

## המרכז האקדמי הרב תחומי ירושלים, החוג למדעי המחשב

### אנליזה של ביג דאטה / חננאל פרל

#### סמסטר א', תשפ"ו

תרגיל 6 / 07.12.2025

תאריך הגשה:

יום חמישי 18.12.2025 בשעה 23:00

מטרות התרגיל:

השוואת זמני ריצה SQLite DuckDB

הוראות:

בעולמות של ביג דאטה, הרבה פעמים יש צורך להשוות זמני ריצות של כל מיני כלים ושיטות. זמני הריצה מושפעים לרוב מגודל הקלט. בתרגיל זה נבחן שני כלים עם כמה אופציות. נריך בכל אחד מהקונפיגורציות כמה שאלות עם כמה קלטים בגדלים שונים.

מה שנבדוק בתרגיל, יהיו 3 קונפיגורציות האלו:

(A) טבלאות שניצור בדאטהבייס DUCKDB ועליהם נריך השאלות.

```
duckdb.connect("file:duckdb")
```

(B) טבלאות בדאטהבייס SQLITE ללא אינדקסים כלל.

```
sqlite3.connect("file:sqlite")
```

(C) טבלאות בדאטהבייס SQLITE עם אינדקסים על עמודות רלוונטיות לפי שיקול דעתכם.

```
sqlite3.connect("file:sqlite")
```

```
CREATE INDEX ..
```

אנו נשתמש במדד TPC-H

<https://www.tpc.org/tpch/>

TPC-H הוא כלי מבחן ביצועים (Benchmark) סטנדרטי להערכת ביצועי מערכות כגון מחסני נתונים ובסיסי נתונים אנליטיים. המטרה העיקרית של TPC-H היא לדמות עומס עבודה אנליטי מציאותי OLAP המאפיין סביבות עסקיות. אנו נבחן את היכולת של המערכת לבצע שאלות מורכבות על כמויות גדולות של נתונים. ונמדוד את זמן ביצוע השאלות (Query Execution Time) ואת יכולת המערכת לעבד נתונים במהירות.

כדי לייצר את הדאטה נשתמש ברכיב ההרחבה tpch של DuckDB:

[https://duckdb.org/docs/stable/core\\_extensions/tpch](https://duckdb.org/docs/stable/core_extensions/tpch)

גרסאות של חבילות ושל פייתון לפי מה שיש באנקונדה גירסה: Anaconda3-2025.06-0

אפשר לימצוא בלינק כאן:

<https://repo.anaconda.com/archive/>

ולמשל עבור חלונות:

[https://repo.anaconda.com/archive/Anaconda3-2025.06-0-Windows-x86\\_64.exe](https://repo.anaconda.com/archive/Anaconda3-2025.06-0-Windows-x86_64.exe)

חבילות נוספות מותרות:

DuckDB כמובן, גירסה 1.4.2

Sqlglot גירסה 28.1.0

יש לייצר את הדאטה בגדלים שונים

נתחיל מקנה מידה של 0.01 (הפרמטר sf = 0.01)

את הנתונים נייצר על ידי DUCKDB, ואז נעביר אותם לטבלאות SQLITE.

נריץ כל שאילתה (מתוך ה 22 של TPCB), נריץ 3 פעמים ונשמור את הזמן החציוני (median).

כל פעם נגדיל את הקנה מידה (לפי שיקול דעתכם בכמה כל פעם). נמחוק את כל הקבצים מהסבב הקודם (הקבצים של DUCKDB והקבצים של SQLITE). נשמור את הנתונים בטבלאות חדשות. נריץ ונשמור את הזמני ריצות של כל השאילתות. נדפיס תוצאות מדגמיות של כל שאילתה.

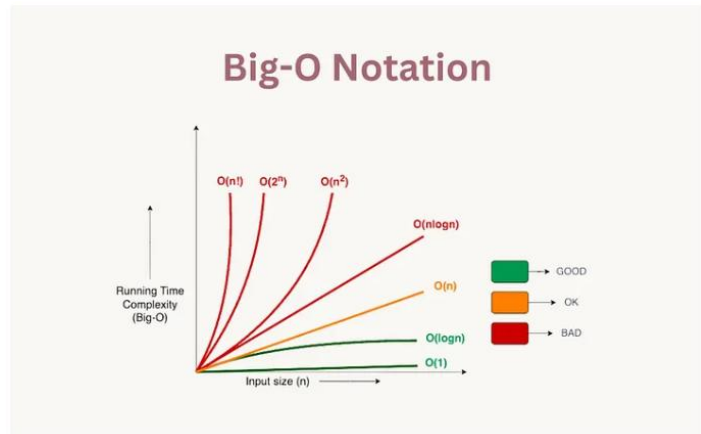
ברגע שהזמני ריצה על SQLITE נהיים איטיים מידי יש להפסיק להריץ את SQLITE (אפשר לבדוק ידנית ולסמן בקוד בצורה קבועה מאיזה קנה מידה להפסיק להריץ SQLITE) ולהמשיך רק על DUCKDB. יש להמשיך להריץ את DUCKDB עוד כמה פעמים עד שהקובץ נהיה גדול מיד, או נגמר הזכרון, או זמן ריצה ארוך מידי לטעמכם (לא יותר מחצי שעה ריצה). כמות הסבבים תהיה סה"כ בערך 10. בהתאם לזה יש לתכנן מה יהיו הקנה מידה בכל ריצה.

את הריצה של השאילתות עצמם יש לבצע ישירות דרך הכלי הנבדק או SQLITE או DUCKDB בהתאמה. לא דרך כלים אחרים (למשל אין להשתמש ב PANDAS בשלב זה) את מדידת זמני הריצה נעשה בפייתון, למשל time.time(), וכדומה..

עבור כל שאילתה שהרצת יש לשמור:

- + מספר השאילתה
- + הקונפיגורציה
- + גודל הדאטהבייס כולו בדיסק
- + כמות זמן הריצה החציוני median
- + זמן נוכחי בשעון (של הריצה האחרונה מתוך ה-3)

יש להדפיס את כל התוצאות למסך,  
וגם לשמור בקובץ בטבלת CSV את כל התוצאות של זמני הריצה לפי כל הסוגים (כאן אפשר ע"י PANDAS או כל משהו דומה)  
כמו כן יש לצייר גרפים עבור כל שאילתה: גרף קווי שבה כל קונפיגורציה מקבלת צבע, ציר X - גודל הקלט (לפי הקנה מידה = המשתנה SF), ציר Y - זמן ריצה.  
לדוגמא:



<https://medium.com/@clebertonfgc/big-o-notation-in-practice-ac192e23c16>

<https://www.bigocheatsheet.com/>

שימו לב, בתוצאות אצלכם, לאיזה עקומה בתמונה לדוגמא ( $n$  |  $2^n$  |  $n^2$  |  $n \log n$  |  $n$  |  $\log n$  | 1), הזמנים שלכם מתאימים.  
(לציור יש להשתמש באחת מהחבילות הסטנדרטיות שמותקנות עם אנקונדה, למשל MATPLOTLIB וכדומה)

כמו כן, יש לצייר גרף נוסף מספר 23, סיכום שיכיל את זמני הריצה הממוצעים של כל השאילתות לכל קונפיגורציה.

יש לשים לב שהמחשב לא עסוק בדברים אחרים, ושאינן תוכנות אחרות רצות ברקע, כדי שנוכל לקבל השוואה אמיתית.

יש לכתוב את סוג המחשב וכמות הזכרון והמעבד וכו  
על ידי שימוש בקוד כזה (ללא החלק של GPU)

<https://thepythoncode.com/article/get-hardware-system-information-python>

psutil | platform | datetime

שימו לב, מבחינת הריצה, יש לייצר את הדאטה הקטן, להכין את כל הקונפיגורציות ואז להריץ את כל השאילתות, ולשמור הזמני ריצה וכו, ואז למחוק כל הקבצים. ולעבור לגודל הבא וכו..

כל הקוד שמוגש צריך להיות שלכם מתחילתו ועד סופו, אין להגיש קוד שמישהו אחר כתב (למעט הקוד של סוג המחשב וכמות הזכרון שצוין לעיל). אפשר להעזר באתרי אינטרנט וצאטים, אך יש להבין ולהיות מסוגל להסביר ולהצדיק את הקוד שמוגש.

זהו תרגיל תכנותי, ועל כן הבדיקות יהיו קפדניות גם מבחינת פייתון, גם מבחינת מבנה הקוד וחלוקה לפונקציות וכו וגם מבחינת צורת ההגשה וכדומה.

סך הכל צריך להגיש קובץ ZIP שמכיל:

- א. קובץ requirements.txt המכיל את כל חבילת הפיתוח הנצרכות להתקנה והרצה.
  - ב. קובץ פייתון שמכין את הדאטה, מייצר ומכניס לבסיסי הנתונים ומריץ את כל השאילתות ושומר תוצאות בסוף.
  - ג. קובץ טקסט המכיל את הפלט של הריצה שלכם.
  - ד. קובץ CSV שמכיל את כל התוצאות של זמני הריצה לפי כל הסוגים.
  - ה. קבצי התמונות של הגרפים – 1+22 קבצים, קובץ לכל שאילתה והסיכום.
  - ו. קובץ README עם הסברים ותוספות והחלטות לגבי דברים שלא מוגדרים בפירוש.
- (אין להגיש את הדאטה שיצרתם! קובץ הפיתוח אמור לייצר את כל הדאטה הנצרך לביצוע הריצות!)

**שם הקובץ ZIP צריך להיות שמות ות"ז של הזוג שמגיש**

**בהצלחה!**