

קורס SQL למתחילים

הרצאה 10 – תרגיל

גיבוי דטאבייסים

פקודה לגיבוי מסד נתונים, למשל בשם Sample, נראית כך:

```
BACKUP DATABASE [Sample] TO DISK = 'c:\tmp\baksSample.bak'
```

[בהרצאה](#) ראינו כיצד ניתן להרכיב רצף משפטי SQL, שמגבה את כל הדטאבייסים של המשתמש באמצעות שימוש ב-SELECT יחיד, ולאחר מכן להריץ אותו באמצעות EXECUTE.

כדי לקבל את שמות כל מסדי הנתונים של המשתמש, השתמשנו בשאילתה הבאה:

```
select name  
from sys.databases  
where database_id>5
```

כתבו קוד שפותח Cursor מול תוצאות השאילתה הזאת, ואז עבור כל שורה (כל שם של DB) מייצר את ה-string של פקודת הגיבוי (ב-syntax שמופיע למעלה) ומריץ אותה באמצעות EXECUTE. לנחיותכם, מצורף בנספח הטקסט של דמו השימוש ב-Cursor שהוצג בהרצאה.

יחס של טוב

אנחנו מגדירים יחס "X ענה ל-Y" בין שני משתמשים X ו-Y, אם X ענה לשאלה כלשהי שנשאלה ע"י Y.

אנחנו יכולים לדמיין גרף מכוון שנוצר ע"י היחס הזה: אם אני נקודת ההתחלה שלו, אז יש לי צלע אל כל האנשים שעניתי לשאלה שלהם. לכל אחד מהם, יש צלע לכל האנשים שהוא ענה לשלה שלהם וכן הלאה.

כתבו שאילתה שבהינתן Id של משתמש מקור ופרמטר "עומק" שולפת את כל המשתמשים שאפשר להגיע אליהם בגרף ממשתמש המקור שניתן עד לעומק שהוגדר.

למשל, אם שחר ענה לגל ולאבישי, גל ענתה לאיציק ואיציק ענה לענת ולשחר, אז התוצאה תראה כך כאשר מתחילים משחר ומוגדר עומק של עד 10:

עומק	משתמש
0	שחר
1	גל
1	אבישי
2	איציק
3	ענת

שימו לב ששחר לא מופיע בעומק 3 (למרות שאפשר להגיע אליו בעומק 3 במסלול שחר-גל-איציק-שחר) כי אנחנו רוצים למנוע הופעת כאלה שכבר הגענו אליהם קודם בטיול בגרף, כדי להימנע ממעגלים (שיגררו למעשה שליפה אינסופית במקרה שלנו).

עשו את השאילתה הזאת מול הטבלה Posts, כאשר פוסט X הוא מענה לפוסט Y אם ה-ParentId שלו הוא ה-Id של Y. שימו לב, שההיררכיה בתשובות היא רק רמה אחת: לשאלה יש מספר תשובות, אבל אין תשובות לתשובות.

התחילו את השאילתה ב-

```
DECLARE @sourceUserId int = 285462  
DECLARE @maxDepth int = 4
```

רמז 1: הפיתרון שלכם צריך למעשה לבנות שאילתה באופן דינמי (בלולאה) ולהריץ אותה. לא נדרש שימוש ב-CURSOR-ים במקרה הזה.

רמז 2: כדי לאגד את התוצאות מכל הרמות, אתם תרצו להגדיר משתנה טבלה. יש להגדיר אותו **בתוך** ה-SQL שנוצר דינמית, כי לא ניתן לגשת מתוך SQL שמורץ עם EXECUTE למשתנה טבלה שמוגדר בחוץ (ניתן לטבלאות זמניות, אבל לא דיברנו עליהם, ואין סיבה שתצטרכו להשתמש בכך בתרגיל הנ"ל).

© כל הזכויות שמורות לשחר גבירץ.

להרצאות, תרגילים ופתרונות ניתן להיכנס [לאתר הקורס](#).

הערה (לידע כללי): ניתן לפתור את זה באמצעות משהו שנקרא RECURSIVE CTE. אנחנו לא דיברנו על זה בקורס, ואין סיבה שתעשו בזה שימוש (המטרה של התרגיל היא להשתמש ב-SQL דינאמי).

נספח – דוגמא לשימוש ב-Cursor

המטרה: להחזיר result-set שכולל שורה עבור כל משתמש שפירסם שאלה עם ה-Id שלו, שמו, וכותרות השאלות שפירסם מופרדות ב-|~|

השיטה:

1. הגדרת משתנה טבלאי שיכיל את התוצאה
 2. הגדרת Cursor מול שאילתה שמביאה את ה-Id וה-DisplayName של כל המשתמשים שפירסמו שאלה
 3. פתיחת ה-Cursor
 4. הגדרת משתנים שמכילים את הערכים של שורה בודדת
 5. הבאת השורה הראשונה
 6. בלולאה – כל עוד הצלחנו להביא מידע
1. ביצוע הפעולה שאנחנו רוצים ויצירת ה-string שאנחנו רוצים (שרשור הכותרות)
 2. הכנסת התוצאה למשתנה של התוצאות
 3. הבאת השורה הבאה (קידום הלולאה)
 7. סגירת ה-Cursor ושחרורו
 8. שליפה של התוצאות

```
--1
DECLARE @results table (UserId int, DisplayName nvarchar(200), SpecialString nvarchar(max))

--2
DECLARE usersCursor CURSOR FOR
SELECT Id, DisplayName
FROM Users
WHERE Users.CreationDate BETWEEN '2016-01-01' AND '2016-02-01' AND
      EXISTS (SELECT NULL FROM Posts WHERE PostTypeId = 1 AND Posts.OwnerUserId = Users.Id)

--3
OPEN usersCursor

--4
DECLARE @currentUserId int, @currentUserDisplayName nvarchar(200)

--5
FETCH NEXT FROM usersCursor INTO @currentUserId, @currentUserDisplayName

--6
WHILE @@FETCH_STATUS=0 --6
BEGIN
  --6.1
  DECLARE @str nvarchar(max) = ''

  SELECT @str = @str + '|~|' + Title
  FROM Posts
  WHERE PostTypeId=1 AND OwnerUserId = @currentUserId
```

```
--6.2  
INSERT INTO @results(UserId, DisplayName, SpecialString)  
VALUES (@currentUserId, @currentUserDisplayName, @str)
```

```
--6.3  
FETCH NEXT FROM usersCursor INTO @currentUserId, @currentUserDisplayName
```

```
END
```

```
--7  
CLOSE usersCursor  
DEALLOCATE usersCursor
```

```
--8  
SELECT *  
FROM @results
```