תרגיל 3 : SQL מתקדם ואינדקסים

תאריך הגשה : 6.12.20, 23:55.

הוראות הגשה:
בתרגיל זה אתם נדרשים להגיש קובץ zip בודד שיכלול את הקבצים הבאים :

- ex3.pdf עם התשובות לשאלות בחלק ב : אינדקסים.
- q1.sql
- q2.sql
- q3.sql
- q4.sql
- q5.sql
- README שמכיל שורה בודדת ובו ה-login של הסטודנט שמגיש את התרגיל. אם התרגיל מוגש בזוגות, על שורה זאת להכיל את שני ה-login מופרדים בפסיק.

שימו לב:

- נא לקרוא על הדרישות המנהליות של הקורס בלינק באתר הקורס כדי למלא אחר ההוראות להגשה של קבצים סרוקים!
- תרגיל מוקלד יזכה ב- 2 נקודות בונוס!

נתונים היחסים הבאים מתוך מסד נתונים של IMDb (זהים ליחסים בתרגיל 2) :

Movies (movieId, title, rating, year, duration, genre)

Actors (actorId, name, byear, dyear)

PlaysIn (movieId, actorId, character)

הערות :

- לכל סרט (Movie) יש מספר מזהה (movieId), כותרת (title), דירוג הסרט (rating), שנת יציאה (year), משך הסרט בדקות (duration) וז'אנר (genre).
- לכל שחקן (Actor) יש מספר מזהה (actorId), שם (name), שנת לידה (byear), ושנת פטירה (dyear). עבור שחקן חי, dyear הוא null .
- לכל משחק בסרט (PlaysIn) יש מספר מזהה של סרט ושחקן (movieId, actorId), ואת שם הדמות שהשחקן שיחק (character).

באתר הקורס יש קובץ create.sql המכיל הגדרות עבור הטבלאות וקובץ drop.sql המכיל פקודות המוחקות את הטבלאות.
כמו כן, נתונים הקבצים :

- actors.csv
- movies.csv
- playsIn.csv

הקבצים מכילים מידע חלקי אך אמיתי אודות סרטים ושחקנים מהאתר IMDb.
את המידע המלא ניתן למצוא ב https://www.imdb.com/interfaces/
הקבצים שלנו מכילים מידע על כ10,000 סרטים, שעבר "ניקוי" והתאמה לצורך התרגיל.

ניתן למצוא את הקבצים גם במערכת המחשבים במעבדה בתיקיה :

~ db/data/

ניתן להעתיק אותם לתיקיה שלכם.

על מנת לבדוק את התרגיל שלכם, יש ליצור את הטבלאות בעזרת create.sql, ולטעון לתוכן נתונים בעזרת הפקודות

```
cat movies.csv | psql -h dbcourse public -c "copy movies from STDIN DELIMITER ',' CSV HEADER"
cat actors.csv | psql -h dbcourse public -c "copy actors from STDIN DELIMITER ',' CSV HEADER"
cat playsIn.csv | psql -h dbcourse public -c "copy playsIn from STDIN DELIMITER ',' CSV HEADER"
```

חלק א: שאילתות SQL (40 נקודות):

כתבו את השאילתות הבאות בSQL. שם הקובץ שבו צריכה להופיע התשובה לכל שאלה נמצא בתחילת השאלה.

בכל התשובות לשאלות בחלק זה:

- השתמשו ב SELECT DISTINCT כדי למנוע כפילויות בתשובות (אם כפילויות עלולות להוויצר בתשובה).
- **שימו לב:** בכל סעיף כתוב באיזה סדר למיין את התוצאות וכן את שמות העמודות בתוצאה.

1. **(q1.sql)** לכל שחקן, החזירו את actorId, את משך הסרט הארוך ביותר ששיחק בו, משך הסרט הקצר ביותר ששיחק בו, וממוצע אורך הסרטים ששיחק בהם.
יש להחזיר טבלה עם העמודות actorId,max,min,avg ממויין לפי actorId.

```
select actorid, max(duration) as max,min (duration) as min,avg(duration) as avg
from playsIn natural join movies
group by actorid
order by actorid;
```

2. **(q2.sql)** החזר את הכותרת ומספר מזהה של כל הסרטים שממוצע הגילאים של השחקנים בשנת יציאת הסרט היה לפחות 70.
יש להחזיר טבלה עם העמודות movieid,title ממויין לפי movieid

```
select movieid,title
from movies natural join playsin natural join actors
group by movieid
having avg(year-byear)>70
order by movieid;
```

3. **(q3.sql)** החזר את שם ומספר מזהה של השחקן שממוצע הדירוג (rating) של הסרטים ששיחק בהם הוא הגבוה ביותר.
אם יש מספר שחקנים כאלו, החזר את כולם.
יש להחזיר טבלה עם העמודות actorId, name, ממויין לפי actorId.

```
with actorAvgRating(actorid,avgRate) as
(select actorid, avg(rating)
from playsin natural join movies
group by actorid)

select actorid,name
from actors natural join actorAvgRating
where avgRate >=(select max(avgRate) from actorAvgRating)
order by actorid;
```

4. **(q4.sql)** החזר את מספר השחקים ששיחקו אך ורק בסרטים עם לפחות 6 שחקנים (כולל עצמם).
יש להחזיר טבלה עם עמודה בודדת הנקראת num.

```
select count(*) as num
From Actors
Where actorid not in (select actorid
                      from playsin natural join (select movieid,count(*) as
                      actorcount
                                                 from playsin
                                                 group by movieid) as T
                      where actorCount<6);
```

5. **(q5.sql)** כיתבו שאילתה רקורסיבית אשר מחזירה את שמות כל השחקנים שמספר Frank Bacon שלהם הוא לכל היותר 5. הסבר על מספר Bacon אפשר למצוא כאן.
שימו לב שבנתונים שלנו לא מופיע השחקן Kevin Bacon, ולכן אנחנו נשתמש בFrank Bacon במקומו.
יש להחזיר טבלה עם 2 עמודות actorid,name ממויינת לפי actorid.

```
with recursive BaconNum(actorid, num) as
(select actorid,0 from actors where name='Frank Bacon'

Union

select p1.actorid, num+1
from playsin P1, playsin p2 natural join BaconNum BN
where p1.movieid=p2.movieid and BN.num<5)

select distinct actorid,name
from BaconNum natural join actors
order by actorid;
```

חלק ב: אינדקסים (60 נקודות): (להגשה בכתב בקובץ ex3.pdf)

בחלק זה של התרגיל אנחנו עדיין נשתמש בסכמה המובאת פה שוב לייתר נוחות :

Movies (movieId, title, rating, year, duration, genre)

Actors (actorId, name, byear, dyear)

PlaysIn (movieId, actorId, character)

שאלה 1:

א. כתבו שאילתה בSQL המחזירה את המספר המזהה של כל השחקנים ששיחקו בסרט כלשוה דמות ששמה 'Sheriff'.

```
select actorid
from Playsin
where character='Sheriff';
```

ב. הריצו את השאילתה עם פקודת explain analyse, שמראה את הquery plan של השאילתה, צרפי אותה לתשובות.

כיתבו כמה זמן לקח להריץ את השאילתה, והסבירו את אופן חישוב השאילתה.

```
QUERY PLAN
-----------------------------------------------------------------------------------------------
 Seq Scan on playsin  (cost=0.00..632.98 rows=53 width=22) (actual time=0.551..3.101
rows=50 loops=1)
   Filter: (("character"):: text = 'Sheriff'::text)
   Rows Removed by Filter: 32602
 Planning Time: 0.075 ms
 Execution Time: 3.147 ms
```

זמן ריצה : 3.222 ms

הסבר :

השאילתה מבוצעת באמצעות מעבר סדרתי על הטבלה playsIn, ועל כל שורה מופעל תנאי הפילטר character='Sheriff'.

ג. כיתבו פקודה אשר תייצר אינדקס **על שדה בודד** שישפר את זמן הריצה של השאילתה.

```
create index on playsin(character);
```

ד. הריצו את פקודת הבנייה של האינדקס ואת השאילתה עם פקודת explain analyse, שמראה את הquery plan של השאילתה, צרפו אותה לתשובות.
כיתבו כמה זמן לקח להריץ את השאילתה, והסבירו את אופן חישוב השאילתה.
אם אתם לא מבינים לגמרי את ה query plan חפשו באינטרנט דוקומנטציה שתעזור לכם להסביר.

QUERY PLAN

```
----------------------------------------------------------------------------------------------------
 Bitmap Heap Scan on playsin  (cost=4.68..125.19 rows=51 width=4) (actual time=0.068..0.147
rows=50 loops=1)
   Recheck Cond: ((''character'')::text = 'Sheriff'::text)
   Heap Blocks: exact=37
   -> Bitmap Index Scan on xxx  (cost=0.00..4.67 rows=51 width=0) (actual time=0.060..0.061
rows=50 loops=1)
        Index Cond: ((''character'')::text = 'Sheriff'::text)
 Planning Time: 0.610 ms
 Execution Time: 0.201 ms
```

זמן ריצה : 0.811 ms

הסבר :

השאילתה מבוצעת באמצעות אינדקס בשני שלבים.

בשלב הראשון (הפנימי) משתמשים באינדקס כדי למצוא את הכתובות של כל השורות בהן מתקיים character='Sheriff'. בשלב השני מביאים את השורות עצמן מהטבלה.

לאחר מציאת הכתובת של כל השורות בהן מתקיים התנאי, הבלוקים הרלוונטים מובאים מהטבלה בעזרת Bitmap Heap Scan כלומר לפני הקריאה מסדרים בעזרת Bitmap איזה שורות מכל בלוק צריך, ואז כל בלוק נקרא רק פעם אחת, והשורות הנצאו באינדקס מוצאות לפלט.

לפעמים הBitmap לא מקדיש ביט ייחודי לכל שורה, אלא לכל בלוק, ואז התנאי נבדק שוב כנגד כל שורה בבלוק שהובא (Recheck Cond:) לפני שמועבר למשתמש.

שאלה 2:

בסעיפים הבאים, יש לכתוב הסבר לדרך הפתרון, ולהדגיש את התוצאה הסופית של כל חישוב!

הנחות:

- גודל בלוק הוא 1,000 בייטים.
- בטבלה Movies יש 10,000 שורות,
- כל שורה תופסת 150 בייטים.
- התכונה movieId תופסת 8 בייט.
- התכונה duration תופסת 8 בייט.
- התכונה genre תופסת 10 בייט.
- מצביע תופס 8 בייט.
- הערכים בduration בטבלה Movies מתפלגים אחיד בטווח [1,200]
- הערכים בgenre בטבלה מחולקים ל4 קטגוריות באופן אחיד.

א. נתונה השאילתה הבאה:

```
SELECT DISTINCT “exists”
FROM Movies
WHERE duration > 100
```

1. מה עלות חישוב השאילתה בהנחה שאין אינדקסים על הטבלה?

בכל בלוק נכנסות $\lfloor 1{,}000 / 150 \rfloor = 6$ שורות.

הטבלה תופסת $\lceil 10{,}000/6 \rceil = 1{,}667$ בלוקים.

עלות קריאת הטבלה כולה הוא **1,667**.

כעת, נתון האינדקס הבא על הטבלה:

```
CREATE index on movies(duration)
```

2. מה תהיה דרגת הפיצול האופטימלית של האינדקס?

כל ערך באינדקס תופס 8 בייט, וכל מצביע תופס גם 8 בייט, גודל בלוק הוא 1000 בייט.

לפי הנוסחא: $d \leq \frac{b+v}{v+p} = \frac{1000+8}{8+8} \rightarrow d = 63$

3. מה תהיה עלות חישוב השאילתה באמצעות האינדקס, בהנחה שדרגת הפיצול היא זו שחושבה בסעיף הקודם?

INDEX UNIQUE SCAN

גובה העץ $h = \left\lceil log_{\lceil \frac{d}{2} \rceil}(T(Movies)) \right\rceil = \left\lceil log_{\lceil \frac{63}{2} \rceil}(10{,}000) \right\rceil = 3$.

מספר העלים=1

סה"כ עלות חישוב עם אינקדס $3 + 1 = \mathbf{4}$

ב. נתונה השאילתה הבאה:

```
SELECT avg (duration)
FROM Movies
WHERE duration > 100
```

1. מה עלות חישוב השאילתה בהנחה שאין אינדקסים על הטבלה?

כמו בסעיף א **1,667**.

כעת, נתון האינדקס הבא על הטבלה:

```
CREATE index on movies(duration)
```

2. מה תהיה דרגת הפיצול האופטימלית של האינדקס?

כמו בסעיף א: $d = 63 \leftarrow \mathrm{d} \leq \frac{b+v}{v+p}$

3. מה תהיה עלות חישוב השאילתה באמצעות האינדקס, בהנחה שדרגת הפיצול היא זו שחושבה בסעיף הקודם?

INDEX RANGE SCAN

אותו אינדקס כמו א לכן $d = 63, h = 3$

כמה ערכים עם duration > 100: $\frac{200-100}{200} \times 10{,}000 = 5{,}000$

בכל עלה נכנסים לכל הפחות $\left\lceil \frac{63}{2} \right\rceil - 1 = 31$ ערכים, וסהכ צריך לעבור על $\left\lceil \frac{5{,}000}{31} \right\rceil = 162$ עלים.

סה"כ עלות חישוב עם אינקדס $\underline{3 + 162 = \mathbf{165}}$

ג. נתונה השאילתה הבאה:

```
SELECT name
FROM Movies
WHERE movieid=200
```

1. מה עלות חישוב השאילתה בהנחה שאין אינדקסים על הטבלה?

כמו בסעיף א $\underline{\mathbf{1,667}}$.

כעת, נתון האינדקס הבא על הטבלה:

```
CREATE index on movies(movieid)
```

2. מה תהיה דרגת הפיצול האופטימלית של האינדקס?

לmovieid אותו גודל כמו לduration והחישוב הוא עדיין $d = 63, h = 3$

3. מה תהיה עלות חישוב השאילתה באמצעות האינדקס, בהנחה שדרגת הפיצול היא זו שחושבה בסעיף הקודם?

INDEX UNIQUE SCAN+TABLE ACCESS BY ROWID

Movieid הוא מפתח ולכן יש לגשת רק לעלה אחד, ורק לבלוק אחד בטבלה

סה"כ עלות חישוב עם אינקדס $\underline{3+1+1=\mathbf{5}}$

ד. נתונה השאילתה הבאה:

```
SELECT avg(duration)
FROM Movies
WHERE genre = 'Drama'
```

1. מה עלות חישוב השאילתה בהנחה שאין אינדקסים על הטבלה?

כמו בסעיף א $\underline{\mathbf{1,667}}$.

כעת, נתון האינדקס הבא על הטבלה:

```
create index on movies(genre)
```

2. מה תהיה דרגת הפיצול האופטימלית של האינדקס?

כל ערך באינדקס תופס 10 בייט, וכל מצביע תופס גם 8 בייט, גודל בלוק הוא 1000 בייט.

לפי הנוסחא: $d = 56 \leftarrow \mathrm{d} \leq \frac{b+v}{v+p} = \frac{1000+10}{10+8}$

3. מה תהיה עלות חישוב השאילתה באמצעות האינדקס, בהנחה שדרגת הפיצול היא זו שחושבה בסעיף הקודם?

INDEX RANGE SCAN+TABLE ACCESS BY ROWID

גובה העץ $h=\left\lceil log_{\left\lceil \frac{56}{2} \right\rceil}(10{,}000) \right\rceil = 3$.

מספר הערכים שמתאימים ל genre = 'Drama' הוא $\frac{10{,}000}{4} = 2500$

בכל עלה נכנסים לכל הפחות $\left\lceil \frac{56}{2} \right\rceil - 1 = 27$ ערכים, וסהכ צריך לעבור על $\left\lceil \frac{2{,}500}{27} \right\rceil = 93$ עלים.

צריך לקרוא מהטבלה 2500 ערכים, שנמצאים בלכל היותר 1667 בלוקים, כמספר הבלוקים בטבלה

סה"כ עלות חישוב עם אינקדס $3 + 93 + 1667 = \mathbf{1763}$

ה. נתונה השאילתה הבאה:

```
SELECT avg(duration)
FROM Movies
WHERE genre = 'Drama'
```

1. מה עלות חישוב השאילתה בהנחה שאין אינדקסים על הטבלה?

כמו בסעיף א **1,667**.

כעת, נתון האינדקס הבא על הטבלה:

```
create index on movies(genre,duration)
```

2. מה תהיה דרגת הפיצול האופטימלית של האינדקס?

כל ערך באינדקס תופס $10 + 8 = 18$ בייט, וכל מצביע תופס גם 8 בייט, גודל בלוק הוא 1000 בייט.

לפי הנוסחא: $d \leq \frac{b+v}{v+p} = \frac{1000+18}{18+8} \rightarrow d = 39$

3. מה תהיה עלות חישוב השאילתה באמצעות האינדקס, בהנחה שדרגת הפיצול היא זו שחושבה בסעיף הקודם?

INDEX RANGE SCAN

גובה העץ $h=\left\lceil log_{\left\lceil \frac{39}{2} \right\rceil}(10,000) \right\rceil = 4$.

מספר הערכים שמתאימים ל genre = 'Drama' הוא $\frac{10,000}{4} = 2500$

בכל עלה נכנסים לכל הפחות $\left\lceil \frac{39}{2} \right\rceil - 1 = 19$ ערכים, וסהכ צריך לעבור על $\left\lceil \frac{2,500}{19} \right\rceil = 132$ עלים.

לא צריך לקרוא מהטבלה

סה"כ עלות חישוב עם אינקדס $\underline{4 + 132 = \mathbf{136}}$

בהצלחה!