

רשות חברתית

דוח פרויקט

מאת:

דניאל יוחנן 322406232

&

אבי פדר 208199638

המרצה: מר יair גlodشتין



תוכן

רשות חברתיות – תיאור המערכת	3
חלק א – ישוויות ותרשיים	4
Yoshiot	4
תרשימים ERD	6
תרשימים Relational schema	7
חלק ב – יצירת הטבלאות והכנסת נתונים	8
פקודות יצירת הטבלאות	8
הכנסת הנתונים לatableות	9
חלק ג – שאלות ואינדקסים	12
שאלות	12
אינדקסים	16
חלק ד - אינטגרציה ושאלות נוספת	17
אינטגרציה	17
שאלות נוספת	18
חלק ה - גרפים וviews	20
גרפים	20
Views	22
חלק ו – פונקציות ופרוצדורות	26
פונקציות	26
פרוצדורות	27
נספח א – קוד הכנסת הנתונים	28
נספח ב – קוד של השאלות	29
נספח ג – קוד של האינדקסים	30
נספח ד – קוד של השאלות הנוספות	31
נספח ה – קוד של הגרף הראשון	32
נספח ו – קוד של הגרף השני	32
נספח ז – קוד עברו שני הVIEW הראשונים	32
נספח ח – קוד עברו שני הVIEW השניים	33
נספח ט – קוד עברו הפונקציות	33
נספח י – קוד עברו הפרוצדורות	34

רשות חברתיות – תיאור המערכת

רשות חברתיות (Social network) היא מבנה חברתי המורכב מקבוצה של מספר גורמים חברתיים (כגון אנשים או ארגונים), המקיימים ביניהם מערכת של קשרים הדדיים בעוצמות שונות. הרשות כוללת קשרים בין גורמים אלו ומתבססת על פעילותם.

רשות חברתיות מקוונת היא פלטפורמה או אתר אינטרנט המתמקד בבניה או שיקוף של קשרים חברתיים בין אנשים. באמצעותן יכול כל אדם לגלות מי החברים של חבריו, לקרוואת הפרופיל שלהם וליצור איתם קשר.

בפרויקט שלנו אנחנו נבנה בסיס נתונים של רשות חברתיות.

כל דוג קיבל מספר ישוויות שהם חלק מרשות חברתיות.

אנחנו אחראים על הישוויות:

profile	.1
post	.2
post_type	.3
post_comment	.4

מהלך הפרויקט נתמקד בעיקר בישיות שתחת אחריותנו.

בין היתר ניתן למצוא עוד ישוויות עיקריות בפרויקט כגון:

SocialGroup	.1
Friendship	.2
Like	.3
Share	.4

את רישימת הישויות המלאה ואת חלוקת העבודה ניתן למצוא [פה](#).

חלק א – יישוות ותרשימים

ישוות
Profile •
ישות זו מייצגת פרופיל משתמש

מספר זהה של המשתמש	INT	Profile_id
שם פרטי של המשתמש	VARCHAR(20)	First_name
שם משפחה של המשתמש	VARCHAR(20)	Last_name
כתובת האימייל של המשתמש	VARCHAR(30)	Email
סיסמת התחברות של המשתמש	VARCHAR(20)	Pass
מין המשתמש	CHAR(1)	Gender

Post •
ישות זו מייצגת פост של משתמש

מספר זהה של הפוסט	INT	Post_id
מספר זהה של המשתמש	INT	Profile_id
תאריך כתיבת הפוסט	DATE	Post_date
נראות הפוסט (פרט או ציבורי)	NUMERIC(10)	Visibility_code
מספר הלייקים לפוסט	INT	Number_of_likes
תוכן הפוסט	VARCHAR(500)	Post_content
סוג הפוסט (פרט או של קבוצה)	VARCHAR(20)	Post_type
אם זה פוסט של קבוצה – מספר הקבוצה	INT	Group_id

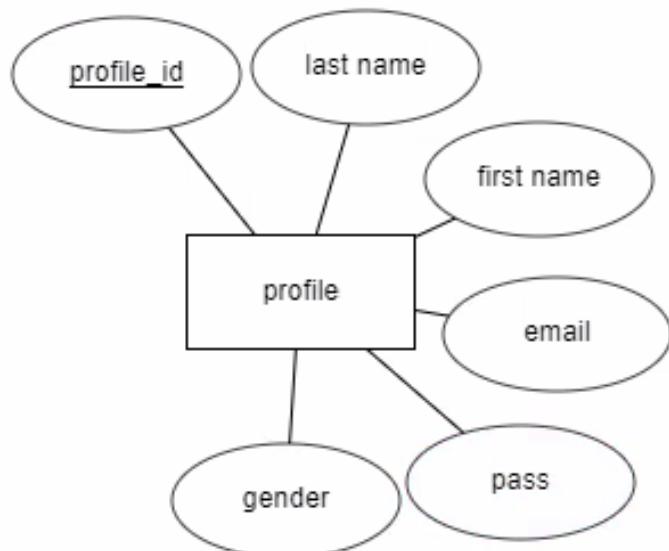
Comment •

ישות זו מייצגת תגובה על פוסטים

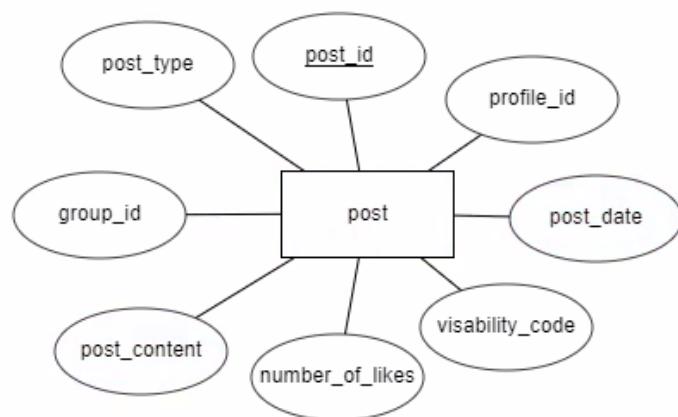
מספר מזהה של התגובה	INT	Comment_id
מספר מזהה של הפוסט	INT	Post_id
מספר מזהה של המגיב	INT	Profile_id
תוכן התגובה	VARCHAR(500)	Comment_content

תרשים ERD

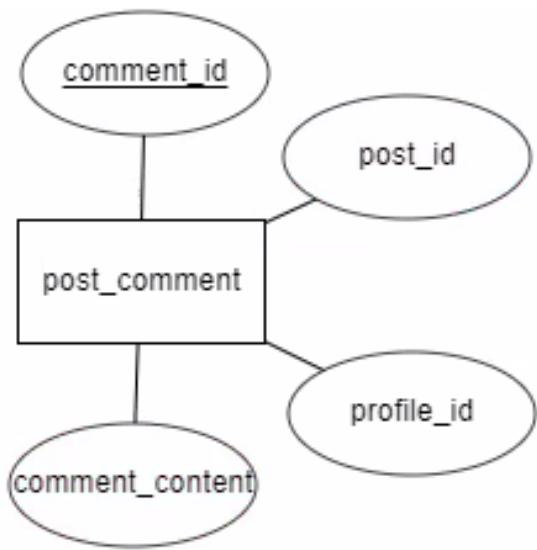
Profile •



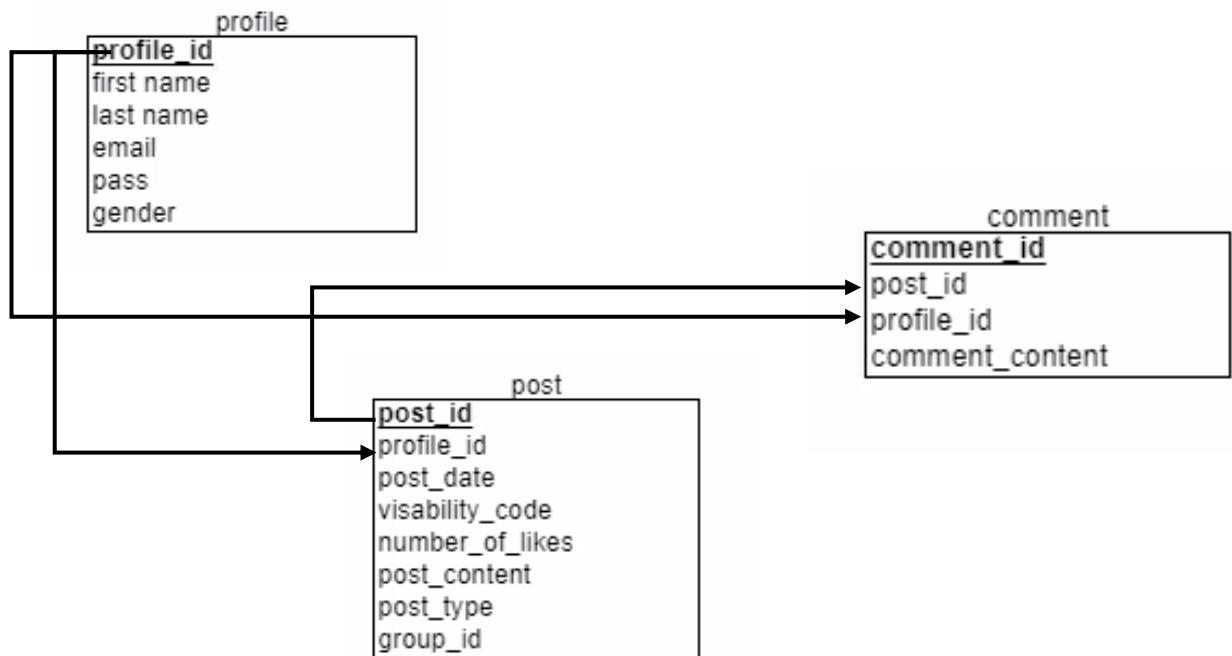
Post •



Comment •



תרשים Relational schema



חלק ב – ייצרת הטבלאות והכנסת נתונים

פקודות ייצרת הטבלאות

מכיוון שבין הבעיות מתקיימים יחסי תלות שונים, יש חשיבות לסדר ייצרת הטבלאות.
הקוד לייצרת הטבלאות הוא:

```
CREATE TABLE profile
(
    profile_id INT NOT NULL,
    first_name VARCHAR(20) NOT NULL,
    last_name VARCHAR(20) NOT NULL,
    email VARCHAR(30) NOT NULL,
    pass VARCHAR(20) NOT NULL,
    gender CHAR(1) CHECK(gender='F' or gender='M') NOT NULL,
    PRIMARY KEY (profile_id)
);

CREATE TABLE post
(
    post_id INT NOT NULL,
    profile_id INT NOT NULL,
    post_date DATE NOT NULL,
    visibility_code NUMERIC(10) NOT NULL,
    number_of_likes INT NOT NULL,
    post_content VARCHAR(500) NOT NULL,
    post_type VARCHAR(20) NOT NULL,
    group_id INT NOT NULL,
    PRIMARY KEY (post_id),
    foreign key(profile_id) references profile(profile_id)
);

CREATE TABLE post_comment
(
    comment_id INT NOT NULL,
    post_id INT NOT NULL,
    profile_id INT NOT NULL,
    comment_content VARCHAR(500) NOT NULL,
    PRIMARY KEY (comment_id),
    foreign key(profile_id) references profile(profile_id),
    foreign key(post_id) references post(post_id)
);
```

הכנסת הנתונים לatableות
(את הקוד של הנתונים ניתן לראות בנספח א' או [בגיט](#))

Profile •

ראשית, נכניס נתונים לטבלה הראשונה שלנו – profile.

לצורך כך נשתמש באתר [mockaroo](#) שמאפשר לנו ליצור נתונים ולהוריד אותם כקובץ CSC.

The screenshot shows the configuration page for generating a dataset. The 'Field Name' column lists 'profile_id', 'first_name', 'last_name', 'email', 'gender', and 'pass'. The 'Type' column includes 'Sequence' for profile_id and 'First Name', 'Last Name', 'Email Address', 'Gender (abbrev)', and 'Password' for the others. The 'Options' column contains settings like 'start at: 9001', 'step: 1', 'repeat: 1', and 'blank: 0 %' for profile_id, and 'blank: 0 %' with a sum icon for the other fields. Below the table is an 'ADD ANOTHER FIELD' button. At the bottom, there are options for '# Rows: 10000', 'Format: SQL', 'Table Name: profile', and a checked 'include CREATE TABLE' checkbox. A dropdown for 'Append Dataset:' is also present.

לאחר הורדת הנתונים, נטען את קובץ CSC לתוכנה ומלא את הטבלה:

The screenshot shows a spreadsheet table with columns: PROFILE_ID, FIRST_NAME, LAST_NAME, EMAIL, PASS, and GENDER. The data consists of 24 rows, each containing a unique ID, first name, last name, email address, password, and gender. The data is color-coded in green and white rows.

	PROFILE_ID	FIRST_NAME	LAST_NAME	EMAIL	PASS	GENDER
► 1	6997	Terrence	Gilley	terrence.gilley@yumbrands.com	887880404	F
2	6998	Richard	Alston	richard.a@httpprint.ca	229526463	O
3	6999	Leo	Sweeney	leo.sweeney@providenceservice.com	707536111	M
4	7000	Taye	Klugh	tklugh@directdata.uk	728477299	F
5	7001	Kenneth	Evanswood	kenneth.evanswood@appriiss.com	828246673	F
6	7002	Lorraine	Callow	lorraine.callow@pinnaclestaffing.lt	985576344	M
7	7003	Bridgette	Fisher	bridgette.f@healthscribe.com	601869273	F
8	6881	Gin	Calle	gin.calle@ptg.com	206373139	O
9	6882	Chalee	Belle	chalee.belle@northhighland.br	735106242	M
10	6883	Hazel	Kotto	hazel.kotto@visionarysystems.ch	529107382	F
11	6884	First	Glover	first.g@refinery.com	468508279	O
12	6885	Jodie	Rhymes	jodie.r@lms.com	786589939	M
13	6886	Marie	Griggs	mriggs@electricalsolutions.com	805261409	F
14	6887	Maggie	Sevigny	msevigny@ibfh.br	587230922	O
15	6888	Regina	Thorton	regina@ccb.com	491590791	O
16	6889	Wesley	Ledger	w.ledger@kingland.be	255143046	F
17	6890	Armand	Dukakis	armand@providentbancorp.com	189400825	M
18	6891	Kyle	Root	k.root@johnkeeler.com	824168823	M
19	6892	Chely	Dourif	chelyd@questrel.com	217946050	M
20	6893	Mary	Spector	mary.spector@venoco.be	471738661	M
21	6894	Angela	Moody	angela@tama.com	937700658	M
22	6895	Harriet	Kelly	harriet.kelly@kitba.nl	889654542	F
23	6896	Rosie	Atlas	rosie.atlas@lemproducts.it	317430747	O
24	6897	Jimmy	Starr	jimmy.starr@integramedamerica.nl	432663749	F

Post •

לצורך מלאו טבלת Post, נוצר טבלה נוספת בשם post_type ונוסיף לה שני סוגי פוסטים, פост רגיל ופוסט קבוצה:

	POST_TYPE_CODE	POST_TYPE_NAME
▶	1	NORMAL
	2	GROUP

לאחר מכן נכתוב סקריפט בפייתון כדי ליצור פקודות insert data ולמלא את ה таблицה post:

```

for x in f:
    c += 1
    x = x.replace("date", "post_date")
    x = x.replace("content", "post_content")
    counter += 1
    count += 1
    if count % 1001 == 0:
        count = 1
    y = x
    profile_id = random.randint(1, 10000)
    arr[profile_id] += 1
    post_id = arr[profile_id]
    visibility = random.randint(0, 1)
    likes = random.randint(0, 2000)
    type = random.randint(0, 1)
    group = 'NULL'
    if type == 0:
        group = random.randint(1, 500)
    lst = y.split("values")
    prop = lst[1].split(',')
    post_id_new = prop[0].replace(str(counter), str(post_id))
    profile_id_new = prop[1].replace(str(count), str(profile_id))
    date_new = change_format_date(prop[2].split(" ")[1])
    visibility_new = prop[3].replace(str(count), str(visibility))
    likes_new = prop[4].replace(str(count), str(likes))
    type_new = prop[6].replace(str(count), str(type))
    group_new = prop[7].replace(str(count), str(group))
    y = "values".join([lst[0], ' '.join([post_id_new, profile_id_new, date_new, visibility_new, likes_new, prop[5], type_new, group_new])])
    p.write(y)

```

PyCharm 2020.2.5 available
Update...

נ裏 את פקודות insert data בתוכנה ונמלא את ה таблицה בתונונים:

	POST_ID	PROFILE_ID	POST_DATE	VISIBILITY_CODE	NUMBER_OF_LIKES	POST_CONTENT	POST_TYPE_CODE	GROUP_ID
▶	1	1	2462 10/05/2020	0	563	vehicula condimentum	1	...
2	1	176	15/04/2020	1	1587	metus vitae ipsum aliquam	0	480
3	1	8102	01/12/2020	1	47	eu interdum eu tincidunt in	1	...
4	1	1649	26/01/2021	1	1026	in hac	1	...
5	1	6786	17/10/2020	1	911	habitasse platea	1	...
6	1	5831	08/03/2021	1	1068	etiam pretium iaculis justo in	0	129
7	1	9330	26/01/2021	1	543	sed lacus	0	298
8	1	5146	27/06/2020	0	1242	arcu libero rutrum ac lobortis	0	208
9	1	2729	01/08/2020	1	1870	mauris vulputate elementum nullam varius	0	480
10	1	2001	06/03/2021	0	826	ipsum aliquam non mauris morbi	0	414
11	1	3169	04/05/2020	1	1652	tempus sit amet	1	...
12	1	9326	29/04/2020	0	1237	mus etiam	1	...
13	1	734	21/05/2020	0	865	nulla nisl	0	427
14	1	9445	21/05/2020	1	1544	ultricies eu nibh quisque	1	...
15	1	7248	30/10/2020	0	1454	lacinia sapien quis libero nullam	1	...
16	1	9143	31/03/2021	0	1599	diam vitae	1	...
17	1	1818	29/03/2021	1	1791	proin leo odio porttitor id	1	...
18	1	9947	19/02/2021	1	1533	nisi at nibh in	0	401
19	1	6615	23/09/2020	0	744	magna bibendum imperdiet nullam orci	1	...
20	2	8253	26/12/2020	1	813	porta volutpat erat quisque	0	361
21	1	885	11/05/2020	0	1184	donec quis orci	1	...
22	1	9232	29/07/2020	1	913	sapien cum	0	455
23	1	3812	23/12/2020	1	1087	faucibus orci luctus et ultrices	0	108
24	1	9480	08/07/2020	0	1800	dolor sit amet consectetur adipiscing	1	...

Comment •

בטבלה הצעת משתמש בSQL שבדואן כדי להוסיף נתונים.

לאחר הרצת הדרישה נקבל טבלה מלאה בתוצאות:

	COMMENT_ID	POST_ID	PROFILE_ID	COMMENT_CONTENT
►	1	1	1	1125 Morbi porttitor lorem id ligula. Suspendisse ornare consequat lectus. In e...
	2	1	1	8901 Mauris sit amet eros. Suspendisse accumsan tortor quis turpis. Sed ante. ...
	3	1	1	7938 Vivamus metus arcu, adipiscing molestie, hendrerit at, vulputate vitae, ni...
	4	1	2	2836 Aenean auctor gravida sem. Praesent id massa id nisl venenatis lacinia. A...
	5	1	1	5473 Nulla ac enim. In tempor, turpis nec euismod scelerisque, quam turpis ac...
	6	1	1	9705 Maecenas tristique, est et tempus semper, est quam pharetra magna, ac ...
	7	1	1	9207 In hac habitasse platea dictumst. Etiam faucibus cursus urna. Ut tellus. N...
	8	1	1	5184 Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere i...
	9	1	1	8969 Vivamus metus arcu, adipiscing molestie, hendrerit at, vulputate vitae, ni...
	10	1	1	1167 Vivamus metus arcu, adipiscing molestie, hendrerit at, vulputate vitae, ni...
	11	1	1	6488 Proin interdum mauris non ligula pellentesque ultrices. Phasellus id sapi...
	12	1	1	5313 Nunc rhoncus dui vel sem. Sed sagittis. Nam congue, risus semper porta ...
	13	1	1	9146 Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ...
	14	1	1	232 Donec ut dolor. Morbi vel lectus in quam fringilla rhoncus. Mauris enim ...
	15	1	1	8985 Nunc rhoncus dui vel sem. Sed sagittis. Nam congue, risus semper porta ...
	16	1	1	9001 Suspendisse potenti. Nullam porttitor lacus at turpis. Donec posuere met...
	17	1	1	1402 Phasellus id sapien in sapien iaculis congue.
	18	1	1	4649 Aenean lectus. Pellentesque eget nunc. Donec quis orci eget orci vehicul...
	19	1	1	4245 Phasellus sit amet erat. Nulla tempus. Vivamus in felis eu sapien cursus v...
	20	1	1	8210 Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ...
	21	1	1	3378 Nulla ut erat id mauris vulputate elementum. Nullam varius. Nulla facilis ...
	22	1	1	5689 Proin eu mi. Nulla ac enim. In tempor, turpis nec euismod scelerisque, qui...
	23	1	1	9944 In blandit ultrices enim. Lorem ipsum dolor sit amet, consectetur adipis...
	24	1	1	8679 In sagittis dui vel nisl. Duis ac nibh. Fusce lacus purus, aliquet at, feugiat ...

חלק ג – שאלות ואינדקסים

שאלות

נציג 8 שאלות מתקדמות על הקוד שלנו.
בכל שאלה נסביר מה השאלה מחזירה, מה מטרת השאלה וכמה ערכים קיבלנו בתוצאה.
(את הקוד של השאלות ניתן לראות בנספח ב או [בגיט](#))

שאילה 1:

- מה השאלה נותנת:

השאלה מחזירה טבלה שפרטת לפי כל מין את מספר הליקים המקורי לפוסט, מספר הליקים הממוצע לפוסט, וכמה פוסטים יש מפרופילים מאותם מין.

כדי שנוכל ללמוד על השוק לפי פולחים שונים.

- כמה שורות יש בתוצאה:

3 שורות.

	GENDER	Avg(Number_of_Likes)	Max(Number_of_Likes)	Min(Number_of_Likes)	Count(*)
► 1	M	1007.81987000929	2000	0	3231
2	O	991.82283481219	2000	2	3381
3	F	1012.82821723731	2000	0	3388

שאילה 2:

- מה השאלה נותנת:

משתמשים שיש להם פוסט שבו מספר הליקים גבוה מהממוצע.

מכיוון שנרצה למצאו ולאחר מכן חדשם.

- כמה שורות יש בתוצאה:

5001 שורות.

	PROFILE_ID	FIRST_NAME	LAST_NAME	POST_ID	NUMBER_OF_LIKES
► 1	7000	Taye	Klugh	...	1416
2	7001	Kenneth	Evanswood	...	1762
3	7001	Kenneth	Evanswood	...	1429
4	7002	Lorraine	Callow	...	1552
5	6881	Gin	Calle	...	1660
6	6883	Hazel	Kotto	...	1765
7	6884	First	Glover	...	1947
8	6892	Chely	Dourif	...	1983

שאילתה 3:

- מה השאלתה נותנת:

פרופילים שיש להם פוטיטים עם מספר ליקויים גבוה מה ממוצע וגם מספר תגיות שהגיבו להם גבוה מה ממוצע.

מכיון שנדיצה למצוא פרופילים פופולריים מאוד.

- כמה שורות יש בתוצאה:
3956 שורות.

	PROFILE_ID	FIRST_NAME	LAST_NAME	GENDER
▶ 1	3	Halle	Rydell	F
2	10	Mitchell	Zellweger	F
3	11	Adrien	Love	F
4	12	Franz	Tsettos	O
5	14	Bonnie	Russell	F
...

שאילתה 4:

- מה השאלתה נותנת:

פרופילים שהגיבו מספר תגיות גבוה מה ממוצע.

מכיון שנדיצה למצוא פרופילים פעילים מאוד.

- כמה שורות יש בתוצאה:
416 שורות.

	PROFILE_ID	FIRST_NAME	LAST_NAME	GENDER
▶ 1	5489	Ramsey	Rooker	O
2	5785	Denise	Tennison	M
3	5865	Carole	Atkins	O
4	6112	Emmylou	McCann	O
5	6193	Taye	McIntyre	M
6	8090	Ann	Sevigny	M
7	8752	Rod	McGriff	O
...	9144	Toni	Sladee	M

שאילתה 5:

- מה השאלה נותנת:

מתוך האנשים שלא הגיעו על אף פוטו, אנחנו שולפים את בעלי מספר הלייקים הרב ביותר ואת מגדרם.

זאת על מנת לנתח את אופי נתינת הלייקים לפרופילים שלא מגיבים ולא משתתפים פעילים בפלטפורמה.

- כמה שורות יש בתוצאה:

5 שורות.

	PROFILE_ID	POST_ID	NUMBER_OF_LIKES
▶	1	7406	1
	2	8202	1
	3	8497	3
	4	3325	3
	5	9832	2

שאילתה 6:

- מה השאלה נותנת:

פרטים על פרופילים מוביילים בעלי ממוצע לייקים בגודל מ 1500.

מכיוון ש00 1500 הינו כ15% מכלל הפרופילים בפלטפורמה וככה נמצא פרופילים בעלי עניין.

- כמה שורות יש בתוצאה:

263 שורות.

	PROFILE_ID	FIRST_NAME	LAST_NAME	AVG(NUMBER_OF_LIKES)	NUM_OF_POST
▶	1	6978	Robert	... Tah ...	1950 2
	2	5565	Seann	... Walken ...	1526.3333333333333 3
	3	5755	Pablo	... Merchant ...	1581.5 2
	4	5775	Jet	... Saucedo ...	1607.6666666666667 3
	5	5913	Antonio	... Salt ...	1730 2
	6	6184	Jake	... Pollak ...	1967 2
	7	9719	Larenz	... Allan ...	1568.5 2
	8	7524	Meryl	... Melvin ...	1804 2
	9	4938	Patricia	... Hatchet ...	1857.5 2

שאילתה 7:

- מה השאלה נותנת:

השאילתה מחייבת פרטים על משתמשים אשר פרסמו פוסטים מכל הסוגים. זאת על מנת לראות ולנתח את השימוש של משתמשים בסוגי פוסטים שונים.

- כמה שורות יש בתוצאה:

1590 שורות.

	PROFILE_ID	FIRST_NAME	LAST_NAME	GENDER
► 1	7001	Kenneth	Evanswood	F
2	6883	Hazel	Kotto	F
3	6897	Jimmy	Starr	F
4	6900	Vonda	Idol	O
5	6902	Nick	Astin	F
6	6906	Madeline	Whitman	O
7	6923	Lorraine	Cale	F
8	6935	Ceili	Tyson	F
9	6956	Tammie	Melville	M

שאילתה 8:

- מה השאלה נותנת:

פרטים על משתמשים שמגיבים על פוסטים "רגעים".

על מנת לקבל אינפורמציה נוספת המשתמשים.

- כמה שורות יש בתוצאה:

592 שורות.

	PROFILE_ID	FIRST_NAME	LAST_NAME	MAX(P.NUMBER_OF_LIKES)
► 1	5532	Phoebe	Sutherland	1576
2	5613	Murray	Irving	142
3	5755	Pablo	Merchant	1950
4	5851	Kylie	Sinatra	930
5	5868	Mindy	Gold	1683
6	6009	Chalee	Lynn	1599
7	6077	Herbie	Lerner	1827
8	6128	Victor	Turturro	974
9	6193	Taye	McIntyre	682

אינדקסים

ע"מ לשפר את זמן הריצה, נוצר אינדקסים לטבלאות.
אינדקס הוא טבלה המאפשרת גישה ישירה לרשומות על-פי מפתח, מבלי שייהי צורך לסרוק את כל הנתונים, באמצעות התאמת בין המפתח של הרשימה לבין כתובתה בזיכרון.

המפתח של הרשימה יכול להיות שדה כלשהו בתוכנה המזהה אותה באופן יחיד ומאפשר את אחיזורה (לדוגמא מספר זהות ברשימה העוסקת באדם, מספר קטלוגי ברשימה העוסקת בפרט במלאי וכדומה).

ע"י יצירת אינדקסים, ניתן לשפר את זמן הריצה של השאלות שלנו.
נוצר אינדקסים עבור עמודות שימושיות בשאלות וນבחן את שינוי הזמן:
(את הקוד של האינדקסים ניתן לראות בנספח ג או [בגיט](#))

```
create index index1 ON AFEDER.PROFILE(profile_id,first_name,last_name,gender);  
create index index2 ON AFEDER.POST(post_id,number_of_likes);  
create index index3 ON AFEDER.POST_COMMENT(comment_id,post_id,profile_id);
```

נתמקד רק בשאלות זמן הריצה שלהם היה מעיל לשניה.

שאילה מס' 8	זמן ריצה לפני יצירת האינדקס	זמן ריצה לאחר יצירת האינדקס	אחוז השינוי בזמן הריצה	האינדקס
רעה של 4.2 שניות כלומר תוספת של 175 אחוז בזמן	6.6	2.4	2	
רעה של 0.2 שניות כלומר תוספת של 4 אחוז בזמן	4.4	4.2	3	
טבה של 1.6 שניות כלומר חיסכון של 65 אחוז בזמן	2.8	4.4	4	
טבה של 1.8 שניות כלומר חיסכון של 40 אחוז בזמן	2.7	4.5	6	
טבה של 2.7 שניות כלומר חיסכון של 57 אחוז בזמן	2.5	5.7	7	
טבה של 4 שניות כלומר חיסכון של 70 אחוז בזמן	1.7	5.7	8	

מכיוון שאינדקס הוא אובייקט בפני עצמו, גם לו זמן ריצה, כפי שניתן לראות לעיתים נקלט הרעה בזמן הריצה.

חלק ד - אינטגרציה ושאלות נוספות

אינטגרציה

ע"מ להמשיך בפרויקט, علينا לבצע אינטגרציה לטללאות של זוגות נוספים ולתת לזוגות אחרים גישה לטללאות שלנו.

נוסף למספר זוגות שפנו אליו גישה לטללאות.

הטבלה PROFILE היא הטבלה המבקשת ביותר מצד זוגות אחרים, מכיוון שהיא מכילה את המשתמשים של הרשות החברתית ומעלה מtabesse הפרויקט:

AFEDER.PROFILE@LABDBWIN										
Grantee	Select	Insert	Update	Delete	References	Alter	Index	Read	Debug	
ASEBBAG	Yes				Yes					
MZUCKERB	Yes				Yes					
NSHUSHAN	Yes				Yes					
YDEMRI	Yes				Yes					

גם לטללה POST קיבלנו בקשה לתת הרשאות:

AFEDER.POST@LABDBWIN										
Grantee	Select	Insert	Update	Delete	References	Alter	Index	Read	Debug	
MZUCKERB	Yes				Yes					

כפי שניתן לראות, נתנו גישה רק לקריאת הנתונים מהטללאות אך לא לבצע שינויים בהם. זאת ע"מ לשמר על עקרון separate privilege, כלומר כל אחד יקבל רק את הרשות שהוא צריך ולא יותר.

אנחנו ביקשנו מזוגות אחרים גישה לטללאות הבאות:

Share .1

Friends .2

Tag .3

על טללאות אלו נכתב שאילתות נוספות.

שאליות נוספות

(את הקוד של השאלות הנוספות ניתן לראות בנספח ד או [בגיט](#))

שאלה 1:

מה השאלה נוتنת:

השאילה מחייבת עבור כל פרופיל, כמה הצעות חברות הוא מקבל שלחו לו בקשת חברות.

השאילה תשתמש את המערכת על מנת לציג חסיפה מיותרת ולמקד את ההצעות למשתמש.

	RECIPIERID	NUM_OF_SUGGESTION
1596	2104	1
1525	2016	1
5898	7807	1
134	183	1
2576	3390	1
5030	6636	0
5021	6623	0
5029	6633	0
5031	6637	0
5019	6621	0

שאלה 2:

מה השאלה נוتنת:

השאילה מחייבת את הפרופילים, שלהם יש הצעות חברות למשתמשים מסוימים כאשר הפרופיל מופיע לאוותם ההצעות גם בהצעות חברות (וכמה אלה יש לו).

(של אחד מופיע לשני בהצעת חברות)

השאילה תשתמש את המערכת על מנת לבדוק את רמת הדיק של ההצעות ולמקד את ההצעות למשתמש.

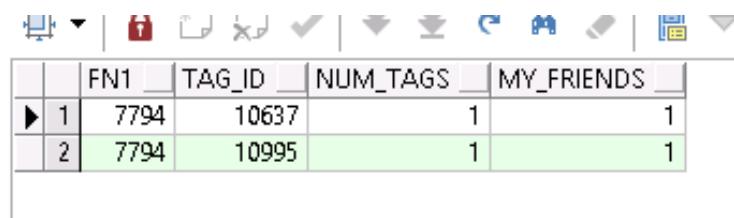
	RECIPIERID	COMMON_SUGGESTION
► 1	9417	1
2	7421	1
3	8636	1

שאילתה 3:

מה השאלתה נותנת:

השאילתה מחזירה את הפרופילים, שתויגו בתמונה שבה כל שאר המתוויגים נמצאים ברשימת החברים שלהם.

השאילתה תשתמש המערכת לזהות מעגלי חברות ולשפר את דיק הוצאות החברות למשתמשים.



	FN1	TAG_ID	NUM_TAGS	MY_FRIENDS
▶	1	7794	10637	1
	2	7794	10995	1

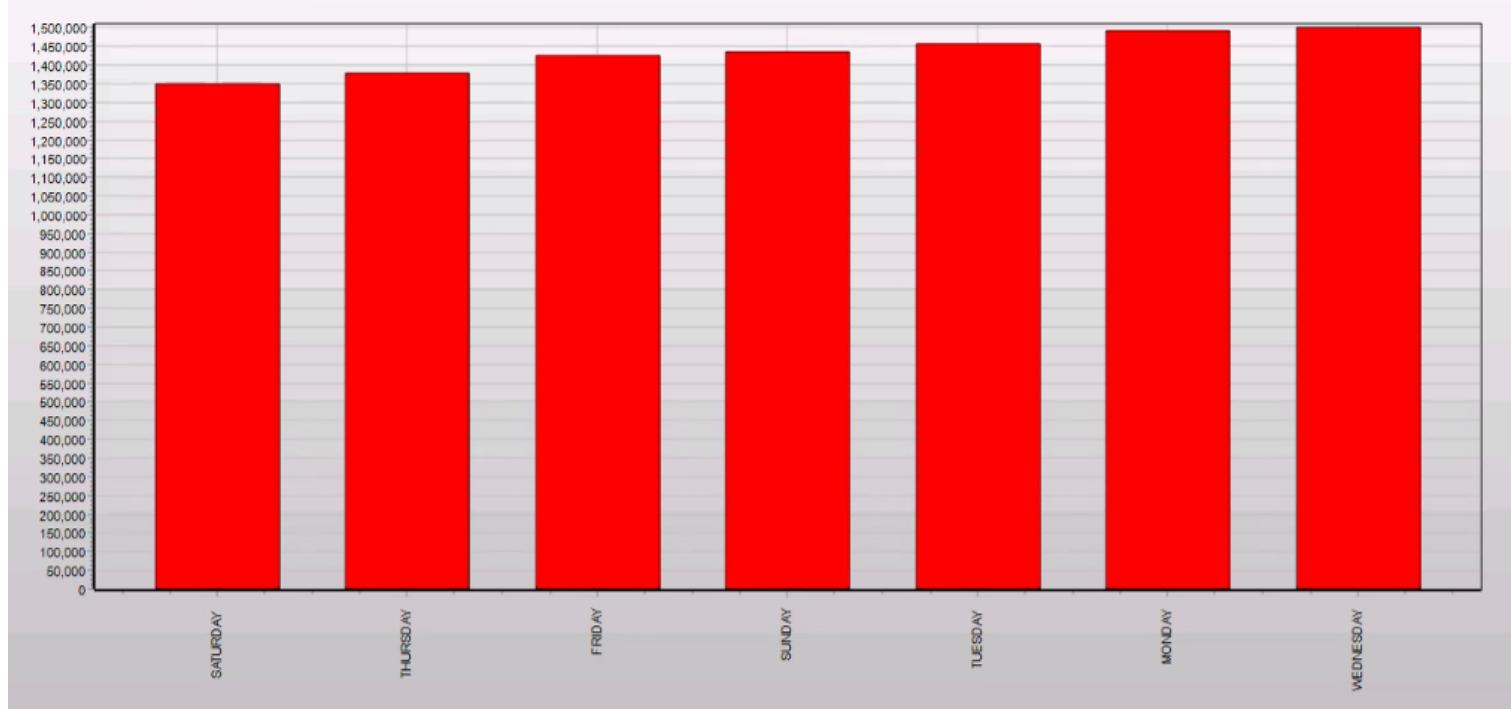
חלק ה - גרפים וviews

גרפים

1. הגרף הראשון שלנו יתמקד בליליקים לפווטים. יציג את מספר הליליקים שניתנו בפלטפורמה לפווטים בחלוקת לפי ימים.

בציר Y נראה את מספר הליליקים ובציר X את יום פרסום הפווט:

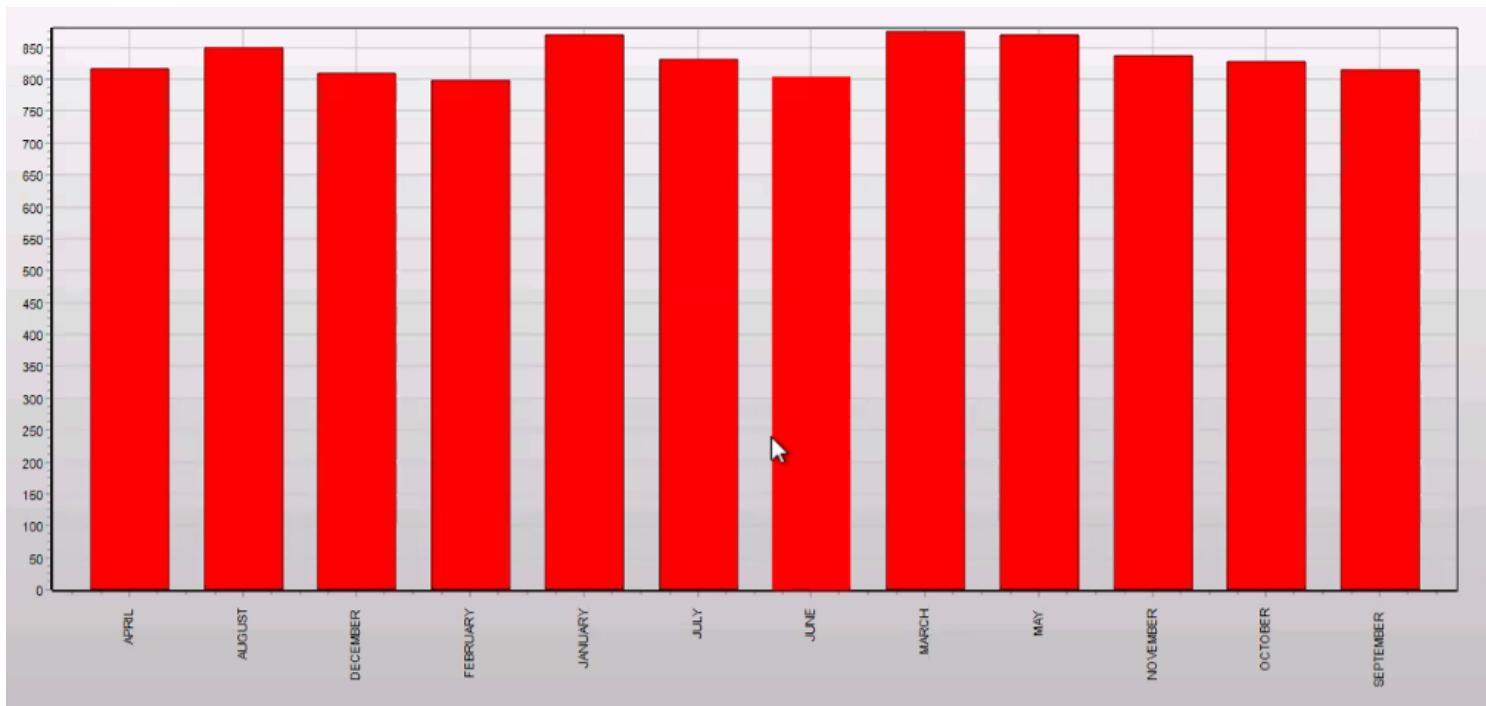
(את הקוד של הגרף ניתן לראות בנספח ה או [בגיט](#))



1. הגרף השני יתמקד בפוסטים שעלו בפלטפורמה ויציג לנו כמה פוסטיםים עלו בכל חודש.

בציר Y נראה את כמות הפוסטים שעלו ובציר X את החודש.

(את הקוד של הגרף ניתן לראות בנספח 1 או [בקיט](#))



Views

זו טבלה דינמית על בסיס הנתונים, שמכילה שאלתה שרצה מחדש כל פעם שה View נקרא.
ניתור 4 Views, עבור שני סוגים משתמשים - 2 עבור כל משתמש.

Views עבור משתמש רגיל

(את הקוד של הViews ניתן לראות בנספח 2 או [בגיט](#))

- ניתור View בשם trand_profile שייתן למשתמשים שיש להם פוסט שבו מספר הליקים גבוה מההמוצע.
כל משתמש בפלטפורמה יוכל לראות מה הטרנדים חדשים.

	PROFILE_ID	FIRST_NAME	LAST_NAME	POST_ID	NUMBER_OF_LIKES
▶	1	3 Halle	Rydell	4	1728
	2	3 Halle	Rydell	2	1407
	3	3 Halle	Rydell	1	1245
	4	10 Mitchell	Zellweger	2	1552
	5	11 Adrien	Love	1	1717
	6	12 Franz	Tsettos	1	1905
	7	14 Bonnie	Russell	1	1019
	8	16 Mira	Norton	1	1233
	9	19 Brad	Thornton	2	1172
	10	22 Hal	Malkovich	2	1563
	11	22 Hal	Malkovich	1	1300
	12	23 Bill	Galecki	1	1056
	13	31 Bobbi	Gore	1	1858
	14	35 Peabo	Osment	1	1828
	15	37 Mekhi	Spears	1	1119
	16	38 Colin	Conley	1	1609
	17	39 Merle	Russell	2	1602
	18	39 Merle	Russell	1	1588
	19	44 Cliff	Hall	1	1100
	20	46 Clea	Shaw	2	1436

```
create or replace view trand_profile as
(
select profile_id, first_name, last_name, post_id, number_of_likes
from post natural join profile
where number_of_likes >= (
    select avg(number_of_likes)
    from post
)
);

select * from trand_profile ;
```

2. ניצור View בשם active_profile שייתן משתמשים שהגיבו מספר תגובות גבוהה מה ממוצע. כל משתמש בפלטפורמה יוכל לראות מי הם הפרופילים הפעילים והטרנדים.

	PROFILE_ID	FIRST_NAME	LAST_NAME	GENDER
▶ 1	383	Phil	Brody	O
2	423	Juliana	Horizon	M
3	546	Daryle	Fisher	O
4	668	Rich	Porter	F
5	743	Vern	Heslov	O
6	908	Wade	Stanley	O
7	1021	Dustin	Bacharach	O
8	1036	Kitty	Stuermer	M
9	1504	Sarah	Marie	O
10	1536	Davey	Collette	O
11	1550	Franz	Folds	F
12	2532	Jody	Day	F
13	3003	Diamond	Ferry	F
14	3041	Isaac	Tillis	F
15	3090	Rutger	Margulies	M
16	3106	Radney	Uggams	O
17	3295	Rodney	Arthur	O
18	3574	Murray	Bragg	M
19	3715	Oro	Chaplin	O
20	3733	Michelle	Bracco	O

```

create view active_profile as
(
select profile_id, first_name, last_name, gender
from profile natural join post_comment
group by profile_id, first_name, last_name, gender
having count(*) > (select avg(c) from (
                                select profile_id, count(*) as c
                                from post_comment
                                group by profile_id
                                ) )
);
select * from active_profile ;

```

(את הקוד של הview View ניתן לראות בסוף ח או [בגיט](#))

1. ניצור View בשם active_friendship שייתן כמה חברים יש לכל משתמש במערכת. באמצעות View זה והview הבא מנהל מערכת יוכל למצוא חברותות לא פעילות.

	USER_ID	FRIENDS
▶	1	1
	2	22
	3	25
	4	30
	5	34
	6	42
	7	43
	8	51
	9	54
	10	57
	11	78
	12	83
	13	87
	14	91
	15	122

```
create view active_friendship as(
select friend_id1 as user_id, count(*) as friends
from bmarcus.friends
group by friend_id1

);

select * from active_friendship;
```

2. ניצור View בשם friendship_request שיתן כמה בקשות חברות יש לכל משתמש במערכת.
באמצעות View זה והView הקודם מנהל מערכת יוכל למצוא חברות לא פעילים.

```
create view friendship_request as
select reciever_id as user_id, count(*) as requests
from ydemri.friendshiprequest
group by reciever_id

;

select * from friendship_request;
```

	USER_ID	REQUESTS
▶	1	6131
	2	6108
	3	2213
	4	6034
	5	4882
	6	5030
	7	3832
	8	1184
	9	8313
	10	1876
	11	701
	12	9470
	13	2039
	14	8263
	15	6731
	16	1412
	17	6904
	18	9510

חלק 1 – פונקציות ופróצדורות

פונקציות

(את הקוד של הפונקציות ניתן לראות בנספח ט או [בגיט](#))

1. פונקציה שתקבל שם (VARCHAR) ותחזיר כמה פרופילים יש עם זהה שם פרטי או משפחה (NUMBER).

- מקרה בדיקה 1

נכניס את השם Gilley ונקלט שיש 5 משתמשים עם זהה שם פרטי או משפחה:

	Variable	Type	Value
☒	result	Float	5
▶☒	user_name	String	Gilley
*			

- מקרה בדיקה 2

נכנייס את השם Avi ונקלט שיש 0 משתמשים עם זהה שם פרטי או משפחה:

	Variable	Type	Value
☒	result	Float	0
▶☒	user_name	String	Avi
*			

2. פונקציה שתקבל מספר פרופיל (NUMBER) ותחזיר כמה חברים של אותו פרופיל, תייגו אותם בפואט כלשהו (NUMBER).

- מקרה בדיקה 1

נכנייס מספר פרופיל 6774 ונקלט שיש 0 משתמשים שעוניים על התנאי הזה:

	Variable	Type	Value
☒	result	Float	0
▶☒	profile_id	Float	6774
*			

- מקרה בדיקה 2

נכנייס מספר פרופיל 8155 ונקלט שיש 1 משתמשים שעוניים על התנאי הזה:

	Variable	Type	Value
☒	result	Float	1
▶☒	profile_id	Float	8155
*			

פרוצדורות

(את הקוד של הפרוצדורות ניתן לראות בסוף ' או [בגיט](#))

1. פרוצדורה שמקבלת מספר פרופיל ומסכמת את המידע על אותו משתמש.
הפרוצדורה_TDיפס למסך את סיכום הנתונים כמו: סכום הליקים, כמות הפווטים, כמות התיאוגים, מספר החברים וכו'.

נכון פרופיל מס' 6775:

	Variable	Type	Value
▶	profile_id	Float	6775
*			

ונקבל:

USER INFO									
ID	POST	LIKES	SHARES	TAGS	FREINDS	SUGGESTION	REQUEST	NOTIFICATION	
6775	1	412	3	1	5	2	1	6	

2. פרוצדורה שתדפיס לי את 10 אחוז המשתמשים המתויגים ביותר במערכת.
הפרוצדורה לא מקבלת ערכים ולא מחזירה ערכים אלא רק סורקת את המסלד נתונים
ומדפיסה את 10 אחוז המשתמשים המתויגים ביותר במערכת בסדר יורד:

USER INFO	
ID	NUM OF TAGS
9097	10
8634	9
6627	9
3081	8
2413	8
1963	8
1366	8
1837	8
4240	8
4800	8

נספה א – קוד הכנסת הנתונים

```
for x in f:
    c += 1
    x = x.replace("date", "post_date")
    x = x.replace("content", "post_content")
    counter += 1
    count += 1
    if count % 1001 == 0:
        count = 1
    y = x
    profile_id = random.randint(1, 10000)
    arr[profile_id] += 1
    post_id = arr[profile_id]
    visibility = random.randint(0, 1)
    likes = random.randint(0, 2000)
    type = random.randint(0, 1)
    group = 'NULL'
    if type == 0:
        group = random.randint(1, 500)
    lst = y.split("values")
    prop = lst[1].split(',')
    post_id_new = prop[0].replace(str(counter), str(post_id))
    profile_id_new = prop[1].replace(str(count), str(profile_id))
    date_new = change_format_date(prop[2].split(' '))[1]
    visibility_new = prop[3].replace(str(count), str(visibility))
    likes_new = prop[4].replace(str(count), str(likes))
    type_new = prop[6].replace(str(count), str(type))
    group_new = prop[7].replace(str(count), str(group))
    y = "values".join([lst[0], ',').join([post_id_new, profile_id_new, date_new, visibility_new, likes_new, prop[5], type_new, group_new])
    p.write(y)
```

PyCharm 2020.2.5 available
Update...

```
commentId = []
for i in range(10001):
    commentId += [0]

for x in comment:
    y = x
    randomPost = random.randint(1, 1000)
    commentId[randomPost] = commentId[randomPost] + 1
    comment_id = commentId[randomPost]
    post_id = postTuple[randomPost][0]
    profile_id = postTuple[randomPost][1]
    part1 = y.split("values")
    part2 = part1[1].split(',')
    part2[0] = '(' + str(comment_id)
    part2[1] = post_id
    part2[2] = profile_id
    end = part1[0] + "values "
    for i in part2:
        end = end + i + ','
    end = end[0:len(end)-1]
    end = end.replace('MOCK_DATA', "post_comment")
    end = end.replace('content ', "comment_content ")
    newComment.write(end)

newComment.close()
```

נספה ב – קוד של השאלה

```
select gender, avg(number_of_likes), max(number_of_likes), min(number_of_likes), count(*)
from post natural join profile
group by gender;

select profile_id, first_name, last_name, post_id, number_of_likes
from post natural join profile
where number_of_likes >= (
    select avg(number_of_likes)
    from post
);

select profile_id, first_name, last_name, gender
from post natural join profile
where post_id in (select post_id
from (select post_id , count(*)
      from post_comment
      group by post_id
      having count(*) >=(
          select avg(c)
          from (select post_id, count(*) as c
                from post_comment
                group by post_id) t
      )
    ) )

intersect

select profile_id , first_name, last_name, gender
from post natural join profile
where number_of_likes>=(
    select avg(number_of_likes)
    from post
)

;

select profile_id, first_name, last_name, gender
from profile natural join post_comment
group by profile_id, first_name, last_name, gender
having count(*)> (select avg(c) from (
    select profile_id, count(*) as c
    from post_comment
    group by profile_id
) );

SELECT profile_id, post_id, number_of_likes
FROM post p1
WHERE number_of_likes =(
    SELECT max(number_of_likes)
    FROM post p2 natural join
        ( SELECT profile_id
          FROM profile
          minus
          SELECT profile_id
          FROM post_comment))
;
```

```

SELECT pr.profile_id, first_name, last_name, avg(number_of_likes), count(*) as num_of_post
FROM profile pr, post po
WHERE pr.profile_id = po.profile_id
GROUP BY pr.profile_id, first_name, last_name
HAVING count(*) > 1 and avg(number_of_likes) > 1500
;
SELECT profile_id, first_name, last_name, gender
FROM profile pr
WHERE NOT EXISTS(
    SELECT post_type_code
    FROM post_type pt
    WHERE NOT EXISTS(
        SELECT po.post_id
        FROM post po
        WHERE po.post_type_code = pt.post_type_code and po.profile_id = pr.profile_id))
;

SELECT profile_id , first_name, last_name, max(p.number_of_likes)
    FROM post p natural join
        (SELECT profile_id
        FROM post_comment natural join (select post_id, post_type_code from post) T1 natural join post_type
        where post_type_name = 'NORMAL') T
    natural join profile
group by profile_id , first_name, last_name
;

```

נספח ג – קוד של האינדקסים

```

create index index1 ON AFEDER.PROFILE(profile_id,first_name,last_name,gender);
create index index2 ON AFEDER.POST(post_id,number_of_likes);
create index index3 ON AFEDER.POST_COMMENT(comment_id,post_id,profile_id);

```

נספה 6 – קוד של השאלות הנוספות

```

select recieverid, count(*) as num_of_suggestion
from ydemri.friendshipsuggestion join ydemri.friendshiprequest
on recieverid=reciever_id and offerid=requester_id
group by recieverid
union
select recieverid, 0
from ydemri.friendshipsuggestion join ydemri.friendshiprequest
on recieverid=reciever_id
where recieverid not in (select recieverid
                           from ydemri.friendshipsuggestion join ydemri.friendshiprequest
                           on recieverid=reciever_id and offerid=requester_id
);

```



```

select f1.recieverid, count(*) as common_suggestion
from ydemri.friendshipsuggestion f1, ydemri.friendshipsuggestion f2
where f1.offerid = f2.recieverid and f1.recieverid = f2.offerid
and not exists
(
  select offerid
  from ydemri.friendshipsuggestion t1
  where t1.recieverid = f1.recieverid
  minus
  select offerid
  from ydemri.friendshipsuggestion t2
  where t2.recieverid = f2.recieverid
)
group by f1.recieverid;

```



```

SELECT *
FROM
  (SELECT FRIEND_ID1 AS FN1, TAG_ID, NUM AS NUM_TAGS
  FROM (
    (SELECT TAG_ID, COUNT(*) AS NUM
    FROM BMARCUS.TAG
    GROUP BY TAG_ID)

    NATURAL JOIN

    (SELECT DISTINCT FRIEND_ID1, TAG_ID
    FROM BMARCUS.TAG, BMARCUS.FRIENDS
    WHERE FRIEND_ID1 = PROFILE_ID2)))

  NATURAL JOIN

  (SELECT FN1, TAG_ID, COUNT(*) AS MY_FRIENDS
  FROM BMARCUS.TAG NATURAL JOIN (
    SELECT DISTINCT FRIEND_ID1 AS FN1, TAG_ID
    FROM BMARCUS.TAG, BMARCUS.FRIENDS
    WHERE FRIEND_ID1 = PROFILE_ID2) -- מי תוויג ואיפה --
    -- מי תוויג מי תוויג איפה תוויג וממי שוד תוויג --
    WHERE PROFILE_ID2 IN (SELECT FRIEND_ID2
    FROM BMARCUS.FRIENDS
    WHERE FRIEND_ID1 = FN1) -- שהמתויג הראשון חבר של המתויג השני --
    GROUP BY FN1, TAG_ID)

  WHERE NUM_TAGS = MY_FRIENDS; -- גזיבוץ לפי המתויג הראשון ומספר תווינ

```

נספה ה – קוד של הגרף הראשון

```
select TO_CHAR(post_date, 'DAY') as day, sum(number_of_likes) as number_of_likes
from post
group by TO_CHAR(post_date, 'DAY')
order by number_of_likes;
```

נספה ו – קוד של הגרף השני

```
select count(*) as num_of_post, TO_CHAR(post_date, 'MONTH') as month
from post
group by TO_CHAR(post_date, 'MONTH')
order by TO_CHAR(post_date, 'MONTH');
```

נספה ז – קוד עבור שני ה-VIEW'ים הראשונים

```
create or replace view trand_profile as
(
  select profile_id, first_name, last_name, post_id, number_of_likes
  from post natural join profile
  where number_of_likes >= (
    select avg(number_of_likes)
    from post
  )
);

select * from trand_profile ;

create view active_profile as
(
  select profile_id, first_name, last_name, gender
  from profile natural join post_comment
  group by profile_id, first_name, last_name, gender
  having count(*) > (select avg(c) from (
    select profile_id, count(*) as c
    from post_comment
    group by profile_id
  ) )
);

select * from active_profile ;
```

נספה ח – קוד עבור שני VIEWים השונים

```
create view friendship_request as
select reciever_id as user_id, count(*) as requests
from ydemri.friendshiprequest
group by reciever_id

;

select * from friendship_request;

create view active_friendship as
select friend_id1 as user_id, count(*) as friends
from bmarcus.friend
group by friend_id1

;

select * from active_friendship;
```

נספה ט – קוד עבור הֆונקציות

```
create or replace function equalName(user_name in VARCHAR2) return number is
    FunctionResult number;
begin

    select count(*)
    into FunctionResult
    from afeder.profile p
    where p.first_name = user_name or p.last_name = user_name;

    return(FunctionResult);
end equalName;

create or replace function howFriendsTag(profile_id in number) return number is
    FunctionResult number;
begin
    select count(*)
    into FunctionResult
    from bmarcus.tag t join bmarcus.friends f on f.friend_id2 = t.profile_id1
    where f.friend_id1 = profile_id and t.profile_id2 = profile_id;

    return(FunctionResult);
end howFriendsTag;
```

נספח י – קוד עבורה הפרוצדורות

```

create or replace procedure allInfo(profile_id in INTEGER) is
    proId INTEGER := profile_id;
    numOfPost number;
    numOfLikes number;
    numOfFriends number;
    numOfShare number;
    numOfTags number;
    numOfSuggestion number;
    numOfRequest number;
    numOfNotification number;
begin
    select count(*)
    into numOfRequest
    from ydemri.friendshiprequest f1
    where f1.reciever_id = proId;

    select count(*)
    into numOfSuggestion
    from ydemri.friendshipsuggestion s1
    where s1.recieverid = proId;

    select count(*)
    into numOfNotification
    from ydemri.notification n
    where n.profile_id = proId;

    select count(*)
    into numOfShare
    from bmarcus.share_post s2
    where s2.profile_id = proId;

    select count(*)
    into numOfFriends
    from bmarcus.friends f2
    where f2.friend_id1 = proId;

    select count(*)
    into numOfTags
    from bmarcus.tag t
    where t.profile_id2 = proId;

    select sum(p1.number_of_likes)
    into numOfLikes
    from afeder.post p1
    where p1.profile_id = proId;

    IF numOfLikes is null then
        numOfLikes := 0;
    end if;

    select count(*)
    into numOfPost
    from afeder.post p2
    where p2.profile_id = proId;

    dbms_output.put_line('-----');
    dbms_output.put_line('                               USER INFO');
    dbms_output.put_line('ID | POST | LIKES | SHARES | TAGS | FREINDS | SUGGESTION | REQUEST | NOTIFICATION');
    dbms_output.put_line('-----');
    dbms_output.put_line(proId || ' | ' || numOfPost || ' | ' || numOfLikes || ' | ' || numOfShare ||
                         ' | ' || numOfTags || ' | ' || numOfFriends || ' | ' || numOfSuggestion ||
                         ' | ' || numOfRequest || ' | ' || numOfNotification );
end allInfo;

```

```

create or replace procedure thePop is

CURSOR top_users IS
select t2.pro_id, t2.c
from
(
    select t1.profile_id2 as pro_id , count(*) as c, ROW_NUMBER() OVER (ORDER BY count(*)) as rnum
    from bmarcus.tag t1
    group by t1.profile_id2
) t2

where t2.rnum >=
(
    select count( distinct t1.profile_id2)
    from bmarcus.tag t1
) * 0.9

order by t2.c desc;

top_user top_users%rowtype;

BEGIN
open top_users;
fetch top_users into top_user;
if top_users%notfound then
    dbms_output.put_line('error');
elsif top_users%found then

    dbms_output.put_line('-----');                                     USER INFO
    dbms_output.put_line('ID | NUM OF TAGS ');
    dbms_output.put_line('-----');
loop
    exit when top_users%notfound ;
    dbms_output.put_line(top_user.pro_id || ' | '|| top_user.c);
    fetch top_users into top_user;
end loop;
close top_users;
end if;

end thePop;

```