קורס SQL למתחילים

הרצאה 4 – תרגיל – פתב"ס

אורך ממוצע של שם מקום

ראינו כבר שהטבלה Users מכילה עמודה בשם Location הכוללת את המיקום של המשתמש. נרצה לדעת מה ממוצע אורך השם של המקומות **כאשר אנחנו מתייחסים לכל המקומות הייחודיים**.

מה זאת אומרת? אם נעשה סתם שליפה של SELECT AVG(LEN(Location)) FROM Users נקבל תוצאות מוטות, כי למשל יש הרבה משתמשים שהמיקום שלהם הוא USA, שכולל 3 אותיות – מה שמטה את הממוצע כלפי מטה.

המטרה שלנו לדעת מה ממוצע אורכי השם הייחודיים של המיקומים השונים – בלי קשר לכמה פעמים הם מופיעים בטבלה (מתייחסים לכל המיקומים כאילו הם מופיעים רק פעם אחת).

תשובה:

```
SELECT AVG(LEN(Location))
FROM
(
	SELECT DISTINCT Location
	FROM Users
) distinctLocations
```

בשאילתה הפנימית אנחנו מביאים את שמות המקומות, כאשר כל שם יופיע רק פעם אחת (בזכות ה-DISTINCT). בשאילתה החיצונית, אנחנו לוקחים את ה- LEN של כל אחד מהמיקומים האלה, כלומר אורך ה- string – ומשתמשים בפונקציית AVG שהיא פונקציה אגרגטיבית שמקבלת אוסף של ערכים (במקרה הזה, ה- LENעבור כל אחד מהמיקומים השונים שמופיעים בצורה ייחודית) ומחזירה מספר שהוא הממוצע.

התגים הכי פופולריים

ב- StackOverflow כל שאלה יכולה להיות משוייכת לתגיות, שמתארות את נושא השאלה. השיוך בין שאלה לתגיות מתבצע לפי הטבלה PostsToTags שמכילה שתי עמודות: PostId שהוא ה- Id של השאלה בטבלת Posts ו- TagId שהוא ה- Id של התג בטבלת Tags. בטבלה Tags מופיע כל תג פעם אחת בלבד, כאשר מופיע ה- Id שלו ושמו (TagName).

כתבו שליפה המחזירה את שמות התגים ועבור כל תג – כמה שאלות נשאלו בו.

תשובה:

```
SELECT Tags.TagName,
            NumOfQuestions =     (
                                                SELECT COUNT(PostsToTags.PostId)
                                                FROM PostsToTags
                                                WHERE PostsToTags.TagId = Tags.Id
                                    )
FROM Tags
ORDER BY NumOfQuestions desc
```

השאלות עם הכי הרבה Upvotes

ב- StackOverflow ניתן לעשות Upvote ("לייק") לכל פוסט, בין היתר לשאלות עצמן. הרישום של ה- Upvotes מתבצע בטבלת Votes, כאשר כל שורה בטבלה כוללת עמודות: PostId – ה- Id של השורה בטבלת Posts שאליה התבצעה ההצבעה, ו- VoteTypeId שהוא מס' שמייצג את סוג ההצבעה, כאשר הערך 2 מייצג Upvote. הערך 3 מייצג את הפעולה ההפוכה, שגם אותה אפשר לעשות, שנקראת Downvote.

סעיף א'

שלפו את 100 **השאלות** (להזכירכם – מזהים שאלה לפי PostTypeId=1) שקיבלו הכי הרבה Upvotes. בכל שורה כללו את העמודות הבאות: ה- Id של השאלה (ה- Id שלה בטבלת Posts), כותרת השאלה (Title), עמודה בשם NumUpvotes שכוללת את מס' ה- Upvotes שקיבלה ועמודה בשם NumOfReplies שכוללת כמה פוסטים נכתבו בתגובה לשאלה הזאת (להזכירכם, פוסט שנכתב לתגובה יכיל בשדה של ה- ParentId שלו את ה- Id של השאלה).

תשובה:

```
SELECT TOP(100) Id,
            Title,
            NumUpvotes =  (
                                    SELECT COUNT(Votes.Id)
                                    FROM Votes
                                    WHERE Votes.PostId = Posts.Id AND VoteTypeId = 2
                                ),
            NumOfReplies= (
                                    SELECT COUNT(p2.Id)
                                    FROM Posts p2
                                    WHERE p2.ParentId = Posts.Id
                                )
FROM Posts
WHERE PostTypeId=1
```

```
ORDER BY NumUpvotes DESC
```

הנקודות שאנחנו צריכים לשים לב אליהם זה השימוש בשאילתות הפנימיות ב- SELECT. שימו לב שבשביל השליפה של NumOfReplies אנחנו עושים שליפה פנימית שגם היא מתבצעת מול הטבלה Posts, ולכן אנחנו נותנים שם שונה ל- Posts בטבלה הפנימית (p2).
בנוסף, שימו לב שאין לנו בעייה **למיין** לפי NumOfUpvotes, זאת בגלל שה- ORDER BY מבוצע אחרי ה- SELECT, ולכן ניתן להשתמש בשם NumOfUpvotes ב- ORDER BY. אם היינו רוצים לעשות WHERE על NumOfUpvotes **לא** היינו יכולים באותה השאילתה (בלי לשכפל שוב את השאילתה הפנימית) והיינו צריכים לעטוף את זה בשאילתה חיצונית או ב- CTE.

סעיף ב'

שנו את השליפה בסעיף א', כך שתחזיר את כל השאלות שקיבלו יותר Upvotes מהממוצע בקרב שאלות בעלי יותר מ-5000 צפיות (להזכירכם, בטבלת Posts יש את העמודה ViewCount שמכילה את מס' הצפיות בשאלה). העמודות שצריכות להיות כלולות הן אותן עמודות כמו סעיף א'.

תשובה:

```
with postsAndUpvotes as (
	SELECT Id,
			Title,
			ViewCount,
			NumUpvotes =  (
					SELECT COUNT(Votes.Id)
					FROM Votes
					WHERE Votes.PostId = Posts.Id AND VoteTypeId = 2
				),
			NumOfReplies= (
					SELECT COUNT(p2.Id)
					FROM Posts p2
					WHERE p2.ParentId = Posts.Id
				)
	FROM Posts
	WHERE PostTypeId=1
)

SELECT *
FROM postsAndUpvotes
WHERE NumUpvotes > (SELECT AVG(p2.NumUpvotes) FROM postsAndUpvotes p2 WHERE p2.ViewCount > 5000)
```

שימו לב שלקחנו למעשה את השאילתה הפנימית מקודם, ועשינו לה קצת שינויים: הוספתי את ViewCount (כי אנחנו צריכים אותו) והורדתי את ה-TOP וה-ORDER BY שלא רלוונטיים כבר. את כל השאילתה הזאת עטפתי ב- CTE שקראתי לו postsAndUpvotes. הסיבה היא שאנחנו רוצה לעשות סינון ללפי NumUpvotes, והשם NumUpvotes לא מוכר בתחום של השאילתה הפנימית (כי ה- WHERE מתבצע לפני ה- SELECT) ולכן הפיתרון הנוח הוא להפריד את זה ל- cte שונה.

חשוב מאד לשים לב שבמקרה הזה, לשים את התוכן של השאילתה הפנימית בתוך ה- FROM (במקום ב- CTE נפרד) לא היה מתאים – כי אנחנו עושים ב-WHERE שליפה שמשתמשת בו פעם נוספת ב-CTE (בשביל להביא את הממוצע), מה שלא יכולנו לעשות באותה הצורה אם זה היה "מודבק" בתוך ה-FROM.