

## הרצאה מס' 9

נושאים לשיעור זה:

▪ תתי שאילתות

סיכום הפקודות עד לשיעור זה

```
SELECT      id, count(Y)
FROM        Students AS S
            LEFT OUTER JOIN Course AS C
ON          S.studentID = C.studentID
WHERE       (id=3) AND (name is not NULL)
GROUP BY   id
HAVING      count(Y) > 1
ORDER BY   id, name
```

## תתי שאילתות - Sub Queries

בהרצאות ובתרגולים האחרונים התחלנו לגעת בנושא הנקרא תתי-שאילתות.

**לדוגמא:** בהרצאה בה למדנו על פקודות ה - INSERT השונות, כאשר באנו להוסיף שורה חדש בפקודת INSERT ראינו שאפשר לכתוב את הפעולה הבאה:

```
INSERT INTO EmpSalary  
VALUES ( 'Average salary', ( SELECT AVG( Employee_Salary )  
FROM Employees) );
```

← השתמשנו כאן בתת-שאילתא שאת תוצאתה הכנסנו לתוך פקודת INSERT

### תת שאילתא:

תת שאילתא הינה שאילתא רגילה הנמצאת בתוך שאילתא אחרת, כך שתוצאת השאילתא הפנימית (טבלה) משמשת להערכת/חישוב השאילתא החיצונית (נוכל גם לראות זאת בדוגמא הקודמת):

```
שאילתא חיצונית  
SELECT ID, NAME  
FROM ( שאילתא פנימית )
```

**דוגמא נוספת:** נניח שקיימות במכללה 2 טבלאות עובדים: טבלת עובדים קבועים (Workers1) וטבלת עובדים זמניים (Workers2). מנכ"ל המכללה ביקש לקבל את מספר העובדים הקבועים ולהוסיף להם את מספר העובדים הזמניים.

Workers1		
ID1	Name1	Dep1
11	Dan	12
12	Ran	23
13	Moshe	23
14	Roni	14
15	Omer	21

Workers2		
ID2	Name2	Dep2
71	Ilan	21
72	Gili	14
73	Gilad	23

SELECT COUNT(ID1) + ( SELECT COUNT(ID2)  
FROM Workers2 )

3

FROM Workers1

SELECT COUNT(ID1) + 3

FROM Workers1

← השתמשנו כאן בשאילתת בסיס שבה ספרנו את מספר העובדים הקבועים (5 עובדים) ובנוסף בפקודת ה SELECT הוספנו עמודה נוספת שהיא בעצם שדה המחושב בתת-שאילתא המחשב את התוצאה של שלילת כמות העובדים הזמניים (3 עובדים) ואת תוצאה זו הכנסנו לתוך פקודת ה SELECT המקורית.

כעת נלמד בצורה מורחבת על השימוש בתתי שאילתות, היכן ניתן למקם אותן ומהן והפקודות החדשות המתווספות לנו לאור השימוש בתתי השאילתות.

על מנת להסביר את נושא תתי השאלות, נגדיר את ה DB שעליו נבצע את הדוגמאות עבור הפעולות לשיעור זה:

Students					
ChnNum	Name	Street	Number	City	Amount
1	Avi Levy	Begin St.	21	Ramat-Gan	1200
2	Avi Cohen	Begin St.	4	Ramat-Gan	3600
3	Ben Levi	Hayarkon St.	147	Tel-Aviv	4000
4	Chen Levin	Herzrl St.	71	Tel-Aviv	2000
5	Debi Dvir	Hashalom St.	93	Givatiim	700
6	Zvi Kaner	Vaitzman St.	17	Givatiim	3500
7	Haim Itzhak	Haela St.	65	Jerusalem	1000

AgudaMembers		
Name	Status	HasAjob
Avi Cohen	Student	1
Ben Levi	Graduated	1
Chen Levin	Student	1
Zvi Kaner	Graduated	1

## סוגי תתי שאלות:

הראנו בעמוד הקודם איך ניתן לשלב תת-שאלתא בפקודת SELECT, בהמשך השיעור אנו נעסוק ב-2 הסוגים העיקריים של תתי שאלות שניתן לשלב במהלך כתיבת שאלות.

(1) תתי שאלות במשפטי WHERE ו HAVING שבתוקן יש 2 תתי-סוגים:

(א) תתי שאלות המחזירות ערך יחיד

(ב) תתי שאלות המחזירות אוסף ערכים

(2) תתי שאלות בפסוק FROM

חשוב לציין שניתן לשלב תתי שאלות במקומות נוספים ובעצם כמעט בכל מקום בשאלתא נתונה (כמו בדוגמא ששילבנו בפקודת ה SELECT).

## 1) תתי שאילתות במשפטי HAVING ו WHERE:

### א) תתי שאילתות המחזירות ערך יחיד

דוגמא 1: רשימת מספרי החשבונות ושמות הסטודנטים שסכום היתרה בהם גבוה מסכום היתרה הממוצעת של כל חשבונות הסטודנטים:

← נוכל לראות שקיימות כאן בעצם שתי שאלות שניתן לענות עליהם בנפרד:

**שלב א':** היתרה הממוצעת של כל חשבונות הסטודנטים

```
SELECT AVG(amount) AS Average  
FROM Students
```

Average
2268

**שלב ב':** רשימת מספרי החשבונות ושמות הסטודנטים שסכום היתרה בהם גבוה מסכום היתרה הממוצעת

```
SELECT ChnNum, Name  
FROM Students  
WHERE Amount > 2268
```

כעת, לאור העובדה שאנו לא רוצים להשתמש במשתנים ובזיכרון בכדי לאחסן את המידע המגיע מתוצאת הממוצע ולפעמים איננו רוצים בכלל לדעת את הערכים שהתקבלו ולכן ניקח את 2 השאלות שבנינו ונשלב אותן לשאלתא אחת:

← שילוב 2 שאלות הבסיס לשימוש בתת-שאלתא:

```
SELECT ChnNum, Name
FROM students
WHERE Amount > (SELECT AVG(amount)
FROM students);
```

AVG(Amount) = 2286

ChnNum	Name
2	Avi Cohen
3	Ben Levi
6	Zvi Kaner

## דוגמא 2:

נרצה למצוא את רשימת הסטודנטים שגרים בעיר של Ben Levi (כולל Ben עצמו) ?  
כזכור לכם השתמשנו בדוגמא זו באלגברת יחסים כאשר דברנו על פקודת הכינוי כאשר נרצה לבצע מכפלה קרטזית של טבלה בעצמה.

שלב א': העיר של Ben Levi

```
SELECT city
FROM students
WHERE name = 'Ben Levi'
```

city
Tel-Aviv

שלב ב': רשימת שמות הסטודנטים

```
SELECT Name AS StudentName
FROM students
```

← שילוב 2 שאילתות הבסיס לשימוש בתת-שאילתא:

```
SELECT Name AS StudentName
FROM students
WHERE City = (
  SELECT city
  FROM students
  WHERE name = 'Ben Levi' );
```

Name
Ben Levi
Chen Levin

**הערה:** אם תת השאילתא לא הייתה מחזירה ערך יחיד, אלא אוסף ערכים הייתה מתקבלת הודעת שגיאה כי לא היה ניתן לבצע את פעולת ההשוואה.

## ב) תתי שאילתות המחזירות אוסף ערכים

תתי שאילתות מסוג זה הן מסובכות יותר כיוון שמכילות אופרטורים המאפשרים לבצע השוואות על אוסף ערכים ולא על ערך בודד ולשם כך נלמד 4 אופרטורים חדשים:

- SOME()
- ALL()
- IN()
- EXISTS()

בעיקרון השימוש באופרטורים אלו מתבצע רק בעבודה מול תתי-שאילתות.

- SOME()**: המילה **SOME()** צריכה לבוא לאחר אופרטור השוואה (סימן =) ותבדוק האם קיים **לפחות ערך אחד** מתוך רשימת מאיברי הקבוצה שנמצאים בסוגריים:

דוגמא: שמות הסטודנטים שבחשבונם סכום השוואה | לסכום כלשהוא הקיים בעיר תל-אביב:

**SELECT** Name

**FROM** students

**WHERE** Amount = SOME( **SELECT** Amount

**FROM** students

**WHERE** City = 'Tel-Aviv' );

4000, 2000

בדוגמא זו רק לסטודנטים  
בתי"א יש סכום השוואה לסכום  
שהם עצמם מחזיקים.

Name
Ben Levi
Chen Levin



- **ALL()** – יופיע גם כן לאחר אופרטור השוואה, מטרתו לבדוק האם מתקיים התנאי **לכל** איברי הקבוצה הנמצאים בסוגריים.

**דוגמא:** מצא את שם העיר בה ממוצע היתרות גדול שווה מכל ממוצעי היתרות של הערים בארץ:

```
SELECT City
FROM Students
GROUP BY City
HAVING Avg(Amount) >= ALL(
    SELECT Avg(Amount)
    FROM students
    GROUP BY City );
```

2400
3000
2100
1000

בעיר ת"א יש את ממוצע היתרות הגבוה ביותר יחסית לשאר הערים.

City
Tel-Aviv

- **IN()** – בודק השתייכות לקבוצת ערכים (באופרטור זה משתמשים גם בעבודה שוטפת עם רשימת ערכים ולא בהכרח רק בעבודה עם תתי-שאליות):

דוגמא: הצג את שמות האנשים הגרים באחת מהערים הבאות:

```
SELECT Name
FROM Students
WHERE City IN ('Eilat', 'Haifa', 'Jerusalem');
```

מר חיים יצחק גר בירושלים שהיא אחת הערים המופיעות ברשימה.

Name
Haim Itzhak

## הערות לשימוש ב IN():

- 1) שימוש ב IN() שקול לשימוש ב =SOME()
- 2) על אותו משקל ניתן גם להשתמש בפקודה NOT IN() שהיא בעצם שקולה לשימוש ב <>ALL()
- 3) ניתן להשתמש ב IN() ע"י ביצוע תתי-שאלית בתוך הפסקית IN(SELECT... שתחזיר אוסף של ערכים או ע"י קבוצת נתונות מראש של אוסף קבועים, כמו בדוגמא שלהלן

- **EXISTS()** – בודק האם קיימות שורות בתוצאת תת השאילתא ואם כן מתחשב בהן בחישוב השאילתא החיצונית.

דוגמא: מה מחזירה השאילתא הבאה:

```
SELECT SUM(Amount)
FROM Students
WHERE EXISTS (
  SELECT *
  FROM AgudaMembers AS AM
  WHERE AM.Name = Students.Name );
```

SUM(Amount)
13,100

בחלק מהתוכנות תנאי **EXISTS** בודק האם תת השאילתא מתקיימת, ז"א האם מוצאת לפחות שורה אחת, בהנחה וכן מוחזר הערך TRUE ואז השאילתא הראשית רצה במלואה, במצב זה ערך התוצאה היה 16,000 (כי יש לפחות סטודנט אחד שהוא חבר אגודה ולכן תתבצע סכימה של כל סכומי הסטודנטים).

**חשוב:** התשובה לתנאי זה נראית בהתחלה מבלבלת !!! במקרה זה לאור שתנאי ה **EXISTS** מחזיר את הערך TRUE יוחזר הסכום הכולל של כל היתרות של כל הסטודנטים ולא רק של הסטודנטים החברים באגודה.

**שימו לב** שבמקרה זה תת-השאילתא אינה יכולה לתפקד כשאילתא עצמאית, שכן יש בה התייחסות לשאילתא החיצונית.

## הערות:

- (1) מה שבעצם חשוב בחלק של תת-השאלתא המופעלת ב (**EXISTS**) הוא לא תוצאת השאלתא אלא האם מוחזר ערך ריק או לא.
- (2) כמובן שעל אותו משקל ניתן להשתמש ב (**NOT EXISTS**)
- (3) במקרה בו השאלתא הפנימית הייתה מחזירה ערך **FALSE** (כאשר שאלתא הפנימית לא מחזירה אפילו שורה אחת) השאלתא הראשית לא הייתה מחזירה תשובה (ז"א הייתה חוזרת סכמה ריקה)

## (2) תתי שאילתות במשפטי FROM:

### צורת השימוש:

```
SELECT ....  
FROM (Sub-Query) AS NewName;
```

חובה לבצע את פעולת הכינוי

נחזור לדוגמא מתחילת השיעור: שמות האנשים שגרים בעיר של Ben Levi הפעם לא כולל

BEN עצמן:

```
SELECT Name  
FROM Students , ( SELECT City  
FROM Students S  
WHERE Name = 'Ben Levi' ) AS ben  
WHERE ( S.City = ben.City ) AND (Name <> 'Ben Levi');
```

מתבצעת כאן  
מכפלה קרטזית

הסבר: תת השאילתא תחזיר טבלה שתכיל ערכים, ועם טבלה זו נבצע מכפלה קרטזית עם טבלת

סטודנטים ועל המכפלה נפעיל את תנאי ה where שמטרתו הינה סינון שורות.

הערה: חלק מגירסאות MySQL החינמיות אינן תומכות בכל סוגי תתי השאילתות.