

דוח מסכם – מערכת חישוב נסחאות דינמיות/ רבקה טהר אפל

ミミשתי מערכת שמריצה נסחאות דינמיות על מיליאון רשומות בשלוש שיטות שונות (SQL, C#, Python), שומרת תוצאות זמני ריצה, ומשווה בין השיטות כדי לבחור את היעילה יותר.

• SQL_SP – חישוב דינמי בצד הנתונים

השיטה מחשבת את הנוסחה ישירות ב-DB Server SQL בעזרת SQL דינמי. החישוב מתבצע כולו בתוך ה-DB על מיליון הרשומות, ללא העברת נתונים לאפליקציה, ולכן מתקבלות תוצאות יציבות ומהירות. יתרון: ביצועים גבוהים וניצול מלא של מען ה-SQL. חיסרון: גמישות מוגבלת ושינויי נסחאות דורש זהירות בעבודה עם SQL דינמי.

• C_SHARP-NCalc – חישוב בצד האפליקציה

האפליקציה טענת את הנתונים מה-DB פעם אחת, מריצה את הנוסחה באמצעות מנוע הביטויים NCalc, וביצעת את כל החישובים בזיכרון. השיטה תומכת בפונקציות מתמטיות מורכבות ובעל יכולת הרחבת גבואה. יתרון: גמישות מלאה, קוד נקי, מהירות גבוהה מאוד בזכות חישוב בזיכרון. חיסרון: דורש טיפול נכון בזכרון ובחיבור DB.

• PYTHON – חישוב דינמי בלולאה

שיטה זו נטענים הנתונים ומתבצע חישוב של כל נסחה בעזרת eval על כל רשומה בנפרד. זה מאפשר גמישות גבוהה ופשטות בקוד, אך דורש פירוש מחדש בכל איטרציה. יתרון: קל להרחבה, גמיש מאוד, מתאים לניסויים. חיסרון: איטי משמעותית, בעיקר בגלל חישוב שורה-אחר-שורה ללא אופטימיזציות פנימיות.

מדידת ביצועים

להלן זמני הריצה שנמדדו עבור כל שיטה על כל רשומות הנתונים:

- #C – זמן ריצה של כ-1–3 שניות
- SQL_SP – זמן ריצה של כ-7–8 שניות
- Python – זמן ריצה של כ-30–33 שניות

	method	avg_run_time
1	C_SHARP	2.25
2	PYTHON	32.86
3	SQL_SP	7.89

- C# הוא המהיר ביותר בזכות חישוב בזיכרון (In-Memory) ומונע ביטויים עיל.
- SQL_SP ביצועי, אך איטי מעט יותר בשל אופטימיזציות SQL פנימיות.
- Python איטי משמעותית בגל פירוש בכל איטרציה.

בדיקת התאמת תוצאות בין שיטות

כתבתי פרוצדורה שמודדת שכל שלוש השיטות הניבו את אותה התוצאה עבור כל נוסחה וכל רשותה:

- בודקת פער בין MIN ל-MAX של כל (`targil_id, data_id`)
- בודקת שכל שלוש השיטות החזירו תוצאה (השלמות)
- מחזירה את מספר הסטיות לכל תרגיל

תוצאה שהתקבלה: **0 סטיות** → כל השיטות הפיקו תוצאות זהות. (הוחזרה טבלה ריקה- 0 סטיות)

<code>targil_id</code>	<code>mismatches</code>

לסיום:

#C - הפתרון מהיר והמדויק ביותר

- חישוב מהיר מאוד בזיכרון
- מנוע ביטויים עיל
- גמישות גבוהה להוספה פונקציות ותנאים
- תוצאות זהות ובטוחנות כמו SQL

ולכן זו השיטה המתאימה ביותר למערכות שביציאות חישובים דינמיים רבים, לאורך זמן.

SQL_SP - מצין כשל החישובים נשארים בתוך מסד הנתונים. מהיר יחסיב, אך פחות גמיש לשינויים בנסיבות.

Python - גמיש מאוד ומתאים לניסויים ופיתוח מהיר. חישוב טורי של מיליון רשומות איטי יחסית, אך הביצועים יכולים להשתפר משמעותית בעזרת כלים כמו Pandas.