SQL

- יצירת יחסים והכנסת רשומות חדשות
- שליפת נתונים מתוך יחס אחד
- ⇐ שליפת נתונים מתוך מספר יחסים
- פעולת הקבצה ופונקציות הקבצה
- פונקציות שימושיות
- טריגרים ופונקציות בפורמט PostgreSQL
- תרגול

שליפת נתונים מתוך מספר יחסים

כאשר אנו רוצים לשלוף נתונים ממספר יחסים שונים, עלינו לחבר בין היחסים לפני ביצוע השאילתה. לצורך כך נלמד על שתי אפשרויות:

מכפלה קרטזית, וצירוף טבעי

התחביר של מכפלה קרטזית הוא פסיק "," בין שני היחסים בשורת ה-from, לדוגמה:

```
SELECT dname, dfamily
FROM doctor, treatment
```

התחביר של צירוף טבעי הוא פשוט "natural join" בין שני היחסים בשורת ה-from, לדוגמה:

```
SELECT dname, dfamily
FROM doctors NATURAL JOIN treatment
```

אז מה ההבדל בין שניהם?

מכפלה קרטזית מחברת בין **כל** שורה הקיימת ביחס אחד עם **כל** שורה הקיימת ביחס השני ויחס התוצאה כולל בתוכו את כל התכונות שיש בשני היחסים יחד. במצב זה, בכדי לגשת לתכונה כלשהי שהשם שלה זהה בשני היחסים, נהיה חייבים לכתוב מאיזה יחס אנו מעוניינים להוציא את התכונה! לדוגמה: doctor.did

צירוף טבעי לעומת זאת מחבר אך ורק בין שורות שהערכים שנמצאים בתכונות בעלות אותו השם שווים זה לזה. שימו לב: הצירוף מתייחס אך ורק לשם של התכונה! לא למשמעות שלה! אם נצרף שני יחסים שבשניהם יש תכונה עם השם did, השורות שנקבל ביחס התוצאה הן רק השורות בהן הערכים של התכונה did זהים בשני היחסים. במקרה זה, בניגוד למכפלה קרטזית, התכונה בעלת השם הזהה מופיעה רק פעם אחת ובכדי לשלוף תכונה מסוימת מיחס התוצאה, אין צורך לרשום מאיזה יחס התכונה הזו מגיעה.
אם יש מספר תכונות בעלות שם זהה, הצירוף יהיה רק לפי שורות שבהן **כל** הערכים של התכונות הללו זהים.

באופן כללי, אם אפשר, תמיד עדיף להשתמש בצירוף טבעי מאשר במכפלה קרטזית, גם כי זה חוסך מקום בזיכרון, וגם כי נצטרך לכתוב הרבה פחות קוד.

שליפת נתונים מתוך מספר יחסים

מכפלה קרטזית של שני יחסים

tdate	clinicid	pid	did
17/1/2019	1	123	111
18/1/2019	2	456	222
22/1/2019	5	789	111
23/1/2019	7	543	222
25/1/2019	3	753	555
28/1/2019	2	456	222

×

ddatestart	dspeciality	dhouse	dstreet	dcity	dphone	dfamily	dname	did
1/8/1948	משפחה	1	האגס	ירושלים	0255555	לוי	יוסי	111
4/8/2011	נשים	5	בוגרשוב	תל אביב	0344444	כהן	משה	222
5/4/2005	נשים	4	רוטשילד	תל אביב	0344885	לוי	רבקה	555

=

treatment.				doctor.								
tdate	clinicid	pid	did	ddatestart	dspeciality	dhouse	dstreet	dcity	dphone	dfamily	dname	did
17/1/2019	1	123	111	1/8/1948	משפחה	1	האגס	ירושלים	0255555	לוי	יוסי	111
18/1/2019	2	456	222	1/8/1948	משפחה	1	האגס	ירושלים	0255555	לוי	יוסי	111
22/1/2019	5	789	111	1/8/1948	משפחה	1	האגס	ירושלים	0255555	לוי	יוסי	111
23/1/2019	7	543	222	1/8/1948	משפחה	1	האגס	ירושלים	0255555	לוי	יוסי	111
25/1/2019	3	753	555	1/8/1948	משפחה	1	האגס	ירושלים	0255555	לוי	יוסי	111
28/1/2019	2	456	222	1/8/1948	משפחה	1	האגס	ירושלים	0255555	לוי	יוסי	111
17/1/2019	1	123	111	4/8/2011	נשים	5	בוגרשוב	תל אביב	0344444	כהן	משה	222
18/1/2019	2	456	222	4/8/2011	נשים	5	בוגרשוב	תל אביב	0344444	כהן	משה	222
22/1/2019	5	789	111	4/8/2011	נשים	5	בוגרשוב	תל אביב	0344444	כהן	משה	222
23/1/2019	7	543	222	4/8/2011	נשים	5	בוגרשוב	תל אביב	0344444	כהן	משה	222
25/1/2019	3	753	555	4/8/2011	נשים	5	בוגרשוב	תל אביב	0344444	כהן	משה	222
28/1/2019	2	456	222	4/8/2011	נשים	5	בוגרשוב	תל אביב	0344444	כהן	משה	222
17/1/2019	1	123	111	5/4/2005	נשים	4	רוטשילד	תל אביב	0344885	לוי	רבקה	555
18/1/2019	2	456	222	5/4/2005	נשים	4	רוטשילד	תל אביב	0344885	לוי	רבקה	555
22/1/2019	5	789	111	5/4/2005	נשים	4	רוטשילד	תל אביב	0344885	לוי	רבקה	555
23/1/2019	7	543	222	5/4/2005	נשים	4	רוטשילד	תל אביב	0344885	לוי	רבקה	555
25/1/2019	3	753	555	5/4/2005	נשים	4	רוטשילד	תל אביב	0344885	לוי	רבקה	555
28/1/2019	2	456	222	5/4/2005	נשים	4	רוטשילד	תל אביב	0344885	לוי	רבקה	555

שליפת נתונים מתוך מספר יחסים

צירוף טבעי של שני יחסים

tdate	clinicid	pid	did
17/1/2019	1	123	111
18/1/2019	2	456	222
22/1/2019	5	789	111
23/1/2019	7	543	222
25/1/2019	3	753	555
28/1/2019	2	456	222

$\bowtie$

ddatestart	dspeciality	dhouse	dstreet	dcity	dphone	dfamily	dname	did
1/8/1948	משפחה	1	האגס	ירושלים	0255555	לוי	יוסי	111
4/8/2011	נשים	5	בוגרשוב	תל אביב	0344444	כהן	משה	222
5/4/2005	נשים	4	רוטשילד	תל אביב	0344885	לוי	רבקה	555

=

tdate	clinicid	pid	did	ddatestart	dspeciality	dhouse	dstreet	dcity	dphone	dfamily	dname
17/1/2019	1	123	111	1/8/1948	משפחה	1	האגס	ירושלים	0255555	לוי	יוסי
22/1/2019	5	789	111	1/8/1948	משפחה	1	האגס	ירושלים	0255555	לוי	יוסי
18/1/2019	2	456	222	4/8/2011	נשים	5	בוגרשוב	תל אביב	0344444	כהן	משה
23/1/2019	7	543	222	4/8/2011	נשים	5	בוגרשוב	תל אביב	0344444	כהן	משה
28/1/2019	2	456	222	4/8/2011	נשים	5	בוגרשוב	תל אביב	0344444	כהן	משה
25/1/2019	3	753	555	5/4/2005	נשים	4	רוטשילד	תל אביב	0344885	לוי	רבקה

שליפת נתונים מתוך מספר יחסים

תיאורטית, תמיד אפשר לעשות את פעולת הצירוף הטבעי באמצעות מכפלה קרטזית (שוב, אם אפשר אחרת זה עדיף, אבל אם שמות התכונות שלפיהן רוצים לצרף לא זהים, נשתמש במכפלה קרטזית או בפטנט בהמשך העמוד).

לדוגמה, אם נתונים היחסים הבאים:

doctor (<u>did</u>, dname, dfamily, dphone, dcity, dstreet, dhouse, dspeciality, ddatestart)
treatment (<u>did, pid, clinicid</u>, tdate)

השאילתות הבאות זהות בתוצאתן:

```
SELECT dname, dfamily
FROM doctor NATURAL JOIN treatment
```

=

```
SELECT dname, dfamily
FROM doctor, treatment
WHERE doctor.did = treatment.did
```

אם יש שתי תכונות עם אותה משמעות בדיוק, אבל השם שלהן שונה בשני היחסים שאנו רוצים לצרף, לדוגמה, אם תעודה הזהות של רופא הייתה נקראת id ביחס "רופאים" והיתה נקראת did ביחס "טיפולים" והיינו רוצים לצרף בין היחסים לפי תכונה זו, היינו יכולים לעשות זאת כך:

```
SELECT dname, dfamily
FROM doctor JOIN treatment ON (id=did)
```

שליפת נתונים מתוך מספר יחסים

אם שמות היחסים ארוכים או מורכבים, נוכל לתת להם שמות חדשים רק עבור השאילתה עצמה באמצעות as באופן הבא:

```
SELECT dname, dfamily
FROM doctor AS d, treatment AS t
WHERE d.did = t.did
```

השימוש החשוב באמת ב-as הוא כאשר מבצעים מכפלה קרטזית של יחס עם עצמו.

מה הכוונה?

אם נרצה לדוגמה למצוא את כל המטופלים שהיו לפחות בשני טיפולים שונים אצל רופאים שונים:

```
SELECT t1.pid
FROM treatment AS t1, treatment AS t2
WHERE t1.pid = t2.pid AND t1.did != t2.did
```

מצאו את כל המטופלים שהיו לפחות בשני טיפולים שונים במרפאות שונות:

```
SELECT
FROM
WHERE
```

שליפת נתונים מתוך מספר יחסים

חיבור נוסף (פחות שימושי) בין שני יחסים הוא שימוש באחד משלושת האופרטורים:

1. איחוד (union) - מחבר את הרשומות של שני היחסים יחד,
2. חיתוך (intersect) - מחזיר את הרשומות שנמצאות בשני היחסים,
3. חיסור (except) - מחזיר את כל השורות של היחס הראשון שלא נמצאות ביחס השני.

בכדי שתתקבל תוצאה, שני היחסים שעליהם אנו מבצעים את הפעולה חייבים להיות:

1. עם אותה כמות עמודות,
2. עם אותו סוג מידע בכל עמודה,
3. העמודות צריכות להיות באותו הסדר

זאת אומרת שאם ננסה לאחד בין שני יחסים עם אותה כמות עמודות, ועם אותו סוג מידע בעמודות השונות אבל הסדר של העמודות יהיה שונה, לדוגמה ביחס אחד העמודה הראשונה תכיל מחרוזות וביחס השני מספרים שלמים, הפעולה לא מוגדרת.

דבר נוסף שחשוב לשים אליו לב: המהדר לא יודע אם הפעולות הגיוניות או לא. הוא רק יכול לבדוק אם טיפוסי הנתונים מתאימים. אם תעודת זהות מוגדרת כמספר שלם (int) וגם שכר חודשי מוגדר כמספר שלם וננסה לחבר ביניהם המהדר לא "יתלונן" ויאפשר לנו לבצע את הפעולה. ולכן האחריות לעשות רק דברים הגיוניים מוטלת על כתפינו.

שליפת נתונים מתוך מספר יחסים

לדוגמה, השאילתה הבאה מחזירה את תעודות הזהות של כל הרופאים שגרים ברעננה ואת תעודות הזהות של כל הרופאים שגרים בכפר סבא:

```
SELECT did
FROM doctor
WHERE dcity = 'raanana'

UNION

SELECT did
FROM doctor
WHERE dcity = 'kfar saba'
```

כמובן שהשאילתה שקולה לשאילתה הבאה:

```
SELECT did
FROM doctor
WHERE dcity = 'raanana' OR dcity = 'kfar saba'
```

doctor (did, dname, dfamily, dphone, dcity, dstreet, dhouse, dspeciality, ddatestart, dsalary)
patient (pid, pname, pfamily, pbirthdate)
clinic (cid, ccity, caddress, cname)
treatment (tdate, did, pid, cid)

שליפת נתונים מתוך מספר יחסים

לדוגמה, השאילתה הבאה מחזירה את כל הפרטים של כל הרופאים שגרים ברעננה ומתמחים ברפואת ילדים:

```
SELECT *
FROM doctor
WHERE dcity = 'raanana'

INTERSECT

SELECT *
FROM doctor
WHERE dspeciality = 'children'
```

כמובן שהשאילתה שקולה לשאילתה הבאה:

```
SELECT *
FROM doctor
WHERE dcity = 'raanana' AND dspeciality = 'children'
```

doctor (did, dname, dfamily, dphone, dcity, dstreet, dhouse, dspeciality, ddatestart, dsalary)
patient (pid, pname, pfamily, pbirthdate)
clinic (cid, ccity, caddress, cname)
treatment (tdate, did, pid, cid)

שליפת נתונים מתוך מספר יחסים

לדוגמה, השאילתה הבאה מחזירה את תעודות הזהות של כל הרופאים שלא ביצעו לעולם שום טיפול:

```
SELECT did
FROM doctor

EXCEPT

SELECT did
FROM treatment
```

כמובן שהשאילתה שקולה לשאילתה הבאה:

```
SELECT *
FROM doctor
WHERE did NOT IN (SELECT did
                  FROM treatment)
```

doctor (did, dname, dfamily, dphone, dcity, dstreet, dhouse, dspeciality, ddatestart, dsalary)
patient (pid, pname, pfamily, pbirthdate)
clinic (cid, ccity, caddress, cname)
treatment (tdate, did, pid, cid)

יש לשים לב כי שלושת הפעולות: איחוד (union), חיתוך (intersect) וחיסור (except), מצמצמות כפילויות, זה אומרת שאם לדוגמה תעודת הזהות 333 מופיעה יותר מפעם אחת באיחוד או בחיתוך או בחיסור, כברירת מחדל היא תופיע רק פעם אחת.

בכדי למנוע מצב של צמצום כפילויות יש לרשום: union all או intersect all או except all

שליפת נתונים מתוך מספר יחסים

אפשרות נוספת לשליפת נתונים מתוך מספר יחסים היא שימוש בשאילתות מקוננות. המשמעות של שאילתות מקוננות היא למעשה כתיבת שאילתה בתוך שאילתה. ישנם 6 אופרטורים שימושיים עבור יצירת שאילתות מקוננות:

1. מכל (all) - מאפשר לבחור שורות אשר אחת מהתכונות בהן גדולה / קטנה / שווה / שונה מכל הערכים בתכונות שבשאילתה שבאה אחרי,
2. מאיזשהו (any) - מאפשר לבחור שורות אשר אחת מהתכונות בהן גדולה / קטנה / שווה / שונה מאיזשהו ערך בתכונות שבשאילתה שבאה אחרי,
3. שייך (in) - מאפשר לבחור שורות אשר הערך בתכונה כלשהי שלהן נמצא ביחס היוצא מהשאילתה שבאה אחרי,
4. לא שייך (not in) - מאפשר לבחור שורות אשר הערך בתכונה כלשהי שלהן לא נמצא ביחס היוצא מהשאילתה שבאה אחרי,
5. קיים (exists) - האופרטור מחזיר "אמת" אם היחס היוצא מהשאילתה שבאה אחריו מכילה לפחות שורה אחת, ומחזיר "שקר" אם היחס ריק,
6. לא קיים (not exists) - האופרטור מחזיר "שקר" אם היחס היוצא מהשאילתה שבאה אחריו מכילה לפחות שורה אחת, ומחזיר "אמת" אם היחס ריק.

בהמשך נראה כיצד משתמשים באופרטורים אלו (שימושי בעיקר בשימוש עם פעולת הקבצה).