**מיני פרויקט בבסיסי נתונים**

**שגב בורשטיין & יקיר דמרי**

**20.6.2021**

תוכן

[מבוא 2](#_Toc74573024)

[הפרויקט שלנו 3](#_Toc74573025)

[תרשים ERD 3](#_Toc74573026)

[תיאור הישויות והקשרים 4](#_Toc74573027)

[ישויות 4](#_Toc74573028)

[קשרים 4](#_Toc74573029)

[נרמול הטבלאות 5](#_Toc74573030)

[תרשים DSD 5](#_Toc74573031)

[יצירת הטבלאות 5](#_Toc74573032)

[אכלוס טבלאות 6](#_Toc74573033)

[שאילתות SQL 7](#_Toc74573034)

[אינדקסים 11](#_Toc74573035)

[הרשאות 13](#_Toc74573036)

[Views 14](#_Toc74573037)

[תרשימים 18](#_Toc74573038)

[פונקציות ופרוצדורות 20](#_Toc74573039)

[סיכום 22](#_Toc74573040)

# מבוא

הרעיון שנבחר במהלך הדיון בכיתה הוא לממש בסיס נתונים המספק את הפונקציונליות על מנת לתמוך **ברשת חברתית**. הרשת שלנו מאפשרת למשתמש מגוון עולמות לנוחותו כגון פיתוח קשרים עם משתמשים אחרים ותקשורת עמהם, גישה לקבוצות ופגישות חברתיות המכילות תוכן משותף וכמובן היכולת לשתף את דעתו האישית של המשתמש עם שאר המשתמשים ברשת ולשמוע את דעתם על כך.

על כן, אלו הן המחלקות השונות המנהלות את הרשת לרוחבה:

* **פרופיל משתמש**
* **פרסום פוסט חדש**
  + סוג הפוסט
  + תגובה על פוסט
* **התקהלות קבוצתית**
  + אירוע
  + קבוצה
  + עמוד עסקי
    - סוג העסק
* **התראה**
* **התפתחות חברתית**
  + הצעת חברות
  + בקשת חברות
* **חבר**
* **נראות**
* **פעולות משתמש אקטיבי**
  + לייק
  + שיתוף
  + תיוג

# הפרויקט שלנו

כחלק מתפקוד של רשת חברתית, יש צורך לספק למשתמש את האפשרות ליצור קשרים חדשים בין משתמשים קיימים במערכת, עדכונו בתהליכים שנעשים סביבו, חברים חדשים שהוא עשוי להכיר דרך קשרים שהוא כבר פיתח ועוד.

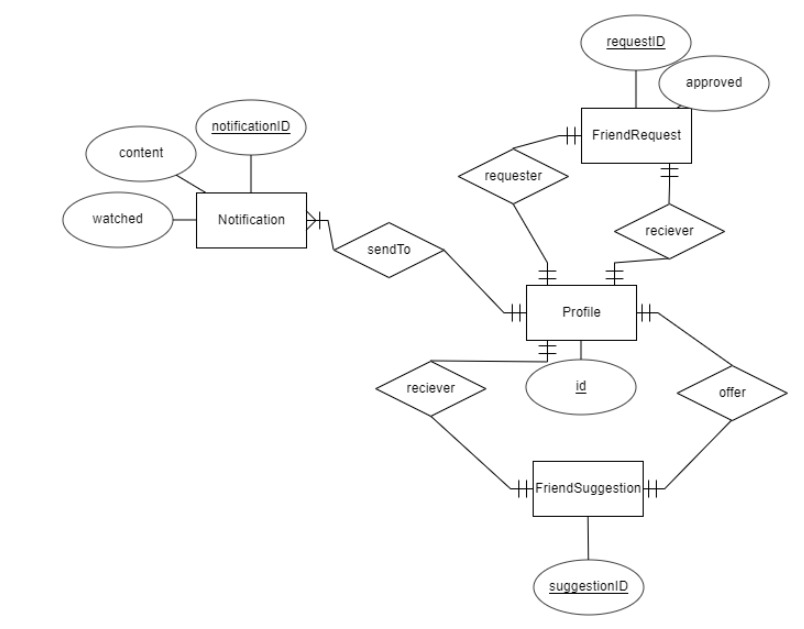
כחלק מתפקידנו במערכת הכוללת, לקחנו על עצמנו את יישומם של הטבלאות הבאות:

* **התראות** משתמש בעדכונים מהנעשה בסביבתו (פוסטים חדשים שהועלו, הודעות חדשות בקבוצות בהם הוא לוקח חלק ועוד)
* **בקשת חברות** באמצעותה המשתמש מסוגל להרחיב את מעגל הקשרים שלו.
* **הצעת חברות**. המלצות מטעם המערכת בדבר חברים חדשים שכדאי למשתמש להכיר בהתבסס על נתונים קודמים.

נתונים אלו מרוכזים בטבלאות שלנו ובאמצעות אינטראקציה מספקים למנהל המערכת אפשרות צפייה ומעקב במתרחש ברשת.

## תרשים ERD

כפי שהזכרנו, במחלקה שלנו, ישנן 3 ישויות: התראות, בקשת חברות והצעת חברות. בשלב הראשון יצרנו תרשים ERD שיתאר את הקשרים בין הישויות הללו ואת התכונות שלהן. בעזרת תרשים זה הבנו את התנהלות המערכת ואת האפשרויות השונות העומדות לרשותנו מבחינת פונקציונליות הניתנת להציע למנהל המערכת. (מנהל מערכת בסיס המידע).



## תיאור הישויות והקשרים

### ישויות

* Notification – ישות זאת אחראית על כל ההתראות בסביבת המשתמש ברשת החברתית.

ישות זאת הינה חזקה, כיוון שיכולה להתקיים ללא תלות בישות אחרת.

* notification\_id – מספר מזהה של ההתראה (PK).
* profile\_id – מזהה הפרופיל שקיבל את ההתראה (FK).
* content – תוכן ההתראה.
* watched – האם ההודעה נצפתה.
* FriendshipSuggestion – ישות זאת אחראית על כל הצעות החברות שהמערכת מסוגלת להציע למשתמש בהתבסס על נתונים ופעולות קודמות שהמשתמש ביצע.

ישות זאת הינה חזקה, כיוון שיכולה להתקיים ללא תלות בישות אחרת.

* suggestion\_id – מספר מזהה של ההצעה (PK)
* profile1\_id – מספר מזהה של הפרופיל **המוצע** (FK)
* profile2\_id – מספר מזהה של הפרופיל **המקבל** את ההצעה (FK).
* Friendship-Request – ישות זאת אחראית על כל בקשות החברות שהמשתמש מפנה כלפי משתמשים אחרים ברשת החברתית על מנת להגדיל את הסביבה האישית שלו.

ישות זאת הינה חזקה, כיוון שיכולה להתקיים ללא תלות בישות אחרת.

* request\_id – מספר מזהה של הבקשה (PK).
* profile1\_id – מספר מזהה של המשתמש **המקבל** (FK).
* profile2\_id – מספר מזהה של המשתמש **המציע** (FK)
* isApproved – האם הבקשה אושרה. [כן/לא/מחכה לתשובה].

### קשרים

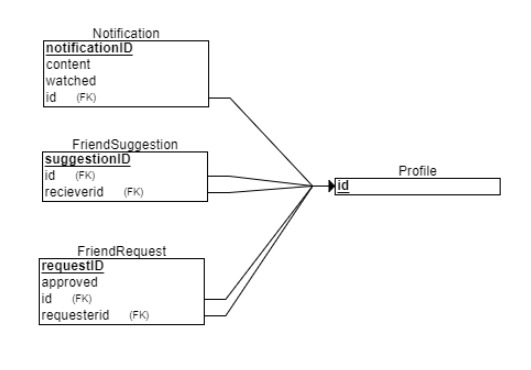
* **sendto** – הקשר בין Notification לבין Profile. הקשר הינו חזק, כיוון שמקשר בין שתי ישויות חזקות. הקשר הוא M:1 משום שיכול להיות הרבה התראות למשתמש בודד אבל לא יכול להיות הרבה משתמשים להתראה בודדת.
* **Receiver** – הקשר בין profile לבין friendshipSuggestion. הקשר הינו חזק, כיוון שמקשר בין שתי ישויות חזקות. הקשר הוא 1:1 משום שלכל הצעה יכול להיות רק אדם אחד שמקבל אותה (אלו הצעות מותאמות אישית פר פרופיל).
* **Receiver** - הקשר בין profile לבין friendshipRequest. הקשר הינו חזק, כיוון שמקשר בין שתי ישויות חזקות. הקשר הוא 1:1 משום שלכל בקשה יכול להיות רק אדם אחד שמקבל אותה.
* **Requester** – הקשר בין profile לבין friendshipRequest. הקשר הינו חזק, כיוון שמקשר בין שתי ישויות חזקות. הקשר הוא 1:1 משום שלכל בקשה יכול להיות רק אדם אחד ששולח אותה.
* **Offer** - הקשר בין profile לבין friendshipSuggestion. הקשר הינו חזק, כיוון שמקשר בין שתי ישויות חזקות. הקשר הוא 1:1 משום שלכל הצעה יכול להיות רק אדם אחד שמציעים אותו בה.

## נרמול הטבלאות

היחסים עומדים ב- NF3 וב- BCNF : מכיוון שבכל טבלה, התלויות הפונקציונאליות הלא-טריוויאליות הן מהמפתח אל תכונות נוספות לכן מתקיים שלכל , הוא מפתח ולכן הם עומדים בתנאים.

## תרשים DSD

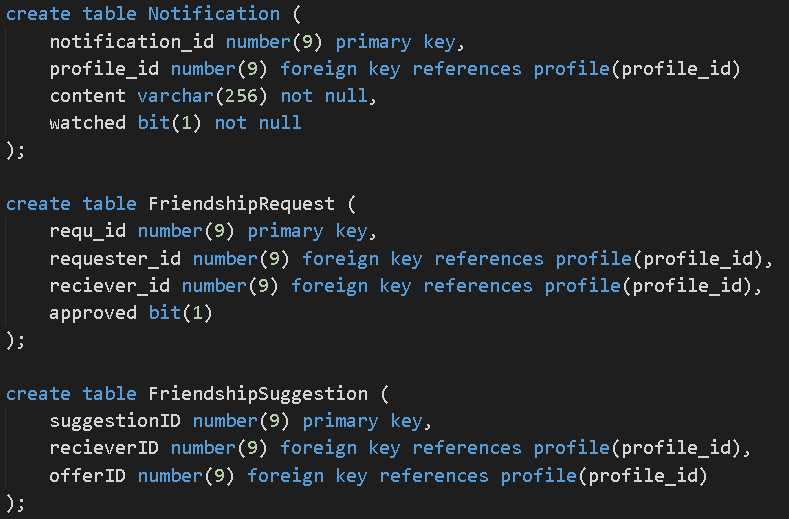
על פי תרשים ה- ERD ועל ידי הבנת הקשרים בין הישויות, יצרנו תרשים DSD עבור החלק שלנו במערכת: מחלקת מתקני המלון.



## יצירת הטבלאות

אחרי שהבנו כיצד בסיס הנתונים צריך להראות בצורה מדויקת, מה תכיל כל טבלה ומהם הקשרים בין כל הטבלאות, ניגשנו ליצירת הטבלאות בפועל בעזרת פקודות הcreate table.

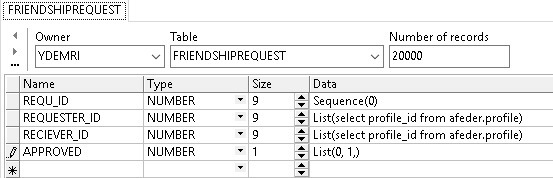
יצרנו קוד לייצור הטבלאות באמצעות export SQL של האתר erdPlus, יצרנו קובץ SQL ואז העתקנו את קוד ה- SQL של כל טבלה אל תוכנת ה-plsql לשם יצירת הטבלאות בפועל.



## אכלוס טבלאות

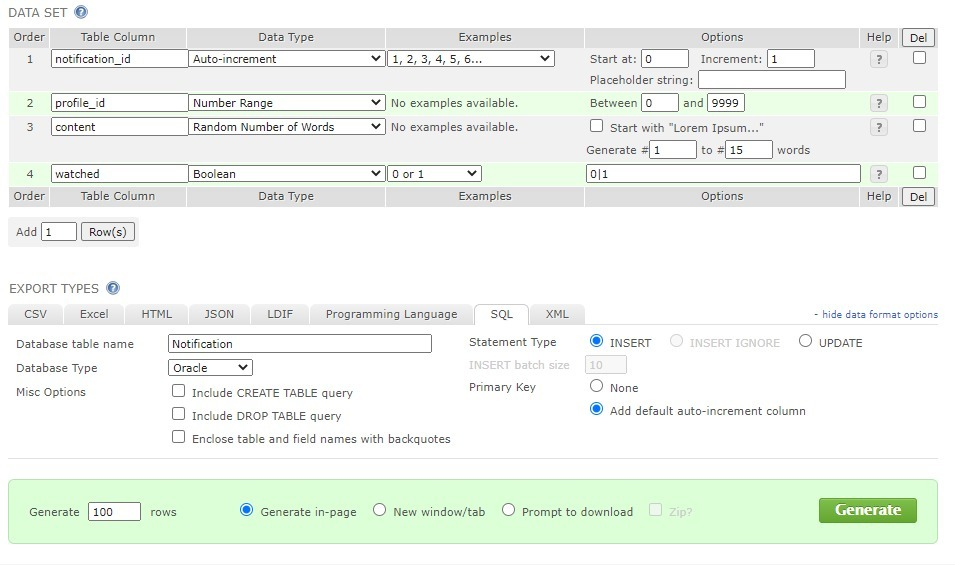
לאחר יצירת הטבלאות, אנו נדרשים למלא אותן במידע. על מנת לעשות זאת, השתמשנו במגוון על מנת לאכלס את הטבלאות. להלן הכלים והתוצאות:

* Data Generator מתוכנת ה-PL/SQL להכנסת ערכים לטבלת ה-FriendshipRequest:

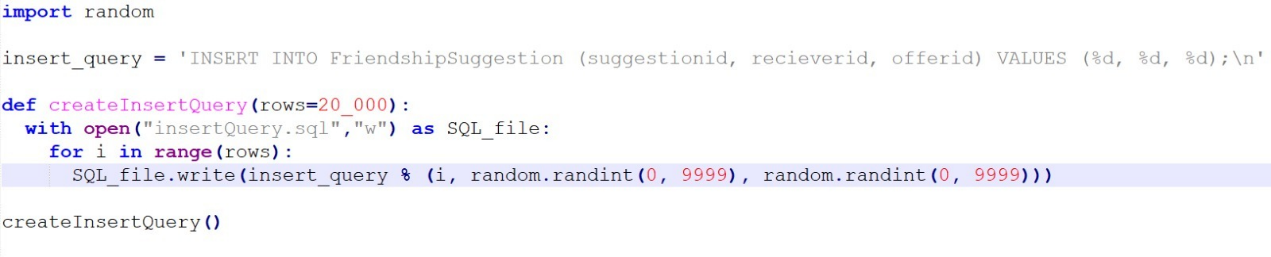


ניתן לראות כי בעמודה Approved הData הוא List(0, 1,). זוהי לא טעות. הפסיק האחרון מסמל שגם null הוא אופציה עבור עמודה זו.

* באמצעות האתר generatedata.com[[1]](#footnote-1)להכנסת ערכים לתוך Notification:



כמו שאפשר לראות, באתר זה ניתן להכין רק 100 מופעים שונים לטבלה. לאור זאת שהיינו זקוקים ל-20 אלף ערכים, כתבנו scriptקצר ב-python הלוחץ על כפתור ה-Generate באופן אוטומטי וכך קיבלנו את מספר הערכים הרצוי.

* סקריפט בפייתון המייצר קובץ עם שאילתות Insert. להלן הסקריפט:

תוצאות סקריפט זה וקובץ הפלט מ GenerateData נמצאים בגיט.

## שאילתות SQL

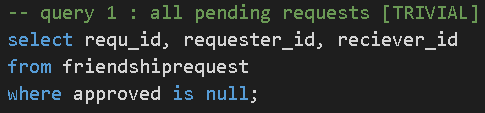
שאילתה 1 - כל הבקשות שמחכות לתשובה:

התרחיש: כחלק מהרצון של ההנהלה לדעת את מצב זמינות המשתמשים, ופנאי השימוש במערכת, ההנהלה ביקשה לדעת את כמות ההתראות שמחכות למענה בקרב המשתמשים.

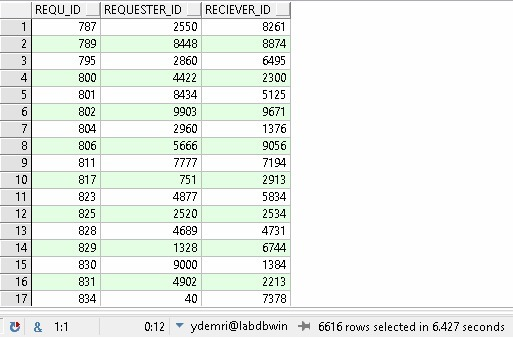
לכן כתבנו שאילתה המחזירה טבלה של כל הבקשות שמחכות למענה.

הגדרה: במערכת המידע שלנו,0/1 מסמנים התקבל/נדחה בהתאמה ו-null מסמן 'ממתין למענה'.

הקוד:



תוצאה וזמן ריצה:

התוצאה: 6616 שורות ב6.427 שניות

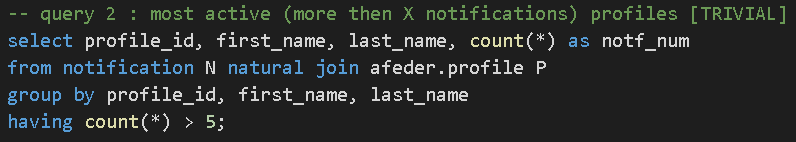
שאילתה 2 - הפרופילים הכי פעילים:

התרחיש: ההנהלה החליטה לפנק במתנה את כל המשתמשים הפעילים ביותר - בסט סטיקרים חדש.

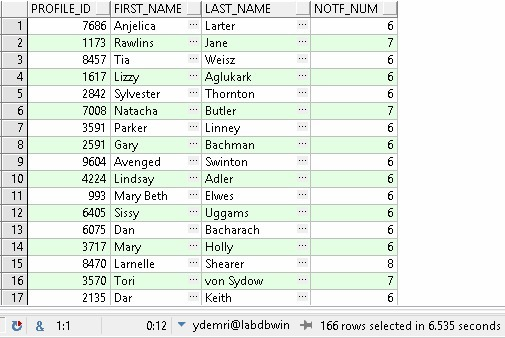
לכן ההנהלה ביקשה רשימה של כל המשתמשים הפעילים. כתבנו שאילתה המבצעת את הבקשה הנ"ל.

הגדרה: פרופיל שמקבל יותר מ-5 התראות נחשב 'משתמש פעיל'.

הקוד:



תוצאה וזמן ריצה:

התוצאה: 166 שורות ב6.535 שניות 

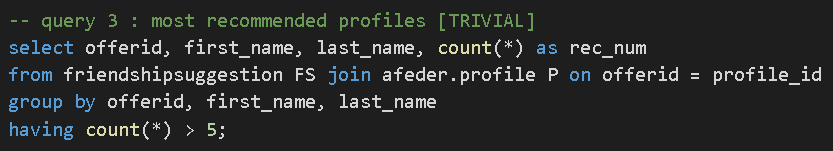
שאילתה 3 - הפרופילים המומלצים ביותר:

התרחיש: כחלק מניסוי כלים של אלגוריתם יצור ההמלצות בניסיון לשפרו במידת הצורך, ביקשו מרכז הפיתוח רשימה של כל המשתמשים המומלצים ביותר.

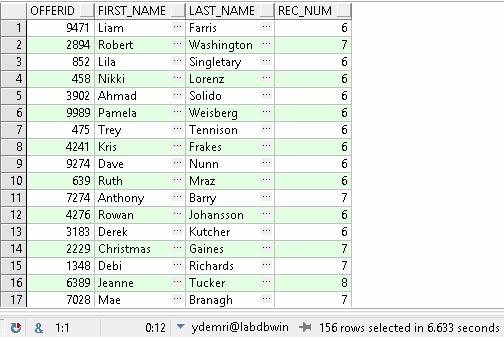
לכן כתבנו שאילתה המחזירה את הפרופילים המומלצים ביותר ע"י המערכת בקרב משתמשים שונים ברשת החברתית.

הגדרה: פרופילים שהוצעו יותר מ-5 פעמים.

הקוד:



תוצאה וזמן ריצה:

התוצאה: 156 שורות ב6.663 שניות 

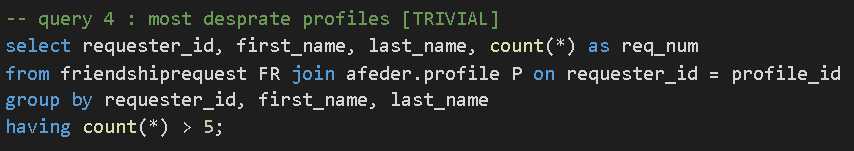
שאילתה 4 – פרופילים שנחשדים כבוטים:

התרחיש: חוקרי אבטחת המידע של הרשת החברתית מנסים להבחין בבוטים חשודים שמפיצים בקשות חברות בצורה מוגזמת על מנת ליצור פרסום בפני קהל אותו הם יכולים להשכיר ללקוחות חיצוניים.

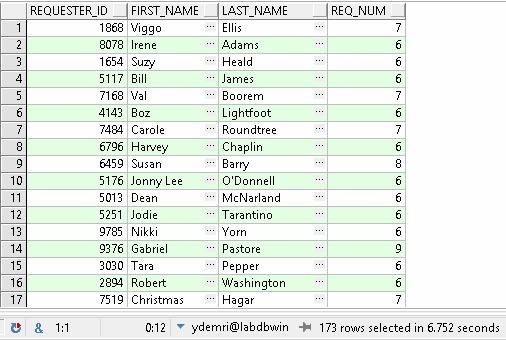
לכן כתבנו שאילתה המחזירה טבלה של כל המשתמשים החשודים. מצב שכזה יועבר לניתוח מעמיק יותר ותפיסת המפעילים.

הגדרה: משתמשים אשר שלחו יותר מ-5 בקשות.

הקוד:



תוצאה וזמן ריצה:

 התוצאה: 173 שורות ב6.752 שניות

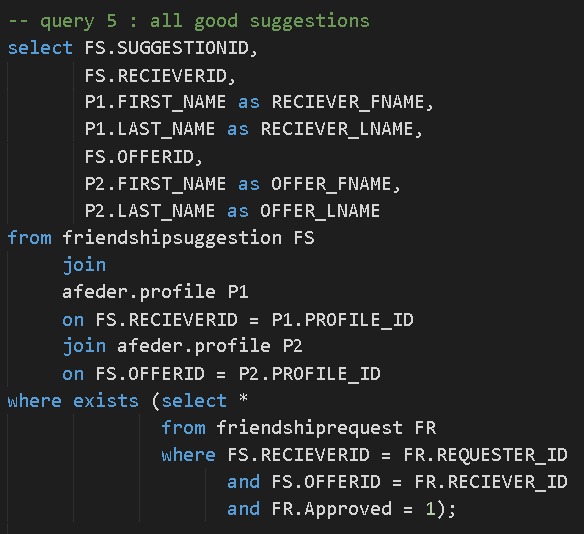
שאילתה 5 - הצעות החברות המוצלחות:

התרחיש: ההנהלה חוששת שאלגוריתם המלצת החברים איננו איכותי ומאפשר ליצור קשרים שמתקיימים בפועל. רוצים לבדוק שאכן הצעה לאדם מסוים יוצרת קשר הדדי בין שני משתמשים.

לכן כתבנו שאילתה המחזירה את כל הצעות החברות המוצלחות שנעשו ע"י המערכת.

הגדרה: מוצלח: הפרופיל שאליו נשלחה ההצעה, שלח את הבקשה לפרופיל המוצע, והנ"ל אישר את הבקשה.

הקוד:



תוצאה וזמן ריצה:

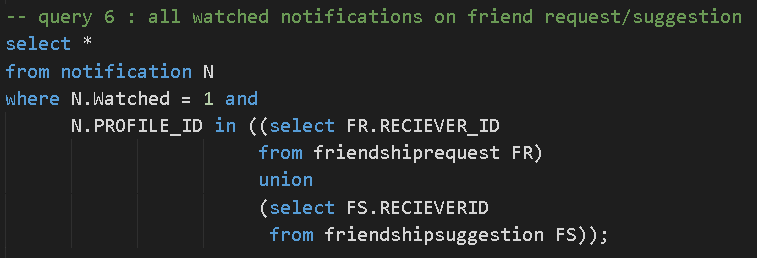
התוצאה: 240 שורות ב6.883 שניות 

שאילתה 6 - כל ההתראות על המלצות/בקשות:

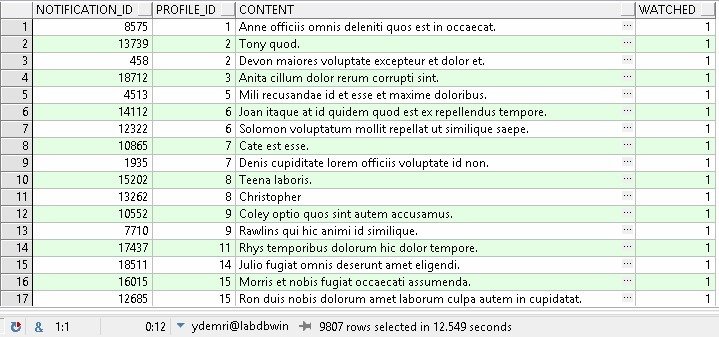
התרחיש: משתמשים רבים העלו פנייה בפני שירות הלקוחות של הרשת אודות התראות סרק לאחר שמקבלים הצעת חברות או שולחים אחת. ההנהלה ביקשה לנתח התראות סרק אלו ולהבין את מקור הבאג.

לכן כתבנו שאילתה המחזירה את כל ההתראות הקשורות להמלצות חברות או בקשות חברות.

הקוד:



תוצאה וזמן ריצה:

התוצאה: 9807 שורות ב12.549 שניות

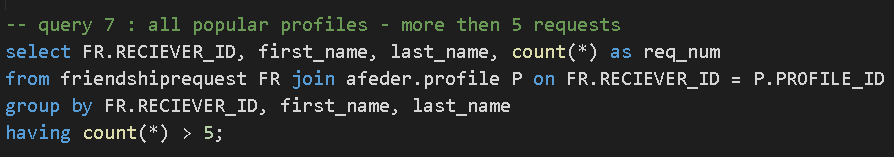
שאילתה 7 - כל הפרופילים הפופולריים:

התרחיש: הנהלת הרשת רוצה לעלות פרסומת חדשה למשוך משתמשים חדשים. לפיכך היא מעוניינת להזמין את המשתמשים הפופולריים ביותר (בדר"כ אלו מפורסמים) על מנת שיציגו את היתרונות וההנאה בשימוש ברשת החברתית.

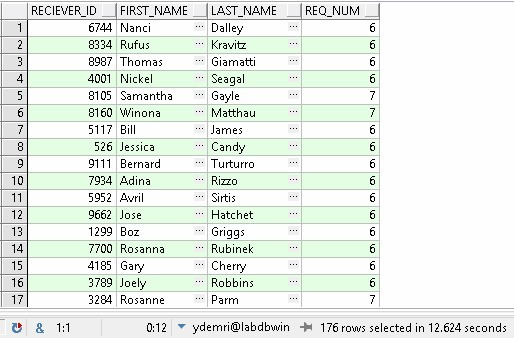
לכן כתבנו שאילתה המחזירה את כל אותם משתמשים מפורסמים.

הגדרה: פופולרי – פרופיל ששלחו אליו יותר מ-5 בקשות חברות.

הקוד:



תוצאה וזמן ריצה:

התוצאה: 176 שורות ב12.624 שניות 

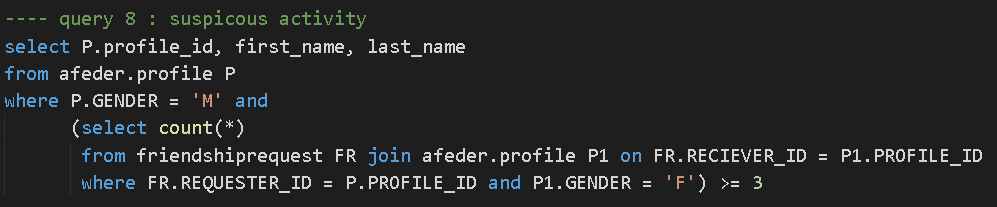
שאילתה 8 - פעילות חשודה:

התרחיש: בתקופה האחרונה עלו פניות רבות בקרב אגף אבטחת המשתמש בשירות הלקוחות של השרת אודות משתמשים שמטרידים משתמשות. ההנהלה, יחד עם המשטרה רוצים לתפוס את החשודים ולהביאם אל עונשם.

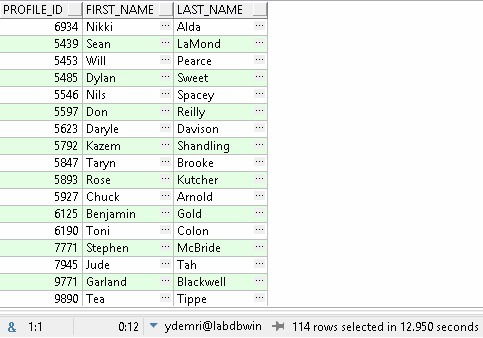
לשם כך כתבנו שאילתה המחפשת את כל המשתמשים הבעייתיים הללו ומחזירה את פרטיהם.

הגדרה: פרופיל ששלח יותר מ-3 הצעות חברות לנשים שונות.

הקוד:



תוצאה וזמן ריצה:

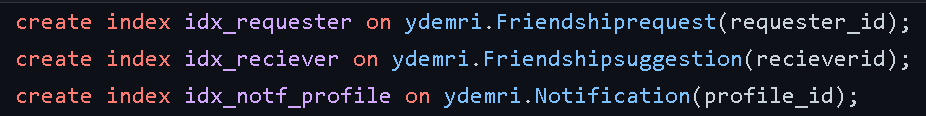
 התוצאה: 114 שורות ב12.950 שניות

## אינדקסים

אינדקסים עוזרים למצוא במהירות גדולה יותר נתונים שנשמרו בטבלאות בבסיס הנתונים. אפשר לדמות את האינדקסים כמו מראה מקום בספר. במקום שנקרא את כל הספר כדי למצוא את מה שאנחנו מחפשים נלך למראה מקום שיראה לנו את כל המקומות שבהם מוזכר הנושא שאנחנו מחפשים. השימוש באינדקסים יחסוך לנו זמן ויהפוך את תהליך החיפוש ליעיל יותר. מהבחינה הזו האינדקסים בטבלאות של ה- SQL זהים לאינדקס בספר.

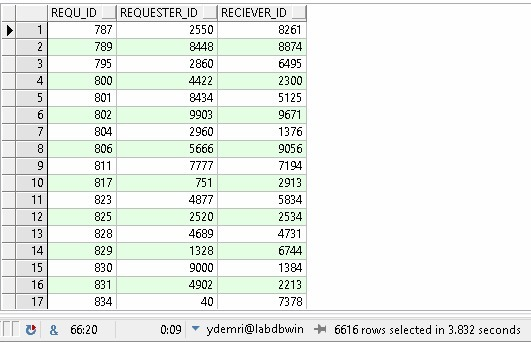
במידה ולא נגדיר אינדקס לטבלה אז בכל שאילתה על הטבלה השאילתה תגרום למעבר על כל הרשומות בטבלה עד שתמצא את כל הרשומות העונות למה שחיפשנו. כשנגדיר אינדקס מתאים אז החיפוש יהיה מהיר יותר כי הפניה לבסיס הנתונים תגרום לזה שמנוע החיפוש בבסיס הנתונים יפנה קודם לאינדקס וילך לרשומות המתאימות על פי מה שרשום באינדקס.

האינדקסים שכתבנו הם (גודלן של הטבלאות – 20K נתונים): #למה בחרנו עמודות אלו + גרף#



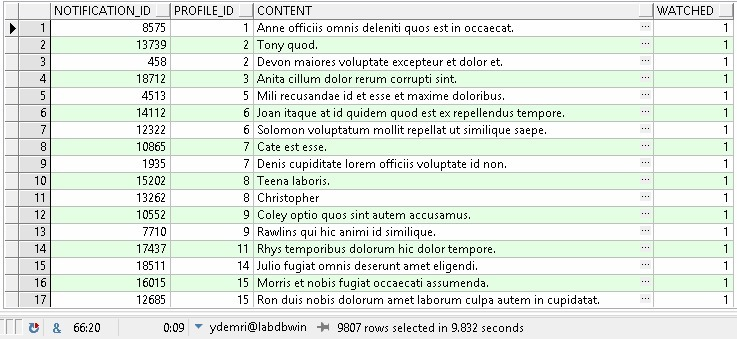
הסיבה לבחירת מאפייני האינדקסים הללו היא לאור זאת שבשאילתות שכתבנו, אלו הן העמודות המבוקשות ביותר, דהיינו, הסינון של מופעי הטבלה מתבצע על ידם. לדוג' אפשר לראות בשאילתה (8) שמתבצע סינון על בסיס המידע המגיע מעמודה requester\_id. לפיכך, חישוב מקדים באמצעות אינדקסים על עמודה זו מאפשר לצמצם את זמן החיפוש בצורה משמעותית. לכן, יצירת אינדקסים תאפשר למערכת לקצר את זמן החיפוש של שורות בבסיס המידע. כעת נציג שיפור של שלושה שאילתות מדגמיות מתוך שמונת השאילתות שציינו בחלק הקודם (עמוד 10).

1. שאילתה מס' (1):



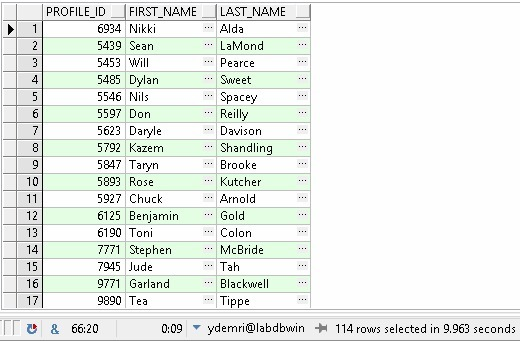
* לפני: 6.427 (sec).
* אחרי: 3.832 (sec).
* מסקנה: מדובר ב-2.595 (sec), 40.377% שיפור.

1. שאילתה מס' (6):



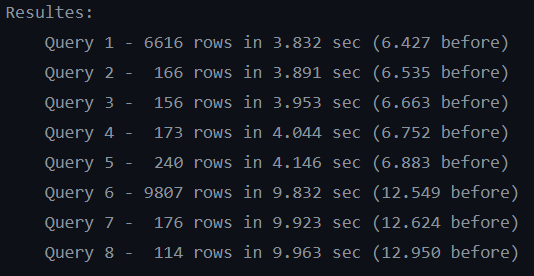
* לפני: 12.549 (sec).
* אחרי: 9.832 (sec).
* מסקנה: מדובר ב-2.717 (sec), 21.652% שיפור.

1. שאילתה מס' (8):



* לפני: 12.950 (sec).
* אחרי: 9.963 (sec).
* מסקנה: מדובר ב-2.987 (sec), 23.066% שיפור.

**להלן תקציר השיפורים בכל 8 השאילתות:**



## הרשאות

כדי שנוכל לעבוד ככיתה שלמה המפתחת בסיס נתונים כאשר כל קבוצה בונה טבלאות שונות לבסיס הנתונים ואנו צריכים לגשת לטבלאות שלהם כדי לקחת מידע אנו צריכים לקבל מיוצרי הטבלאות הרשאה כדי שנוכל להשתמש בטבלאות שלהם.

במקרה שלנו לא היינו צריכים לתת הרשאות לאף אחד מכיוון שהטבלאות שלנו מייצגות מאפיינים ייחודים המתארים חלק ספציפי במודל הרשת החברתית. מהסיבה הזו אנחנו היינו תלויים באחרים.

קיבלנו הרשאות לקריאה מהטבלאות הבאות:

* פרופיל Profile
* קבוצה חברתית SocialGroup
* עמוד עסקי BuisnessPage
* אירוע Event
* נראות Visibility
* סוג עסק BuisnessType

## Views

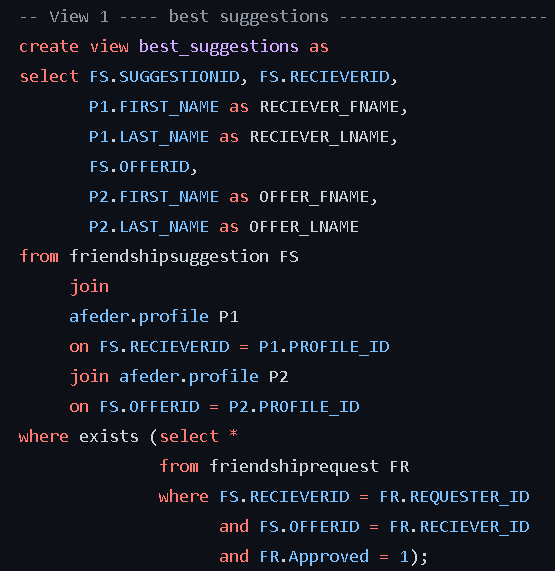
VIEWS הם טבלאות וירטואליות. VIEWS מכילים הגדרות של עמודות וסוגי מידע שאותן עמודות יכולות להכיל. ההבדל בין הטבלאות לבין ה- VIEWS הוא שבטבלאות נשמרים נתונים באופן פיזי ואילו ב- VIEWS הנתונים לא נשמרים באופן פיזי בתוכם אלא הם רק מציגים נתונים הנשמרים בטבלאות. לכן לא ניתן לעדכן או להוסיף נתונים ל- VIEWS כפי שעושים לטבלאות.

ארבעת התצוגות הבאות שנציג מאפשרות למשתמשים שונים המנהלים את התפתחות הרשת החברתית ותפקודה לקבל אינדיקציה על מגוון רחב של אלמנטים שונים.

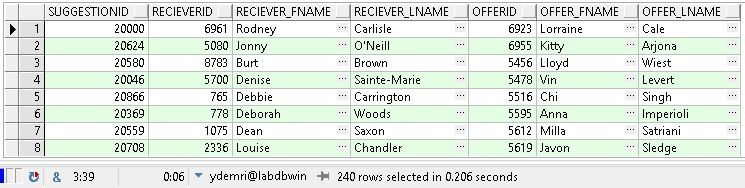
* מחלקת פיתוח – AI & Data

1. תצוגה זו מאפשרת למפתחים לצפות בשינוי פלטי האלגוריתם המקשר בין שני משתמשים שאינם מכירים על בסיסי מאפיינים ומכנים משותפים. לאור זאת שהשינויים תדירים בזמן הפיתוח, יכולת זו חיונית מאוד לעיבוד הנתונים ושיפורו של ה-suggestions המסופקים למשתמשים ע"י המערכת המתבססת על האלגוריתם לאיתור צמדים מתאימים.

הקוד:

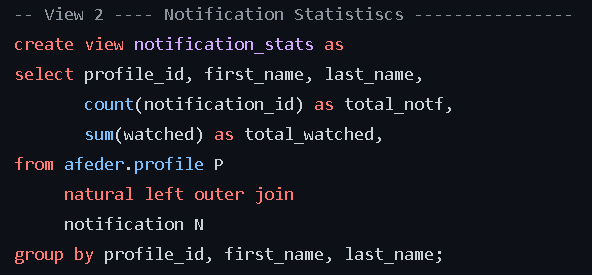


הפלט:

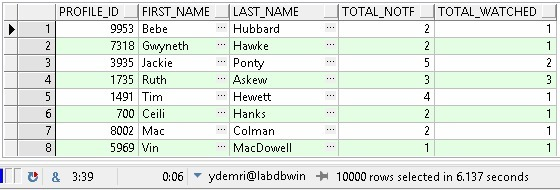


1. על מנת לאבחן את החשיפה של כלל המשתמשים להתראות המסתובבות ברשת ולאפשר התאמה אישית לתדירות השליחה, המפתחים מעוניינים, בשלבי אימון מודל ה-AI, לנתח את הקשרים שבין כמות התראות שאדם מקבל לאלו שהוא אכן השקיע זמן לצפות בהם.

הקוד:



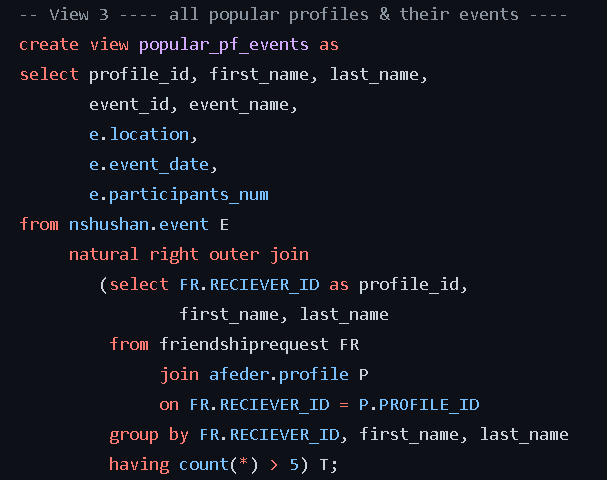
הפלט:



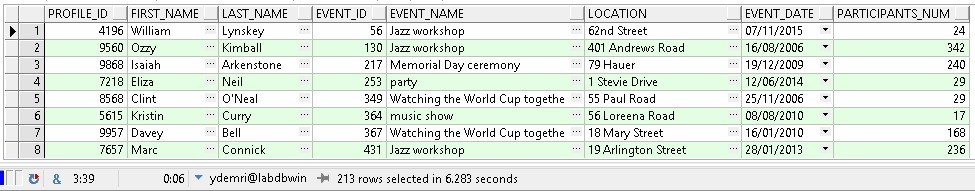
* מחלקת שיווק

1. נתח שוק גדול שפונה לרשת החברתית הן חברות הפרסום. לאור הבנה זו, מעוניינים למצוא את האזורים "החמים" ביותר בהם כדאי להכניס תכני פרסומות ומודעות. אזורים אלו בדר"כ מאופיינים **במפגשים ואירועים** המנוהלים בידי ישויות פופולרית ובהן מרוכזות מסה של משתמשים. להלן האזורים הללו ברשת החברתית.

קוד:

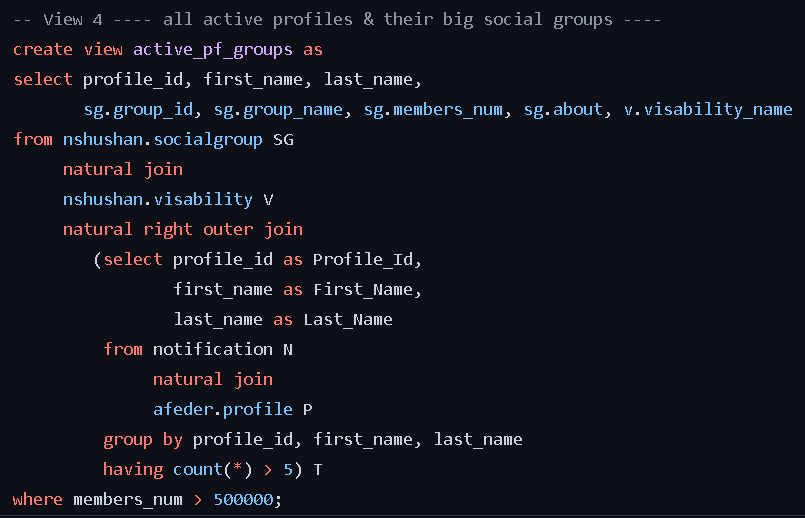


הפלט:

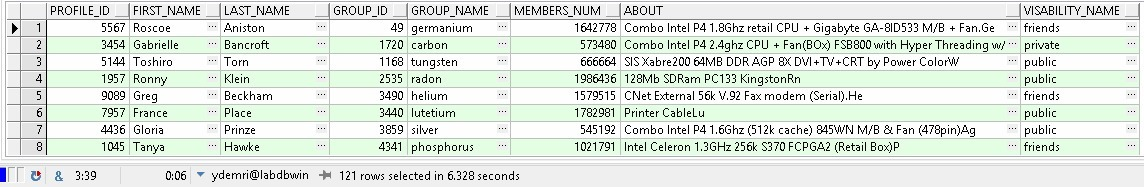


1. לאור ההצלחה בפרסום, עלייה ניכרת בחשיפה של משתמשים לתכני פרסומות ורווחים, החליטה ההנהלה על המשך פריסת הפרסומות גם **בקבוצות** משתמשים גדולות המאופיינים בישות מנהלת פעילה. להלן אזורים אלו ברשת.

הקוד:



הפלט:

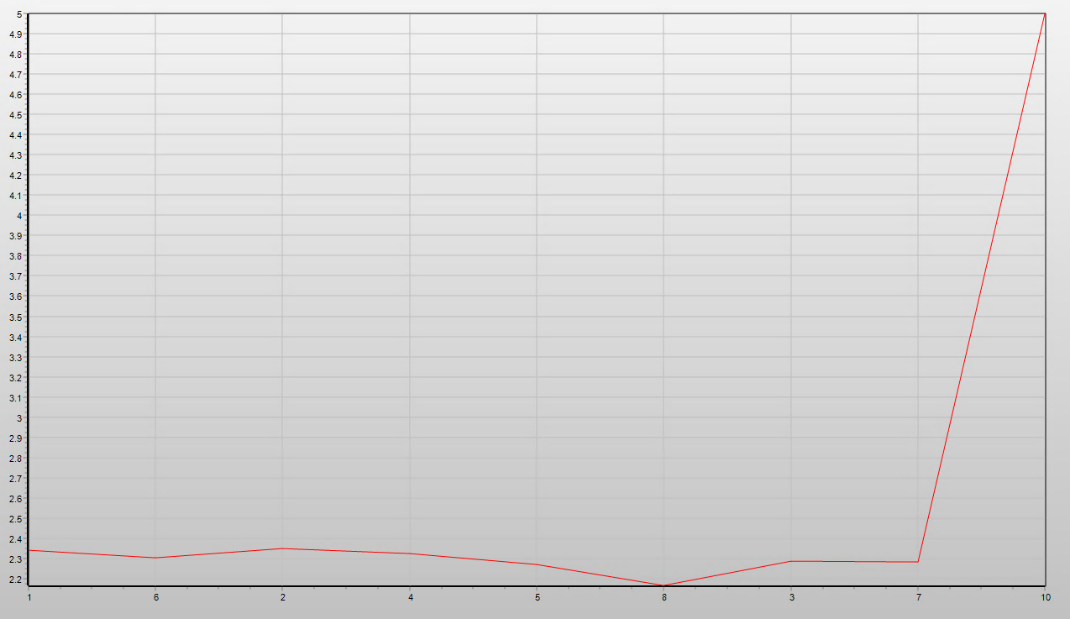


## תרשימים

לאחר שאספנו די מידע אודות החלק שלנו, נגשנו לראות כיצד מתנהגת הרשת החברתית שלנו מבחינת הדינמיקה שבין הנתונים ומסקנות מעניינות אחרות. לכן, הבאנו כאן שני תרשימים הממחישים נקודות מעין אלו.

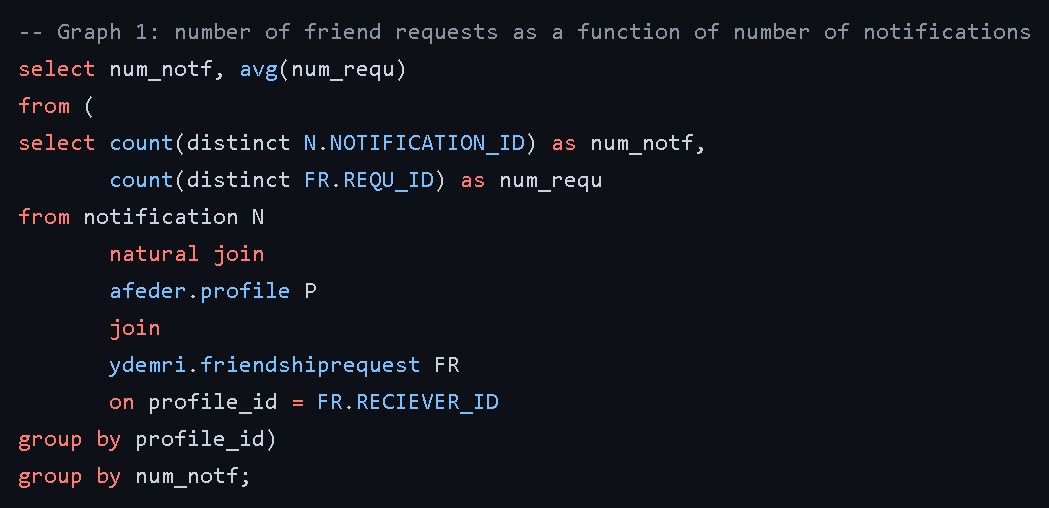
תרשים א' – כמות בקשות החברות כפונקציה של מספר ההתראות

כידוע, משתמשים שמתמצאים ברשת החברתית ומבלים בה רבות בדר"כ מגיעים למפגשים חברתיים באופן פיזי/ דרך המחשב. השערה זו רצינו לבסס באמצעות המידע שנתון לנו בבסיס הנתונים העומד לרשותנו.



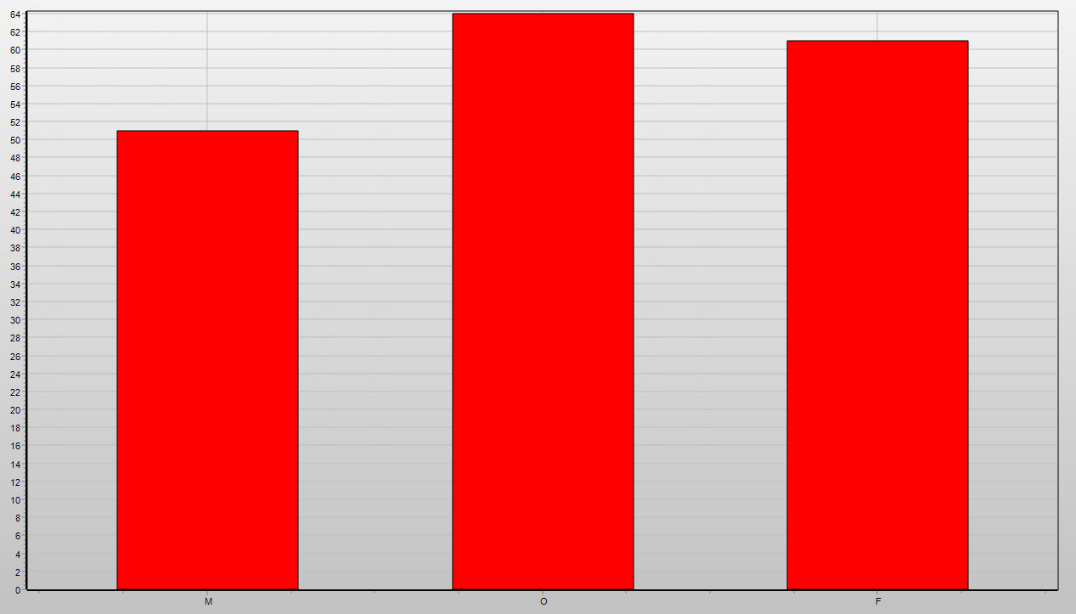
מסקנה: כפי שאפשר לראות, משתמשים בעלי תדירות התראות נמוך מאופיינים בבקשות חברות מעוטות. הערך איננו משתנה עבור מס' התראות נמוך מ-10 כפי שאפשר לראות אבל כאשר מתקרבים ל-10, אכן משתנה הערך ומשתמשים אלו דווקא מאופיינים בכמות בקשות חברות מרובות.

הקוד:



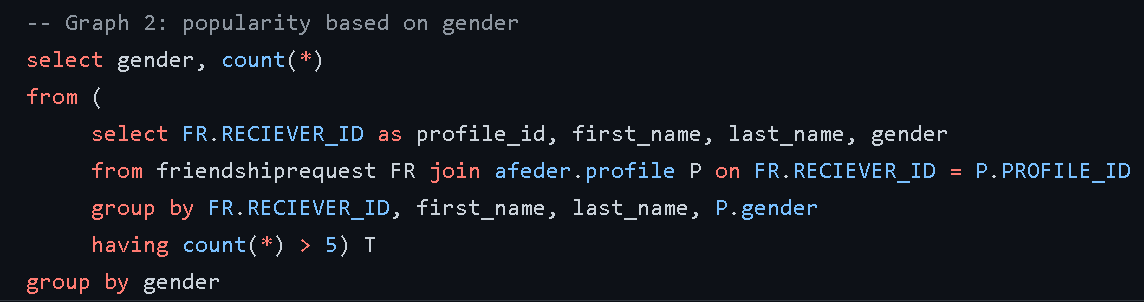
תרשים ב' – פופולריות כפונקציה של מגדר

לאור חופש הדעות והבמה הפתוחה לכל משתמש שמאפשרת הרשת החברתית, רצינו לבדוק האם קיים איזה שהוא קשר בין האופן שמגדיר עצמו בן-אדם לבין הפופולריות שלו בחברה. כפי שכבר הגדרנו, **פופולריות** של משתמש ניתנת להגדרה כסה"כ בקשות החברות שמקבל דרך קבע. להלן התוצאה:



מסקנה: כפי שאפשר לראות, אנשים הנוטים לאפיין עצמם באופן שונה [**O**ther] מהדרך הרגילה, נוטים להיות פופולריים יותר ממגדרים אחרים. אחריהם מגיעות הנשים [**F**emale], באחוז לא גדול בכלל ולבסוף – גברים [**M**ale].

הקוד:



## פונקציות ופרוצדורות

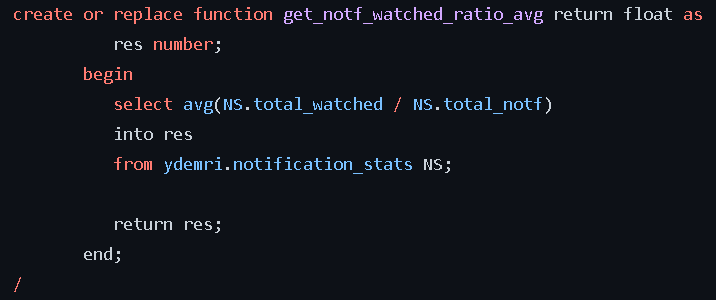
פונקציה בשפת SQL  היא צורה מיוחדת של פקודה אשר מאפשרת ביצוע שאילתה מורכבת בצורה כזו שמשלבת פעולות שונות על המידע בבסיס הנתונים כדוג' הוספה, עדכון, מחיקה ושליפה. ההבדל בין פרוצדורה לפונקציה היא העובדה שפרוצדורה שאינה מחזירה ערך בגמר הפעולה לעומת פונקציה שכן מחזירה ערך כלשהו.

תוספות אלו למערך ניהול המידע מאפשר לנו, כמפתחי ומתחזקי בסיס הנתונים לבצע מניפולציות על ערכים המוחזרים משאילתות מה שמאפשר ניתוח מעמיק יותר ושיפור השירות.

* **Functions:**

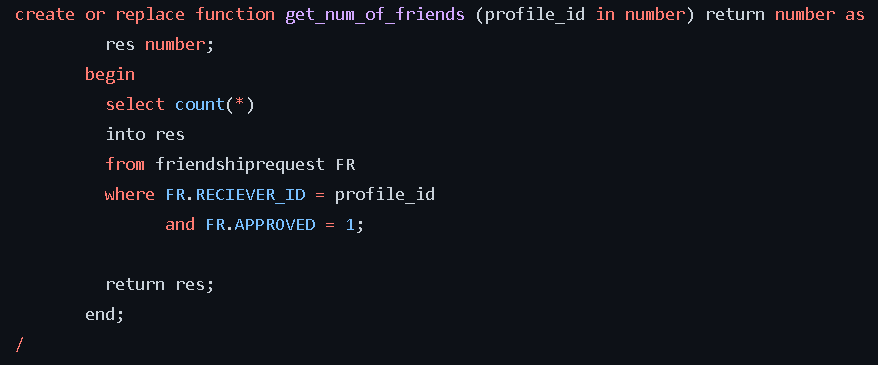
1. מפתחי ה-AI מעוניינים לקבל נתוני חשיפה להתראות על מנת לתחזק את האלגוריתם להצגת התראות. להלן פונקציה המחזירה **את היחס הממוצע בין מספר ההתראות שנקראו לסך ההתראות שהתקבלו**.

הקוד:



1. צוות ה-UI רוצה לדעת כיצד לתכנן את תצוגת החברים של המשתמש. לשם כך, עליהם לקבל את כמות החברים שיש למשתמש מסוים. פונקציה זו מקבלת מס' מזהה של משתמש ומחזירה את **מס' חבריו**.

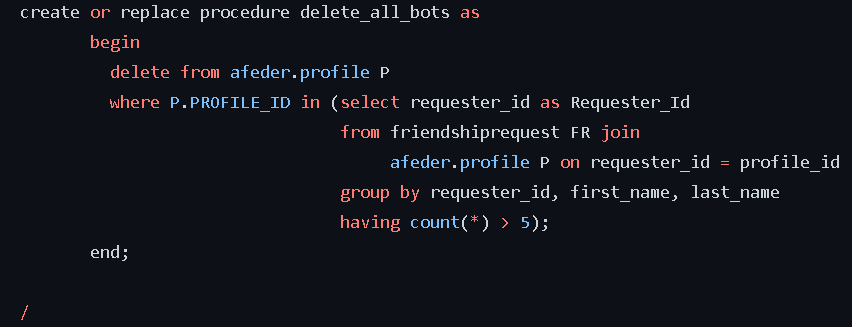
הקוד:



* **Procedures:**

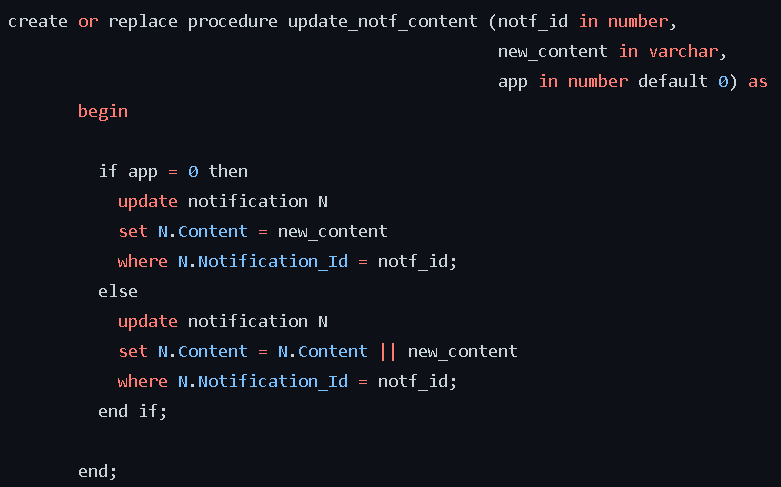
1. צוות האבטחה של הרשת מעוניין ביכולת להסיר משתמשים החשודים כבוטים (ע"פ ההגדרה שהצגנו בפרקים הקודמים). פרוצדורה זו עוברת על בסיס הנתונים **ומסירה** את כל המשתמשים ששולחים יותר מדי בקשות חברות.

הקוד:



1. מחלקת שירות משתמש הציעה להוסיף בתקופות מסוימות של השנה כותרות מיוחדות ומצחיקות להתראה ("שנה טובה", "שבת-שלום" וכו'). פרוצדורה זו עוברת על כלל ההתראות ומאפשרת החלפה של כלל ההתראות או שרשור של מחרוזת חדשה.

הקוד:



# סיכום

לאחר מימוש חלקנו בפרויקט במהלך הסמסטר, למדנו להכיר את תהליך האפיון של הטבלאות ואת חשיבותו כבסיס לפיתוח איכותי בהמשך. הבנו כיצד מתבצעת חלוקת העבודה בין משתתפים שונים וכיצד מתאפשרת אינטגרציה בין חלקי הפיתוח השונים לכדי מכלול שלם.

אנחנו מרגישים שהקורס הנ"ל תרם רבות לידע האישי שלנו בתכנון והקמת בסיס נתונים כמו גם תחזוקו לאורך זמן ומרגישים כעת בטחון רב יותר בעיסוק עם מאגרי נתונים.

כמובן, על גבי שלל היכולות שבנינו במהלך הסמסטר, ניתן כעת להוסיף ולשפר את הרשת החברתית כך שתכלול יותר אינפורמציה, לרתום את כלל שכבת הנתונים למנוע לוגי המבצע מניפולציות שונות ומיוחדות וכמו כן לממשק משתמש נוח להצגת הנתונים.

את הקבצים השונים המכילים את קוד הפרויקט כמו גם את הדו"ח הזה ניתן למצוא באתר ה-GitHub כקוד פתוח בכתובת:

<https://github.com/yakir0/minipDB>

1. https://www.generatedata.com/ [↑](#footnote-ref-1)