ДОДАТОК А

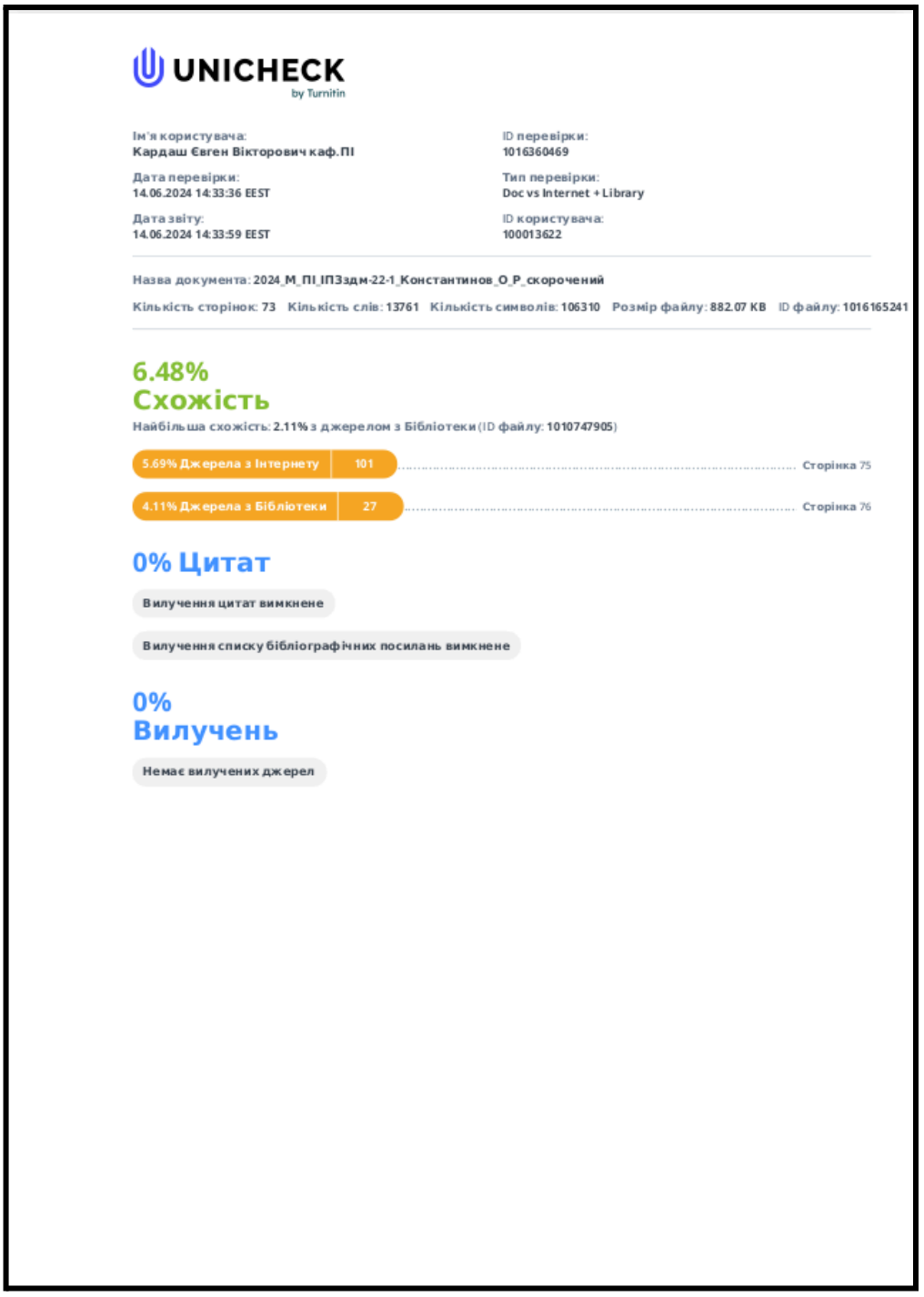
Перелік джерел посилання за науковими напрямами керівника та науковців кафедри програмної інженерії

21. Константинов Олександр, Вечур Олександр. «Дослідження та оптимізація продуктивності баз даних у розподілених обчислювальних мережах». Збірник 25ої Міжнародної науково- технічної конференції «Current Trends in the Development of Scientific Research in Today's Conditions» (ICT-2024) Комп'ютерна інженерія секція (дата звернення: 15.05.2024).

22. Chetverikov G., Puzik O., Vechirska I. Multiple-valued structures of intellectual systems //Proceedings of the with Internations Computer Sciences and Information Technologies (CSIT). 2016, 7589907. -pp. 204-207 (дата звернення: 19.05.2024).

ДОДАТОК Б

Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ



UNICHECK
by Turnitin

Ім'я користувача:
Кардаш Євген Вікторович каф.ПІ

ID перевірки:
1016360469

Дата перевірки:
14.06.2024 14:33:36 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
14.06.2024 14:33:59 EEST

ID користувача:
100013622

Назва документа: 2024_М_ПІ_ІПЗздм-22-1_Константинов_О_Р_скорочений

Кількість сторінок: 73 Кількість слів: 13761 Кількість символів: 106310 Розмір файлу: 882.07 KB ID файлу: 1016165241

6.48%
Схожість

Найбільша схожість: 2.11% з джерелом з Бібліотеки (ID файлу: 1010747905)

5.69% Джерела з Інтернету 101 ..... Сторінка 75

4.11% Джерела з Бібліотеки 27 ..... Сторінка 76

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Рисунок Б.1 – Титульний аркуш звіту результатів перевірки на унікальність тексту в базі ХНУРЕ

ДОДАТОК В

Слайди презентації

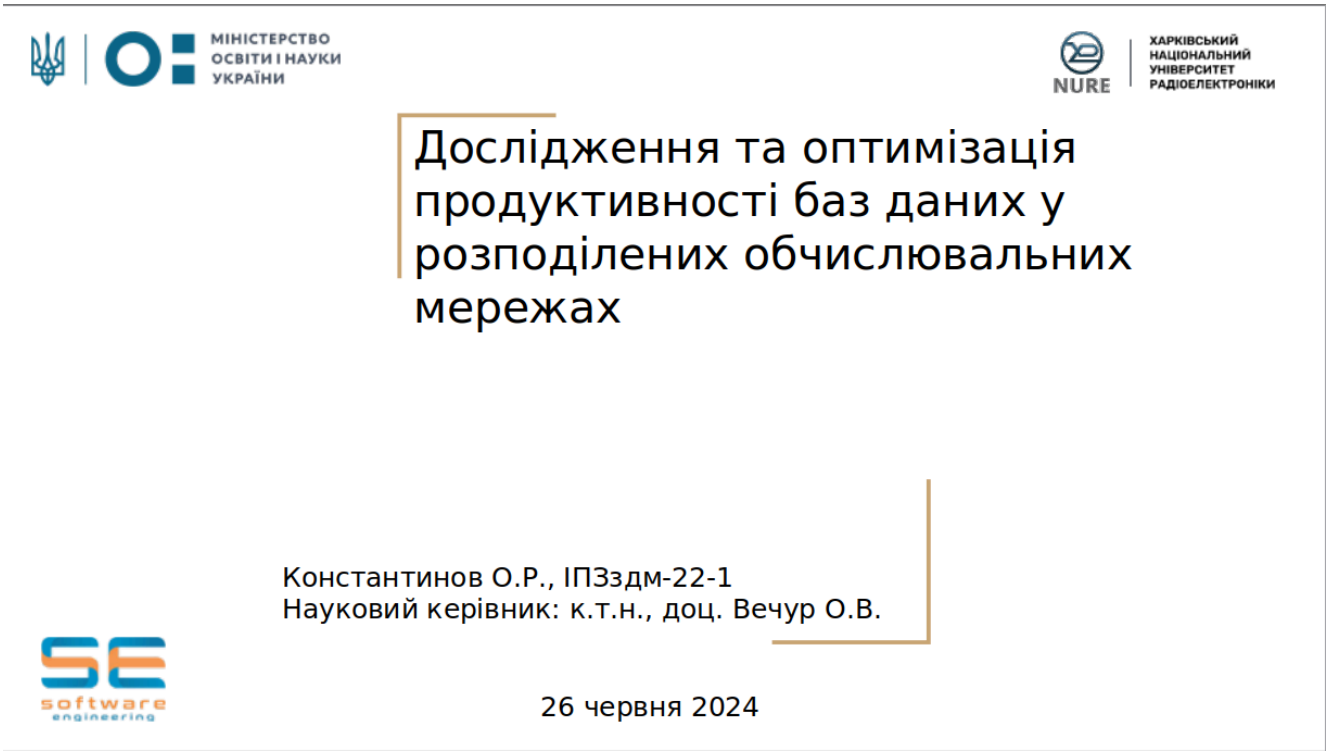


Рисунок В.1 – Перший слайд презентації

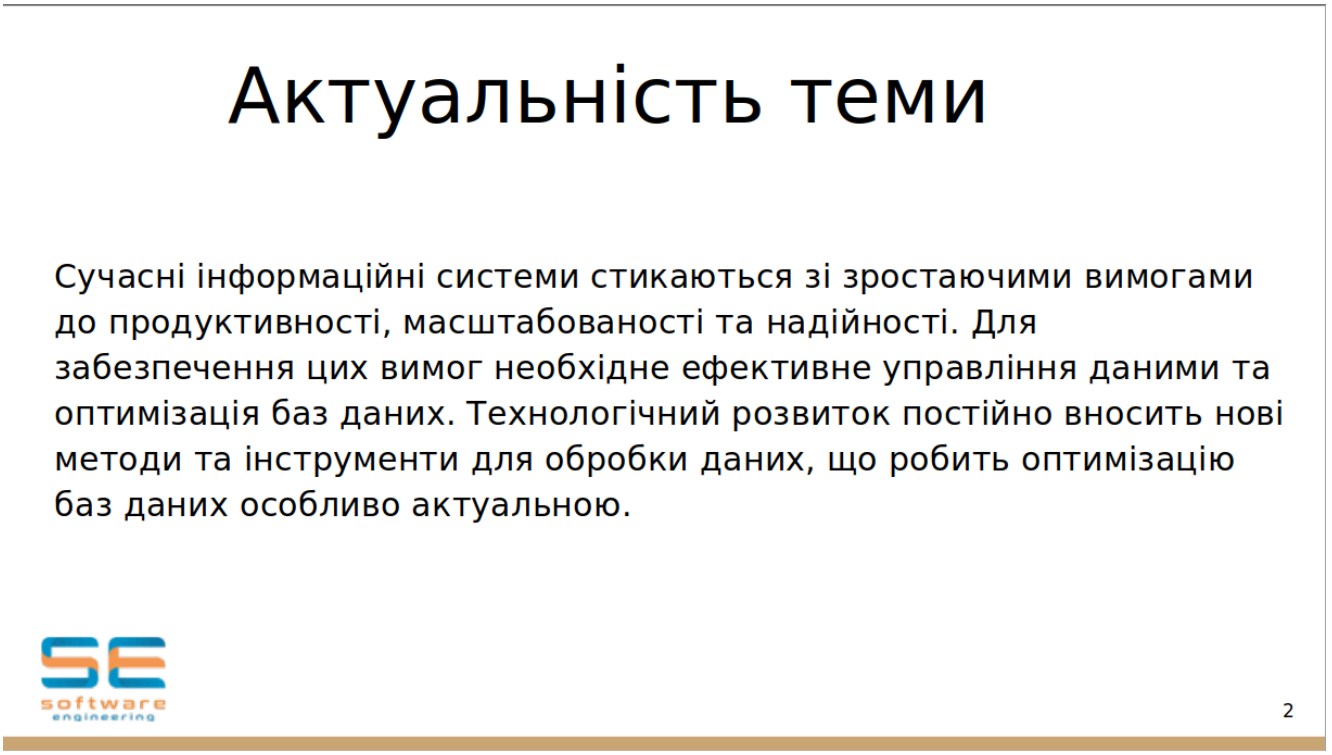


Рисунок В.2 – Другий слайд презентації

Стан розвитку галузі

На сьогоднішній день існує два основних типи баз даних, які широко використовуються в інформаційних системах:

- Реляційні бази даних (PostgreSQL, MySQL): Традиційно використовуються для управління структурованими даними. Вони забезпечують високу надійність та підтримку складних запитів.
- NoSQL бази даних (MongoDB): Здобули популярність завдяки гнучкості та здатності обробляти великі обсяги неструктурованих даних. Вони дозволяють досягти високої продуктивності та масштабованості в розподілених системах.

3

Рисунок В.3 – Третій слайд презентації

Напрям та об’єкт дослідження

Основний напрям дослідження полягає в розробці та впровадженні методів оптимізації баз даних для підвищення їх продуктивності та ефективності в інформаційних системах.

Об'єктом дослідження є бази даних різних типів, включаючи реляційні бази даних (PostgreSQL, MySQL) та NoSQL бази даних (MongoDB).

4

Рисунок В.4 – Четвертий слайд презентації

Основні джерела літератури

1) MongoDB Documentation
2) JSQLParser Documentation
3) "Designing Data-Intensive Applications" by Martin Kleppmann
4) "MongoDB: The Definitive Guide" by Kristina Chodorow and Michael Dirolf
5) "NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence" by Pramod J. Sadalage and Martin Fowler

SE software engineering

5

Рисунок В.5 – П’ятий слайд презентації

Постановка задачі

Сучасні інформаційні системи часто стикаються з проблемами низької продуктивності та ефективності через неефективне управління даними та використання застарілих методів оптимізації. Це негативно впливає на швидкість доступу до даних, їх обробку та загальну продуктивність системи.

SE software engineering

6

Рисунок В.6 – Шостий слайд презентації

Опис очікуваних результатів

- Підвищення продуктивності баз даних: Впровадження оптимізованих структур та алгоритмів, які забезпечать швидший доступ до даних та їх обробку.
- Покращення масштабованості: Використання мікросервісної архітектури та методів горизонтального і вертикального масштабування для ефективної обробки великих обсягів даних.
- Зменшення часу доступу до даних: Впровадження кешування для швидкого доступу до часто використовуваних даних, що знизить навантаження на базу даних.
- Оптимізація запитів у розподілених базах даних: Розробка та впровадження методів для ефективної обробки запитів, що підвищить продуктивність і знизить час відповіді системи.
- Трансляція запитів з реляційної структури у формат NoSQL: Створення алгоритмів для конвертації запитів, що забезпечить гнучкість системи та можливість використання різних типів баз даних в залежності від потреб конкретного застосування.
-

software engineering

7

Рисунок В.7 – Сьомий слайд презентації

Методологія

- У даному дослідженні основна мета полягає в конвертації SQL запитів в еквівалентні запити для MongoDB, що використовують агрегаційні функції. Основні кроки дослідження включають:
- Розбір SQL запиту: Використання JSQLParser для синтаксичного розбору SQL запитів та витягнення необхідних елементів запиту, таких як SELECT, FROM, WHERE, GROUP BY, HAVING, ORDER BY та JOIN.
- Формування словника запиту MongoDB: Створення структури, що зберігає всі елементи запиту у зручному для обробки форматі.
- Визначення структури бази даних MongoDB: Використання MongoDB Java Driver для з'єднання з MongoDB та визначення доступних колекцій.
- Синтез MongoDB запиту: Використання даних із словника для створення агрегаційного запиту для MongoDB за допомогою MongoDB Aggregation Framework.

software engineering

8

Рисунок В.8 – Восьмий слайд презентації

Інструментарій та технології, використані в роботі

1. Мова програмування Java: Основна мова програмування для реалізації конвертації.
2. JSQLParser: Бібліотека для розбору SQL запитів, що дозволяє витягувати структуру запиту та його елементи.
3. MongoDB Java Driver: Офіційний драйвер для роботи з MongoDB, що надає можливості для виконання запитів та управління базою даних.
4. Середовище розробки:
   1) IntelliJ IDEA: Використовується як інтегроване середовище розробки (IDE) для написання та тестування Java коду.
   2) Maven: Система керування проектами та залежностями, що використовується для завантаження необхідних бібліотек, таких як JSQLParser та MongoDB Java Driver.
5. MongoDB: NoSQL база даних, що використовується для зберігання даних у форматі JSON-подібних документів.

software engineering

9

Рисунок В.9 – Дев’ятий слайд презентації

Опис використаних методів

1. Розбір SQL запиту:
2. Метод extractNonKeyTokens використовується для розбору SQL запиту та витягнення неключових токенів.
3. Метод populateMongoDBDictionary відповідає за формування словника запиту MongoDB, де зберігаються всі елементи SQL запиту
4. Формування словника запиту MongoDB:
5. Клас MongoDBQueryDictionary використовується для зберігання елементів запиту.
6. Додаткові методи для додавання SELECT, FROM, WHERE, JOIN, GROUP BY, HAVING та ORDER BY умов до словника.
7. Визначення структури бази даних MongoDB:
8. Клас MongoDBStructure відповідає за підключення до MongoDB та визначення доступних колекцій.
9. Синтез MongoDB запиту:
10. Метод buildMongoDBQuery використовується для створення агрегаційного запиту MongoDB на основі даних із словника.
11. Додатковий метод parseCondition для перетворення умов SQL в Bson.

software engineering

10

Рисунок В.10 – Десятий слайд презентації

Ключові компоненти архітектури

1. Клієнтське середовище: Інтерфейс користувача для введення SQL запитів та отримання результатів конвертації.
2. Серверне середовище: Серверна частина, що обробляє SQL запити, використовуючи JSQLParser для аналізу та формування MongoDB запитів.
3. MongoDB база даних: NoSQL база даних для зберігання та обробки даних.
4. Інтерфейс користувача: Графічний інтерфейс для введення SQL запитів та відображення результатів.
5. Обробка запитів (Java та JSQLParser): Логіка серверної частини для аналізу та конвертації SQL запитів.
6. Формування запиту MongoDB: Компонент для створення MongoDB запитів з використанням даних, отриманих від серверної частини.
7. Виконання запиту в MongoDB: Виконання сформованого запиту в MongoDB та повернення результатів у клієнтське середовище.

Ця архітектура забезпечує ефективний обмін даними та обробку запитів від користувача до бази даних MongoDB через проміжні етапи обробки та конвертації.

software engineering

11

Рисунок В.11 – Одинадцятий слайд презентації

Зміст проведеного експерименту

Методи:
- Аналіз SQL запитів
- Конвертація SQL до MongoDB запитів
- Виконання та оцінка результатів

Вхідні дані:
- Набір SQL запитів різної складності

Критерії:
- Точність конвертації SQL запитів до MongoDB
- Час виконання кожного етапу конвертації

Послідовність:
- Аналіз SQL запитів для визначення структури та умов.
- Формування відповідних MongoDB запитів.
- Виконання та оцінка ефективності конвертації.

Вимірювання:
- Час виконання кожного етапу.
- Точність відповідності результатів конвертації.

software engineering

12

Рисунок В.12 – Дванадцятий слайд презентації

Результати експерименту

Таблиця 1: Точність конвертації SQL до MongoDB

Номер запиту	SQL Запит	MongoDB Запит	Успішно конвертовано
1	SELECT Customer, Product_name, SUM(Quantity) FROM Orders GROUP BY Customer, Product_name	...	Так
2	SELECT Customer, Product_name, Date_order FROM Orders WHERE Date_order > '2024-01-01' ORDER BY Date_order DESC	...	Так
3	SELECT Customer, COUNT(*) AS TotalOrders FROM Orders GROUP BY Customer HAVING TotalOrders > 10	...	Так
4	SELECT o.Customer, p.Product_name, p.Price FROM Orders o JOIN Products p ON o.Product_id = p.Product_id	...	Так
5	SELECT o.Customer, o.Date_order, p.Product_name FROM Orders o LEFT JOIN Products p ON o.Product_id = p.Product_id	...	Так
6	SELECT Customer, Product_name, SUM(Quantity) FROM Orders WHERE Date_order BETWEEN '2023-01-01' AND '2024-12-31' GROUP BY Customer, Product_name	...	Так
7	SELECT Customer, Product_name, COUNT(*) AS TotalOrders FROM Orders GROUP BY Customer, Product_name HAVING TotalOrders > 5 ORDER BY TotalOrders DESC	...	Так

Таблиця 2: Час виконання етапів конвертації

Етап конвертації	Час виконання (сек)
Аналіз SQL запитів	0.5
Формування MongoDB запитів	1.2
Виконання MongoDB запитів	4.3

13

Рисунок В.13 – Тринадцятий слайд презентації

Аналіз отриманих результатів

1. Співставлення з цілями дослідження:
2. Конвертація різноманітних SQL запитів у відповідні запити MongoDB була успішною і точною. Всі ключові елементи SQL, такі як SELECT, FROM, WHERE, JOIN, GROUP BY, HAVING та ORDER BY, були адекватно перетворені у відповідні структури MongoDB, що відповідають логіці документ-орієнтованої бази даних.
3. Висновки:
4. Виявлено, що MongoDB підтримується добре інтегрованими бібліотеками та інструментами, які дозволяють ефективно виконувати перетворення складних SQL запитів у запити MongoDB без втрати функціональності.
5. Процес конвертації дозволяє врахувати особливості та обмеження MongoDB, зокрема, підтримку транзакцій та обмежень щодо зовнішніх ключів.

14

Рисунок В.14 – Чотирнадцятий слайд презентації

Аналіз отриманих результатів

3. Інтерпретація результатів:
4. На основі результатів можна зробити висновок, що MongoDB є відмінним вибором для проектів, де важлива гнучкість схеми та масштабованість.
5. Проте перед впровадженням MongoDB в проект необхідно врахувати додаткові витрати на навчання та підготовку команди, а також переглянути архітектурні рішення, оскільки перехід від реляційної бази даних може потребувати адаптації існуючих процесів.
6. Вплив на існуючі теорії та практики:
7. Використання MongoDB впливає на стратегії розробки та управління даними, оскільки вимагає нового підходу до моделювання даних та оптимізації запитів.
8. Зміни в архітектурних рішеннях можуть включати перегляд розподілу даних, оптимізацію запитів та використання відповідних індексів для покращення продуктивності і скалабельності системи.

Цей детальний аналіз демонструє позитивні аспекти використання MongoDB і підкріплює необхідність обережного підходу до впровадження нових технологій в проєкт.

software engineering

15

Рисунок В.15 – П’ятнадцятий слайд презентації

Публікація результатів

software engineering

16

Рисунок В.16 – Шістнадцятий слайд презентації

Підсумки

Отримані результати підтвердили реалістичність та корисність конвертації SQL запитів до формату MongoDB за допомогою використання бібліотеки JSQLParser та інструментів MongoDB Java Driver. Виявлено, що MongoDB може ефективно обробляти різноманітні SQL запити, забезпечуючи гнучкість у схемі та масштабованість управління даними.

Подальші можливості дослідження включають детальне порівняння продуктивності MongoDB з реляційними базами даних, а також оптимізацію запитів і використання індексів для підвищення ефективності. Майбутні дослідження також можуть розширити аналіз впливу вибору бази даних на архітектурні рішення та стратегії розробки додатків, що спрямовані на оптимальне управління даними та забезпечення масштабованості системи.

SE software engineering

17

Рисунок В.17 – Сімнадцятий слайд презентації

Дякую за увагу!

18

Рисунок В.18 – Вісімнадцятий слайд презентації

ДОДАТОК Г

Апробація результатів роботи

Список використаних джерел

1. API Governance best practices, Gravetee.io 2023. Kelsey Ellis. URL: https://www.gravitee.io/blog/api-governance-best-practices (дата звернення: 16.02.2024).

RESEARCH AND OPTIMIZATION OF DATABASE PERFORMANCE IN DISTRIBUTED COMPUTING NETWORKS

Vechur Oleksandr
Ph.D., Associate Professor
alexander.vechur@nure.ua
Konstantynov Oleksandr
Master student
oleksandr.konstantynov.cpe@nure.ua
Department of Software Engineering
Kharkiv National University of Radioelectronics, Ukraine

Ціль дослідницької роботи полягає у створенні формалізованих методів для трансляції даних з реляційної бази даних у формат NoSQL типу «ключ-документ», а також в оптимізації її структури та запитів для покращення швидкості обробки та зберігання даних.

У сучасному цифровому світі обробка великих обсягів даних стала не тільки нормою, але й ключовим елементом успіху для багатьох організацій та підприємств різних галузей. Підтримка високої продуктивності баз даних у розподілених обчислювальних мережах важлива для забезпечення швидкого доступу до інформації та ефективного виконання операцій.

Дослідницька робота спрямована на дослідження та оптимізацію продуктивності баз даних у розподілених обчислювальних мережах. Це дослідження стало актуальним через стрімкий розвиток технологій, який супроводжується збільшенням обсягів даних, а також через поширення використання розподілених обчислювальних середовищ у великих корпораціях та малих підприємствах.

Розподілені обчислювальні мережі використовуються для обробки великих обсягів даних, розподілу завдань обробки між різними вузлами та забезпечення доступності даних в різних місцях одночасно. Проте, ефективність роботи баз даних у таких середовищах залежить від різноманітних факторів, включаючи алгоритми реплікації даних, стратегії маршрутизації запитів та механізми синхронізації даних.

Метою даної дослідницької роботи є проведення комплексного дослідження продуктивності баз даних у розподілених обчислювальних

Рисунок Г.1 – Тези магістерського дослідження з 25ої Міжнародної науково-технічної конференції «Current Trends in the Development of Scientific Research in Today's Conditions» (ICT-2024) Комп'ютерна інженерія секція. Друга сторінка

мережах та розробка оптимальних методів їх оптимізації. Для досягнення цієї мети передбачається:

1. Аналіз сучасного стану технологій баз даних у розподілених середовищах: огляд існуючих методів реплікації, розподілення та доступу до даних у розподілених системах, виявлення їх переваг та недоліків.

2. Експериментальне дослідження продуктивності: виконання серії експериментів для оцінки продуктивності різних алгоритмів та стратегій роботи баз даних у розподілених обчислювальних мережах, а також для виявлення чинників, що впливають на їх ефективність.

3. Розробка оптимальних підходів до оптимізації: на основі отриманих результатів розроблення нові стратегії та методи оптимізації роботи баз даних у розподілених середовищах з урахуванням специфіки завдань та вимог користувачів.

Ця робота має важливе теоретичне та практичне значення. Результати дослідження можуть бути використані для покращення продуктивності баз даних у розподілених обчислювальних мережах, що сприятиме підвищенню ефективності обробки даних та забезпеченню більш високої якості обслуговування користувачів. Враховуючи складність та масштабність проблеми, дослідження, яке виконано в рамках цієї роботи, є кроком до подальших досягнень в області оптимізації обробки даних у розподілених обчислювальних мережах.

Наукова частина дослідницької роботи зосереджена на створенні та описі методу трансформації запитів з SQL до NoSQL. Цей метод реалізований з використанням баз даних PostgreSQL та MongoDB та складається з чотирьох етапів:

- розбір(парсинг) початкового SQL-запиту;
- створення словника з елементів запиту для MongoDB;
- переробка представлень словника з урахуванням структури бази даних MongoDB;
- формування вихідного запиту з елементів словника.

Метод був реалізований на мові програмування Java, а кожен етап був детально описаний. Також розглянуто приклади використання розробленого методу для трансформації різних запитів до бази даних MongoDB з різними структурами даних.

Під час перетворення запитів з однієї мови на іншу можуть виникати різноманітні труднощі, наприклад, відмінності у синтаксисі мов запитів. Це зумовлено тим, що кожен тип баз даних має свою власну граматику для обробки даних. Ще однією складністю є різниця у моделях зберігання даних кожної бази даних. Наприклад, реляційні дані зберігаються у вигляді таблиць, які пов'язані між собою, тоді як дані в MongoDB організовані у колекціях, де кожен рядок вважається окремим документом.

Далі у процесі перетворення бази даних із формату SQL у формат NoSQL, такий як MongoDB, виконується створення методу автоматичного перекладу запиту з SQL у формат NoSQL, який задовольняє такі умови: Запит до бази даних

Рисунок Г.2 – Тези магістерського дослідження з 25ої Міжнародної науково-технічної конференції «Current Trends in the Development of Scientific Research in Today's Conditions» (ICT-2024) Комп'ютерна інженерія секція. Друга сторінка

MongoDB автоматично буде створений з урахуванням структури використовуваної бази даних MongoDB. Результати виконання запиту в MongoDB після перекладу мають збігатися з результатами виконання вихідного SQL-запиту.

Метод автоматичної трансляції запитів з формату SQL в формат MongoDB, розроблений у рамках дослідницької роботи, враховує структуру баз даних PostgreSQL і MongoDB. Вхідні дані включають код запиту до бази даних PostgreSQL та метадані про їхню структуру. Вихідні дані - це код запиту до бази даних MongoDB.

Першим кроком в реалізації задумки є парсинг. Парсинг - це процес аналізу комп'ютерних мов програмування та запитів за допомогою формальної граматики. Мета цього етапу полягає в тому, щоб розбити код вхідного запиту на токени(слова). Токенами запитів є складові частини запитів (ключові слова, параметри, числа, знаки). Парсинг запитів має на меті не тільки розділити запит на окремі токени, але й визначити роль кожного токена. Один SQL-запит може мати кілька варіантів написання. Визначення правильної ролі кожного токена під час парсингу є складним завданням, яке неможливо вирішити за допомогою стандартних функцій обробки рядків у класичних високорівневих мовах програмування, таких як C++ або Java, а також за допомогою регулярних виразів. У зв'язку з цим у дослідницькій роботі був розроблений парсер, написаний на Java.

Граматика SQL-запитів є ключовим елементом у процесі парсингу вхідного запиту PostgreSQL. Для парсингу запиту його необхідно розділити на граматичні підзапити PostgreSQL, такі як: SELECT, FROM, WHERE, GROUP BY, HAVING та ORDER BY.

Нижче перераховані та схематично описані дії стосовно "розбору" основних запитів.

Рисунок 1. Граматика "SELECT"

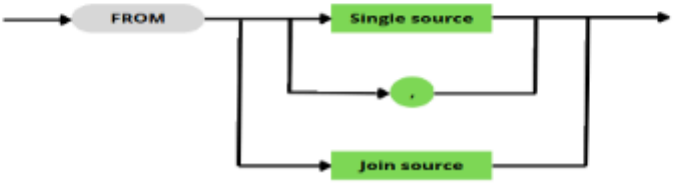


Рисунок 2. Граматика "FROM"

Рисунок Г.3 – Тези магістерського дослідження з 25ої Міжнародної науково-технічної конференції «Current Trends in the Development of Scientific Research in Today's Conditions» (ICT-2024) Комп'ютерна інженерія секція. Друга сторінка

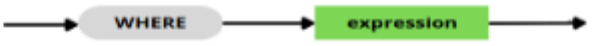


Рисунок 3. Граматика "WHERE"

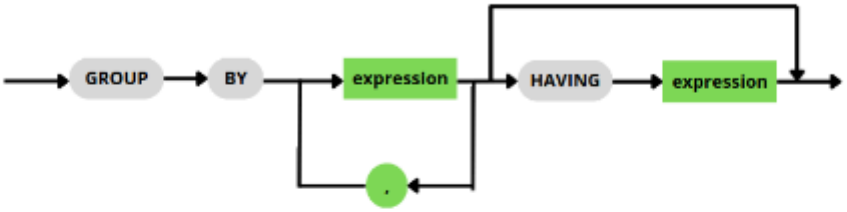


Рисунок 4. Граматика "GROUP BY HAVING"

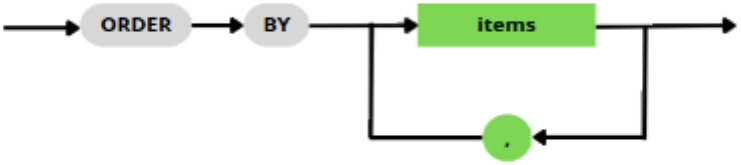


Рисунок 5. Граматика "ORDER BY"

Після парсингу початкового запиту отримується список токенів, де неключові токени не прив'язані до ключових слів SQL. Наступним етапом є створення списків неключових токенів із загального списку токенів.

Нижче приведений приклад розбиття запросу.

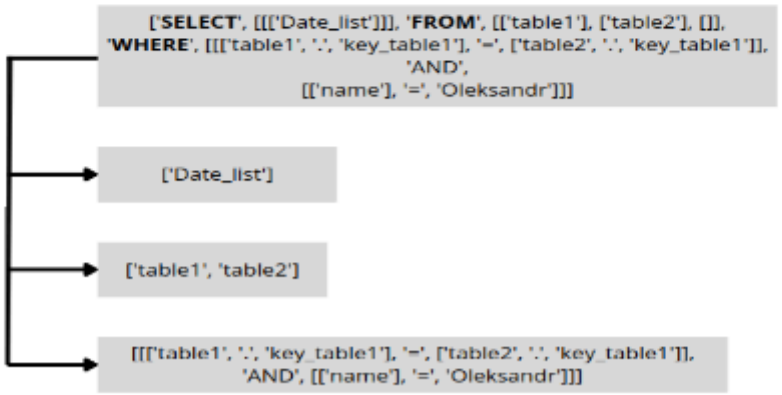


Рисунок 6. Приклад результату парсингу.

Рисунок Г.4 – Тези магістерського дослідження з 25ої Міжнародної науково-технічної конференції «Current Trends in the Development of Scientific Research in Today's Conditions» (ICT-2024) Комп'ютерна інженерія секція. Друга сторінка

Формування словника пропозицій запиту MongoDB: На цьому етапі формується словник для побудови запиту до MongoDB на основі результатів парсингу на першому кроці. У базі даних MongoDB для отримання даних використовується метод find(), котрий є аналогом оператора SELECT у PostgreSQL. Однак проста заміна оператора SELECT на метод find() у деяких випадках може призвести до неправильних результатів, наприклад, при виконанні запитів до колекцій з вкладеними документами. Тому замість методу find() доцільніше буде використати метод aggregate().

Враховуючи синтаксис процедури агрегування даних в MongoDB, структура словника пропозицій повинна бути такою як на рисунку 7.

```
S = {
    "collection": ["collection 1", ..., "collection n"],
    "fields": {"field 1": 1|0, ..., "field n": 1|0},
    "spec": {"expression 1", ..., "expression n"},
    "group_by": false|true,
    "having": false|true,
    "sort": false|true
}
```

Рисунок 7. Структура словника пропозицій

У словнику пропозицій запиту MongoDB кожній пропозиції призначається одне з двох значень: "True" чи "False". Тільки пропозиції зі значенням "True" будуть включені до кінцевого запиту MongoDB.

Після створення словника на кроці 2 необхідно переглянути пропозиції, враховуючи структуру бази даних MongoDB. Це означає визначення полів даних, що використовуються у запиті до бази даних MongoDB, їхню належність до колекцій та розгляд можливості їхнього розташування у самому документі або вкладеному. Для цього спочатку слід вивчити структуру бази даних MongoDB. Це можна зробити, уявивши колекцію у формі дерева, де вузлами будуть поля в колекції. Кожний вузол матиме три параметри: назва вузла, батьківський вузол (якщо він існує) та дочірні вузли (якщо вони існують).

Глибина колекції - це кількість рівнів вкладених документів у ній. Якщо n = 1, це означає, що у колекції відсутні вкладені документи.

Далі наведений алгоритм реалізації полів даних в вузлі(рисунок 8) з назвою колекції як вхідними даними та вузлами полів як результатом.

```
1. Отримати шаблон даних з колекції;
2. Поки кожен ключ та його значення в шаблоні даних виконується, робити:
3.     Якщо тип даних(значення) == словник, то
4.         sub_node = рекурсивно_компілювати(значення, ключ, поточний_вузол);
5.         Додати sub_node до списку sub_nodes поточного вузла;
6.     Якщо тип даних(значення) == список, то
7.         sub_node = рекурсивно_компілювати(значення[0], ключ, поточний_вузол);
8.         Додати sub_node до списку sub_nodes поточного вузла;
9.     В іншому випадку,
10.        new_node = Створити_вузол(ключ, поточний_вузол);
11.        Додати new_node до списку sub_nodes поточного вузла;
12. Повернути поточний_вузол.
```

Рисунок 8. Алгоритм переміщення полів даних в вузлі

Рисунок Г.5 – Тези магістерського дослідження з 25ої Міжнародної науково-технічної конференції «Current Trends in the Development of Scientific Research in Today's Conditions» (ICT-2024) Комп'ютерна інженерія секція. Друга сторінка

Кінцевим кроком буде синтез вихідного запиту. На цьому етапі буде складатися запит з частин словника, відповідно до структури запиту "aggregate" в MongoDB. Кінцевий результат - це запит до бази даних MongoDB. Оскільки "aggregate" не вимагає використання всіх компонентів, процес буде відбуватися послідовно: спочатку перевіряється наявність цих частин у словнику, а потім вони додаються до синтаксису запиту.

Роботу можна описати як метод, що дозволяє перетворювати запити з SQL на MongoDB, враховуючи структуру бази даних. Цей процес включає чотири етапи: аналіз вихідного SQL-запиту, створення словника запитів MongoDB, конвертацію цього словника з врахуванням структури бази даних MongoDB та формування остаточного запиту. Цей метод відіграє важливу роль у вирішенні завдань консолідації різнотипових баз даних.

Висновок: у дослідницькій роботі була розроблена сукупність формалізованих методів для трансляції баз даних із формату реляційної БД у формат NoSQL та оптимізації їх структури з метою прискорення обробки даних і зменшення пам'яті, що використовується для зберігання даних. Усі поставлені задачі були вирішені.

Рисунок Г.6 – Тези магістерського дослідження з 25ої Міжнародної науково-технічної конференції «Current Trends in the Development of Scientific Research in Today's Conditions» (ICT-2024) Комп'ютерна інженерія секція. Друга сторінка

ДОДАТОК Д

Експертний висновок результатів перевірки кваліфікаційної роботи на відповідність оформлення вимогам ДСТУ 3008: 2015

1

Експертний висновок результатів перевірки кваліфікаційної роботи

студент (посада) | програмної інженерії (кафедра) | ІПЗздм-22-1 (група)

Константинов Олександр Романович
(прізвище, ім'я, по батькові)

Зауваження

Пункт ДСТУ 3008-2015	Зміст пункту	Сторінка кваліфікаційної роботи
1	2	3
	7.1 Загальні положення	
	7.3 Нумерація сторінок звіту	
	7.4 Нумерація розділів, підрозділів, пунктів, підпунктів	
	7.5 Рисунки	
	7.6 Таблиці	
	7.7 Переліки	
	7.8 Примітки	
	7.9 Виноски	
	7.10 Формули та рівняння	
	7.11 Посилання	
	7.13 Список авторів	
	7.14 Скорочення та умовні позначки	
	7.15 Додатки	

зауважень немає

Експерт _____ (підпис) Олена ОЛІЙНИК (прізвище, ініціали)

23.06.2024