

JOIN של טבלה על עצמה חזרה על שאילתות מסוג GROUP BY, המשך הצפנה ו- colission attack

הזכרנו מצב שבו רוצים אינפורמציה מורכבת מטבלה, הדורשת חיבור טבלה לעצמה בעזרת alias.

JOIN של טבלה על עצמה

נתונה הטבלה:

employee

emp_id	emp_name	manager_id
1	Hattie	4
2	Henry	4
3	Harry	5
4	Helen	NULL
5	Heidi	4
6	Hazel	1

רוצים לקבל מידע על מי מדווח למי, כלומר לכל עובד או עובדת לתת את המנהל שלו או שלה. זהו מצב שרוצים לקבל שני סטים של אינפורמציה מטבלה אחת.

```
select e.emp_name as employee,
       m.emp_name as manager
from   employee e, employee m
where  e.manager_id = m.emp_id
```

תוצאה:

employee	manager
Hattie	Helen
Henry	Helen
Harry	Heidi
Heidi	Helen
Hazel	Hattie

שמתם לב שחסרה שורה?

עשינו inner join, ו- Helen היא בעלת החברה כך שאין לה מנהל. תנסו לעשות outer join

כאשר מעוניינים בנתונים מטבלה שמקבצים מספר שורות יחד תחת קריטריון מסוים, ניתן להשתמש במילות המפתח- GROUP BY.

כפי שכבר הזכרנו, בשאיתה מסוג "הקבצה", אנחנו בעצם רוצים לדעת או כמה רשומות השוות ביניהן בשדה (או שדות) מסויים, או לסכם שדה שהוא שדה נומרי, על פני כל שורות הקבוצה.

לדוגמה, השאלה שניתנה לשיעורי הבית ביקשה רשימה של סופרים שכתבו יותר מספר אחד, לכן ביצענו על הטבלה: book את השאילתה הבא:

```
select author, count(*) as Titles
```

from book
group by author
having count(*) > 1

זו דוגמה לסוג הראשון של שאילתות מסוג הקבצה (שסופרת את מספר הרשומות)
דוגמה נוספת עם סיכום של שדה נומרי יכולה להיות כלהלן:

```
select isbn, sum(noOfCopies) as 'Total copies'
from available
group by isbn
```

בדוגמה זאת, השאילתה תסכם את כמויות הספרים הזמינות עבור אותו ספר בכל הספריות בהן הוא נימצא ותוציא עבורנו שורות סיכום שבהן לכל מספר זיהוי של ספר (isbn) יכתב מספר העותקים הזמין.

שאלה נוספת לשיעורי הבית
לכתוב שאילתה שנותנת את צמדי הסטודנטים ששאלו אותו ספר, כאשר ה-ID של אחד קטן מה-ID של השני.

הצפנה

דיברנו על פונקציית Hash. הצורה שהפונקציה עובדת, והיא בד"כ עם מחרוזות ביטים של 128, 160 או 256 או יותר ביטים. הפונקציה מערבבת את הביטים של ההודעה המקורית בצורות שונות ולבסוף לוקחת נניח 128 ביטים שמאליים אחרי הערבוביה. הרעיון הוא שאי אפשר לעשות reverse לפונקציית הערבוב, כלומר מהתוצאה לא ניתן לנחש את ההודעה. התוצאה של ההאש $h(x)$ ניקראת החתימה (signature)

יש למעשה 3 (התקפות) קריטריונים שבהם פונקציית ההאש צריכה לעמוד.

1. בהינתן מצב שבו ידוע ה-digest של הודעה $y=h(M)$ צריך להיות מאד קשה למצוא M_1 , כך ש-
 $y=h(M_1)$ זו ניקראת התקפת first preimage. כלומר שבהינתן y אפשר למצוא את M המקורי או אחר.

2. התקפה אחרת היא במצב שידועה גם ההודעה M וגם $y=h(M)$ ואפשר למצוא M_1 (שונה מ- M) כך שיתקיים מצב ש- y יהיה גם שווה ל- $h(M_1)$ זה ניקרא: second preimage attack (או resistance) זהו מצב שאפשר לזייף הודעה.

3. התקפה שלישית ואולי החשובה מכולם ניקראת: Collision attack זהו מצב שניתן למצוא שתי הודעות (לא קיימות קודם) M ו- M_1 , כך שיתקיים: $h(M)=h(M_1)$ זהו סוג התקפה הקל ביותר, כי אפשר לבדוק בצורה מתודית אפשרויות ולכן עם מהירות חישוב גבוהה זה יכול להצליח. כל הרעיון ביצירת ה-digest (או תוצאת ההאש) שזה יהיה קצר מההודעה עצמה ולכן יתכן מצב ששתי הודעות יתורגמו לאותו digest

כדי להבין את הקלות הכמעט בלתי ניסבלת לבצע תקיפה מהסוג השלישי (collision), בואו נתבונן במה שניקרא פרדוקס ימי ההולדת. כמה תלמידים לדעתכם צריכים להיות בכיתה על מנת שהסיכויים שיהיו לפחות שניים עם יום הולדת באותו יום, גדולים מ-50%?

התשובה המפתיעה היא: 23. ההסבר ההגיוני זה שישנם הרבה צירופים שיכולים לענות על התנאי ומרחב המדגם שלנו הוא יחסית קטן (365 בקירוב). צורת החישוב היא על ידי שיטת האלימינציה, כלומר מציאת הסיכוי שלא יהיה יום הולדת משותף ואז לעשות אחד פחות זה. לדוגמה אם יש 2 תלמידים הסיכוי הוא 364/365 שלא יהיה משותף. אם יש שלושה אז: $(363/365)*(364/365)$ וכך הלאה.

התשובה המתמטית הפורמלית שזוהי התפלגות נורמלית של משתנים בלתי תלויים, למי שלמד קצת הסתברות וסטטיסטיקה והנוסחה לחישוב היא (בקירוב):

$$P = 1 - e^{-k^2/2N}$$

כאשר k זה מספר התלמידים (המשתנים הבלתי תלויים), N זה מספר ימי השנה (365) ו- e זה הבסיס של הלוגריתם הטבעי (ln) בקירוב 2.718 (P הסתברות) - זה נובע מהחישוב של טור טיילור:

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

אם נחלץ מהמשוואה את k , (כאשר נציב $P=1/2$) ומבלי להכביר כאן במתמטיקה, נקבל: $k=1.18 \times N^{1/2}$ או במילים אחרות $k=23$.

הווה אומר שאם יש לנו בכתה 23 תלמידים או יותר, הסיכוי שיהיו כאן תלמידים עם יום הולדת משותף הוא גבוה מ-50%.

קעת נחזור חזרה לבעייה של ה- Collision attack. באופן כללי בפונקציה האש קורית הבעיה שניקראת pigeonhole. ברגע שיש לנו N חורי יונים (כמו במגרש גולף) ובתוכן מסתתרות $N+1$ יונים, הרי ברור שלפחות בחור אחד ישנה יותר מיונה אחת. ואם יש $kN+1$, יהיה לנו לפחות חור אחד עם $k+1$. גם הזכרנו כבר שפונקציה האש שמעבירה הודעה בעלת אורך לא מוגבל ל- digest באורך מוגבל, בהינתן מספיק הודעות, היא בהכרח תייצר אותו digest עבור שתי הודעות שונות.

בואו נניח שיש אלגוריתם שמייצר עבורנו את ה- digest עבור כל הודעה שמגיעה בצורה הבאה: ישנה טבלה ובה 2 עמודות, אחת עבור הודעות והשניה עבור ה- digest שמתאים לה. יושב לו מגיד העתידות (oracle) עם מטבע מאוזן. הנה דוגמה לטבלה עם שתי הודעות (אורך ה digest הוא 16 ביט מיוצגים על ידי 4 ספרות הקסה-דצימליות)

message	digest
BA200EFFCBCDEF	13AB
AAAAEEEEFBDDCC	A38B

מגיעה הודעה חדשה, מגיד העתידות בודק אם היא בטבלה, אם כן (נניח: BA200EFFCBCDEF), הוא מחזיר לנו את ה- digest שלה: 13AB. אם אינה בטבלה, הוא מטיל את המטבע 16 פעמים ולכל עץ הוא רושם 0, לכל פאלי רושם 1, ויוצר בכך digest רנדומלי לגמרי. זהו המודל הראנדומלי של ה- Oracle.

שיעורי בית
 לכתוב אלגוריתם של התקפה מסוג collision על הפונקציה של ה- Oracle. כלומר לייצר שתי הודעות M ו- $M1$, כך ש: $h(M)=h(M1)$