הרצאה מספר 10

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
```


רואי זרחיה 2009-2010 - כל הזכויות שמורות ©

Sub Queries - תתי שאילתות

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
                                )
```


רואי זרחיה 2009-2010 - כל הזכויות שמורות ©

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 )
FROM   Workers1
```

3

```
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 ב**תת-שאילתא** המחשב את התוצאה של שליפת כמות העובדים הזמניים (3 עובדים) ואת תוצאה זו הכנסנו לתוך פקודת SELECT המקורית.

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

רואי זרחיה 2009-2010 - כל הזכויות שמורות ©

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

Students					
ChnNum	**Name**	**Street**	**Number**	**City**	**Amount**
1	Avi Cohen	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
Avi Cohen	Student	1
Ben Levi	Graduated	1
Chen Levin	Student	1
Debi Dvir	Student	0
Zvi Kaner	Graduated	1
Haim Itzhak	Student	0

סוגי תתי שאילתות:

אנו נעסוק ב-2 הסוגים העיקריים של תתי שאילתות שניתן לשלב במהלך כתיבת שאילתות.

(1) תתי שאילתות במשפטי WHERE ו HAVING שבתוכן יש 2 תתי-סוגים:

א) תתי שאילתות המחזירות ערך יחיד

ב) תתי שאילתות המחזירות אוסף ערכים

(2) תתי שאילתות בפסוק FROM

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

רואי זרחיה 2009-2010 - כל הזכויות שמורות ©

1) תתי שאילתות במשפטי WHERE ו HAVING:

- **תתי שאילתות המחזירות ערך יחיד**

דוגמא1: 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 השאילתות שבנינו ונשלב אותן לשאילתא אחת:

שילוב 2 שאילתות הבסיס לשימוש בתת-שאילתא:

```
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

© כל הזכויות שמורות - 2009-2010 רואי זרחיה

דוגמא2:

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.

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

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.

רואי זרחיה 2009-2010 - כל הזכויות שמורות ©

ב) תתי שאילתות המחזירות אוסף ערכים

תתי שאילתות מסוג זה הין 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 השווה 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 בת"א יש סכום השווה לסכום שהם עצמם מחזיקים.

Name
Ben Levi
Chen Levin

רואי זרחיה 2009-2010 - כל הזכויות שמורות ©

- **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 גבוה 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

רואי זרחיה 2009-2010 - כל הזכויות שמורות ©

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

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 תת-שאילתא בתוך הפסוקית IN(SELECT… שתחזיר אוסף של ערכים או ע"י קבוצות נתונות מראש של אוסף קבועים , כמו בדוגמא שלהלן

רואי זרחיה 2009-2010 - כל הזכויות שמורות ©

- **EXISTS()** – מחזיר את הערך TRUE אם קיימות שורות בתוצאת תת השאילתא (כולל שורות שמכילות NULL).

דוגמא: מה מחזירה השאילתא הבאה:

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

TRUE

SUM(Amount)
16,000

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-השאילתא המופעלת ב EXSIST() הוא לא תוצאת השאילתא אלא האם מוחזר ערך ריק או לא.

2) כמובן שעל אותו משקל ניתן להשתמש ב **NOT EXSIST()**

3) במקרה בו השאילתא הפנימית הייתה מחזירה ערך FALSE (כאשר שאילתא הפנימית לא מחזירה אפילו שורה אחת) השאילתא הראשית לא הייתה מחזירה תשובה (ז"א הייתה חוזרת סכמה ריקה)

רואי זרחיה 2009-2010 - כל הזכויות שמורות ©

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
                  WHERE    Name = 'Ben Levi' ) AS ben
WHERE            ( 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.

רואי זרחיה 2009-2010 - כל הזכויות שמורות ©