף BNL ונקבל:

$$Read\,(E_R) + Read\,(E_S) \cdot \left\lceil \frac{B(E_R)}{M-2} \right\rceil = 4,000 + 1,200 \cdot \left\lceil \frac{200}{70-2} \right\rceil = 7,600$$

כלומר עלות אלגוריתם זה היא **7,600** פעולות I/O.

- INL - אין לנו אינדקסים ולכן לא נוכל להשתמש באלגוריתם.
- HJ - נבדוק האם ניתן להשתמש באלגוריתם זה, כלומר אם מתקיים: $\left\lceil \frac{B(E_S)}{M-1} \right\rceil < M-1$ או $\left\lceil \frac{B(E_R)}{M-1} \right\rceil < M-1$. נציב באפשרות הראשונה ונקבל כי $\left\lceil \frac{200}{70-1} \right\rceil < 70-1$, אי שוויון זה אכן מתקיים וזהו תנאי מספיק לשימוש ב-HJ. כעת, נחשב את עלות השימוש באלגוריתם:

$$Read\,(E_S) + Read\,(E_R) + 2B(E_S) + 2B(E_R) = 1,200 + 4,000 + 2 \cdot 264 + 2 \cdot 200 = 6,128$$

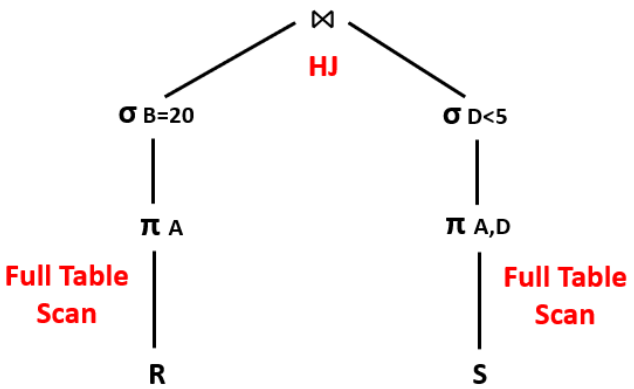
כלומר עלות אלגוריתם זה היא **6,128** פעולות I/O.

- SM - נבדוק האם ניתן להשתמש באלגוריתם זה, כלומר האם מתקיים $\left\lceil \frac{B(E_S)}{M} \right\rceil < M$ וגם $\left\lceil \frac{B(E_R)}{M} \right\rceil < M$. נציב ונקבל כי אכן מתקיים $\left\lceil \frac{264}{70} \right\rceil < 70$ וגם $\left\lceil \frac{200}{70} \right\rceil < 70$, לכן נוכל להשתמש ב-$SMJ$ זה עבור פעולת הצירוף. נשים לב כי מתקיים גם $\left\lceil \frac{B(E_S)}{M} \right\rceil + \left\lceil \frac{B(E_R)}{M} \right\rceil < M$, ולכן נוכל לבצע את האלגוריתם בעלות אופטימלית תוך המנעות ממיון מלא של הטבלאות. נחשב את עלות השימוש באלגוריתם:

$$Read\,(E_S) + Read\,(E_R) + 2B(E_S) + 2B(E_R) = 1,200 + 4,000 + 2 \cdot 264 + 2 \cdot 200 = 6,128$$

כלומר עלות אלגוריתם זה היא **6,128** פעולות I/O.

סך הכל, קיבלנו כי **האלגוריתמים** $\mathbf{HJ}, \mathbf{SM}$ הם בעלי העלות הנמוכה ביותר, ולכן יהיו היעילים ביותר לחישוב התוצאה. עץ ה-$query\ plan$:



ד. כפי שחושב בסעיף הקודם, עלות החישוב הנה **6,128** פעולות I/O.

שאלה 4

א. הרצת השאילתה לקחה **יותר מ־2 דקות**.

הרצת שאילתה עם הפקודה $explain$:

```
                                        QUERY PLAN
-----------------------------------------------------------------------------------------
 Unique  (cost=54723477.96..54723482.33 rows=250 width=44)
   ->  Sort  (cost=54723477.96..54723478.58 rows=250 width=44)
         Sort Key: m1.movieid, m1.title, m1.rating, m1.year, m1.duration, m1.genre
         ->  Seq Scan on movies m1  (cost=0.00..54723468.00 rows=250 width=44)
               Filter: (duration = (SubPlan 1))
               SubPlan 1
                 ->  Aggregate  (cost=1094.44..1094.45 rows=1 width=4)
                       ->  Seq Scan on movies m2  (cost=0.00..1093.00 rows=575 width=4)
                             Filter: (year = m1.year)
 JIT:
   Functions: 10
   Options: Inlining true, Optimization true, Expressions true, Deforming true
(12 rows)
```

ב. השאילתה החדשה:

```
select movieId, title, rating, year, duration, genre
from Movies M1 Natural Join
(
        select year, min(duration) as duration
        from Movies M1
        group by year
) M
```

זמן הרצת השאילתה הוא $142.537\,\mathrm{ms}$.

לדעתנו, מה שגרם לשיפור בזמן הריצה הוא שלא עברנו על כל הטבלה $Movies$ עבור כל סרט. לאחר השינוי עוברים על הטבלה $Movies$ פעם אחת, קיבצנו סרטים שיצאו באותה השנה, בחרנו את אורך הסרט המינימלי של כל קבוצה, והשתמשנו בתוצאה על מנת לחלץ מתוך הטבלה כולה את הסרטים הרלוונטים באמצעות $Natural\ Join$. בנוסף, מכיוון שאנחנו מחזירות שורה עם שדה מפתח, לא יכולות להיות שורות כפולות, ולכן הורדנו את ה־$distinct$ בשאילתה החיצונית (פעולה נוספת החוסכת בזמן ריצה). הורדנו את ה־$distinct$ גם מהשאילתה הפנימית מכיוון שקיבצנו לפי שנים, ולכן שנה לא תחזור פעמיים ולא יהיו שורות כפולות.

הרצת שאילתה עם הפקודה $explain\ analyze$:

```
                                                                QUERY PLAN
-----------------------------------------------------------------------------------------------------------------------------------
-
 Hash Join  (cost=1221.04..2451.57 rows=173 width=44) (actual time=73.420..142.237 rows=116 loops=1)
   Hash Cond: ((m1.year = m1_1.year) AND (m1.duration = (min(m1_1.duration))))
   ->  Seq Scan on movies m1  (cost=0.00..968.00 rows=50000 width=44) (actual time=0.013..33.374 rows=50000 loops=1)
   ->  Hash  (cost=1219.74..1219.74 rows=87 width=8) (actual time=73.395..73.396 rows=88 loops=1)
         Buckets: 1024  Batches: 1  Memory Usage: 12kB
         ->  HashAggregate  (cost=1218.00..1218.87 rows=87 width=8) (actual time=73.247..73.312 rows=90 loops=1)
               Group Key: m1_1.year
               ->  Seq Scan on movies m1_1  (cost=0.00..968.00 rows=50000 width=8) (actual time=0.004..33.612 rows=50000 loops=1)
 Planning Time: 0.169 ms
 Execution Time: 142.368 ms
(10 rows)
```

ג. אפשר לשפר את זמן הריצה של השאילתה ע"י הוספת אינקס. ניסינו להריץ את השאילתה עם האינדקסים הבאים: אינדקס על הזוג $(year, duration)$, אינדקס על $year$ בלבד, אינדקס על $duration$ בלבד.

כפי שצפינו, יצירת אינדקס בודד על השדה $year$ או על השדה $duration$ השאירה את זמן הריצה זהה לסעיף הקודם (מסד הנתונים בחר שלא להשתמש באינדקס). האינדקס ששיפר את זמן הריצה בצורה הטובה ביותר היה האינדקס על הזוג $(year, duration)$:

```
                                                    QUERY PLAN
----------------------------------------------------------------------------------------------------------------------------------
---------------
 Nested Loop  (cost=1218.29..1959.83 rows=173 width=44) (actual time=71.980..73.502 rows=116 loops=1)
   ->  HashAggregate  (cost=1218.00..1218.87 rows=87 width=8) (actual time=71.899..71.972 rows=90 loops=1)
         Group Key: m1_1.year
         ->  Seq Scan on movies m1_1  (cost=0.00..968.00 rows=50000 width=8) (actual time=0.011..33.396 rows=50000 loops=1)
   ->  Index Scan using movies_year_duration_idx on movies m1  (cost=0.29..8.49 rows=2 width=44) (actual time=0.013..0.014 ro
ws=1 loops=90)
         Index Cond: ((year = m1_1.year) AND (duration = (min(m1_1.duration))))
 Planning Time: 0.573 ms
 Execution Time: 73.699 ms
(8 rows)
```

לפני הוספת האינדקס זמן הריצה הכולל היה $142.537\,ms$, ולאחר הוספת האינדקס זמן הריצה הכולל היה $74.272\,ms$.

נשים לב, כי לפני הוספת האינדקס, מערכת ה־DB בחרה להשתמש באלגוריתם $hash\,join$ על מנת לבצע את השאילתה. לאחר הוספת האינדקס, מערכת ה־DB בחרה להשתמש באלגוריתם INL שכן הוספת האינדקס אפשרה שימוש באלגוריתם זה ועלותו הייתה נמוכה יותר מזו של $hash\,join$. כמו כן, מכיוון שהגדרנו את האינדקס על שני שדות הבחירה, אין צורך לגשת לשורות בטבלה $Movies$, אלא רק להגיע לרוץ על העלים המתאימים בעץ - דבר שחוסך זמן ריצה.