DOI: https://doi.org/10.33216/2222-3428-2019-17-1

УДК 351.863.658

Бакітько Д.Е., Нестеров М.В.

ОПТИМІЗАЦІЯ ПОВІЛЬНОГО SQL-ЗАПИТУ

У статті розглядаються етапи, які направлені на фіксування повільних запитів та їх оптимізацію. Визначені та обрані основні метрики, які дають інформацію про стан системи під час обробки запитів. Були встановлені порогові значення для метрик, які необхідні для запуску методики фіксування повільних запитів. Проаналізовані методи оптимізації запитів. Для розрахунків ймовірності оптимізації запиту обраними методами оптимізації було обрано математичний метод DSmT. Обраний метод дає теоретичні дані про стан системи, які найбільш ефективні за показники метрик. З використанням плану виконання запиту (EXPLAIN PLAN) встановлено який з методів оптимізації запитів використовує менше ресурсів. Для аналізу стану показників метрик з обраними методами оптимізації було розраховане теоретичне та практичне навантаження та вартість (cost) кожного запиту. Встановлено, що розраховані навантаження різняться, що свідчить про те, що практичні показники враховують стани системи. З обраним методом фіксувався відсоток даних, який потрапляє в кеш бази даних.

Ключові слова: база даних, метрики, cost, DSmT, індексування, секціонування, ймовірність

Актуальність дослідження. Повільний запит негативно впливає на систему, виникають навантаження, які зменшують продуктивність бази даних. Зростає час обробки запитів, виникають ситуації з простоями, які марнують обчислювальний потенціал системи.

Постановка проблеми. Виявлення повільних запитів, розрахунок ймовірностей метрик та встановлення їх порогових значень для фіксування повільних запитів. Вибір методів оптимізації та розрахунок їх впливу на систему. Вибір методу з меншими показниками впливу на систему та виявлення кількості даних, які потраплять до кешу.

Теоретичний аналіз дослідження. Проблема оптимізації запитів є дуже актуальною, про що свідчить багато наукових робіт за останні 5 років. Дослідники намагаються знайти різні способи вирішення проблеми. William Charles Eidson та Jesse Collins в патенті [1] звертали увагу на те, що швидкість обробки запиту до бази даних можна пришвидшити за допомогою індексування. John F. Hornibrook, Dieu Quang La, Calisto Paul Zuzarte патенті [2] звертають увагу на те, що прискорити запити до бази даних можна шляхом об'єднання запитів. Xing Chen, Qing Yun Hao, Yi Jin, Wan Chuan Zhang в патенті [3] звертають увагу на те, що покращити оптимізацію бази даних можна за допомогою запиту та його плану. На основі плану запиту будується оптимізаційний план доступу до ресурсів. План зберігається в зовнішньому ресурсі даних та містить в собі очікування розподілу зовнішніх даних. Спосіб також включає в себе отримання інформації про розподіл даних, пов'язаної з очікуваним розподілом зовнішніх даних, на основі оптимізованого плану доступу, передачу інформації про розподіл даних у зовнішній джерело даних, так що зовнішнє джерело даних розділяє зовнішні дані відповідно до розподілу даних та повертає розділені зовнішні дані паралельно, і виконує пов'язану із запитом обробку розділених зовнішніх даних відповідно до оптимізованим планом доступу.

Мета статті. Виконати дослідження, яке направлене на оптимізацію запитів. З'ясувати вплив методів оптимізації на систему, розрахувати навантаження методів.

Задачі дослідження. Виявлення основних метрик баз даних, вибір методу оптимізації, який має менший вплив на систему, створення методики виявлення повільних запитів на основі ймовірностей навантаження на систему.

Основний матеріал статті. Для виявлення проблем в роботі системи треба переконатися в тому, в чому саме проблема: проблеми в роботі бази даних чи в самому запиті. Для аналізу проблеми на боці серверу можна користуватися метриками, з використанням яких можна виявити джерело проблем. Якщо показники метрик не перебувають в критичних станах, проблема в самому запиті. При великих об'ємах даних треба заздалегідь продумувати структуру бази даних, щоб звести до мінімуму пошуку та обробки інформації. Для пришвидшення операцій з базою даних в таких випадках треба користуватися засобами оптимізації. Оптимізація полегшує пошук інформації за рахунок позитивних сторін того, чи іншого методу оптимізації.

А. Вибір метрик

В статті [4] в якості метрик продуктивності використовуються показники навантаження на центральний процесор (ЦП), навантаження на пам'ять (RAM) та значення якості мережі.

В патенті [5] приділяють велике значення метриці зчитування/запису. Оптимізація запитів ведеться за моніторингом значень метрики запам'ятовуючого пристрою.

Найбільш важливими інтегральними (тобто враховують кілька різних факторів) метриками є пропускна здатність і час відгуку системи. Обидві характеристики вимірюються на певному навантаженні системи. Обидві метрики мають сенс для дуже широкого класу систем; уточнення того, які навантаження має сенс розглядати, залежить, звичайно, від класу системи і від вимог до неї. Для систем управління базами даних це може бути деяка суміш запитів або інших дій різної складності. Коли таке навантаження визначена, можна виміряти середню

кількість подібних дій, які виконуються за одиницю часу (пропускна здатність), або середній час виконання однієї дії (час відгуку). У багатьох випадках має сенс оцінювати час відгуку окремо для кожного класу дій (в залежності від їх складності для системи) [6].

Згідно з аналізу метрик, основними метриками для оцінювання стану системи є навантаження на центральний процесор (CPU) та зчитування/запис (Disk I/O).

В. Розрахунок ймовірностей та порогових значень метрик

Для розрахунку ймовірності станів зчитувань (Disk I/O) слід проаналізувати скільки запитів всього виконалось та скільки запитів мають однакову кількість зчитувань.

Кількість повторів значень зчитувань на загальну кількість зчитувань приведено в таблиці 1.

Таблиця 1 – Значення повторів станів зчитувань.

Кількість зчитувань одним запитом	Кількість повторів зчитувань одного запиту
1694416	32
1696607	23
2897824	8
2900015	7
1694418	5
1983527	4

Слід визначити, які саме значення кількості зчитувань впливають на навантаження системи. Для цього необхідно визначити, скільки саме системних ресурсів необхідно для кожного стану зчитування. За допомогою SQL Script було виявлено значення системних ресурсів, які використовуються для обробки запитів:

Рисунок 1 - Значення системних ресурсів для значення зчитування 1694416

Рисунок 2 - Значення системних ресурсів для значення зчитування 1696607

Встановлено, що два стани зчитування потребують багато системних ресурсів для обробки запитів. Інші стани потребують значно менше ресурсів.

Розрахунок ймовірності навантаження на зчитування/запис буде здійснюватися [7]:

$$P(A) = \frac{m}{n} \quad (1)$$

де n – загальна кількість повторів станів, m – загальна кількість повторів, на обробку яких необхідно багато системних ресурсів.

Ймовірність навантаження на зчитування/запис становить:

$$P_{\max}(A) = \frac{55}{79} = 0.69$$

де 55 – значення повторів запитів, які використовують багато ресурсів.

Ймовірність навантаження на зчитування/запис під час обробки запиту для повільного запиту становить 0.69. Ймовірність мінімального навантаження зчитування/запису залежить від навантаження становить:

$$P_{\min}(A) = \frac{15}{79} = 0.19$$

За допомогою журналу повільних запитів та монітору (Статус/Монітор) будо відстежено навантаження на центральний процесор серверу (CPU). В таблиці 2 приведено діапазон станів центрального процесору та кількість повторів станів, в яких перебував процесор під час обробки запиту.

Таблиця 2 – Стани навантаження процесору.

Навантаження, %	Кількість повторів навантаж.
40	110
51	94
76	86
49	72
68	70
22	63

З таблиці 2.4 можна зробити висновок, що навантаження на процесор належить діапазону від 22 до 76%. Серед цих станів необхідно обрати ті, які гальмували продуктивність бази даних. Серед всіх показників навантаження можна виділити такі стани:

а) 76%.

б) 68%.

в) 63%.

Мінімальне значення навантаження – 22%.

Максимальне навантаження на процесор можна розрахувати, склавши кількість повторів виділених станів та розділити на загальну кількість повторів станів.

$$P_{\max}(A) = \frac{m}{n} = \frac{203}{596} = 0.34$$

Ймовірність мінімального навантаження процесору під час виконання запиту [7]:

$$P_{\min}(A) = \frac{63}{596} = 0.1$$

В таблиці 3 приведені всі розраховані ймовірності розглянутих метрик.

Таблиця 3 – Ймовірності метрик.

Метрика	Max навантаження	Min навантаження
Disk I/O	0.69	0.19
CPU	0.34	0.1

Порогові значення для метрик зчитування/запису та центрального процесору (Disk I/O) є середнім арифметичним від значень ймовірностей мінімального та середніх навантажень. В таблиці 4 приведені розраховані порогові значення ймовірностей метрик.

Таблиця 4 – Порогові значення ймовірностей метрик

Метрика	Ймовірність
Disk I/O	0.44
CPU	0.22

Для розрахунку ймовірностей метрик та методів оптимізації під час обробки повільного SQL-запиту використовується метод Дезерта-Смарандаша (DSmT) [8]. Злиття ймовірностей дає більш чітку інформацію про стан системи під час обробки проблемних запитів, ніж моніторинг однієї з метрик. Злиття ймовірностей метрик розраховується по формулі:

$$m_{PCR5} = \sum_{\substack{x_1 \in 2^x \\ x_1 \cap x_2 = X}} m_1(X_1) m_2(X_2) + \sum_{\substack{x_2 \in 2^x \\ x_2 \cap x = \varnothing}} \left[\frac{m_1(X_1)^2 m_2(X_2)}{m_1(X_1) + m_2(X_2)} + \frac{m_2(X_1)^2 m_1(X_2)}{m_2(X_1) + m_1(X_2)} \right] \quad (2)$$

де, m_1,m_2 – маси (ймовірності) методів; X_1,X_2 – порядкові номери

Для злиття двох метрик, які сигналізують про стан бази даних в момент обробки запиту та в подальшому не перевіряти стан цих метрик, необхідно зробити злиття їх показників. Для цього необхідно використовувати їх максимальні показники навантаження.

Згідно з формули 2, злиття ймовірностей навантаження зчитування/запису (Disk I/O) по правилу DSmT становить:

$$M_{PCR5}(X) = 0.69 * 0.34 + \left[\frac{0.69^2 * 0.34}{0.69 + 0.34} + \frac{0.34^2 * 0.69}{0.34 + 0.69} \right] = 0.4676$$

Згідно з формули 2.3, злиття ймовірностей навантаження центрального процесору (CPU) по правилу DSmT становить:

$$M_{PCR5}(X) = 0.1 * 0.19 + \left[\frac{0.1^2 * 0.19}{0.1 + 0.19} + \frac{0.19^2 * 0.1}{0.19 + 0.1} \right] = 0.0374$$

В таблиці 5 приведені ймовірності метрик та їх розраховані значення з використанням методу DSmT.

Таблиця 5 – Розраховані значення по двом математичним методам.

Фокальний елемент	m_1	m_2	m_{PCR5}
A (Disk I/O)	0.69	0.19	0.4676
B (CPU)	0.34	0.1	0.0374

Згідно з [9] теоретичне навантаження на системні ресурси (cost) методом індексування розраховується за допомогою формул 3, 4, та 5. Розглядаються випадки кластерного індексування, індексування стовпця та таблиці без індексів.

Розрахунок індексованої таблиці відбувається за допомогою формули 3.

$$cost = HT_I + 1 \quad ((3)$$

де HT_I – рівень індексу в B-Three дереві.

Cost для індексної таблиці становить:

$$cost = 65 + 1 = 66$$

Cost для кластерного індексу розраховується за допомогою формули 4.

$$cost = HT_I + \left\lceil \frac{SC(A,R)}{F_R} \right\rceil \quad ((4)$$

де SC(A,R) - середня кількість кортежів R, які задовольняють умові рівності; F_R - кількість кортежів, які вміщуються в один блок даних.

Cost для кластерного індексу становить:

$$cost = 65 + \left\lceil \frac{42.477}{22.234} \right\rceil = 66.91$$

Cost для таблиці без індексів розраховується згідно з формули 5.

$$cost = HT_I + SC(A,R) \quad (5)$$

Cost для таблиці без індексів становить:

$$cost = 65 + 42.477 = 107.477$$

C. Розрахунок оптимізації запиту методом індексування

План оптимізації повільного запиту методом індексування передбачає такі етапи:

Рисунок 3 - Схема плану виконання оптимізації методом індексування

Для аналізу методу необхідно відстежувати план виконання запиту для кожного окремого індексу та фіксувати кількість про сканованих рядків та значення системних ресурсів для обробки запиту.

В таблиці 6 приведені значення використання системних ресурсів для індексування кожного з розглянутих стовпців. Індексування окреме, тобто кожного разу індексувався лише один стовпець. Після індексування стовпця необхідно визначити значення використаних системних ресурсів для обробки запиту.

Таблиця 6 – Значення системних ресурсів для індексування кожного розглянутого стовпця.

Назва стовпця	Значення системних ресурсів
ID_COURSE	1938.636227
YEAR_PL	1977.893508
CODE_SPECIALITY	2031.171247
SEMESTR_V	2045.191704
LEARN_TIME	1664.563064
ID_STAFF	1791.954881
ID_SPECIALITY_CATH	1977.893508

Індексування кожного стовпцю по різному використовує системні ресурси, але кожен з них оптимізує запит. Згідно з формули 1 ймовірність більшого використання системних ресурсів з розглянутих стовбців становить [7]:

$$P_{\max}(A) = \frac{m}{n} = \frac{5}{7} = 0.71$$

де n – загальна кількість розглянутих стовбців, m – стовбці з меншими показниками системних ресурсів.

Згідно з формули 1 ймовірність мінімального використання системних ресурсів методом індексування:

$$P_{\min}(A) = \frac{1}{7} = 0.14$$

В таблиці 7 приведені ймовірності оптимізації запиту методом індексування.

Таблиця 7 – Ймовірності індексування.

Твердження	Ймовірність
Мінімальне використання системних ресурсів з розглянутих стовбців.	0.14
Максимальне використання системних ресурсів з розглянутих стовбців.	0.71

Важливо стежити за тим, щоб індексовані стовбці використовувалися в запитах. Можна створити індекс у будь-якому стовбці; однак, якщо стовпець не використовується в жодній із цих ситуацій, створення індексу на стовбці не збільшує продуктивність і індекс забирає ресурси без потреби [10].

Згідно з формули 2.3 можна розрахувати ймовірність оптимізації методом індексування:

$$M_{PCR5}(X) = 0.71 * 0.14 + \left[\frac{0.71^2 * 0.14}{0.71 + 0.14} + \frac{0.14^2 * 0.71}{0.14 + 0.71} \right] = 0.859$$

Значення навантаження на зчитування/запис є значенням від кількості значень в індексованому стовпці поділеному на кількість записів цього стовпця в індексному файлі.

D. Розрахунок оптимізації запиту методом секціонування

План оптимізації повільного запиту методом секціонування передбачає такі етапи:

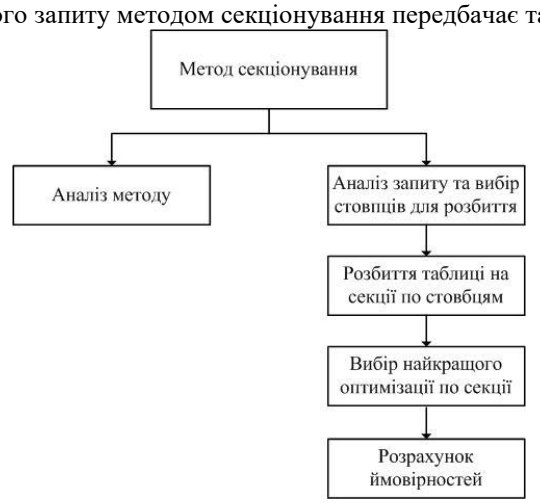


Рисунок 4 - Схема плану виконання оптимізації методом секціонування

Секціонування передбачає розбиття основної таблиці на менші по певним критеріям. Фізично секційна таблиця є одним цілим. Звернення відбуваються до секцій цієї таблиці. Це дозволяє не зчитувати всі рядки таблиці, а шукати інформацію в секціях. Для секціонування треба визначитись з тим, по якому стовпцю проводити розбиття таблиці.

Показники секціонування гірші, ніж секціонування. В таблиці 8 наведені показники секціонування для різних стовпців.

Таблиця 8 – Показники системних ресурсів секціонування

Назва стовпця	Значення системних ресурсів
ID_COURSE	2805.399000
YEAR_PL	3505.414820
SEMESTR_V	3511.282008
ID_STAFF	3413.399000

В таблиці 9 приведено дисковий простір, який займають секції для зберігання даних та загальних розмір таблиць, для яких йде розбиття по секціям.

Таблиця 9 – Дисковий простір для створених секцій

Секція	Розмір таблиці, КіВ	Розмір секцій, КіВ
ID_COURSE	42	416.3
SEMESTR_V	42	406.3
YEAR_PL	42	428.8

Секціонувати інші стовбці не має сенсу, їх значення досить часто повторюються та можуть спричинити складну структуру даних.

Секціонування лише одного стовпця зменшує використання системних ресурсів від не оптимізованого стану. Згідно з формули 1 ймовірність оптимізації запиту методом секціонування становить та мінімального значення використаних системних ресурсів [8]:

$$P_{\min}(A) = \frac{m}{n} = \frac{1}{4} = 0.25$$

Згідно з формули 1 ймовірність максимального використання системних ресурсів методом секціонування:

$$P_{\min}(A) = \frac{m}{n} = \frac{2}{4} = 0.5$$

Ймовірності оптимізації запиту методом секціонування наведено в таблиці 10.

Таблиця 10 – Ймовірності оптимізації запиту методом секціонування

Твердження	Ймовірність
Мінімальне використання системних ресурсів з стовбців.	0.25
Максимальне використання системних ресурсів з розглянутих стовбців.	0.5

Згідно з формули 2 можна розрахувати ймовірність оптимізації методом індексування:

$$M_{PCR5}(X) = 0.25*0.5 + \left[\frac{0.25^2*0.5}{0.25+0.5} + \frac{0.5^2*0.25}{0.5+0.25}\right] = 0.285$$

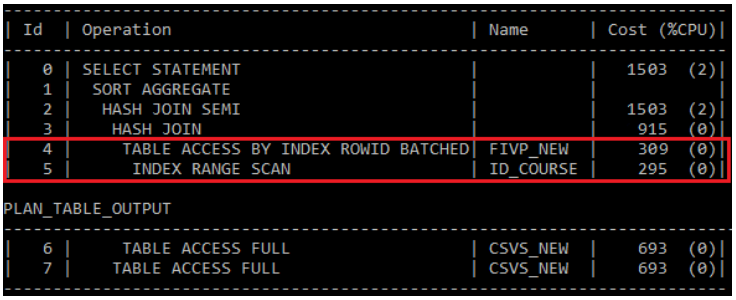
Секціонування показало, що розмір секцій більше розміру самої таблиці та оптимізація запиту цим методом має невелику ймовірність.

Е. Розрахунок оптимізації запиту методом EXPLAIN PLAN

План оптимізації повільного запиту методом планування запитів передбачає такі етапи:

Рисунок 5 - Схема плану виконання оптимізації методом планування запитів

Аналізуючи плани виконання методів (SQL Plus BEST PRACTISE) можна зробити певні висновки про обробку індексування та секціонування. Під час обробки запитів було проаналізовано кожен етап виконання та виявлено найбільшу вартість виконання запиту.

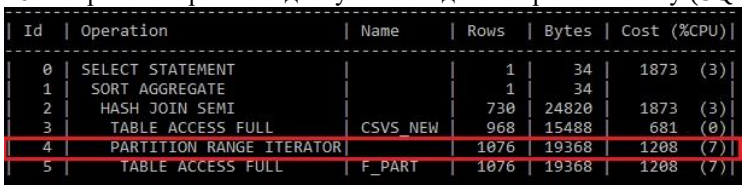


Id	Operation	Name	Cost (%CPU)
0	SELECT STATEMENT		1503 (2)
1	SORT AGGREGATE		
2	HASH JOIN SEMI		1503 (2)
3	HASH JOIN		915 (0)
4	TABLE ACCESS BY INDEX ROWID BATCHED	FIVP_NEW	309 (0)
5	INDEX RANGE SCAN	ID_COURSE	295 (0)

PLAN_TABLE_OUTPUT

6	TABLE ACCESS FULL	CSVS_NEW	693 (0)
7	TABLE ACCESS FULL	CSVS_NEW	693 (0)

Рисунок 6 – Вартість обробки індексування під час обробки запиту (SQL Plus)



Id	Operation	Name	Rows	Bytes	Cost (%CPU)
0	SELECT STATEMENT		1	34	1873 (3)
1	SORT AGGREGATE		1	34	
2	HASH JOIN SEMI		730	24820	1873 (3)
3	TABLE ACCESS FULL	CSVS_NEW	968	15488	681 (0)
4	PARTITION RANGE ITERATOR		1076	19368	1208 (7)
5	TABLE ACCESS FULL	F_PART	1076	19368	1208 (7)

Рисунок 7 – Вартість обробки секціонування під час обробки запиту (SQL Plus)

F. Порівняння розрахованих значень

В таблиці 11 приведені теоретичні та практичні значення навантаження системи з використанням методу індексування. Отримані значення різняться, що свідчить про те, що теоретичні розрахунки не включають в себе навантаження метрик (центральний процесор, зчитування/запис) та інші фактори.

Таблиця 11 – Практичні та теоретичні показники навантаження на систему (cost) з використанням індексування

Твердження	Практичне значення, cost	Теоретичне значення, cost
Вартість запиту з індексуванням	146.348	66
Вартість запиту без індексування	3365.399	107,477

Метод індексування під час обробки запитів використовує менше системних ресурсів ніж секціонування. Індексування зменшує навантаження на зчитування/запис та центральний процесор. Секціонування впливає на дисковий простір бази даних та має гірші показники оптимізації.

Висновки. Під час виконання запиту можуть стати непередбачувані стани системи. Використання системних ресурсів, неправильна побудова схеми бази даних, складність запиту – фактори, які впливають на зниження продуктивності. Для виявлення проблем з базою даних використовуються метрики стану. За допомогою метрик можна переконатися в тому, що система перебуває не в продуктивному стані. З використанням журналу повільних запитів є можливість виявлення повільних запитів та з'ясування час виконання цих запитів, кількості перевірених рядків під час виконання запитів, загальну кількість блокувань. Для отримання точніших даних про стан системи виконується злиття ймовірностей методів оптимізації запиту.

Вплив методів оптимізації запитів було розраховано. Був розрахунок теоретичних та практичних значень навантажень. Практичні дані містять в собі стан системи, тому ці розрахунки показують реальні дані навантаження. Проблема поганої продуктивності бази даних не завжди залежить від показників системи. Необхідно заздалегідь планувати структуру бази даних та кожного запиту.

Література

1. Methods and systems for joining indexes for query optimization in a multi-tenant database [Електронний ресурс]: United States Patent / William Charles Eidson, Jesse Collins // U.S. PATENT DOCUMENTS – 2016 – С.13-25 - Режим доступу до патенту https://patents.google.com/patent/US9405797B2/en

2. Optimization of aggregate queries in database management systems using an early out join when processing min and max functions. [Електронний ресурс]: United States Patent / Edward Gust BRANISH, John F. Hornibrook, Dieu Quang, La Calisto, Paul Zuzarte // U.S. PATENT DOCUMENTS – 2016 – С.7-14 - Режим доступу до патенту https://patents.google.com/patent/US9524317B2/en

3. Database query in a share-nothing database architecture [Електронний ресурс]: United States Patent / Xing Chen, Qing Yun Hao, Yi Jin, Wan Chuan Zhang // U.S. PATENT DOCUMENTS –2015 – С.8-15 - Режим доступу до патенту https://patents.google.com/patent/US9146959B2/en - 2015.

4. How Well do Graph-Processing Platforms Perform? An Empirical Performance Evaluation and Analysis / Yong Guo, Marcin Biczak, Ana Lucia Varbanescu [та ін.] // 2014 IEEE 28th International Parallel & Distributed Processing Symposium – 2014 – С.1-9.

5. Database join optimized for flash storage [Електронний ресурс]: Hewlett Packard Enterprise Development LP. United States Patent / Janet L. Wiener, Stavros Harizopoulos, Mehul A. Shah, Goetz Graefe // U.S. PATENT DOCUMENTS – 2015 – С.13-20 - https://patents.google.com/patent/US20100205351A1/en
6. Основы технологий баз данных: учеб. пособие / Б. А. Новиков, Е. А. Горшкова: под ред. Е. В. Рогова. - М.: ДМК Пресс – 2019 – 21-22с.
7. Теорія ймовірностей та математична статистика. Частина 1. Випадкові події [Електронний ресурс]: Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського» / Веригіна І. В., Островська О. В. // Методичні вказівки – 2018 – С.15-16. – Режим доступу до ресурсу http://ela.kpi.ua/bitstream/123456789/23501/1/NP%28T_Ym%29_1.pdf
8. Non Bayesian Conditioning and Deconditioning / Jean Dezert, Florentin Smarandache // Advances and Applications of DSmT for Information Fusion. Collected Works. Volume 4 – 2015 – С.11-12.
9. UC Davis. Query Processing and Optimization - [Електронний ресурс] / Dept. of Computer Science – 2018 – С.139-143 - Режим доступу до ресурсу: https://web.cs.ucdavis.edu/~green/courses/ecs165a-w11/8-query.pdf
10. Oracle® Database Application Developer's Guide [Електронний ресурс]: Fundamentals 10g Release 1 (10.1) // Oracle Corporation - 2019 - Режим доступу до ресурсу: https://docs.oracle.com/cd/B12037_01/appdev.101/b10795/adfns_in.htm/

References

1. Methods and systems for joining indexes for query optimization in a multi-tenant database [Elektronnyi resurs]: United States Patent / William Charles Eidson, Jesse Collins // U.S. PATENT DOCUMENTS – 2016 – S.13-25 - Rezhym dostupu do patentu https://patents.google.com/patent/US9405797B2/en
2. Optimization of aggregate queries in database management systems using an early out join when processing min and max functions. [Elektronnyi resurs]: United States Patent / Edward Gust BRANISH, John F. Hornibrook, Dieu Quang, La Calisto, Paul Zuzarte // U.S. PATENT DOCUMENTS – 2016 – S.7-14 - Rezhym dostupu do patentu https://patents.google.com/patent/US9524317B2/en
3. Database query in a share-nothing database architecture [Elektronnyi resurs]: United States Patent / Xing Chen, Qing Yun Hao, Yi Jin, Wan Chuan Zhang // U.S. PATENT DOCUMENTS –2015 – S.8-15 - Rezhym dostupu do patentu https://patents.google.com/patent/US9146959B2/en - 2015.
4. How Well do Graph-Processing Platforms Perform? An Empirical Performance Evaluation and Analysis / Yong Guo, Marcin Biczak, Ana Lucia Varbanescu [ta in.] // 2014 IEEE 28th International Parallel & Distributed Processing Symposium – 2014 – S.1-9.
5. Database join optimized for flash storage [Elektronnyi resurs]: Hewlett Packard Enterprise Development LP. United States Patent / Janet L. Wiener, Stavros Harizopoulos, Mehul A. Shah, Goetz Graefe // U.S. PATENT DOCUMENTS – 2015 – S.13-20 - https://patents.google.com/patent/US20100205351A1/en
6. Osnovы tekhnolohyi baz dannыkh: ucheb. posobye / B. A. Novykov, E. A. Horshkova: pod red. E. V. Rohova. - M.: DMK Press – 2019 – 21-22s.
7. Teoriia ymovirnostei ta matematychna statystyka. Chastyna 1. Vypadkovi podii [Elektronnyi resurs]: Natsionalnyi tekhnichnyi universytet Ukrainy «Kyivskyi politekhnichnyi instytut imeni Ihoria Sikorskoho» / Veryhina I. V., Ostrovska O. V. // Metodychni vkazivky – 2018 – S.15-16. – Rezhym dostupu do resursu http://ela.kpi.ua/bitstream/123456789/23501/1/NP%28T_Ym%29_1.pdf
8. Non Bayesian Conditioning and Deconditioning / Jean Dezert, Florentin Smarandache // Advances and Applications of DSmT for Information Fusion. Collected Works. Volume 4 – 2015 – S.11-12.
9. UC Davis. Query Processing and Optimization - [Elektronnyi resurs] / Dept. of Computer Science – 2018 – C.139-143 - Rezhym dostupu do resursu: https://web.cs.ucdavis.edu/~green/courses/ecs165a-w11/8-query.pdf
10. Oracle® Database Application Developers Guide [Elektronnyi resurs]: Fundamentals 10g Release 1 (10.1) // Oracle Corporation - 2019 - Rezhym dostupu do resursu: https://docs.oracle.com/cd/B12037_01/appdev.101/b10795/adfns_in.htm/

The article discusses the steps that are aimed at fixing slow queries and optimizing them. Basic metrics are identified and selected that provide information about the system status during query processing. Thresholds were set for the metrics required to run the slow query fixation technique. Query optimization methods are analyzed. The DSmT mathematical method was selected to calculate the probability of query optimization by the selected optimization methods. The chosen method gives theoretical data on the state of the system, which are most effective in metrics. Using a query execution plan (EXPLAIN PLAN), which query optimization method uses fewer resources. To analyze the status of metrics with the selected optimization methods, the theoretical and practical load and cost of each request were calculated. It is established that the calculated loads differ, which indicates that the practical indicators take into account the state of the system. With the method selected, the percentage of data that goes into the database cache was recorded.
Keywords: database, metrics, cost, DSmT, indexing, partitioning, probability

Нестеров М.В. – старший викладач кафедри комп'ютерних наук та інженерій СНУ ім. В.Даля
Бакітько Д.Е. – магістр кафедри комп'ютерних наук та інженерій СНУ ім. В.Даля