פתרון תרגיל מספר 4 - מסדי נתונים

שם: מיכאל גרינבאום, **ת.ז:** 211747639

30 בדצמבר 2019

1. **פתרון:**

(א) **צ"ל:** עלות השאילתה
הוכחה:
מספר בלוקים ב$Visit$: $B\left(Visit\right)=\frac{T(Visit)}{\text{tuples per block}}=\frac{500000}{40}=12500$

i. **ללא אינדקס:** נצטרך לבצע $\boxed{12500}$ פעולות $I\backslash O$ כי נצטרך לעבור על כולם ולבדוק האם יש אחד שמקיים את התנאי

ii. **עם אינדקס:**

א'. נצטרך לבצע $\log_{10}500000$ פעולות כדאי לטייל בעץ

ב'. נצטרך סך הכל לבדוק עלה אחד ולבדוק האם הוא מכיל ערך גדול מ- 990

ג'. לא נצטרך לקרוא עוד מהזיכרון

לכן עם אינדקס נצטרך $\boxed{\log_{10}500000+1=7}$ פעולות $I\backslash O$

מ.ש.ל.א.☺

(ב) **צ"ל:** עלות השאילתה
הוכחה:
תחילה נשים לב שיש $\frac{1000-990}{1000}=\frac{10}{100}=1\%$ של ערכים שה- fee שלהם הוא גדול מ- 990,
בנוסף לזה, נשים לב כי יש $T\left(Visit\right)\cdot 1\%=5000$ שורות שמקיימים את התנאי ש- $fee>990$,
לכל עלה בעץ יש 10 ערכים, ולכן ה- 5000 שורות הללו יהיו ב- $\frac{5000}{10}=500$ עלים.

i. **ללא אינדקס:** נצטרך לבצע $\boxed{12500}$ פעולות $I\backslash O$ (חושב בסעיף הקודם) כי נצטרך לעבור על כולם ולסכום את כל אלה שמקיימים את התנאי

ii. **עם אינדקס:**

א'. נצטרך לבצע $\log_{10}500000$ פעולות כדאי לטייל בעץ

ב'. נצטרך לבדוק את כל העלים שמקיימים את התנאי, מהחישוב לפני יש 500 כאלה

ג'. לא נצטרך לקרוא עוד מהזיכרון

לכן עם אינדקס נצטרך $\boxed{\log_{10}500000+500=506}$ פעולות $I\backslash O$

מ.ש.ל.ב.☺

(ג) **צ"ל:** עלות השאילתה
הוכחה:

i. **ללא אינדקס:** נצטרך לבצע $\boxed{12500}$ פעולות $I\backslash O$ (חושב בסעיף הקודם) כי נצטרך לעבור על כולם ולסכום את כל אלה שמקיימים את התנאי

ii. **עם אינדקס:**

א'. נצטרך לבצע $\log_{10}500000$ פעולות כדאי לטייל בעץ

ב'. נצטרך לבדוק את כל העלים שמקיימים את התנאי, מהחישוב (בסעיף הקודם) יש 500 כאלה

ג'. בסעיף הקודם חישבנו שיש 5000 שורות שעונות על התנאי,

לכן עם אינדקס נצטרך $\boxed{\log_{10} 500000 + 500 + 5000 = 5506}$ פעולות $I\backslash O$

מ.ש.ל.ג.☺

(ד) **צ"ל:** עלות השאילתה

הוכחה:

מספר בלוקים ב־ $Patient$: $B\left(Patient\right) = \frac{T(Patient)}{\text{tuples per block}} = \frac{40000}{12} = 3334$

i. **ללא אינדקס:** נצטרך לבצע $\boxed{3334}$ פעולות $I\backslash O$ (חושב בסעיף הקודם) כי נצטרך לעבור על כולם ולסכום את כל אלה שמקיימים את התנאי

ii. **עם אינדקס:**

א'. נצטרך לבצע $\log_{10} 40000$ פעולות כדאי לטייל בעץ

ב'. נצטרך לבדוק רק עלה אחד עם ה־ pid המתאים

ג'. נצטרך לקרוא את הבלוק שבו הוא מופיע בשביל השם שלו

לכן עם אינדקס נצטרך $\boxed{\log_{10} 40000 + 1 + 1 = 7}$ פעולות $I\backslash O$

מ.ש.ל.ד.☺

2. **פתרון:**

(א) **צ"ל:** יעילות $R \bowtie S$

הוכחה:

נשים לב כי $B\left(R\right) = \frac{T(R)}{100} = \frac{2000000}{100} = 20000$, וגם $B\left(S\right) = \frac{5000}{20} = 250$

i. Block nested loop:

$$\min\left\{Read\left(R\right) + Read\left(S\right) \cdot \left\lceil \frac{B\left(R\right)}{M-2} \right\rceil, Read\left(S\right) + Read\left(R\right) \cdot \left\lceil \frac{B\left(S\right)}{M-2} \right\rceil\right\}$$
$$= \min\left\{20000 + 250 \cdot \left\lceil \frac{20000}{100} \right\rceil, 250 + 20000 \cdot \left\lceil \frac{250}{100} \right\rceil\right\}$$
$$= 250 + 20000 \cdot \left\lceil \frac{250}{100} \right\rceil = 250 + 20000 \cdot 3 = \boxed{60250\ I\backslash O}$$

ii. sort merge join:

נבדוק האם התנאי של $sort - merge$ מתקיים

$$\left(\left\lceil \frac{B\left(S\right)}{M} \right\rceil < M\right) \wedge \left(\left\lceil \frac{B\left(R\right)}{M} \right\rceil < M\right)$$

תחילה נבדוק את R,

$$\left\lceil \frac{B\left(R\right)}{M} \right\rceil < M \Leftrightarrow \left\lceil \frac{20000}{102} \right\rceil < 102 \Leftrightarrow 197 < 102$$

התנאי **לא** מתקיים, כלומר לא נוכל לעשות $sort - merge$ בזמן יעיל :(

iii. hash join:

נבדוק האם התנאי של $hash - merge$ מתקיים

$$\left(\left\lceil \frac{B\left(R\right)}{M-1} \right\rceil < M-1\right) \vee \left(\left\lceil \frac{B\left(S\right)}{M-1} \right\rceil < M-1\right)$$

נבדוק את S,

$$\left\lceil \frac{B\left(S\right)}{M-1} \right\rceil < M-1 \Leftrightarrow \left\lceil \frac{250}{101} \right\rceil < 101 = 3 < 101$$

התנאי מתקיים, ולכן נוכל למיין עם $hash-join$ ויעילות המיון היא

$$Read\left(S\right)+Read\left(R\right)+2\cdot\left(B\left(R\right)+B\left(S\right)\right)=20000+250+2\cdot\left(20000+250\right)=\boxed{60750I\backslash O}$$

מ.ש.ל.א.☺

(ב) **צ"ל:** יעילות $R\bowtie S$

הוכחה:

נשים לב כי $B\left(R\right)=\frac{T(R)}{100}=\frac{2000000}{100}=20000$, וגם $B\left(S\right)=\frac{5000}{20}=250$

i. Block nested loop:

$$\begin{aligned}\min\left\{Read\left(R\right)+Read\left(S\right)\cdot\left\lceil\frac{B\left(R\right)}{M-2}\right\rceil,Read\left(S\right)+Read\left(R\right)\cdot\left\lceil\frac{B\left(S\right)}{M-2}\right\rceil\right\}\\=\min\left\{20000+250\cdot\left\lceil\frac{20000}{298}\right\rceil,250+20000\cdot\left\lceil\frac{250}{298}\right\rceil\right\}\\=250+20000\cdot\left\lceil\frac{250}{298}\right\rceil=250+20000=\boxed{20250\ I\backslash O}\end{aligned}$$

ii. sort merge join:

נבדוק האם התנאי של $sort-merge$ מתקיים

$$\left\lceil\frac{B\left(S\right)}{M}\right\rceil+\left\lceil\frac{B\left(R\right)}{M}\right\rceil<M$$

נשים לב כי

$$\left\lceil\frac{B\left(S\right)}{M}\right\rceil+\left\lceil\frac{B\left(R\right)}{M}\right\rceil<M\Leftrightarrow\left\lceil\frac{250}{300}\right\rceil+\left\lceil\frac{20000}{300}\right\rceil<300\Leftrightarrow68<300$$

התנאי מתקיים, כלומר נוכל לעשות $sort-merge$ בזמן יעיל ביחד אפילו, וזמן הריצה הוא

$$Read\left(S\right)+Read\left(R\right)+2\cdot\left(B\left(R\right)+B\left(S\right)\right)=20000+250+2\cdot\left(20000+250\right)=\boxed{60750I\backslash O}$$

iii. hash join:

ראינו שבסעיף א' גודל ה־ buffer היה מספיק גדול למיון האש, לכן אם נגדיל אותו התנאי עדיין יתקיים ולכן יעילות המיון היא:

$$Read\left(S\right)+Read\left(R\right)+2\cdot\left(B\left(R\right)+B\left(S\right)\right)=20000+250+2\cdot\left(20000+250\right)=\boxed{60750I\backslash O}$$

מ.ש.ל.ב.☺

(ג) **צ"ל:** דרישה מינימלית על הbuffer

הוכחה:

i. Block nested loop:

נשים שצריך בלוק 1 לקריאת יחס אחד , לפחות 1 לקריאת היחס השני ולפחות 1 לכתיבת הזיכרון, לכן צריך לפחות 3 בלוקים בbuffer, כלומר המינימלי הוא $\boxed{M=3}$

ii. sort merge join:

נרצה למצוא M מינימלי כך ש$\left\lceil\frac{B(R)}{M}\right\rceil<M\wedge\left\lceil\frac{B(S)}{M}\right\rceil<M$, נשים לב כי $B\left(R\right)>B\left(S\right)$ ולכן מספיק לדרוש $\left\lceil\frac{B(R)}{M}\right\rceil<M$, כלומר $B\left(R\right)<M^2$ בערך. נציב $B\left(R\right)=20000$ ונקבל $M=142$, נשים לב כי $\left\lceil\frac{20000}{142}\right\rceil=141<142$, כלומר המינימלי הוא $\boxed{M=142}$

iii. hash join:

נרצה למצוא M מינימלי כך ש$\left\lceil \frac{B(S)}{M-1} \right\rceil < M-1 \vee \left\lceil \frac{B(R)}{M-1} \right\rceil < M-1$, נשים לב כי $B(R) > B(S)$ ולכן מספיק לדרוש $\left\lceil \frac{B(S)}{M-1} \right\rceil < M-1$, כלומר $B(S) < (M-1)^2$ בערך. נציב $B(S) = 250$ ונקבל $M = 17$, נשים לב כי $16 < \left\lceil \frac{250}{16} \right\rceil = 16$ לא עובד, לכן נציב $M = 18$ ונקבל $15 = \left\lceil \frac{250}{17} \right\rceil < 17$, כלומר המינימלי הוא $\boxed{M = 18}$

מ.ש.ל.ג.☺

3. **פתרון:**

(א) **צ"ל:** גודל $\sigma_{C<3}(S(B,C))$

הוכחה:

נשים לב כי אין לנו נתונים על C, לכן נגיד שכל בלוק שלישי מקיים את הנדרש, נשים לב כי $T(S) = 5 \cdot B(S) = 1,500$ ולכן $T(\text{select S}) = \left\lceil \frac{1500}{3} \right\rceil = 500$, ולכן מספר הבלוקים שמקיימים את הנדרש הוא $B(\text{select S}) = \left\lceil \frac{T(\text{select S})}{5} \right\rceil = \left\lceil \frac{500}{5} \right\rceil = \boxed{100}$ בלוקים

מ.ש.ל.א.☺

(ב) **צ"ל:** גודל $\sigma_{A=11}(R(A,B))$

הוכחה:

נשים לב כי יש 100 ערכים שונים ו־ 5000 בלוקים. נניח שההתפלגות של הערכים אחידה ולכן 1% מהערכים יהיה עם $A = 11$, נשים לב כי $T(R) = 10 \cdot B(R) = 50000$ ולכן $T(\text{select R}) = \left\lceil \frac{50000}{100} \right\rceil = 500$, לכן מספר הבלוקים שיוחזרו מהשאילתה הוא $B(\text{select R}) = \left\lceil \frac{T(\text{select R})}{10} \right\rceil = \left\lceil \frac{500}{10} \right\rceil = \boxed{50}$ בלוקים.

מ.ש.ל.ב.☺

(ג) **צ"ל:** גודל $\sigma_{A=11 \wedge C<3}(R(A,B) \bowtie S(B,C))$

הוכחה:

נשים לב כי ראינו שהנוסחא לחישוב זה הינה

$$\frac{T(\text{select R}) \cdot T(\text{select S})}{\max\{V(S,B), V(R,B)\}} = \frac{T(\text{select R}) \cdot T(\text{select S})}{\max\{20, T(\text{select R})\}} = \frac{T(\text{select R}) \cdot T(\text{select S})}{T(\text{R})} = \frac{500 \cdot T(\text{select S})}{50000} = 5$$

כלומר השאילתה תחזיר $\boxed{5}$ שורות

מ.ש.ל.ג.☺

(ד) **צ"ל:** עץ ג'וין

הוכחה:

נחשב את $block-nested-loop$:

נשים לב כי לעשות $select$ על R יעלה הרבה יותר עם קריאה של כל הבלוקים מאשר שימוש באינדקס הנתון, לכן נשתמש באינדקס על R שמכיל 500 שורות ולשמור איפשהו בזיכרון ב־50 בלוקים.

$$\min\left\{Read(R) + Read(S) \cdot \left\lceil \frac{B(R)}{M-2} \right\rceil, Read(S) + Read(R) \cdot \left\lceil \frac{B(S)}{M-2} \right\rceil\right\}$$
$$= \min\left\{500 + 300 \cdot \left\lceil \frac{50}{8} \right\rceil, 300 + 5000 \cdot \left\lceil \frac{100}{8} \right\rceil\right\} = 500 + 2100 = \boxed{2600 I\backslash O}$$

נחשב את $sort-merge-join$: נשים לב כי $\frac{B(\text{select S})}{10} = \frac{100}{10} = 10 \not< 10$, ולכן לא נוכל לעשות $sort-merge-join$

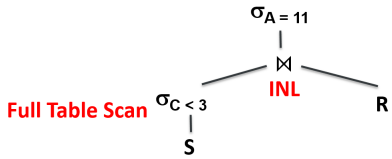
נחשב את $hash-join$: נשים לב כי $\frac{B(\text{select R})}{9} = \frac{50}{9} < 9$, ולכן נוכל לעשות $hash-join$ ולכן היעילות היא

$$Read(R) + Read(S) + 2 \cdot (B(S) + B(R)) = 5000 + 300 + 2 \cdot (50 + 100) = \boxed{5600 I\backslash O}$$

נחשב את $index-loop-join$:

$$Read(S) + T(\text{select S}) \cdot \text{access index R} = 300 + 500 \cdot 1 = \boxed{800 I\backslash O}$$

כלומר הג'וין הכי יעיל במקרה זה הוא $index-loop-join$



מ.ש.ל.ד.☺

(ה) **צ"ל:** יעילות
הוכחה:
מהסעיף הקודם ראינו שהיעילות היא $\boxed{800I\backslash O}$

מ.ש.ל.ה.☺

4. **פתרון:**

(א) **צ"ל:** מספר שורות
הוכחה:
תחילה נשים לב כי כמות השורות בבלוק בR הוא $\frac{3000}{3\cdot 10} = 100$, לכן $T(R) = 100 \cdot B(R) = 100000$, מהיות ואין נתונים, נסיק כי $T(\text{R select}) = \frac{100000}{3} = 33334$, נשים לב כי,
עתה נשים לב שלהמשך השאילתה נצטרך רק 2 מתוך ה3 עמודות ולכן נוכל לשמור רק את A, B ולכן ייכנסנו לנו $\frac{3000}{20} = 150$ שורות בבלוק ולכן $B(\text{R select}) = \frac{33334}{150} = 223$.
תחילה נשים לב כי כמות השורות בבלוק בS הוא $\frac{3000}{2\cdot 10} = 150$, לכן $T(S) = 150 \cdot 90 = 13500$, מהיות ואין נתונים, נסיק כי $T(\text{S select}) = \frac{13500}{3} = 4500$.
נשים לב כי $B(\text{S select}) = \frac{4500}{150} = 30$
עתה נקבל כי מספר השורות בסוף הוא

$$\frac{T(\text{R select}) \cdot T(\text{S select})}{\max\{V(S,B), V(R,B)\}} = \frac{T(\text{R select}) \cdot T(\text{S select})}{T(R)} = \frac{3334 \cdot T(\text{S select})}{100000} = 1500$$

לכן יש $\boxed{1500}$ שורות

מ.ש.ל.א.☺

(ב) **צ"ל:** מספר בלוקים
הוכחה:
נשים לב שבשורה ג'וין יש 20 בטים בשורה, ולכן בבלוק יש $\frac{3000}{20} = 150$ שורות.
לכן התוצאה תשמר ב־ $\boxed{\frac{1500}{150} = 10}$ בלוקים.

מ.ש.ל.ב.☺

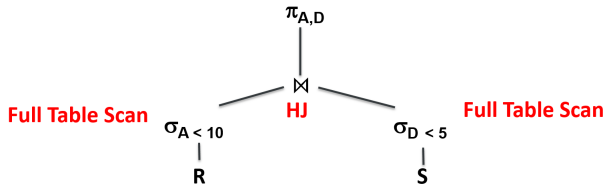
(ג) **צ"ל:** חישוב הכי יעיל
הוכחה:
$hash - merge$: נשים לב כי $\frac{B(\text{S select})}{21} = \frac{30}{21} < 5 < 21$, לכן אפשר וזה יעלה

$$B(S) + B(R) + 2 \cdot (B(\text{R select}) + B(\text{S select})) = 1000 + 90 + 2 \cdot (223 + 30) = \boxed{1596I\backslash O}$$

מהיות $hash$ ביעילות טובה \ שווה ל$sort - merge$, לא נבדוק אותו
$block - nested - loop$:
נשים

$$\min\left\{1000 + 90 \cdot \left\lceil \frac{B(\text{R select})}{20} \right\rceil, 90 + 1000 \cdot \left\lceil \frac{B(\text{S select})}{20} \right\rceil\right\} = 1000 + 90 \cdot 12 = \boxed{2080I\backslash O}$$

לכן הג'וין היעיל במקרה זה הוא $hash$ הכי יעיל.



מ.ש.ל.ג.☺

(ד) **צ"ל:** חישוב הכי יעיל

הוכחה:

לפי מה שראינו קודם $hash$ הכי יעיל ביעילות $\boxed{1596I\backslash O}$

מ.ש.ל.ד.☺

(ה) **צ"ל:** חישוב הכי יעיל

הוכחה:

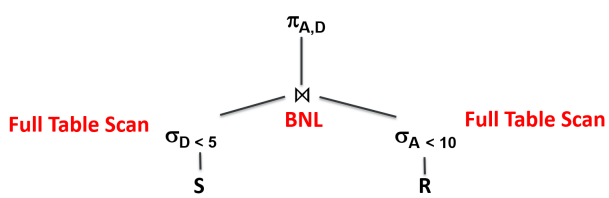
$hash - merge$: נשים לב כי $\frac{B(\text{S select})}{21} < \frac{20}{21} < 5 < 21$, לכן אפשר וזה יעלה

$$B(S) + B(R) + 2 \cdot (B(\text{R select}) + B(\text{S select})) = 1000 + 60 + 2 \cdot \left(223 + \left\lceil \frac{60}{3} \right\rceil\right) = \boxed{1546I\backslash O}$$

מהיות $hash$ ביעילות טובה \ שווה ל$sort - merge$, לא נבדוק אותו $block - nested - loop$:

$$\min\left\{1000 + 60 \cdot \left\lceil \frac{223}{20} \right\rceil, 60 + 1000 \cdot \left\lceil \frac{\left\lceil \frac{60}{3} \right\rceil}{20} \right\rceil\right\} = 60 + 1000 \cdot \left\lceil \frac{\left\lceil \frac{60}{3} \right\rceil}{20} \right\rceil = \boxed{1060I\backslash O}$$

לכן הג'וין היעיל במקרה זה הוא $block - nested - loop$ הכי יעיל כשR הוא יחס פנימי וS חיצוני ומהיות זה עלות הקריאה של 2 הקבצים ואין אינדקס, הוא יהיה אופטימלי.



מ.ש.ל.ה.☺

5. **פתרון:**

(א) רץ יותר מ־2 דקות

```
public=> EXPLAIN ANALYZE
public-> select distinct *
public-> from People P1
public-> where bdate = (select min(bdate)
public(> from People P2
public(> where P2.country = P1.country);
                                                    QUERY PLAN
-----------------------------------------------------------------------------------------------------------------------------
 HashAggregate  (cost=69027634.38..69027636.88 rows=250 width=81) (actual time=322106.629..322106.858 rows=246 loops=1)
   Group Key: p1.id, p1.name, p1.phone_number, p1.city, p1.country, p1.job_title, p1.bdate
   ->  Seq Scan on people p1  (cost=0.00..69027630.00 rows=250 width=81) (actual time=4328.979..322105.325 rows=246 loops=1)
         Filter: (bdate = (SubPlan 1))
         Rows Removed by Filter: 49754
         SubPlan 1
           ->  Aggregate  (cost=1380.52..1380.53 rows=1 width=4) (actual time=6.437..6.438 rows=1 loops=50000)
                 ->  Seq Scan on people p2  (cost=0.00..1380.00 rows=206 width=4) (actual time=0.032..6.247 rows=208 loops=50000)
                       Filter: ((country)::text = (p1.country)::text)
                       Rows Removed by Filter: 49792
 Planning time: 0.209 ms
 Execution time: 322107.138 ms
(12 rows)
```

(ב) הניסיון שלי לשפר זמן ריצה: (רץ בערך בשנייה)

```
WITH temp(country, bdate) as
    (SELECT country, min(bdate)
     FROM People P
     GROUP BY country)
SELECT distinct *
FROM People P1
WHERE EXISTS(SELECT *
             FROM temp T
             WHERE T.country = P1.country AND T.bdate = P1.bdate
            );
```

```
public=> EXPLAIN ANALYZE
public-> WITH temp(country, bdate) as
public->     (SELECT country, min(bdate)
public(>   FROM People P
public(>   GROUP BY country)
public-> SELECT distinct *
public-> FROM People P1
public-> WHERE EXISTS(SELECT *
public(>     FROM temp T
public(>     WHERE T.country = P1.country AND T.bdate = P1.bdate
public(>    );
                                                        QUERY PLAN
-------------------------------------------------------------------------------------------------------------------------------
 HashAggregate  (cost=3367.28..3492.28 rows=12500 width=81) (actual time=199.688..199.983 rows=246 loops=1)
   Group Key: p1.id, p1.name, p1.phone_number, p1.city, p1.country, p1.job_title, p1.bdate
   CTE temp
     ->  HashAggregate  (cost=1505.00..1507.43 rows=243 width=15) (actual time=103.962..104.193 rows=243 loops=1)
           Group Key: p.country
           ->  Seq Scan on people p  (cost=0.00..1255.00 rows=50000 width=15) (actual time=0.009..45.338 rows=50000 loops=1)
   ->  Hash Join  (cost=11.07..1641.11 rows=12500 width=81) (actual time=106.747..199.226 rows=246 loops=1)
         Hash Cond: (((p1.country)::text = (t.country)::text) AND (p1.bdate = t.bdate))
         ->  Seq Scan on people p1  (cost=0.00..1255.00 rows=50000 width=81) (actual time=0.020..44.411 rows=50000 loops=1)
         ->  Hash  (cost=8.07..8.07 rows=200 width=36) (actual time=105.463..105.463 rows=243 loops=1)
               Buckets: 1024  Batches: 1  Memory Usage: 20kB
               ->  HashAggregate  (cost=6.08..8.07 rows=200 width=36) (actual time=104.985..105.201 rows=243 loops=1)
                     Group Key: (t.country)::text, t.bdate
                     ->  CTE Scan on temp t  (cost=0.00..4.86 rows=243 width=36) (actual time=103.969..104.685 rows=243 loops=1)
 Planning time: 0.171 ms
 Execution time: 200.323 ms
```

(ג) אינדקסים:

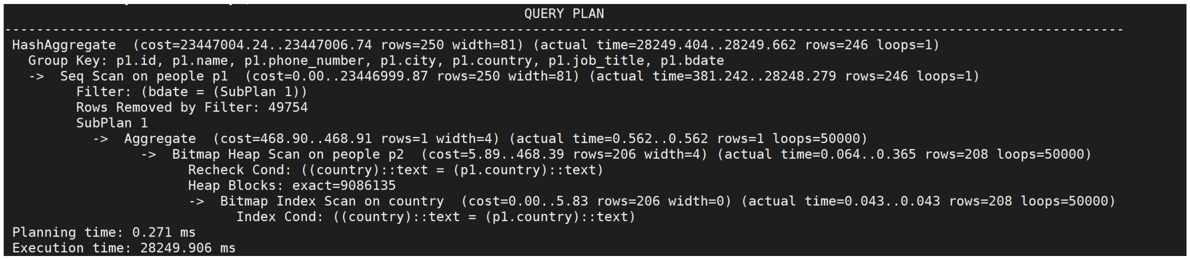
i. האינדקס ההגיוני של עיר ואז תאריך יום הולדת:

```
                                                        QUERY PLAN
-------------------------------------------------------------------------------------------------------------------------------
---
 HashAggregate  (cost=201301.33..201303.83 rows=250 width=81) (actual time=1284.832..1285.074 rows=246 loops=1)
   Group Key: p1.id, p1.name, p1.phone_number, p1.city, p1.country, p1.job_title, p1.bdate
   ->  Seq Scan on people p1  (cost=0.00..201296.95 rows=250 width=81) (actual time=20.510..1284.262 rows=246 loops=1)
         Filter: (bdate = (SubPlan 2))
         Rows Removed by Filter: 49754
         SubPlan 2
           ->  Result  (cost=3.99..4.00 rows=1 width=0) (actual time=0.023..0.024 rows=1 loops=50000)
                 InitPlan 1 (returns $1)
                   ->  Limit  (cost=0.41..3.99 rows=1 width=4) (actual time=0.019..0.020 rows=1 loops=50000)
                         ->  Index Only Scan using country_bdate on people p2  (cost=0.41..736.52 rows=206 width=4) (actual time=0.018..0.018 rows=1 loops=5000
0)
                               Index Cond: ((country = (p1.country)::text) AND (bdate IS NOT NULL))
                               Heap Fetches: 50000
 Planning time: 0.276 ms
 Execution time: 1285.303 ms
(14 rows)
```

ii. האינדקס על תאריך יום הולדת:

```
-------------------------------------------------------------------------------------------------------------------------------
 HashAggregate  (cost=1126980.92..1126983.42 rows=250 width=81) (actual time=6023.223..6023.501 rows=246 loops=1)
   Group Key: p1.id, p1.name, p1.phone_number, p1.city, p1.country, p1.job_title, p1.bdate
   ->  Seq Scan on people p1  (cost=0.00..1126976.54 rows=250 width=81) (actual time=83.463..6022.232 rows=246 loops=1)
         Filter: (bdate = (SubPlan 2))
         Rows Removed by Filter: 49754
         SubPlan 2
           ->  Result  (cost=22.50..22.51 rows=1 width=0) (actual time=0.117..0.118 rows=1 loops=50000)
                 InitPlan 1 (returns $1)
                   ->  Limit  (cost=0.29..22.50 rows=1 width=4) (actual time=0.114..0.115 rows=1 loops=50000)
                         ->  Index Scan using date on people p2  (cost=0.29..4575.95 rows=206 width=4) (actual time=0.112..0.112 rows=1 loops=50000)
                               Index Cond: (bdate IS NOT NULL)
                               Filter: ((country)::text = (p1.country)::text)
                               Rows Removed by Filter: 251
 Planning time: 0.204 ms
 Execution time: 6023.746 ms
```

iii. האינדקס על העיר:



```
                                                                  QUERY PLAN
------------------------------------------------------------------------------------------------------------------------------------------------
HashAggregate  (cost=23447004.24..23447006.74 rows=250 width=81) (actual time=28249.404..28249.662 rows=246 loops=1)
  Group Key: p1.id, p1.name, p1.phone_number, p1.city, p1.country, p1.job_title, p1.bdate
  ->  Seq Scan on people p1  (cost=0.00..23446999.87 rows=250 width=81) (actual time=381.242..28248.279 rows=246 loops=1)
        Filter: (bdate = (SubPlan 1))
        Rows Removed by Filter: 49754
        SubPlan 1
          ->  Aggregate  (cost=468.90..468.91 rows=1 width=4) (actual time=0.562..0.562 rows=1 loops=50000)
                ->  Bitmap Heap Scan on people p2  (cost=5.89..468.39 rows=206 width=4) (actual time=0.064..0.365 rows=208 loops=50000)
                      Recheck Cond: ((country)::text = (p1.country)::text)
                      Heap Blocks: exact=9086135
                      ->  Bitmap Index Scan on country  (cost=0.00..5.83 rows=206 width=0) (actual time=0.043..0.043 rows=208 loops=50000)
                            Index Cond: ((country)::text = (p1.country)::text)
Planning time: 0.271 ms
Execution time: 28249.906 ms
```

מ.ש.ל.☺