מסדי נתונים – 89-281
תרגול 10 - SQL 4 – תתי שאילתות

עמיעד רוזנברג

```
int getRandomNumber()
{
    return 4;  // chosen by fair dice roll.
               // guaranteed to be random.
}
```

תתי שאילתות

- שימוש בתתי שאילתות מאפשר לנו לפרק בקשה אחת למספר בקשות שונות.

- בשיעור זה נלמד כיצד להשתמש ב:
  - תתי שאילתות המחזירות ערך יחיד.
  - תתי שאילתות המחזירות אוסף ערכים.
    - אופרטורים המיועדים לשימוש עם אוסף ערכים.
  - תתי שאילתות בפסוק FROM.

- עקרונית, אפשר לשלב תתי שאילתות כמעט בכל מקום בשאילתא.

customers					
accountID	name	street	number	city	amount
1	Adam Cohen	Begin St.	21	Ramat-Gan	1200
2	Adam 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	David Dvir	Hayarkon St.	93	Givatiim	700
6	Eli Kaner	Vaitzman St.	17	Givatiim	3500
7	Haim Izhak	Haela St.	65	Jerusalem	1000

customersData			
accountID	name	status	hasALoan
1	Adam Cohen	employed	1
2	Adam Cohen	employed	1
3	Ben Levi	unemployed	1
4	Chen Levin	employed	1
5	David Dvir	employed	0
6	Eli Kaner	unemployed	1
7	Haim Izhak	employed	0

הרעיון

- עד כה, כאשר עשינו בדיקות מסויימות – תמיד ביצענו את הבדיקות מול ערכים מוחלטים או מול ערכים החוזרים מפונקציות.

- תתי שאילתות מאפשרות לנו לבצע בדיקות מול תשובות המתקבלות משאילתות אחרות.

- נוכל בעצם לפרק את השאילתא שקיבלנו לתתי שאילתות, ולענות על כל אחד מהם בנפרד.

תתי שאילתות המחזירות ערך יחיד

- דוגמא 1 :
  - הצג את מספרי החשבונות ושמות הלקוחות שסכום היתרה בחשבונם גדול מסכום היתרה הממוצע של לקוחות הבנק.

- נשים לב שנוכל לפרק את השאילתא שקיבלנו ל-2 שאילתות שונות :
  - X = סכום היתרה הממוצע של לקוחות הבנק.
  - מספרי החשבונות ושמות הלקוחות שסכום היתרה שלהם גדול מ-X.

- נפתור כל שאילתא בנפרד :

תתי שאילתות המחזירות ערך יחיד

- דוגמא 1 :
  - הצג את מספרי החשבונות ושמות הלקוחות שסכום היתרה בחשבונם גדול מסכום היתרה הממוצע של לקוחות הבנק.

  - ראשית : נתייחס ל-X (ממוצע היתרות) כאל נעלם ונפתור את השאילתא.
    - מספרי החשבונות ושמות הלקוחות שסכום היתרה שלהם גדול מ-X.
    - SELECT accountID, name
      FROM customers
      WHERE amount > X ;

תתי שאילתות המחזירות ערך יחיד

- דוגמא 1 :
  - הצג את מספרי החשבונות ושמות הלקוחות שסכום היתרה בחשבונם גדול מסכום היתרה הממוצע של לקוחות הבנק.
  - לאחר מכן, ניצור שאילתא שתחשב את X.
    - X = סכום היתרה הממוצע של לקוחות הבנק.
- SELECT AVG(amount) AS average
  FROM customers;

תתי שאילתות המחזירות ערך יחיד

- דוגמא 1 :
  - הצג את מספרי החשבונות ושמות הלקוחות שסכום היתרה בחשבונם גדול מסכום היתרה הממוצע של לקוחות הבנק.

- לסיום, נחבר את השאילתות שלנו לשאילתא אחת :

```
SELECT accountID, name
FROM customers
WHERE amount > (SELECT AVG(amount) AS average
                FROM customers);
```

תתי שאילתות המחזירות ערך יחיד

- דוגמא 2:
  - הצג את רשימת הלקוחות שגרים בעיר של דוד דביר.

- בניגוד לשיטת העבודה שפעלנו בה בעבר, נשים לב לכך שגם כאן נוכל לפרק את השאילתא שקיבלנו ל-2 שאילתות שונות:
  - X = העיר שבה גר דוד דביר.
  - רשימת הלקוחות שגרים בעיר X.

- נפתור כל שאילתא בנפרד:

תתי שאילתות המחזירות ערך יחיד

- דוגמא 2 :
  - הצג את רשימת הלקוחות שגרים בעיר של דוד דביר.
  - ראשית : נתייחס ל-X כאל נעלם ונפתור את השאילתא.
    - רשימת הלקוחות שגרים בעיר X.
- SELECT accountID, name
  FROM customers
  WHERE city = X;

תתי שאילתות המחזירות ערך יחיד

- דוגמא 2 :
  - הצג את רשימת הלקוחות שגרים בעיר של דוד דביר.
  - לאחר מכן, ניצור שאילתא שתחשב את X.
    - X = העיר שבה גר דוד דביר.
- SELECT city
  FROM customers
  WHERE name = ‘David Dvir’;

תתי שאילתות המחזירות ערך יחיד

- דוגמא 2:
  - הצג את רשימת הלקוחות שגרים בעיר של דוד דביר.
  - לסיום, נחבר את השאילתות שלנו לשאילתא אחת:

```
SELECT name
FROM customers
WHERE city = (SELECT city
              FROM customers
              WHERE name = 'David Dvir');
```

תתי שאילתות המחזירות ערך יחיד

- שימו לב לכך שתת השאילתא מחזירה ערך יחיד.
- SELECT city
  FROM customers
  WHERE name = ‘David Dvir’
- אם היו לנו 2 לקוחות שונים ששמם הוא דוד דביר?
  - היינו מקבלים מתת השאילתא קבוצת ערכים ולא היינו יכולים לבצע את ההשוואה!

תתי שאילתות המחזירות קב' ערכים

- כאשר תת שאילתא מחזירה קב' ערכים, ברור שלא נוכל לבצע מולה השוואה.
- נצטרך אופרטורים חדשים לעבודה מול קבוצת ערכים:
  - SOME()
  - ALL()
  - IN()
  - EXISTS()

תתי שאילתות המחזירות קב' ערכים

- האופרטור SOME() מאפשר לבדוק האם קיים לפחות ערך אחד מתוך איברי הקבוצה שבסוגריים העונה לתנאי.
- לדוגמא:
  - הצג את שמות הלקוחות שסכום הכסף בחשבונם שווה לסכום כלשהו הקיים בעיר ת"א.
    - ראשית – נחשב את רשימת סכומי הכסף שיש בחשבונם של הלקוחות הגרים בתל אביב:
    - SELECT amount FROM customers
      WHERE city='Tel-Aviv';

תתי שאילתות המחזירות קב' ערכים

- האופרטור SOME() מאפשר לבדוק האם קיים לפחות ערך אחד מתוך איברי הקבוצה שבסוגריים העונה לתנאי.
- לדוגמא:
  - הצג את שמות הלקוחות שסכום הכסף בחשבונם שווה לסכום כלשהו הקיים בעיר ת"א.
    - כעת, נבדוק מיהם הלקוחות שחשבונם שווה לערך כלשהו מהטבלה שקיבלנו:
- SELECT name FROM customers
  WHERE amount = SOME(SELECT amount FROM customers
  WHERE city='Tel-Aviv');

תתי שאילתות המחזירות קב' ערכים

- האופרטור ()ALL מאפשר לבדוק האם כל איברי הקבוצה שבסוגריים עונים לתנאי.
- לדוגמא:
  - הצג את העיר בה ממוצע היתרות הוא הגבוה ביותר מבין כל הערים.
    - ראשית – נחשב את כל ממוצעי היתרות בערים השונות:
    - SELECT AVG(amount) FROM customers
      GROUP BY city;

תתי שאילתות המחזירות קב' ערכים

- האופרטור ()ALL מאפשר לבדוק האם כל איברי הקבוצה שבסוגריים עונים לתנאי.
- לדוגמא:
  - הצג את העיר בה ממוצע היתרות הוא הגבוה ביותר מבין כל הערים.
    - כעת נבדוק באיזו עיר ממוצע היתרות הוא גבוה (או שווה) לכל ממוצעי היתרות בערים השונות:
      - SELECT city FROM customers
        GROUP BY city
        HAVING AVG(amount) >= ALL(SELECT AVG(amount)
        FROM customers
        GROUP BY city);

תתי שאילתות המחזירות קב' ערכים

- האופרטור IN() מאפשר לבדוק השתייכות לקבי ערכים.
- לדוגמא:
  - הצג את שמות הלקוחות הגרים באילת, חיפה וירושלים.
- SELECT name
  FROM customers
  WHERE city IN('Eilat', 'Haifa', 'Jerusalem');
- הערות:
  - שימוש ב-IN() שקול לשימוש ב-SOME()=
  - ניתן להשתמש בפקודה NOT IN() (שקול לפקודה ALL()<>)
  - אוסף הערכים ב-IN() יכול להיות גם טבלה החוזרת מתת-שאילתא.

תתי שאילתות המחזירות קב' ערכים

- האופרטור EXISTS() מאפשר לבדוק האם קיימים ערכים בתשובה החוזרת.
- לדוגמא:
  - מה מחזירה השאילתא הבאה:
    - SELECT SUM(amount) FROM customers
      WHERE EXISTS (SELECT *
      FROM customersData AS cd
      WHERE cd.name=customers.name);
  - סכום הכסף בחשבונות של <u>כלל הלקוחות</u>!

תתי שאילתות המחזירות קב' ערכים

- SELECT SUM(amount) FROM customers
  WHERE EXISTS (SELECT *
  FROM customersData AS cd
  WHERE cd.name=customers.name);

- החלק הפנימי של השאילתא מחזיר טבלה עם קב' הלקוחות שהשם שלהם בטבלאת customers זהה לשם שלהם בטבלאת customersData.
  מכיוון שהטבלה החוזרת אינה ריקה – התנאי EXISTS() מחזיר TRUE ולכן כל השורות נבחרות!

תתי שאילתות המחזירות קב' ערכים

- SELECT SUM(amount) FROM customers
  WHERE EXISTS (SELECT *
  FROM customersData AS cd
  WHERE cd.name=customers.name);

- שימו לב שבשאילתא הפנימית השתשמנו גם בטבלה customers. הסיבה היא שניתן להשתמש בשאילתות הפנימיות גם בטבלאות המופיעות בשאילתות החיצוניות.
- השאילתא הפנימית אינה יכולה לתפקד בנפרד, שכן יש בה התייחסות לטבלה הנמצאת בשאילתא החיצונית.

תתי שאילתות במשפטי FROM

- ניתן להשתמש בתתי שאילתות גם במשפטי FROM.
- צורת השימוש הכללית היא:
- SELECT ...
  FROM (Sub-Query) AS newName ...

תתי שאילתות במשפטי FROM

- נחזור לשאלה שהייתה לנו בתחילת השיעור:
- הצג את שמות האנשים הגרים בעיר של דוד דביר.
- בשיטה זו – ראשית ניצור טבלה שתכיל עבורנו את שם העיר שבה גר דוד דביר:
- SELECT city FROM customers
  WHERE name = ‘David Dvir’;

תתי שאילתות במשפטי FROM

- נחזור לשאלה שהייתה לנו בתחילת השיעור:

- הצג את שמות האנשים הגרים בעיר של דוד דביר.
- את התוצאה (העיר) נכניס כטבלה בשאילתא הגדולה:

```
SELECT name
FROM customers , (SELECT city FROM customers
                  WHERE name = 'David Dvir') AS dvir
WHERE customers.city = dvir.city;
```

- הערה כללית:
  - חלק מתתי השאילתות לא נתמכות ע"י הגירסאות החינמיות של MySQL!

איחוד - UNION

- בדומה לאלגברת יחסים, גם ב-SQL יש לנו אופציה לבצע איחוד בין טבלאות.

- נניח שבוצע איחוד בין הבנק שלנו לבנק אחר.
  קיבלנו את בסיס הנתונים של הבנק השני, אבל שם הנתונים שמורים בפורמט שבו כל הנתונים על הלקוחות שמורים באותה הטבלה. כלומר, יש לנו טבלה אחרת שנקראת bankCustomers שהסכמה שלה היא: (accountID, name, street, number, city, amount, status, hasALoan).
- נרצה להציג למנהל את המידע על כלל הלקוחות.

איחוד - UNION

- customers (accountID, name, street, number, city, amount)
- customersData (accountID, name, status, hasALoan)
- bankCustomers (accountID, name, street, number, city, amount, status, hasALoan)

- אם נרצה להציג את המידע בפורמט של הבנק שלנו:
  - לדוגמא, נציג את טבלאת customers:
    - SELECT * FROM customers
      UNION
      SELECT accountID, name, street, number, city, amount
      FROM bankCustomers;

איחוד - UNION

- customers (accountID, name, street, number, city, amount)
- customersData (accountID, name, status, hasALoan)
- bankCustomers (accountID, name, street, number, city, amount, status, hasALoan)

- אם נרצה להציג את המידע בפורמט של הבנק החדש :
  - SELECT * FROM bankCustomers
    UNION
    SELECT * FROM customers NATURAL JOIN customersData ;

- פקודת UNION רגילה מבטלת רשומות כפולות (כלומר, רשומה זהה שקיימת ב-2 הטבלאות תופיע בתוצאה הסופית פעם אחת בלבד).
אם נרצה שרשומות כפולות אכן יופיעו פעמיים נשתמש בפקודה UNION ALL