שנקר

בית הספר לתעשייה וניהול
בסיס נתונים 31-028-24

# הרצאה מס' 7

***נושאים לשיעור זה:***

- פקודות איחוד בשפת SQL (המשך)
- פקודות חיתוך בשפת SQL
- מכפלה קרטזית ופקודות JOIN לחיבור בין טבלאות
- ויתור על השימוש בפקודת ה WHERE לצירוף טבלאות
- פקודת ON ופקודת USING
- צירוף חיצוני Outer Join

***סיכום אוסף הפקודות שלמדנו עד כה:***

```
SELECT      X, count(Y)
FROM        Students AS S , Course AS C
WHERE       S.studentID = C.studentID
            AND (X>100)
GROUP BY    X
HAVING      count(Y) > 2500
ORDER BY    X
```

# פקודת איחוד UNION (חזרה והמשך...)

**דוגמא:** נניח שקיימות טבלאות סטודנטים שונות באוניברסיטת בר אילן ובאוניברסיטת תל-אביב ושר החינוך היה רוצה לקבל רשימה אחת של כל הסטודנטים בשני המוסדות:

**סכמת הסטודנטים בבר-אילן:**

| StudentID | StudentName | StudentCity | Department |
|---|---|---|---|
| 1 | a | RG | MA |
| 2 | b | BB | CS |

**סכמת הסטודנטים בתל-אביב:**

| SID | SName | Faculty | Department | City | Age |
|---|---|---|---|---|---|
| 3 | c | ES | PH | TA | 22 |
| 4 | d | ES | MA | TA | 24 |

הבעיה הנראית לעין היא ששתי הסכמות כאן שונות ובכדי לאחד נתונים נצטרך סכמות זהות ולכן נכתוב את השאילתא הבאה:

```
SELECT    StudentID AS ID, StudentName AS Name
FROM      BarIlanStudents

UNION

SELECT    SID AS ID, SName AS Name
FROM      TelAvivStudents
```

| ID | Name |
|---|---|
| 1 | A |
| 2 | B |
| 3 | C |
| 4 | D |

**הערה חשובה:**

פקודת UNION בוחרת את השורות המופיעות בשתי הטבלאות ומבצעת על תוצאה זו אוטומטית את פעולת ה Distinct (הסרת שורות כפולות), על מנת לאפשר שורות כפולות בתוצאה המוחזרת מפעולת האיחוד יש להשתמש בפקודת **UNION ALL** שמטרתה היא לבצע את פעולה האיחוד ללא הסרת השורות הכפולות:

**דוגמא:** עבור 2 טבלאות נתונים **R1 , R2** נראה איך משתנה התוצאה כאשר נשתמש פעם בפקודת **UNION** ופעם בפקודת **UNION ALL**:

| R1 | | |
|---|---|---|
| X | Y | Z |
| 1 | A | 3 |
| 2 | B | 4 |
| 3 | C | 5 |

| R2 | | |
|---|---|---|
| X | Y | Z |
| 2 | B | 4 |
| 3 | C | 5 |
| 4 | D | 6 |

```
SELECT  *
FROM    R1
UNION
SELECT  *
FROM    R2
```

| R1 UNION R2 | | |
|---|---|---|
| X | Y | Z |
| 1 | A | 3 |
| 2 | B | 4 |
| 3 | C | 5 |
| 4 | D | 6 |

```
SELECT  *
FROM    R1
UNION ALL
SELECT  *
FROM    R2
```

| R1 UNION ALL R2 | | |
|---|---|---|
| X | Y | Z |
| 1 | A | 3 |
| 2 | B | 4 |
| 2 | B | 4 |
| 3 | C | 5 |
| 3 | C | 5 |
| 4 | D | 6 |

# ***פקודת חיתוך Intersect***

פקודה המקבילה לפעולת החיתוך באלגברת היחסים (והפוכה לפעולת האיחוד) שמטרתה ליצור טבלה המכילה רק את השורות המופיעות בטבלה הראשונה וגם בטבלה השנייה.

**הערות**: כמו שאמרנו קודם בנושא "האיחוד" גם כאן, שתי הטבלאות צריכות להיות תואמות מבחינת דרגת היחס ומבחינת ה type של כל תכונה.

**דוגמא**: נניח שנתונות שתי טבלאות ונרצה לקבל את השורות המשותפות לשתיהן:

***Table1:***

| ID | Name |
|---|---|
| 1 | a |
| 2 | b |
| 3 | c |
| 4 | d |

***Table2:***

| ID | Name |
|---|---|
| 2 | b |
| 3 | c |
| 9 | d |

```
SELECT      *
FROMTable1

INTERSECT

SELECT      *
FROMTable2
```

| ID | Name |
|---|---|
| 2 | b |
| 3 | c |

```
SELECT      Name
FROM        Table1

INTERSECT

SELECT      Name
FROM        Table2
```

| Name |
|---|
| b |
| c |
| d |

# מכפלה קרטזית ופקודות JOIN לחיבור בין טבלאות

המכפלה הקרטזית הינה פעולה שמכפילה 2 טבלאות (פעולה בינארית) כפי שראינו בנושא אלגברת היחסים.
פעולת המכפלה הקרטזית הינה אחת משש הפעולות הפשוטות כאשר ראינו שקיימת פקודה מורכבת JOIN המבוססת על פעולת המכפלה הקרטזית הנ"ל.

לפקודת ה JOIN יש מטרה פשוטה והיא לבצע מכפלה בין 2 טבלאות כאשר בטבלת התוצאה יתקבלו רק רשומות "**בעלי משמעות**", ז"א רשומות שיש להם קשר כלשהוא (לפחות עמודה משותפת אחת) בטבלאות הבסיס המוכפלות.

כזכור מאלגברת היחסים, בעת ביצוע פעולת הצירוף ניתן להגדיר את תנאי החיבור בין הטבלאות (העמודה המשותפת).

על מנת להמחיש את שתי הפעולות הנ"ל בעבודה למול טבלאות בשפת SQL נגדיר סכמות של טבלאות מידע פשוטות שעליהן נבצע את הדוגמאות עבור השיעור:

| Students | | |
|---|---|---|
| FirstName | LastName | StudentID |
| Avi | Cohen | 111 |
| Dan | Israeli | 222 |
| Ofer | Bar | 333 |

| Courses | | | |
|---|---|---|---|
| StudentID | CourseNumber | CourseName | Grade |
| 111 | 289 | DB | 96 |
| 111 | 281 | Algo | 85 |
| 222 | 281 | Algo | 78 |

**קיימות שתי אפשרויות לבצע מכפלה קרטזית ב SQL:**

1) רשימת שמות טבלאות (בפסוקית ה FROM) כאשר יש הפרדה של פסיקים בין הטבלאות:

```
SELECT    *
FROM      Students, Courses
```

2) רשימת שמות הטבלאות כאשר ביניהם נרשום את המילה JOIN (צירוף) או INNER JOIN:

```
SELECT    *
FROM      Students JOIN Courses
```

← בשני המקרים נקבל טבלה חדשה בת 7 עמודות ובעלת 9 שורות (טבלה זו תכיל את כל האפשרויות).

**3 עמודות + 4 עמודות**

| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|
| 2 | | | | | | |
| 3 | | | | | | |
| 4 | | | | | | |
| 5 | | | | | | |
| 6 | | | | | | |
| 7 | | | | | | |
| 8 | | | | | | |
| 9 | | | | | | |

**3 שורות X 3 שורות**

לאור העובדה שאמרנו כבר בעבר שמכפלה קרטזית הינה הבסיס לכל פעולות ה JOIN (צירוף בין טבלאות) הקיימות ב SQL, אזי כל 4 הפעולות שלמדנו באלגברת יחסים ניתנות לתרגום לשפת SQL ע"י שימוש בפקודת ה JOIN:

- ✓ JOIN , INNER JOIN - מכפלה קרטזית פשוטה
- ✓ LEFT OUTER JOIN - שמירת מידע שמאלה
- ✓ RIGHT OUTER JOIN - שמירת מידע ימינה
- ✓ FULL OUTER JOIN - שמירת מידע דו כיוונית

## ***ויתור על השימוש בפקודת ה WHERE לצירוף טבלאות:***

עד היום כאשר הינו רוצים לבצע מכפלה בין 2 טבלאות היינו משתמשים בפסיק בין הטבלאות הנמצאות בפקודת ה WHERE, לדוגמא:

```
SELECT      FirstName, CourseName
FROM        Students AS S , Course AS C
WHERE       (S.studentID = C.studentID)
            AND
            (FirstName = 'Dan')
```

נוכל לראות שלפקודת ה WHERE קיים שימוש כפול:

1) הגדרת תנאי החיבור בין הטבלאות (שעליהן נבצע מכפלה קרטזית)
2) הגדרת תנאי סינון לטבלאות הבסיס

בשל העובדה **שמכפלה קרטזית** מחזירה לנו את כל הפרמוטציות (האפשרויות) הקיימות, אזי עד היום כשרצינו לבצע סינון שורות היינו מוספים תנאי Where.

כעת כאשר נשתמש בפעולת **צירוף** נוכל להפעיל **פקודת סינון ייעודית** שתאפשר לנו לסנן שורות שאינן בעלות "ערך" תוך כדי ביצוע הצירוף עצמו (ולמנוע חיבור של "רופא" עם "מורה"), לאור כך ניתן להוסיף תנאי לביצוע הצירוף (מייד לאחר פסוקית ה FROM) שיאפשר לנו לבצע פעולת JOIN בין שתי הטבלאות ולהחזיר רק את השורות הרלוונטיות.

עבור התנאי הספציפי הזה הוגדרו ב SQL שתי פקודות שבהן ניתן להשתמש בכדי להגדיר את תנאי החיבור

1) פקודת ON
2) פקודת USING.

## 1) אופרטור ON

מטרת פקודת ה ON הינה להפעיל תנאי סינון על פקודת ה JOIN המתבצעת לפניה.

***מעבר מפקודת מכפלה קרטזית לפקודת צירוף:***

עד ללימוד פקודת ה JOIN, כשרצינו לחבר 2 טבלאות היינו משתמשים במכפלה קרטזית הבאה ואת תנאי החיבור ל-2 הטבלאות הכנסנו לתוך פקודת ה WHERE, הבעיה הייתה שפקודת ה WHERE גם הכילה תנאי סינון (לפקודת ה WHERE היו 2 תפקידים).

```
SELECT    FirstName, CourseName, Grade
FROM      Students AS S , Courses AS C
WHERE     (S.StudentID = C.StudentID)
          AND
          (C.CourseName = 'Algo')
```

כעת, כשנשתמש בפקודת JOIN לצירוף טבלאות, נוכל להוסיף רכיב ה ON שיכיל את תנאי החיבור ל-2 הטבלאות, ובפקודת ה WHERE יישארו רק תנאי סינון. ע"י כך יצרנו הפרדה בין פקודת החיבור לפקודת סינון המידע.

```
SELECT    FirstName, CourseName, Grade
FROM      Students AS S JOIN Courses AS C
ON        S.StudentID = C.StudentID
WHERE     (C.CourseName = 'Algo')
```

| FirstName | CourseName | Grade |
|---|---|---|
| Avi | Algo | 85 |
| Dan | Algo | 78 |

בפקודת ה - ON יכול להופיע תנאי פשוט או מורכב שיופעל על הצירוף המופיע ב בפקודת ה – FROM שיכול להכיל גם את האופרטורים AND , OR, NOT.

**<u>הסבר</u>:**

השילוב של פקודת ה ON לאחר פקודת ה JOIN יגרור ביצוע של מכפלה קרטזית כאשר תוחזרנה רק השורות המקיימות את תנאי הצירוף (שקול לפקודת צירוף-טטה מאלגברת היחסים).

← נשתמש בפסוקית ON כדי לרשום תנאים שמציינים כיצד לבצע את הצירוף בין הטבלאות.

**<u>הערות:</u>**

1) למרות שהפעלת מכפלה קרטזית מחזירה מספר עמודות השווה לסכום העמודות ב 2 הטבלאות (בדוגמא זו - 7 עמודות) נוכל לראות שקיבלנו כאן רק 3 עמודות. הסיבה לכך אינה בשל העובדה ש **JOIN** אינו מחזיר 7 עמודות אלא בשל העובדה שבפקודת ה **SELECT** בחרנו להציג רק 3 מתוך ה 7 העמודות הקיימות.
2) נוכל לראות שקיבלנו חזרה 3 שורות מתוך 9 השורות האפשריות, וזאת כי בביצוע הצירוף היו 6 שורות שאינן הגיוניות מבחינת ביצוע הצירוף והן אלו שנפלו ("סוננו") בתנאי ה **ON**.

## 2) אופרטור USING

פקודת USING שקולה לפקודת ON אך יתרונה הוא כאשר שם העמודה/ות זהה בשתי הטבלאות המצורפות (על אותו עיקרון של צירוף טבעי מאלגברת יחסים).

**דוגמא:**

ניקח את הדוגמא הקודמת ונשנה אותה כך שתשתמש בפסוקית USING:

```
SELECT    FirstName, CourseName, Grade
FROM      Students JOIN Courses
USING     (StudentID)
WHERE     (C.CourseName = 'Algo')
```

נוכל לראות שנקבל בדיוק את אותה התוצאה:

| FirstName | CourseName | Grade |
|---|---|---|
| Avi | Algo | 85 |
| Dan | Algo | 78 |

*** בפועל תתבצע מכפלה קרטזית ותבחרנה השורות בהן יש ערכים זהים בעמודות אלו.

**הבדלים בשימוש בין פקודות USING ו ON:**

- בפקודת USING נרשום את שם העמודה/העמודות הקיימות בשתי הטבלאות שעליהן נבצע את פעולת הצירוף **פעם אחת בלבד**, ז"א לא נוכל להפעיל USING כאשר שמות העמודות שונים (אפילו אם תוכן העמודות זהה) - **(כמו בפעולת הצירוף-טבעי)**.
- בפקודת ON נוכל לרשום עמודות בעלי שמות שונים ולחבר ביניהם - **(שקול לצירוף-טטה)**.

## OUTER JOIN – צירוף חיצוני

כשלמדנו אלגברת יחסים ראינו שבמעבר ממכפלה קרטזית פשוטה לשימוש בפעולת הצירוף (JOIN) יכול להיווצר מצב של איבוד מידע עקב פעולת ההכפלה בשילוב של תנאי סינון.
לשם כך קיימת פקודת צירוף מתוחכמת יותר המאפשרת לשמור מידע מהטבלאות המשתתפות בצירוף.

על מנת להמחיש את עיקרון שמירת המידע בעבודה למול טבלאות בשפת SQL נחזור לטבלאות מהדוגמא הקודמת.

```
SELECT    FirstName, CourseName, Grade
FROM      Students JOIN Courses
USING     (StudentID);
```

התוצאה שהתקבלה הייתה:

| FirstName | CourseName | Grade |
|---|---|---|
| Avi | DB | 96 |
| Avi | Algo | 85 |
| Dan | Algo | 78 |

**שאלה**: מה קרה ל Ofer מהדוגמא הקודמת – למה הוא לא הופיע בתוצאה הסופית בביצוע פעולת הצירוף?

**תשובה**: נוכל לראות ש Ofer שהוא סטודנט במכללה אך אינו רשום בשום קורס ומכאן בפעולת הצירוף אין הוא עונה לתנאי הצירוף (בפקודת ON/USING) ולכן "נעלם" מטבלת התוצאה הסופית.

**פתרון**: על מנת לקבל גם את המידע על Ofer (ז"א בכדי לקבל את המידע גם על הסטודנטים שלא רשומים לקורסים המופיעים בטבלת קורסים) נשתמש בפקודת ***הצירוף חיצוני*** שמטרתה לאפשר שמירת מידע מטבלאות המשתתפות בפעולת הצירוף.

## ישנם 3 סוגים של צירוף חיצוני:

- LEFT JOIN – **שמירת מידע מהטבלה השמאלית**, ז"א ששורות מהטבלה השמאלית שלא מתאימות לאף שורה מהטבלה הימנית (לפי התנאי שמצוין ב USING/ON) יתווספו גם הן לתוצאה המוחזרת, כאשר בשאר העמודות (הנוספות) יופיעו ערכי NULL.
- RIGHT JOIN – **שמירת מידע מהטבלה הימנית** (כמו LEFT רק הפוך).
- FULL JOIN – **שמירת מידע משתי הטבלאות** (איחוד של שמאלה וימינה).

**דוגמא:** שמות **כל** הסטודנטים ומספרי הקורס (אם לקחו קורס כלשהוא).

* לא כל סטודנט חייב להיות רשום לקורס, יכול להיות שהוא עדיין רק מועמד.

```
SELECT    FirstName, LastName, CourseNumber
FROM      Students LEFT OUTER JOIN Courses
USING     (StudentID);
```

| FirstName | LastName | CourseNumber |
|---|---|---|
| Avi | Cohen | 289 |
| Avi | Cohen | 281 |
| Dan | Israeli | 281 |
| Ofer | Bar | Null |

**הערות חשובות:**

1) **איבוד מידע יכול להתרחש רק אם יש תנאי על הצירוף**, לכן אי אפשר להשתמש בצירוף חיצוני מבלי לרשום פסוקית ON או פסוקית USING.
2) נשים לב שהשלמת המידע מתבצעת עבור שורות ש"נפלו" בתנאי ה ON או ה USING, ז"א שתנאי הסינון שרשמנו יחושב ואז שורות שלא עמדו בתנאי זה תרופדנה בערכי NULL בהתאם לכיוון הצירוף.
3) חשוב להבין שפקודת ה - WHERE מתבצעת אחרי פעולת שמירת המידע ולכן **אין שמירת מידע על שורות שסוננו בתנאי ה WHERE** (ראו דוגמא בעמוד הבא).

## סיכום נושא צירופי טבלאות:

| X | Y |
|---|---|
| a | 1 |
| b | 2 |

| Y | Z |
|---|---|
| 2 | f |
| 3 | m |

| X | R1.y | R2.y | Z |
|---|---|---|---|
| a | 1 | 2 | f |
| a | 1 | 3 | m |
| b | 2 | 2 | f |
| b | 2 | 3 | m |

```
SELECT *
FROM  R1 , R2
```

```
SELECT *
FROM  R1 JOIN R2
```

| X | R1.y | R2.y | Z |
|---|---|---|---|
| b | 2 | 2 | f |

```
SELECT *
FROM  R1 JOIN R2
ON      R1.y = R2.y
```

| X | R1.y | R2.y | Z |
|---|---|---|---|
| a | 1 | NULL | NULL |
| b | 2 | 2 | f |

```
SELECT *
FROM  R1 LEFT OUTER JOIN R2
ON      R1.y = R2.y
```

| X | R1.y | R2.y | Z |
|---|---|---|---|
| B | 2 | 2 | f |
| NULL | NULL | 3 | m |

```
SELECT *
FROM  R1 RIGHT OUTER JOIN R2
ON      R1.y = R2.y
```

**FULL OUTER JOIN = RIGHT OUTER JOIN UNION LEFT OUTER JOIN**