קורס SQL למתחילים

הרצאה 7 – תרגיל – פתב"ס

המילים הפופולרית ביותר

כתבו שאילתה המחזירה את רשימת כל המילים השונות, בהן נעשה שימוש לפחות 10 פעמים בכותרות של שאלות שנשאלו בין 01/01/2016 וה-07/01/2016. העמודות שצריכות להיות כלולות בתוצאות זה המילה עצמה, מס' הפעמים בהם נעשה בה שימוש ומס' השאלות השונות בהן נעשה בה שימוש בכותרת השאלה.
הרשימה צריכה להיות ממויינת לפי מס' השימושים, מהמילה הפופולרית ביותר כלפי מטה.
מבחינתנו, התו שמפריד בין מילים הוא רווח.

יש להתעלם מהסימנים ? (סימן שאלה) . (נקודה) ו- , (פסיק). כדי להתעלם, השתמשו בפונקציה REPLACE כדי להחליף אותם ב- string ריק. למשל, SELECT REPLACE('red;green;blue', ';', '') יחזיר redgreenblue (מחליף את הנקודה-פסיק ב-string ריק).

להזכירכם – ניתן לעשות שימוש בפונקציה STRING_SPLIT שראינו בהרצאה.

פיתרון:

```
with relevantTitles as (
SELECT Id, FixedTitle = REPLACE(REPLACE(REPLACE(Title, '.',''),',',''), '?','')
FROM Posts
WHERE PostTypeId=1 AND CreationDate BETWEEN '2016-01-01' AND '2016-01-07'
), tokens as (
SELECT Id, token = strs.value
FROM relevantTitles
OUTER APPLY   (
                        SELECT  value
                        FROM STRING_SPLIT(relevantTitles.FixedTitle, ' ')
                    ) strs
)
SELECT token,
             NumRepeats = count(*),
             NumDistinctQuestion = COUNT(DISTINCT(Id))
FROM tokens
GROUP BY token
HAVING COUNT(*) >= 10
ORDER BY NumRepeats DESC
```

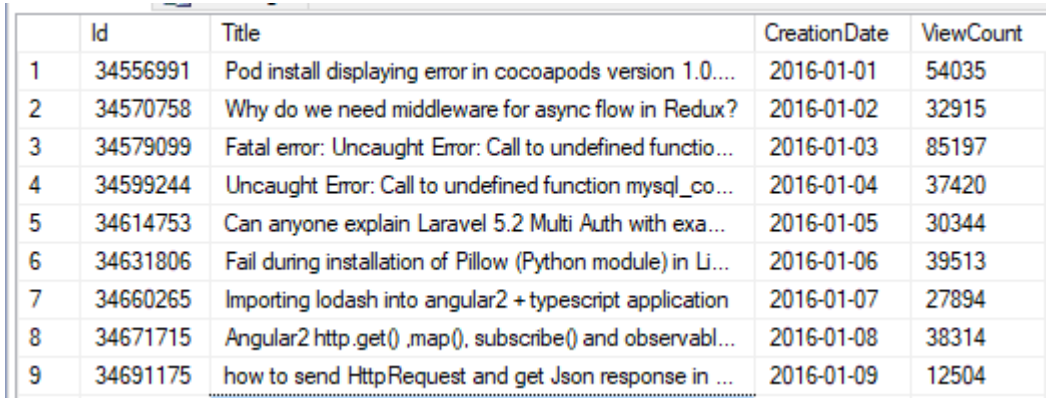
נעקוב אחרי מה שקורה פה: בהתחלה, ה- CTE ששמו relevantTitles כולל את ה- Id והכותרת של השאלות (לאחר שהועפו מהן נקודות, פסיקים וסימני שאלה) שנשאלו בטווח התאריכים המתאים.
ה- CTE ששמו tokens מבצע שליפה מה- CTE הקודם, ומצרף לכל Id את התוצאות של הפרדת המשפט למספר מילים (לפי רווחים) תוך שימוש בפונקציית STRING_SPLIT.
לאחר מכן, בגוף השאילתה, אנחנו עושים GROUP BY על ה- tokens, כפי שראינו בשיעור קודם.

להרצאות, תרגילים ופתרונות ניתן להיכנס לאתר הקורס.

ALL/ANY

שימוש ב- ALL

כתוב שאילתה שמחזירה את ה- Id, הכותרת, תאריך היצירה ומספר הצפיות של השאלה עם מספר הצפיות (ViewCount) הגדול ביותר מבין השאלות שהתפרסמו באותו היום.
בשלב זה של הקורס אתם מסוגלים לכתוב את השאילתה הזאת ביותר מדרך אחת. לטובת התרגול, כתבו אותה תוך שימוש באופרטור ALL.



	Id	Title	CreationDate	ViewCount
1	34556991	Pod install displaying error in cocoapods version 1.0....	2016-01-01	54035
2	34570758	Why do we need middleware for async flow in Redux?	2016-01-02	32915
3	34579099	Fatal error: Uncaught Error: Call to undefined functio...	2016-01-03	85197
4	34599244	Uncaught Error: Call to undefined function mysql_co...	2016-01-04	37420
5	34614753	Can anyone explain Laravel 5.2 Multi Auth with exa...	2016-01-05	30344
6	34631806	Fail during installation of Pillow (Python module) in Li...	2016-01-06	39513
7	34660265	Importing lodash into angular2 + typescript application	2016-01-07	27894
8	34671715	Angular2 http.get() ,map(), subscribe() and observabl...	2016-01-08	38314
9	34691175	how to send HttpRequest and get Json response in ...	2016-01-09	12504

פיתרון:

```
SELECT Id,
            Title,
            CreationDate = CAST(CreationDate as date),
            ViewCount
FROM Posts
WHERE  PostTypeId = 1 AND
            ViewCount >= ALL     (
                                                SELECT p2.ViewCount
                                                FROM Posts p2
                                                WHERE CAST(p2.CreationDate as date) = CAST(Posts.CreationDate AS date)
                                          )
ORDER BY CreationDate
```

שימוש ב-ANY

כתבו שאילתה המחזירה את ה- Id וה- DisplayName של כל המשתמשים שפירסמו לפחות שאלה אחת שזכתה ביותר מ-50,000 צפיות. בשלב זה של הקורס אתם מסוגלים לכתוב את השאילתה הזאת במספר דרכים. לטובת התרגול, כתבו אותה תוך שימוש באופרטור ANY.

פיתרון:

```
SELECT Id, DisplayName
FROM Users
WHERE Id = ANY       (
                                SELECT OwnerUserId
                                FROM Posts
                                WHERE Posts.ViewCount > 50000
                        )
```

השאלות שפופולריות ביותר בכל התגים שלהן

כתבו שאילתה המחזירה את כל השאלות שנצפו הכי הרבה פעמים בכל התגים שמשוייכים לאותה השאלה.

כלומר, אם יש שאלה שמתוייגת בתגים SQL (שהשאלה הכי נצפית בו זכתה לצורך הדוגמא ב-1000 צפיות) ובתג C# (שלצורך הדוגמא, השאלה הכי נצפית בו זכתה ל-15,000 צפיות) – כללו אותה בתוצאות רק אם היא נצפתה לפחות 15,000 פעמים (אם היא נצפתה 2000 פעמים, שזה יותר מהכמות של SQL אבל פחות מהכמות של C# - אל תכללו אותה בתשובות).

שלפו את מזהה השאלה ואת הכותרת שלה, ומיינו בסדר הפוך לפי מספר הצפיות.

פיתרון:

```
with tagsMaximums as (
       SELECT TagId, MaximumViews = MAX(ViewCount)
       FROM PostsToTags
       JOIN Posts ON Posts.Id = PostsToTags.PostId AND Posts.PostTypeId=1
       GROUP BY TagId
)
SELECT Id, Title, ViewCount
FROM Posts
WHERE  PostTypeId = 1 AND
             ViewCount >= ALL     (
                                                SELECT MaximumViews
                                                FROM tagsMaximums
                                                WHERE tagsMaximums.TagId IN (SELECT TagId FROM PostsToTags pt WHERE pt.PostId = Posts.Id)
                                        )

ORDER BY ViewCount DESC
```

בשלב הראשון, אנחנו יוצרים לנו CTE שמכיל עבור כל מזהה של תג, את מספר הצפיות המקסימלי של שאלה באותו התג.
לאחר מכן, אנחנו מסתכלים על כל השאלות בטבלת Posts, ומביאים את אלה שה- ViewCount שלהם גדול מכל הערכים שחזרו בשאילתה הפנימית. ומה בדיוק אנחנו שמים בשאילתה הפנימית? את ה- MaximumViews שאנחנו מביאים מה- CTE שעשינו קודם, עבור כל תג אליו משוייכת השאלה שהיא השורה הנוכחית בטבלת Posts.

שימוש ב- INTERSECT ו-EXCEPT

השתמשו ב- INTERSECT ו-EXCEPT כדי להביא את ה- Id וה- DisplayName של כל המשתמשים שכתבו גם שאלה בנושא #c (כלומר שהיא תחת התג הזה) וגם שאלה בנושא sql (כלומר שהיא מתוייגת תחת sql) אולם לא כתבו אף תגובה ב- StackOverflow (בטבלה Comments).

השליפה הזאת לא אמורה לכלול משתמשים שכתבו שאלה בנושא c# אבל לא כתבו שאלה בנושא sql או להיפך.

למעשה, אמורות לחזור מעט מאד שורות (270 אם מסתכלים על הדטאבייס של שנת 2016).

פיתרון:

```
(
SELECT Users.Id, Users.DisplayName
FROM Posts
JOIN Users ON Users.Id = Posts.OwnerUserId
JOIN PostsToTags ON PostsToTags.PostId = Posts.Id
JOIN Tags ON Tags.Id = PostsToTags.TagId
WHERE Tags.TagName='c#'
INTERSECT
SELECT Users.Id, Users.DisplayName
FROM Posts
JOIN Users ON Users.Id = Posts.OwnerUserId
JOIN PostsToTags ON PostsToTags.PostId = Posts.Id
JOIN Tags ON Tags.Id = PostsToTags.TagId
WHERE Tags.TagName='sql'
)
EXCEPT
SELECT Users.Id, Users.DisplayName
FROM Comments
JOIN Users ON Users.Id = Comments.UserId
```